Spectrum Hole Prediction And White Space Ranking For Cognitive Radio Network Using An Artificial Neural Network

Sunday Iliya, Eric Goodyer, Mario Gongora, John Gow

Abstract: With spectrum becoming an ever scarcer resource, it is critical that new communication systems utilize all the available frequency bands as efficiently as possible in time, frequency and spatial domain. rHowever, spectrum allocation policies most of the licensed spectrums grossly underutilized while the unlicensed spectrums are overcrowded. Hence, all future wireless communication devices beequipped with cognitive capability to maximize quality of service (QoS); require a lot of time and energartificial intelligence and machine learning in cognitive radio deliver optimum performance. In this paper, we proposed a novel way of spectrum holes prediction using artificial neural network (ANN). The ANN was trained to adapt to the radio spectrum traffic of 20 channels and the trained network was used for prediction of future spectrum holes. The input of the neural network consist of a time domain vector of length six i.e. minute, hour, date, day, week and month. The output is a vector of length 20 each representing the probability of the channel being idle. The channels are ranked in order of decreasing probability of being idleminimizing We assumed that all the channels have the same noise and quality of service; and only one vacant channel is needed for communication. The result of the spectrum holes search using ANN was compared with that of blind linear and blind stochastic search and was found to be superior. The performance of the ANN that was trained to predict the probability of the channels being idle outperformed the ANN that will predict the exact channel states (busy or idle). In the ANN that was trained to predict the exact channels states, all channels predicted to be idle are randomly searched until the first spectrum hole was found; no information about search direction regarding which channel should be sensed first.

Index Terms: Cognitive Radio, Neural Network, Primary User, White Space

1 Introduction

SINCE most of the licensed spectrums are grossly underutilized while the unlicensed spectrums are overcrowded; the present spectrum scarcity is the direct consequence of spectrum allocation policy and not the fundamental lack of spectrum. As a result, the US Federal Communication Commission (FCC) has approved the use of some selected channels from 54MHz to 862MHz of VHF/UHF TV band for cognitive radio transmission [1],[2],[3]. In order to circumvent the problem of spectrum scarcity and underutilization, a new paradigm of wireless communication need to be adopted. Advanced Cognitive Radio (CR) or Adaptive Spectrum Sharing (ASS) is one of the ways to optimize our wireless communications technologies for high data rates while maintaining users desired quality of service (QoS) requirements. CR is a radio equipped with the capability of awareness of it radio frequency (RF) environment, perception, adaptation and learning [4]. Irrespective of the definition of CR, it has the followings basic characteristics: observation, adaptability and intelligence. CR is the key enabling technology for dynamic spectrum access and a promising solution for the present problem of spectrum scarcity and underutilization. Cognitive radio network consists of two users i.e. the primary users (PU) who are the incumbent licensed owners of the spectrum and the cognitive radio commonly called the secondary users (SU) who intelligently and opportunistically access the unused licensed spectrums on no interference base (the overlay) or as long as the interference to the PU is below a given threshold which will not be harmful to the PU nor degrade the QoS requirements of the PU (the underlay) [5]. White spaces or spectrum holes referred to the band of frequencies assigned to primary or licensed owners which at that time and in a particular geographical location are currently idle (not used by the licensed owners). There are four things involves in cognitive radio network, these are: spectrum sensing, spectrum decision, spectrum sharing, and spectrum mobility [6],[7]. In spectrum sensing, the CR senses the PU spectrum using either energy detector, cyclostationary features detector, match filter detector, eigen values detector, etc, to sense the occupancystatus of the PU. Base on the sensing results, the SU (CR) will take a decision using binary classifier to classify the PU channels (spectrums) as either busy or idle there by identifying the white spaces. Spectrum sharing deal with how to efficiently share the available white spaces (spectrum holes) among other CR (SU) within a given geographical location at a given period of time while spectrum mobility is the ability of the CR to vacate the channel when the PU reclaimed ownership of the channel and search for another spectrum hole to communicate. During the withdrawal period, the CR should maintain seamless communication.There are different types of artificial intelligence and machine learning that can be applied in CR, such as fuzzy logic, genetic algorithm, neural network, hidden Markov model, support vector machine, Bayesian inference based prediction, linear regression and linear predictors, game theory, etc some for optimization of certain transmission parameters, some for decision making while some for learning and prediction [8],[9],[10]. To the best of our knowledge this is the paper that predicts spectrum holes using artificial neural network (ANN) that consist of time domain inputs with different rate of change. Each input is meant to

- *Sunday Iliyais a PhD student with Centre for Ccomputational Intelligence (CCI), De Montfort University, UK, sundayiliyagoteng@yahoo.com*
- *Professor Eric Goodyer is a member of CCI, De Montfort University, UK eg@dmu.ac.uk*
- *Dr. Mario Gongora is the deputy director of CCI, De Montfort University, UK, mgongora@dmu.ac.uk*
- *Dr. John Gow is a member of school of engineering, media and sustainable development, De Montfort University, UK, jgow@dmu.ac.uk*

enable the ANN to keep track of the trends of variation of spectrum activities due to that input. The time domain component that changes more often in this paper is the minute followed by the hour, day, date, week and the month. During the prediction, after obtaining the output of the ANN for the exact current time inputs, a cluster is form around the current minute with values very close to and above and below it, a maximum of 1.15 minutes. The average of the exact time and the cluster outputs was used to draw conclusion on the probabilities of the channels being idle. This improved the performance than just using the exact current time. During the prediction, the channels are ranked in order of decreasing probability of being idle and a search for spectrum hold commence from the channel with the highest probability of being idle until the first spectrum hold is found, thus minimizing search time and energy. The input of the ANN consist of only time domain without any radio related parameters such as signal to noise ratio, modulation type, etc thus making the model robust where the cognitive radio has no a priory knowledge of such parameters. The rest of the paper is organized as follows: Related works are presented in Section 2. This will be followed Section 3which discuss the neural network implemented. The training methods and architecture of the ANN are presented in Sections 4 and 5 respectively. Simulation data is presented inSection 6 while results and conclusion are covered in sections 7 and 8 respectively.

2 RELATED WORKS

Since 2000 when Joseph Mitola coined the word cognitive radio based on software defined radio (SDR) capability in his PhD dissertation [4], cognitive radio has attract the attention of many researchers as a promising solution to many problems in wireless communication including spectrum scarcity and underutilization. Artificial neural network (ANN) for predicting the data rate that can be achieved by CR in adopting a specific radio configuration such as: centre frequency, bandwidth, modulation, power etc; with reference to the radio frequency (RF) environment was proposed [11] [12]. An exponential moving average algorithm was used to generate the input data used in training the network with a normalized output ranging from -1 to 1. Spectrum holes prediction using ANN, multilayer perceptron (MLP) was modelled and simulated using MATLAB [13]. The PU traffic is assumed to follow Poisson distribution while the on/off time of the channel is drawn from geometric distribution. The channel statuses, busy and idle were denoted by binary symbols 1 and -1 respectively. Using the binary series, the NN predictor was trained to predict the channel status in the next slot based on the slot status history. In multi-channel system, a predictor is assigned to each channel. ANN was used for sensing primary signals using various digital modulation techniques parameters, since every PU employed one modulation scheme [14]. The NN consist of four inputs nodes fed with spectral power density and three signal standard deviations while the output consist of five digital modulation schemes i.e. 2FSK, 2ASK, 4ASK, BPSK and QPSK. The NN input consists of maximum spectral power density, which is used to distinguish signals with amplitude information (2ASK, 4ASK, BPSK and QPSK) from those without amplitude information (2FSK). The three standard deviations for absolute phase, direct instantaneous phase and normalized amplitude are used to further classify the various signal types (modulations). The model was simulated using MATLAB, but the fact that specific modulations were detected does not depict any information about white spaces (spectrum holes or idle channels). In [15], methods of evaluation and decision making in CR using ANN and GA were proposed. Cognitive engine using ANN and prediction of bit error rate (BER) in CR for four modulations: 64QAM, 16QAM, QPSK and BPSK [16]. ANN is used to predict the status of channels either busy or idle, such that all channels predicted to be idle are randomly searched for spectrum holes leaving those that were predicted to be busy [17]. There is no sense of search direction as-per which channel should be searched first and in what order among those predicted to be idled. Different types of machine learning HMM, Fuzzy logic, genetic algorism, support vector machines, neural network, Bayesian inference based prediction, moving average based prediction, autoregressive model based prediction, static neighbor graph based prediction, were presented in [6]

3 NEURAL NETWORKS

Neural networks are composed of simple elements operating in parallel. These elements are inspired by biological nervous systems [18]. The ANN used in this research consists of an input layer called source or sensory nodes, one hidden layer and an output layer. Each of the layers with the exception of the input layer consists of processing units called neurons. The input layer was cast unto a high dimensional hidden layer for effective features selection. Two nonlinear functions: hyperbolic tangent and log sigmoidal functions were used as the hidden and output layer activation functions respectively, though other functions were used for comparison. These functions implement a nonlinear transformation into the network. Log sigmoidal andhyperbolic tangent functions are given by equations 1 and 2 respectively. Back-propagation algorithm was used to train the network. Back-propagation algorithm consists of two phases, the forward and the backward phase. In the forward phase, the input (function) signal propagates from the input; layer by layer up to the output, in this phase, only the activation potential and the outputs of neurons that changes while the synaptic weights and biases remain constant. In the backward phase, the error signal is computed and propagates from the output backward layer by layer. In this phase, only the synaptic weights and biases changes i.e. they are updated in order to minimize the average mean square error while the activation potential and the outputs of the neurons remain unchanged.

4 TRAINING METHODS

4.1 Supervised training

Neural networks can be trained using supervised or non-supervised training method. A supervised training is used in this study, i.e. the time domain (minute, hour, date, day, week and month) where presented at the input and the output of the neural network is compared with the desired channels states commonly referred to as the target. The error is obtained from the difference between the neural network output and the actual channels states (target). The weights and the biases are adjusted or updated such that it minimizes the average mean square error every epoch using back-propagation algorithm. The neural network consists of 20 output neurons each representing the idle probability of one channel making a total of 20 channels. During the training, the actual states of the channels i.e. either idle or busy represented by 1 or 0 respectively; are presented as the target outputs at a particular time instance represented by the normalized values of the minute, hour, date, day, week and month; which served as the inputs. The input consists of vector of length six i.e. (minute, hour, date, day, week and month) while the output consists of vector of length twenty i.e. the number of channels. After normalizing the input data, they are reprocessed using MATLAB inbuilt preprocessing functions. The simulation data is assumed to be collected with reference to one fix geographical location; hence the spatial domain data (longitude, latitude and altitude or three dimensional position coordinate) were ignored. Subsequence simulations will involves normalized latitude and longitudes or position coordinate as part of the inputs. The altitude will not be considered assuming all is done on the earth surface.

4.2 Sequential training

Neural network can be trained in two ways either using the batch training or incremental training (sequential training) method. In the batch training, the synaptic weights and the biases of the neural network are updated only after all the training inputs/outputs patterns are presented. The weights and the biases are updated once every epoch. When a training pattern is presented to the network, the mean square error for that pattern is computed and stored in a memory. When the entire training patterns has being presented, the average of the mean square errors will be obtained and use to update the weights and the biases using back-propagation algorithm. This complete one epoch, one iteration or one training cycle. The training is repeated until one of the stopping criteria is met. The order in which the training pattern occurred in time does not matter. That is, batch training does not keep track of the order in which the training patterns were presented. In this research, since the order in which the training patterns occurred in time or the state of the channels with respect to time domain is of optimum important, batch training is not used rather sequential training is adopted. In sequential training, the order in which the training patterns occurred in time or are presented to the network is very important. This enables the network to adapt to the pattern or trends of changes of the channel states with respect to time which can be used for future spectrum holes prediction or prediction of the probability of a channel being idle. In sequential training, each time a training pattern is presented to the network, the mean square error is obtained and used to update the weights and biases. That is, the weights and biases are updated each time a pattern is presented. Within one epoch, the weights and biases will be updated a number of times equal to the number of the training patterns. The training will terminate if any of the stopping criteria is met; i.e. either the required number of epoch is reach or the acceptable error (goal) is attained or when over-fitting is about to set in.

5 ARCHITECTURE OF THE NEURAL NETWORK AND PERFORMANCE

5.1 Architecture

Neural network architecture can broadly be classified as either feed forward or recurrent type. Each of these two classes can be structured in different configurations. A feed forward network is one in which the output of one layer is connected to input of the next layer via a synaptic weight and so on while the recurrent type may have at least one feedback connection or connections between neurons within the same layer or other layers depending on the architecture. The training time of the feed forward is less compare to that of the recurrent type but the recurrent type has better memory capability for recalling past events. In this study, four configurations were used i.e. feed forward (FF), cascaded feed forward (CFF), layered recurrent (LR) and feed forward with output feedback (recurrent) (OFB). Since there is no exiting algorithms or means of selecting the best ANN architecture or topology for a given non-linear problem, different architectures were adopted and their performances were compared. The cascaded feed forward, feed forward with output feedback and feed forward perform relatively better than the layered recurrent network. All the networks topologies used consist of only one hidden layer. The hidden layer serve as features detector i.e. during the training, it tends to learn some salient features that characterize the training samples. The FF architecture implemented in this work is as shown in Fig. 1.

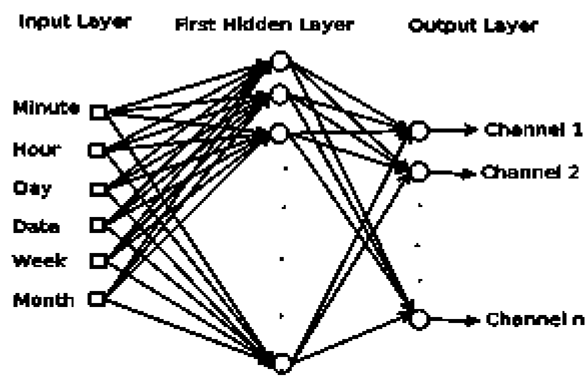

Fig. 1: *FF ANN architecture*

5.2 Activation functions

Different activation functions for the hidden and output layer were used; these are: hard limit (hardlim), symmetric hard limits (hardlims), symmetric straight line (purelin), straight line (poslin), hyperbolic tangent function (tansig) and logistic or log sigmoidal function (logsig). The input layer has no activation functions or processing units since they are just source or sensory units which does not process data but only serve as the link between the input data and the network. It was found that when hyperbolic tangent function (range from -1 to 1) is used as the activation function of the hidden layer and logistic function (range from 0 to 1) in the output layer, it gives a better performance than with any other combinations. During the training, the target output used consist of the actual states of the channels 1 or 0 i.e. idle or busy respectively. The training is based on certain events or states but during the prediction, when a time domain input is presented to the network, the output is probabilistic ranging from 0 to 1. The output of the neural network gives the likelihood of the channels being idle. These channels are ranked starting from the one with the highest probability of being idle to the least. The channels are search for spectrum hole starting from the one with the highest probability of being idle. If the first ranked channel is confirm to be actually idle, the search stop and the channel will be explored for cognitive radio communication. But if the first ranked channel is confirmed to be busy, the next one is search and so on. The search stop when the first vacant channel is detected or sufficient continuous or discontinuous spectrum holes exists within the channel. In order to classify the channels as idle or busy directly, hard limit and symmetric hard limit were used as the activation functions of the output layer. The output of hard limit is either 1 or 0 while that of symmetric hard limit is 1 or -1 representing idle or busy respectively. The performance with hard limit as the output activation function was poor as compared to the probabilistic approach using log sigmoid function. For the hard limit case, the channels with 1s predicted to be idled are randomly searched first; follow by those with zero depending on when the first white space is gotten. But there is no sense of search direction i.e. among those channels with 1 as output, there is no information about which one should be searched first and so on. When symmetric hard limit was used as the activation function for the hidden layer with hyperbolic tangent sigmoid as the output layer activation function, the performance was relatively batter.

5.3 Mean square error and number of channels searched

Though there is a relationship between the number of channels searched before getting the first spectrum holes and the average of the mean square error of miss classification i.e. as the average of the mean square error increases, the number of channels searched before getting the first spectrum hole also increases, but this is not always the case. It was observed that sometimes even though the averaged mean square error is high, white spaces were located within the first or second search as depicted in Fig. 1. This shows that even though some were wrongly classified (ranked), yet the ones assigned with the highest probability of being idle were actually idle and since we are only looking for one spectrum hole to use, such classification can be considered satisfactory. From the simulation result, about 80% of the total search, locates spectrum hole within the first or at most the second search though there are cases of poor performance below the stated percentage since no prediction that is always perfect.

5.4 Updating of the weights using back-propagation algorithm

Considering neuron k located in the output layer with an activation function given by(1) [18], the change in synaptic weight $\Delta w_{kj}(n)$connecting neuron k in the output layer and neuron j in the hidden layer in n iteration is obtain as follow; where: $\mathrm{d}_k(\mathrm{n})$is the desired or the target output, $\mathrm{y_k(n)}$ is the actual output (activation function), $v_k(n)$ is the induced local field or activation potential, $\mathrm{e_k(n)}$ is the error, $\varphi(\mathrm{n})$ is the error energy or mean square error, η is the learning rate, $\mathrm{y_j(n)}$ and $\mathrm{v}_j(\mathrm{n})$ are the activation function and the induced local field of the hidden layer respectively while $\delta_k(n)$ is the local gradient[18].

$$y_k(v_k(n))=\frac{1}{1+\exp(-av_k(n))} \quad (1)$$

$$y_j\left(v_j(n)\right) = d\tanh(bv_j(n)) \quad (2)$$

$$\varphi(n) = \frac{1}{2}\sum_k e_k^2(n) \quad (3)$$

$$e_k(n) = d_k(n) - y_k\big(v_k(n)\big) \quad (4)$$

$$\mathrm{v_k(n)} = \sum_{k=0}^{m} \mathrm{w}_{jk}(\mathrm{n})\, \mathrm{y}_j\big(\mathrm{v_k(n)}\big) \quad (5)$$

$$\mathrm{y_k(n)} = \mathrm{y}_k\big(\mathrm{v_k(n)}\big),\ y_j(n) = y_j\left(v_j(n)\right)$$

$$\frac{\partial\varphi(n)}{\partial e_k(n)} = e_k(n),\quad \frac{\partial e_k(n)}{\partial y_k(n)} = -1\ ,$$

$$\frac{\partial \mathrm{y}_k(\mathrm{n})}{\partial \mathrm{v}_k(\mathrm{n})} = \mathrm{a}\mathrm{y}_k(\mathrm{n})[1-\mathrm{y}_k(\mathrm{n})]\ ,\ \frac{\partial \mathrm{v}_k(\mathrm{n})}{\partial \mathrm{w}_{kj}(\mathrm{n})} = \mathrm{y}_j(\mathrm{n})$$

$$\frac{\partial\varphi(\mathrm{n})}{\partial \mathrm{w_{kj}(n)}} = \frac{\partial\varphi(\mathrm{n})}{\partial \mathrm{e_k(n)}}\frac{\partial \mathrm{e_k(n)}}{\partial \mathrm{y_k(n)}}\frac{\partial \mathrm{y_k(n)}}{\partial \mathrm{v_k(n)}}\frac{\partial \mathrm{v_k(n)}}{\partial \mathrm{w_{kj}(n)}} \quad (6)$$

$$\Delta w_{kj}(n) = -\eta\frac{\partial\varphi(\mathrm{n})}{\partial \mathrm{w_{kj}(n)}} \quad (7)$$

$$\Delta w_{kj}(n) = -\eta\delta_k(n)\mathrm{y_j(n)} \quad (8)$$

$$\delta_k(n) = \frac{\partial\varphi(\mathrm{n})}{\partial \mathrm{v_k(n)}} = \frac{\partial\varphi(\mathrm{n})}{\partial \mathrm{e}_k(\mathrm{n})}\frac{\partial \mathrm{e_k(n)}}{\partial \mathrm{y_k(n)}}\frac{\partial \mathrm{y}_k(\mathrm{n})}{\partial \mathrm{v}_k(\mathrm{n})}$$

$$\delta_k(n) = -\mathrm{e}_k(\mathrm{n})\frac{\partial \mathrm{y}_k(\mathrm{n})}{\partial \mathrm{v_k(n)}}$$

$$\delta_k(n) = \mathrm{a}\mathrm{y}_k(\mathrm{n})[\mathrm{d}_k(\mathrm{n}) - \mathrm{y_k(n)}][1-\mathrm{y_k(n)}]$$

$$\Delta w_{kj}(n) = \eta a\mathrm{y_j(n)}y_k(n)[\mathrm{d_k(n)} - \mathrm{y_k(n)}][1-\mathrm{y_k(n)}]$$

When a momentum is included, the change in weight $\Delta w_{kj}(n)$ becomes

$$\Delta w_{kj}(n) = \propto \Delta w_{kj}(n-1) + \eta a\mathrm{y_j(n)}y_k(n)[\mathrm{d_k(n)} - \mathrm{y_k(n)}][1 - \mathrm{y_k(n)}]$$

The new weight is obtained as:

$$w_{kj}(n+1) = w_{kj}(n) + \Delta w_{kj}(n)$$

Where $\propto$ is the momentum constant and $\Delta w_{kj}(n-1)$ is the previous change in weight. In this work, the learning rate η was kept at 0.2 while $\propto = 0$ based on the chosen training function (trainlm); other parameters were left at matlab default settings. For a neuron j located in the hidden layer, there is no assigned or known desired response, hence the error cannot be computed directly, and rather it has to be determined recursively using the error of the output layer. For neuron j in the hidden layer, the local gradient $\delta_j(n)$ is given as:

$$\mathrm{y_j}\left(\mathrm{v_j(n)}\right) = \mathrm{d}\tanh(\mathrm{b}\mathrm{v}_j(\mathrm{n}))$$

$$\mathrm{y_j(n)} = \mathrm{y_j}\left(\mathrm{v}_j(\mathrm{n})\right)$$

$$\delta_j(n) = -\frac{\partial\varphi(\mathrm{n})}{\partial \mathrm{y}_j(\mathrm{n})}\frac{\partial \mathrm{y}_j(\mathrm{n})}{\partial \mathrm{v}_j(\mathrm{n})}$$

$$\frac{\partial \mathrm{y}_j(\mathrm{n})}{\partial \mathrm{v}_j(\mathrm{n})} = dbsec^2\left(bv_j(n)\right) = db\left(1 - tanh^2\left(bv_j(n)\right)\right)$$

$$= \frac{b}{d}\left(d^2 - y_j^2(n)\right)$$

$$\frac{\partial \mathrm{e_k(n)}}{\partial \mathrm{v}_k(\mathrm{n})} = \frac{\partial \mathrm{e_k(n)}}{\partial \mathrm{y_k(n)}}\frac{\partial \mathrm{y}_k(\mathrm{n})}{\partial \mathrm{v}_k(\mathrm{n})}$$

$$\frac{\partial\varphi(\mathrm{n})}{\partial \mathrm{y}_j(\mathrm{n})} = \frac{\partial\varphi(\mathrm{n})}{\partial \mathrm{e}_k(\mathrm{n})}\frac{\partial \mathrm{e}_k(\mathrm{n})}{\partial \mathrm{y_k(n)}}\frac{\partial \mathrm{y}_k(\mathrm{n})}{\partial \mathrm{v}_k(\mathrm{n})}\frac{\partial v_k(\mathrm{n})}{\partial \mathrm{y}_j(\mathrm{n})}$$

From(3), (4) and (5) we have

$$\frac{\partial\varphi(\mathrm{n})}{\partial \mathrm{y}_j(\mathrm{n})} = -\sum_k \mathrm{e}_k(\mathrm{n})y_k'(n)w_{kj}(n)$$

$$\frac{\partial\varphi(\mathrm{n})}{\partial \mathrm{y}_j(\mathrm{n})} = -\sum_k \delta_k(n)w_{kj}(n)$$

$$\delta_j(n) = \frac{b}{d}\left(d^2 - y_j^2(n)\right)\sum_k \delta_k(n)w_{kj}(n)$$

$$\Delta w_{ji}(n) = \propto \Delta w_{ji}(n-1) - \eta\delta_j(n)\mathrm{y_i(n)}$$

Where $\Delta w_{ji}(n)$ is the changed in the synaptic weight connecting the sensory or input node i and a neuron j in the hidden layer while $\mathrm{y_i(n)}$ is the input state at sensory node i. the new weight is given in terms of the general delta rule as:

$$w_{ji}(n+1) = w_{ji}(n) + \Delta w_{ji}(n)$$

6 SIMULATION DATA

We try to obtain real life data but could not find any within our search domain, so we model some real life scenarios to generate the data for the training and simulation. Our assumption was based on the fact that majority (i.e. over 50%) of the assigned or licensed spectrums are not efficiently utilize with respect to time and space (geographical location). The simulation was carried out using different traffic intensities which represent the percentage utilization of the channels. All spectrum activities in this study are man oriented and sometimes follows certain scenarios and sometimes stochastic in nature. Some spectrum may be busier during the day time than night or at other time instances. Some may be busier during weekends than working days or certain dates or period of the month or year and verse versa. Some may be busier during certain seasons, events, etc. As a result of economic and other factors, some spectrums are more utilize in urban areas than rural areas which also vary in developed and developing countries; while some are only confined to certain geographical locations e.g. the aeronautical spectrum. Considering these factors, we assigned some weights values representing the percentage of idle period for each channel with respect to six elements i.e. minute, hour, date, day, week and month. These weights give information about the channels utilization or the traffic intensity. To account for the contribution of each of these six elements, crisp values were obtained by finding the average of the weights. These crisp values represent the proportion of idle state of the channel for a given number of samples within a given period of time. Random 1s (idle) and 0s (busy) were generated using random number generator in matlab in proportion to the crisp values and the number of samples. These 1s and 0s are used to train the neural network. The simulation was carried out many times using different weights (traffic intensities). Each time the program is run, even for the same traffic intensities, the position of the 1s and 0s is randomized.

7 RESULTS

The results of the spectrum holes search using ANN was compared with that of blind linear and blind stochastic search and was found to be superior as shown in Fig. 2 to 10 for different traffic intensity. Blind linear search method, always start searching from the first channel serially until it found the first spectrum hole if there exist one; while the blind stochastic search searches at random all the channels until it found a spectrum hole if any. The blind stochastic search was included to know if there is a sense of intelligent that the neural network exhibited not just behaving in a stochastic way. The performance of the ANN that was trained to give the probability of the channels being idle was compared and found to outperform the ANN that predict the exact channel states (busy or idle). In the ANN that was trained to give the exact channels states, all channels predicted to be idle are randomly searched until the first spectrum hole was found; no information about search direction regarding which channel should be sensed first among the predicted idle channels and so on. While in ANN that gives the idle probability, the channels are ranked in order of decreasing probability of being idle and search for spectrum hole starting from the channel having the highest probability of being idle. About 80% of the total search, the ANN was able to locate spectrum hole within the first or second search; though there are cases of higher and lower performance below the stated percentage as depicted in Fig. 3 to 11. Different ANN topologies (architecture) were adopted during the simulation and the results were compared. The performance improves as the number of hidden neurons increases from 100, 200, 300 and remain almost constant for 400 and 500 but the time of training keeps increasing with increase in number of neurons. For lower traffic intensities up to 0.50 or %50, 200 and 300 hidden neurons gives a good results, Fig. 4. The simulation was carried out using matlab.

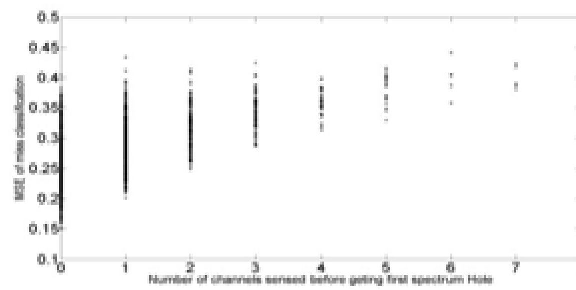

Fig. 2: *Variation of mean square error and number of channels sensed*

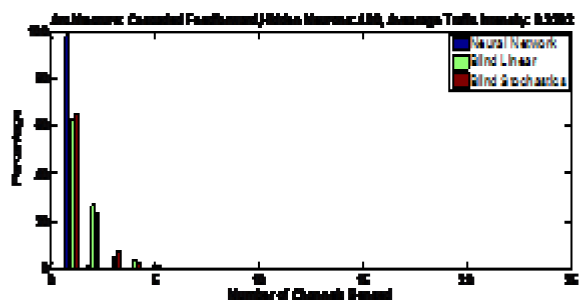

Fig. 3: *CFF ANN, hidden neurons: 400, traffic intensity: 0.3262*

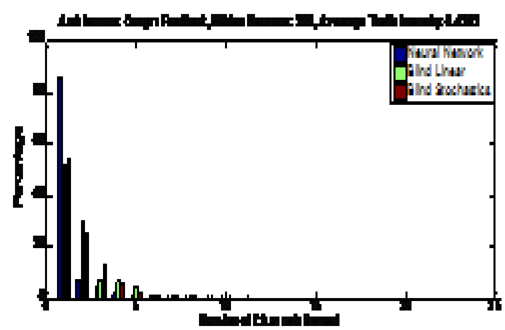

Fig. 4: *OFB ANN, hidden neurons: 200, traffic intensity: 0.4262*

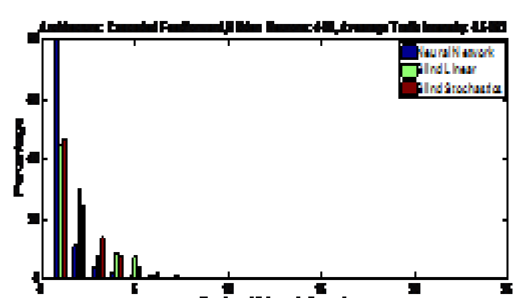

Fig. 5: *CFF ANN, hidden neurons: 400, traffic intensity: 0.4262*

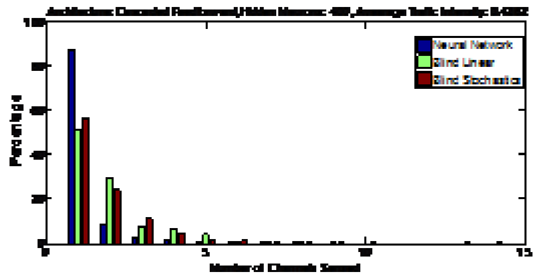

Fig. 6: *CFF ANN, hidden neurons: 400, traffic intensity: 0.5062*

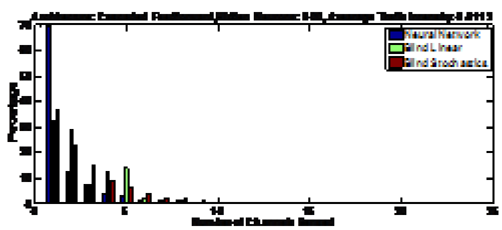

Fig. 7: *CFF ANN, hidden neurons: 300, traffic intensity: 0.6112*

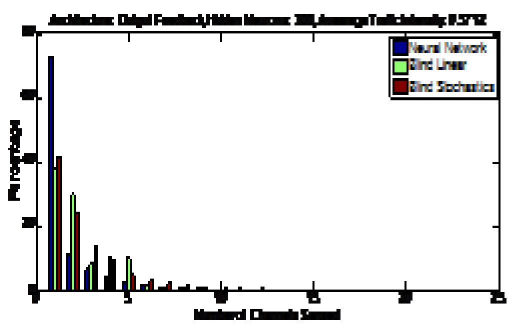

Fig. 8: *OFB ANN, hidden neurons: 300, traffic intensity: 0.5712*

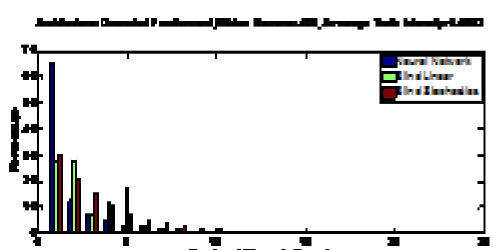

Fig. 9: *CFF ANN, hidden neurons: 400, traffic intensity: 0.6662*

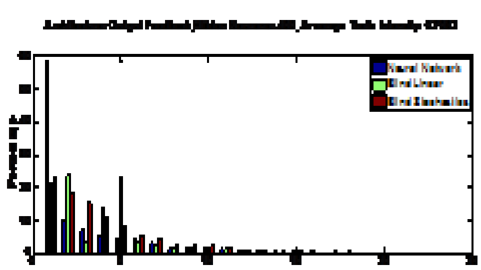

Fig. 10: *CFF ANN, hidden neurons: 400, traffic intensity: 0.7062*

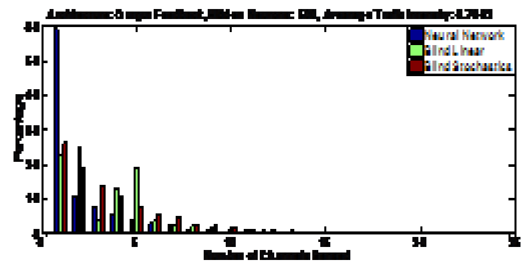

Fig. 11: OFB ANN, hidden neurons: 500, traffic intensity: 0.7062

8 CONCLUSION

Spectrum hole prediction using ANN is found to be far better than the traditional blind linear and blind stochastic search. The ANN that is trained to give the probability of a channel being idle is better than the ANN that will give the exact channel states (busy or idle) since the former give a sense of search direction. The performance was improved by forming a cluster around the fast changing time domain component (minute) where the final judgment was based on the average of the outputs. Our next work will focus on the use of ANN for white space prediction using real life RF data and data from GPS receiver such that the ANN will be trained to learn spectrum activities in frequency, time and spatial domain.

ACKNOWLEDGMENT

This work is supported in part by Petroleum Technology Development Fund (PTDF) Scholarship, Nigeria.

REFERENCES

[1] C. Stevenson, G. Chouinard, Z. Lei, W. Hu, J. Stephen and W. Caldwell, “IEEE 802.22: The First Cognitive Radio Wireless Regional Area Network Standard”, IEEE 2009.

[2] Z. Yan, Z. Ma, H. Cao, G. Li and W. Wang, “Spectrum sensing, Access and Coexistence testbed for Cognitive Radio Using USRP”. IEEE, 2008

[3] Y. lian and A. Hoang, “Cognitive Radio on TV Bands: A New Approach to Provide Wireless Connectivity to Rural Areas”, IEEE, 2008.

[4] J. Mitola III, “Cognitive Radio: An integrated agent architecture for software defined radio”. PhD Thesis, Royal Institute of Technology, Stockholm, Sweden, May 2000.

[5] J. Oh and W. Choi, “A Hybrid Cognitive Radio System: A Combination of Underlay and Overlay Approaches”, IEEE, 2010.

[6] X. Xing, T. Jin, W. Cheng and X. Cheng, “Spectrum Prediction in Cognitive Radio Network.

[7] M. Wyglinski, M. Nekovee and Y. Thomas Hou, “Cognitive Radio Communications and Networks”, pp. 85-109, 2009

[8] R. Azmi, “Support Vector Machine Based White Space Predictors for Cognitive Radio,” Master Theses, 2011.

[9] J. Proakis and D. Manolakis, “Digital Signal Processing Principles, Algorithms, and Applications,” Pearson Prentice Hall, Upper Saddle Rever, New Jersey 07458, 4th Edition 2007.

[10] J. Proakis and M. Salehi, “Digital Communications,” Mcgraw-Hill, 5th Edition 2008.

[11] Katidiotis, K. Tsagkaris and P. Demestichas, “Performance Evaluation of Artifirtial Neural Network-Based Learning Schemes for Cognitive Radio Systems,” Computer Communication, 2008.

[12] K. Tsagkaris, A. Katidiotis and P. Demestichas, “Nueral Network based Learning Schemes for Cognitive Radio Systems”, Computer Communications Journal 2008.

[13] V. Krishna, P. Wang and D. Niyato, “A Neural Network Based Spectrum Prediction Scheme for Cognitive Radio” IEEE ICC 2010.

[14] J. Julius and R. Van, “Application of Neural Network for Sensing Primary Radio Signals in a Cognitive Radio,” IEEE Africon 2011, September 2011.

[15] Z. Zhan and X Xie, “Intelligence Cognitive Radio: Research on Learning and Evolution of Cognitive Radio Based on Neural Network.

[16] X. Dongi, Y Lii, C. Wui and Y. Cai, “A Learner Based on Neural Network for Cognitive Radio”, IEEE International conference of computer Communication 2010 .

[17] Z. Jianli, W. Mingwei and Y. Jinsha, “Based on Neural Network Spectrum Prediction of Cognitive Radio”, IEEE 2011

[18] S. Haykin, “Neural Networks and Learning Machines”, pp. 122- 44, 3rd Edition 2008.