IJASRM International Journal of Advanced Scientific Research and Management, Volume 4 Issue 6, June 2019
www.ijasrm.com
ISSN 2455-6378

ANN Application in Sensing-Transmission Scheduling in Cognitive Radio

Inderdeep Kaur Aulakh

Associate Professor (IT), UIET, Panjab University,
Chandigarh , India

Abstract

Artificial Neural Networks have been applied on a threshold-basedsensing-transmission scheduling in cognitive radio. In Cognitive Radio technology first spectrum is sensed and then on finding it free from licensed subscribers-Primary Users, it is utilized by others- Secondary Users. A sensing-transmission structurebasedon thresholdshas been developed earlier and, in this paper, ANNs have been applied on it for automating the decision making of sensing or transmittingby a secondary user.

Keywords**:** *artificial neural networks; cognitive radio networks; cost of collision; sensing time; transmission time;sensing–transmission scheduling;*

1 Introduction

Usage of the RF spectrum is regulated by the authorities in every country and is given on payment of very high fee. The free frequency bands form a very small part of the RF spectrum. Due to the ever -rising need for spectrum, many efficient techniques for spectrum sharing are being developed. One such technique is of Cognitive Radios. [1],[2].

The word Cognitive is derived from 'recognizing'. Under this technology the free/unused frequency bands are recognized or sensed and then used by other users.

Being efficient classifiers, Artificial Neural networks can be applied to the sensing-transmission scheduling model based on utility thresholddeveloped earlier. The ANN are used to decide whether it is beneficial for the Secondary Userto transmit based on the six metrics of Cost of Collision, SU sensing time, SU transmission time, False Alarm Probability, Detection Probability and Reward for transmitting.

2 Literature Review

Earlier researchers [3],[4],[5],[6] have assumed periodic sensing, deriving optimal balance between throughput and false-alarm probability for a particular sensing time. In [7], a scheme has been proposed, which studies the sensing time and its sequence in order to increase the detection of spectrum gaps in the RF spectrum. In [8], a threshold-based strategy for channel sensing and transmission has been given. In [9],[10]both slotted and un-slotted Primary User transmission has been studied for finding an optimal spectrum utilization plan. The approach in [11] is not suitable for all idle time distributions as the PU state is changing continuously. In [12],[13],[14],[15] and [16] the threshold based model has been developed and the results of varying the Reward for transmitting, false alarm probability, SU transmitting time, SU sensing time and channel noise with respect to cost of collision have been analyzed. Authors in [17] proposed ANN-based learning schemes for finding data rate for a particular radio system. Multiple AI schemes have been analyzed in [18] based on their responsiveness, complexity, security, robustness and stability. In [19] ANNs have been used for spectrum sensing and have been found suitable for even transmissions having low SNR and high channel noise. In [20] the learning issue in CR has been studied with the significance of AI in achieving practical cognitive networks. The learning issues in CRs have been considered and classified under two main categories: Decision-making and feature classification. Supervised and unsupervised learning algorithms have been applied to non-markovian environments and decentralized networks. The conditions under which each of them can be useful is analyzed. The combination of Cost of Collision, SU sensing time, SU transmitting time, False alarm probability, Detection probability and Reward for transmitting on the optimal threshold in sensing-transmission scheduling hasn't been analyzed. These have been taken as the main metrics in this paper. Firstly seven ANNs have been used to find the optimal threshold for different values of all these metrics.

The Neural Network Architecture which takes the least time and gives the least error under different traffic conditions has been thus identified.

3 Sensing-Transmission Scheduling Model

Consider one SU node sensing a PU channel. PU is not aware of the SU so follows a transmission model dependingon the semi-markov renewal theory. Its idle and busy durations have $f_I(.)$ and $f_B(.)$ probability distributions with means $\underline{i}$ and $\underline{b}$. Secondary User may sense or transmit many times in continuation, hence increasing its access efficiency. The Primary User can be protected by controlling the cost of collision. Thus, the utility achieved is benefit minus the cost of collision. It is imperative that SU be aware of PU idle time distribution. Extensive measurements using various methods [7],[21-22] can estimate this idle time distribution function This assumption is like assuming that PU idle time has exponential distribution with knowledge of its parameter, which has been done in earlier works.
This paper assumes that the SU is using any one the many possible spectrum sensing algorithms [21-22]. The detection probability and false alarm probability are performance metrics of these algorithms and is obtained during their implementation or simulation. In this study they have been given typical values obtained in efficient and practical sensing algorithms. For better utilization of vacant spectrum in PU transmission, the SU either senses number of times or transmits many packets.

- Probability of finding Primary User busy when it is idle: False Alarm Probability : Pr_{false}
- Probability of correctly detecting the presence of the Primary User on the channel: Detection Probability: Pr_{busy}
- Sensing duration of the Secondary User: Secondary User sensing time: T*sns*
- Transmission duration of the Secondary User: Secondary User transmission time: T*pkt*
- Gain for reliable Secondary User transmission: Benefit to Secondary User: Reward x Tpkt
- Penalty for collision with Primary User: Cost of Collision: *CCOST*

3.1 Optimality Criterion

The sensing-transmission scheduling scheme that maximizes the average utility per time would be optimal. Hence the following has to be maximized [16]:

$$lim_{L\to\infty} \frac{(\sum_{l=1}^{L}\sum_{t=0}^{N_l} Utility_t(Pr_t, action_t))/L}{(\sum_{l=1}^{L}(I_l+B_l))/L}$$

Where:
$action_t$: Activity of the Secondary User at any instant t, this may be sensing or transmitting
L: The total number of idle-busy periods; L is large
N_l: The number of activities undertaken in the lth idle-busy period,
I_l : Duration of lth idle period
B_l: Duration of lth busy period

From the law of large numbers this implies the maximization of:

$$Y(t, Pr_t) = \frac{E[\sum_{t=0}^{N_l} Utility_t\,(Pr_t, action_t)]}{\bar{\imath} + \bar{b}}$$

for an allocation plan for Secondary User which defines a mapping of Secondary User's detection of Primary User being idle or busy to Secondary User's activity to sense or transmit. Since $\bar{\imath} + \bar{b}$ is constant. The issue is to maximize the numerator. The maximum expected utility obtained by Secondary User at any instant t is:

$$Y(t, Pr_t) = \max_{\{sense, transmit\}} \{Sns(t, Pr_t), Tx(t, Pr_t)\}$$

Instantaneous values of expected utilities obtained by the Secondary User when it senses the channel or transmits on it are $Sns(t,Pr_t)$ and $Tx(t,Pr_t)$ respectively[20].

$$Sns(t, Pr_t) =$$

$$\sum_{i=\frac{idle}{busy}} Pr[result_{t+Tsns}^{sns} = i]\; Y\big(t + Tsns,\, Pr_{t+Tsns}(i)\big)$$

$$Tx(t, Pr_t) =$$

$$\sum_{j=\frac{ack}{nack}} Pr\big[result_{t+Tpkt}^{Tx} = j\big]\, Y\Big(t + Tpkt, Pr_{t+Tpkt}(j)\Big) + Utility_t(Pr_t, Tx)$$

Pr_t: Primary User's conditional probability of being idle at instant t.
The threshold (Pr_t) becomes optimal (Pr_t^*), when expected utility for transmitting exceeds that for sensing.

Lower values of Pr_t^* are beneficial for Secondary User, as the SU is allowed to start transmitting at lower values of probability of Primary User being idle. Cost of collision, is the penalty imposed on SU to safe guard the Primary User transmission from collisions. This can be adjusted according to the importance of Primary User data.

4 ANN Based Testing and Optimization of Sensing-Transmission Structure

The amalgamated influence of Cost of Collision, SU sensing time, SU transmission time, False alarm Probability, Detection Probability and Reward of transmissionon the threshold in sensing-transmission schedulingis analyzed by the application ofArtificial Neural Networks. These metrics are taken as the primary constraints and the model is simulated to determine the threshold for various values of all the above. The six metrics are given a range of variation according to the channel and traffic conditions. If each metric can have three possible values then for the 729 (3^6) cases the threshold value is calculated. Then the conducive optimal threshold for SU to transmit has to be defined. The problem on hand then is to use artificial neural networks to decide if an SU can transmit or not given the various values of the above 6metrics. The neural networks classifiers are then used for making this decision, because they are proficient classifiers and are particularly well suited for addressing non-linear problems.The six metrics act as inputs to a neural network and the decision to transmit or not is the target. For an input made up of any combination of the 6 metrics, the ANN has to give the decision of transmission or sensing.

For a neural network, data associated to the classification problems is arranged by using two matrices, matrix Y which is the input matrix has been named as “cognitiveparameters” and matrix O which is the target matrix has been named as “result”. Each column of the input matrix has 6 entries which represent all the 6 metrics. Each column of the target matrix has 2 entries, first represents SU transmission and the second represents SU sensing or no transmission. If the computed threshold is lower than the one need for SU transmission (optimal threshold) then a 1 is recorded in the first entry of the target matrix but if the computed threshold is above the condition upto which SU should transmit (optimal threshold), then a one is recorded in the second entry of the target matrix. All other entries are zero. Every neural network’s inputs are the 6 metrics and target is SU transmission or sensing. For every input of the 6 CR metrics, the ANN has to predict the SU action.

Seven different neural networks have been used for the classification. As each neural network takes random initial weights, the results are different for every execution. A random seed can be set to prevent this, then the outcome would be same for every execution of each network. The ANN with the least error and minimum time is found for different traffic conditions.

The network architectures considered train themselves using the back-propagation algorithm. A neural network is trained using the inputs and their target results until it implements the function with least error. Thus, it should be able to form the association between the inputs with their outputs. The method of computing the gradient for nonlinear multilayer networks gives the term of back propagation. Proper training of the back-propagation networks givesmore accurate results even if given hitherto unseen inputs. They will then give a fairly accurate result for all possible input/output pairs even without training the network on those inputs. Thus, specifically a new input would give an output which is like the right output for inputs used in training, which are similar to the new input being given. The seven neural network architectures considered are discussed next.

4.1 Feed-Forward Back Propagation Network

The two-layer feed-forward network is the most commonly used network having back propagation. A 2-layer network with 20 neurons within the hidden layer is created by the following call to newff. The number of neurons in the output layer is automatically set to two as the target matrix (result) can have two columns.

net = newff(cognitiveparameters,result,20)

In this type of networks there are one or more hidden layers of sigmoid neurons succeeded by a linear neurons output layer. The purpose of the linear output layer is to allow the network to give values beyond the range –1 to +1. But if the output needs to be restricted between 0 and 1 then the output layer has to use a sigmoid transfer function (like logsig) as seen in Fig.1.The network then needs to be trained, validated and tested.

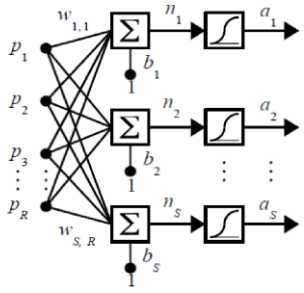

Fig. 1: Configuration of Feed Forward Network

4.2 Cascade-Forward Back-Propagation Network

While 2-layer feed-forward networks are capable of learning almost every input-output mapping, yet feed-forward networks exceeding 2 layers might be

able to learn complex mappings faster. In this network there is a weightedlink from the input to every layer and from every layer to the subsequent layers. As an example assume a network with 4 layers, and then it will have connections from the input to each of the 4 layers along with connections from layer 1 to layer 2, layer 2 to layer 3, layer 3 to layer 4, layer 1 to layer 3, layer 1 to layer 4 and layer 2 to layer 4. Speed of learning for a required relationship gets improved by considering extra connections. The function used create a cascade forward network is:
newcf(cognitiveparameters,result,[20,10]).

This creates a 3-layer cascade forward network with 20 neurons in the first hidden layer, 10 neurons in the second hidden layer and 2 neurons in the output layer. The hidden layers have the sigmoid transfer function and the output layer has the linear transfer function.

Next 5 dynamic neural networks are used for the testing of the model. Dynamic back propagation which is more complicated has to be used to compute the gradients in order to consider this indirect effect. Hence dynamic back propagation also takes more time to train. The steps are creation, training and application of these networks to modelling, detection, and forecasting problems. Dynamic back propagation may be required by only some these networks for computing the gradients and not by networks. The decision whether to apply dynamic back propagation or not need not be made by the user. The NN tool box software automatically determines this besides deciding on the best type of dynamic back propagation to apply. The user needs to only build the network and then call the train command.

4.3 Focused Time-Delay Neural Network

This is the most straight forward dynamic network. Focused Time-Delay Neural Network is a feed forward network having a tapped delay line at the input. It is type of a focused network and also a general class of dynamic networks. It can be seen in Fig. 2 that the input layer of the multilayer static feed forward network has dynamism.

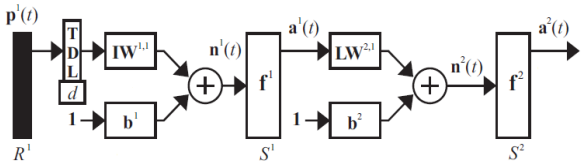

Fig. 2: Configuration of Focused Time Delay Network

The following call to newfftd function is used to create the FTDNN network with a tapped delay line having delays from 1 to 2 and a hidden layer with 20 neurons.

net =newfftd(cognitiveparameters,result,1:2,20).

One good feature of the FTDNN is that it does not contain loops of feedback or parameters which are adjustable. The tapped delay line appears only at the input of the network so it doesn't need dynamic back propagation to find the network gradient. Hence this network should train in a lesser time than other dynamic networks.

4.4 Distributed Time-Delay Neural Network

It is another dynamic neural network. The DTDNN has distributed tapped delay lines throughout the network as opposed to the FTDNN. The Fig.3 depicts a general 2-layer DTDNN. The DTDNN network used here is created using the newdtdnn function as following:
net=
newdtdnn(cognitiveparameters,result,[20,8],{1:2,1:2}).

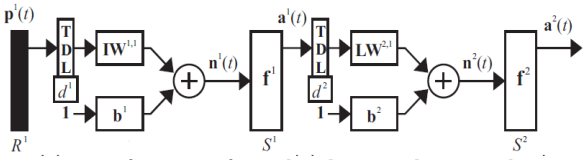

Fig. 3: Configuration of Distributed Time Delay Network

Thus a 3 layer DTDNN network with the 1st hidden layer of 20 neurons, the 2nd hidden layer of 8 neurons and the output layer of 2 neurons is built. The hidden layers contain sigmoid transfer functions and the output layercontains the linear transfer function. The distributed TDNN has to use dynamic back propagation so training in DTDNN takes more time than in FTDNN.

4.5 Elman Back-Propagation Network

This is the simplest dynamic network introduced by Elman and there are only 2 layers in it. A sigmoid (tansig) transfer function is used in the hidden layer while a linear (purelin) transfer functionis used in the output layer. The first version of Elman network was trained by approximating it to the back-propagation algorithm. The following call creates a hidden layer of 20 neurons and a output layer of 2 neurons having feedback from its output to its input:
net = newelm(cognitiveparameters,result,20)

4.6 Nonlinear Autoregressive Network with Exogenous Inputs (NARX)

The NARX is a recurrent dynamic network, having feedback loops encompassing the many layers of the network. The NARX model is based on the linear ARX model mostly found in time-series modeling. The NARX network is modeled through

IJASRM
International Journal of Advanced Scientific Research and Management, Volume 4 Issue 6, June 2019
www.ijasrm.com
ISSN 2455-6378

a feed forward neural network to implement the function *f*. A diagram of the resultant network is depicted in Fig.4, wherein a 2-layer feed forward network is used for the implementation.

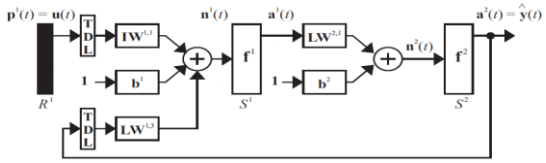

Fig.4: Configuration of NARX Network

4.7 Layer Recurrent Network (LRN)

This is a dynamic neural network considered to be a black box system used for analyzing a non-linear system infused with non-linear input signal. There is a feedback loop in this network and a single delay in each layer of the network but not in the last layer. The Fig.5 illustrates a 2-layer LRN.

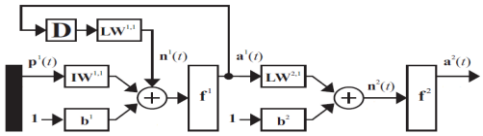

Fig.5: Configuration of Layer Recurrent Network

It is an advancement of the Elman Back Propagation Network. The LRN is trained by the NN toolbox utilizing the gradient-based algorithms. The newlrn command is used to create a general Elman network. It can contain any number of layers and have any transfer function in each layer. The function used is:
newlrn(cognitiveparameters,result,[20,10]).

These specific arguments create a 3-layer LRN with the first hidden layer having 20 neurons with tansig transfer function, second hidden layer having 10 neurons with tansig transfer function and the output layer using a purelin transfer function and 2 neurons.

5 Simulation Results

5.1 Preparing the Data

The threshold depends on the 6 metrics: cost of collision, SU sensing time, SU transmission time, False alarm probability, Detection probability and Reward of transmitting. Each of these metrics is given 3 values as in Table 1. Thus, there can be 3^6 cases.Threshold is obtained for every case. The optimal threshold at which the SU should transmit is fixed at 0.95. If threshold is above this optimal value then SU should make the decision of not transmitting and vice-versa.

Table 1: Metrics Values for ANN optimization

S.No.	Metrics	Values
1.	False alarm Probability (Pr_{false})	0.01,0.02,0.03
2.	Detection Probability (Pr_{busy})	0.97,0.98,0.99
3.	SU sensing time (T*sns*)	8,9,10
4.	SU transmission time (T*pkt*)	10,20,30
5.	Cost of Collision cost (*CCOST*)	3,4,5
6.	Reward for transmission (*Reward*)	0.7,0.8,0.9

The issue here is to decide if a SU should sense or transmit for the above values of these 6 metrics. Here the application of neural networks is an efficient means for decision making. Data is organized into two matrices for a neural network to implement the data classification problem. In the input matrix each column has 6 elements for all the above 6 metrics. Each column in the target matrix has 2 elements, depicting SU's action of transmitting or not (sensing). If threshold is less than 0.95(condition when SU should transmit) then a one is recorded in the first element but if threshold is more than 0.95 (conditions when SU should not transmit) then a one is recorded in the second element. For any one combination of the 6 CR network metrics, the ANN has to decide whether the SU should sense or transmit. To simulate real time scenario of wireless traffic in any frequency band, the data can be assumed to follow a periodically similar pattern. Hence it was repeated thrice, to give in all 729 (3^7) entries.

5.2 Developing the Classifier

The next step is to build a neural network which learns to determine the SU action. The random seed is set: rand('seed', 491218382) to avoid different outcomes for each execution as the ANN will not start with random initial weights now. Results for 7 neural network architectures have been obtained. The samples get divided into training, validation and test sets automatically.

5.3 Training- Neural Network

First step is to initialize the network weights and biases so that the network is ready for training. A sample set defining proper network behavior with

network inputs and target outputs is required for the process of training. This step to train the network on the data, uses the function:
net = train(net,cognitiveparameters,result).

The Performance plot in the training window depicts the mse of the network starting at a high level and reducing to a lower level. This implies that the network is learning. There are 3 traces in the plot, as the input and targets vectors are split into 3 sets. One set trains the network, one validates how well the network generalized and last one tests it. Network training continues till it decreases the error.

5.4 Testing the Classifier

An independent value of accuracy for a neural network is provided by a test set. Data from real world is then used to find out the performance of the network for training, validation and test sets. The epoch which gives the least Mean Square Error during validation (in green colour) is identified and presented in a tabular form. Fig. 6, 7,8,9,10,11 and 12 show the training results of the 7 Neural Networks:

Table 2: Least MSE for different Neural Networks

S.No	Neural Network Architecture	Least mse
1.	Feed-Forward Back Propagation	0.0091743
2.	Cascade-Forward Back Propagation	0.0091743
3.	Focused Time-Delay	0.23488
4.	Distributed Time-Delay	0.24717
5.	Elman Back Propagation	0.019099
6.	NARX Network	0.088502
7.	Layer Recurrent	5.8653e-7

The MSE plots of the seven ANNs while training, testing and validating are presented in Fig. 6, 7, 8, 9, 10, 11 and 12. Also, from these plots and from the Table 2 it is clear that the LRNN gives the least mean squared error and the Feed-forward, Cascade-forward and Elman neural networks also train to a very low level of mse.

The error histogram plot for each of the 7 neural networks is presented in the Fig.13,14,15,16,17,18 and 19. These figures and Table 3 reiterate that the LRN gives the least error, Feed-forward and Cascade-forward neural networks also give very low level of error.

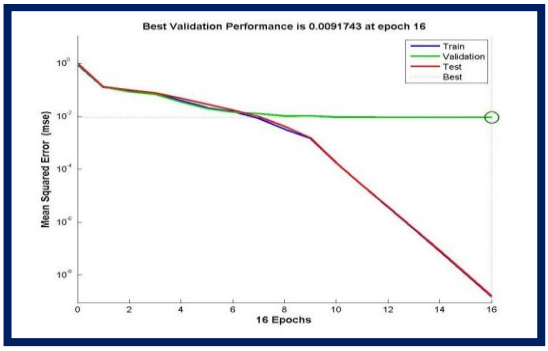

Fig. 6: Training Result of Feed-Forward Network

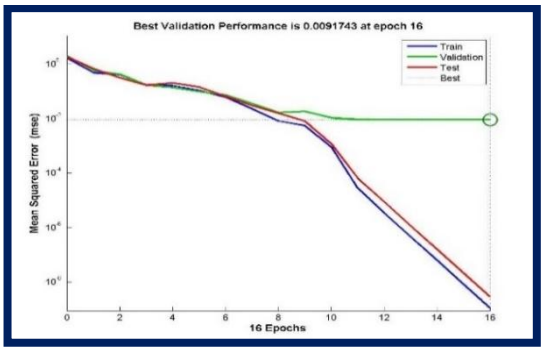

Fig.7: Training Result of Cascade-Forward Back Propagation Network

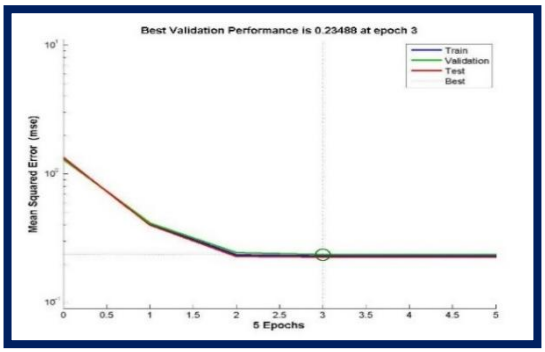

Fig.8: Training Result of Focused Time-Delay Neural Network

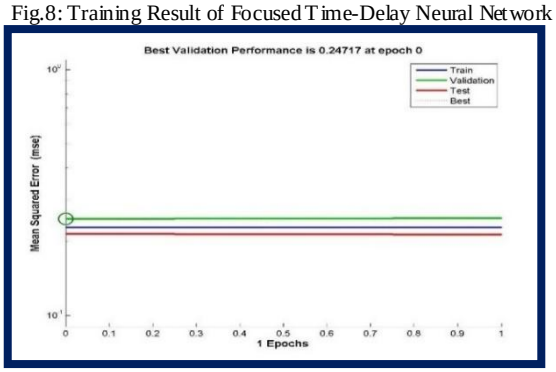

Fig.9: Training Result of Distributed Time-Delay Neural Network

IJASRM
International Journal of Advanced Scientific Research and Management, Volume 4 Issue 6, June 2019
www.ijasrm.com
ISSN 2455-6378

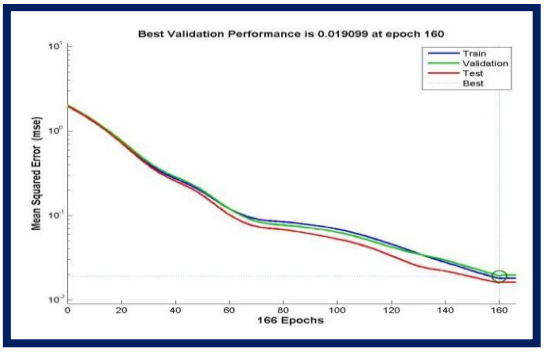

Fig.10: Training Result of Elman Back Propagation Network

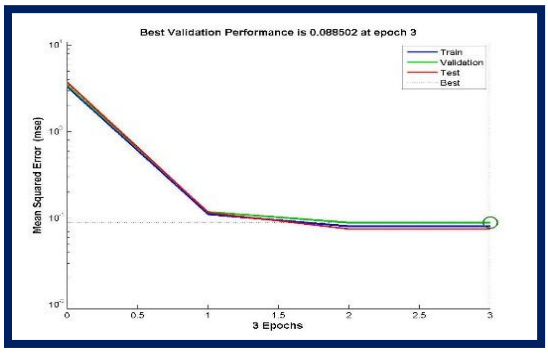

Fig.11: Training Result of NARX Network

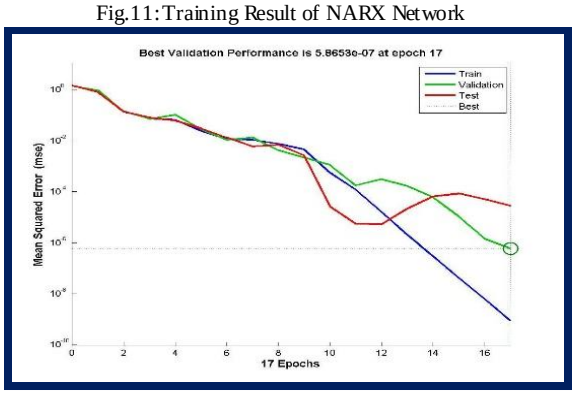

Fig.12: Training Result of Layer Recurrent Network

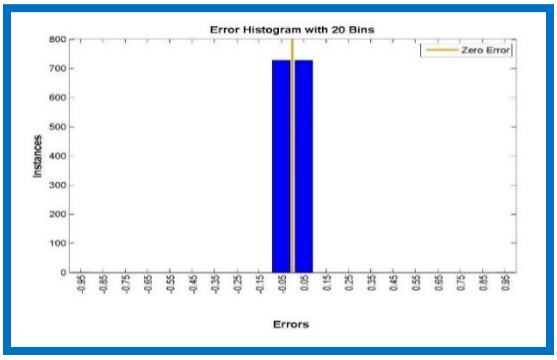

Fig.13: Error Histogram of Feed-Forward Network

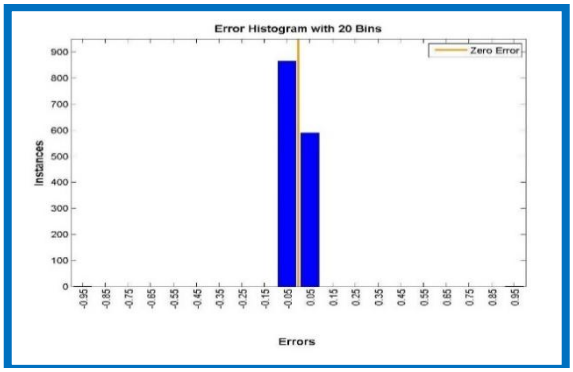

Fig.14: Error Histogram of Cascade-Forward Back Propagation Network

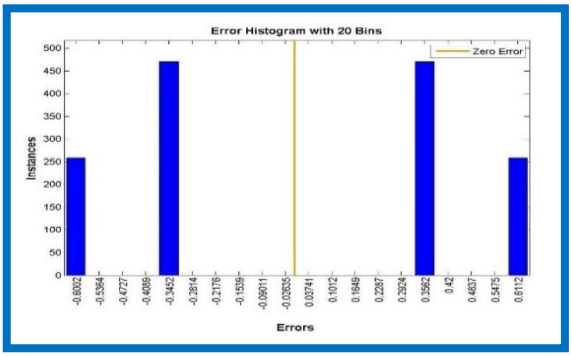

Fig.15: Error Histogram Result of Focused Time-Delay Neural Network

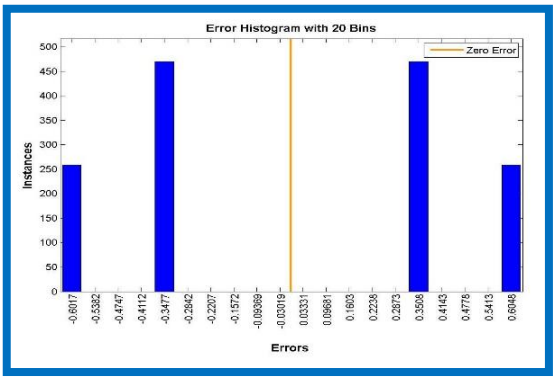

Fig.16: Error Histogram of Distributed Time-Delay Neural Network

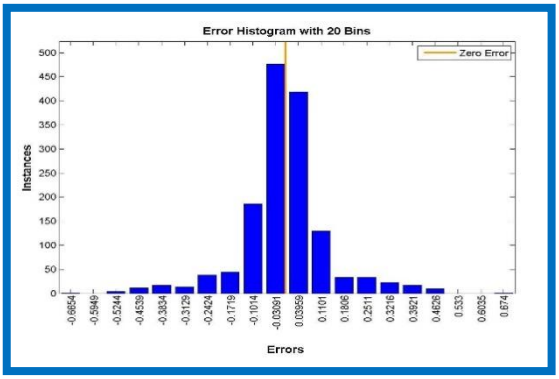

Fig.17: Error Histogram of Elman Back Propagation Network

IJASRM
International Journal of Advanced Scientific Research and Management, Volume 4 Issue 6, June 2019
www.ijasrm.com
ISSN 2455-6378

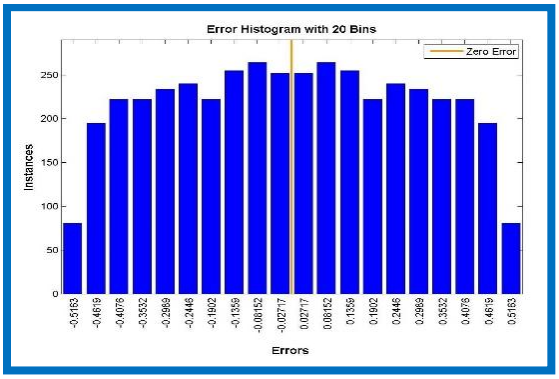

Fig.18: Error Histogram of NARX Network

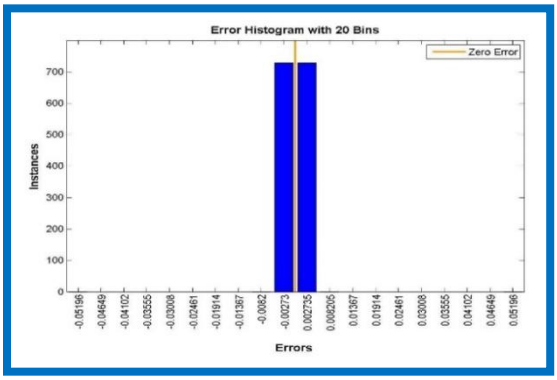

Fig.19: Error Histogram of Layer Recurrent Network

Table 3: Maximum Error Range

S.No	Neural Network Architecture	Maximum Error for 20 Bins
1.	Feed-Forward Back Propagation	± 0.05
2.	Cascade-Forward Back Propagation	± 0.04397
3.	Focused Time-Delay	± 0.6112
4.	Distributed Time-Delay	± 0.6048
5.	Elman Back Propagation	± 0.1101
6.	NARX Network	± 0.4619
7.	Layer Recurrent	± 0.002735

When the simulation of each neural network is run, its architecture is displayed on the screen. The same for each network is presented in the Fig.20,21,22,23,24,25 and 26. These show the architecture of the network with all the layers-input, hidden and output, neurons in each layer and the transfer function used in each layer. They also depict the training algorithm, data division algorithm, the performance parameter chosen and the variation of gradient. The number of layers and number of neurons in each layer considered are optimum. The screen shots of the simulation have been presented below:

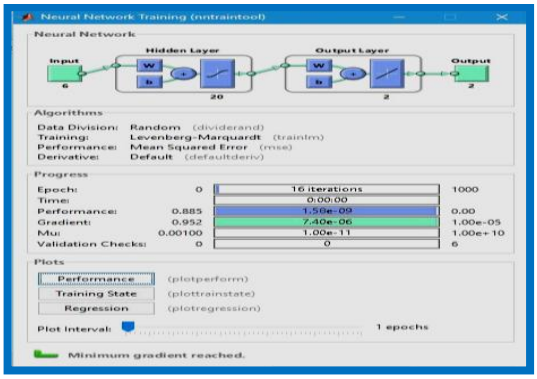

Fig.20: Feed-Forward Network Architecture Simulation

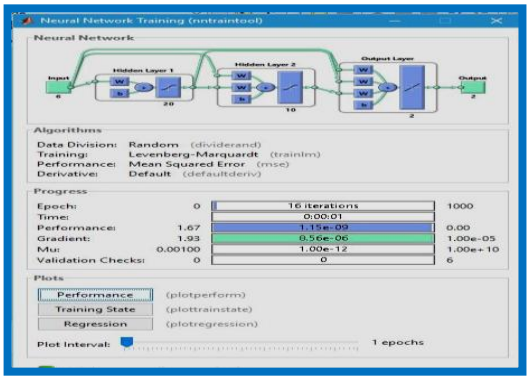

Fig.21: Cascade-Forward Back Propagation Network Architecture Simulation

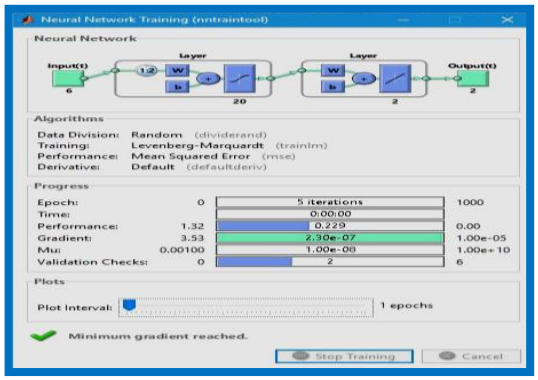

Fig.22: Focused Time-Delay Neural Network Architecture Simulation

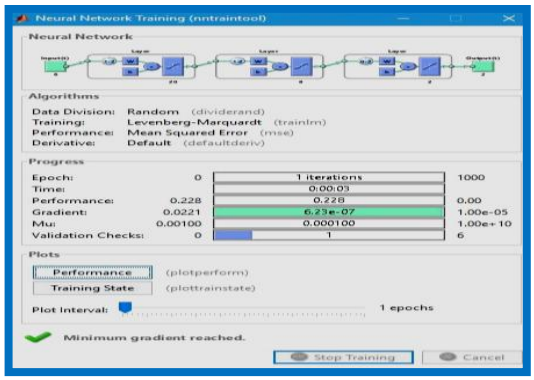

Fig.23: Distributed Time-Delay Neural Network Architecture Simulation

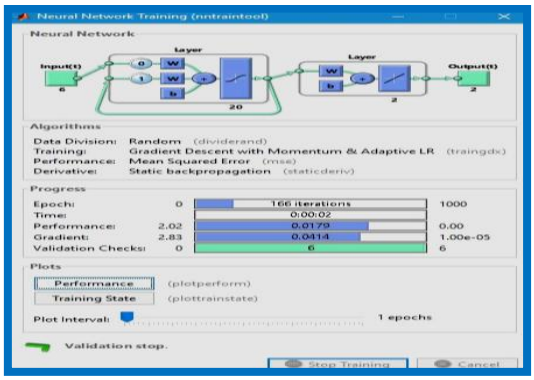

Fig.24: Elman Back Propagation Network Architecture Simulation

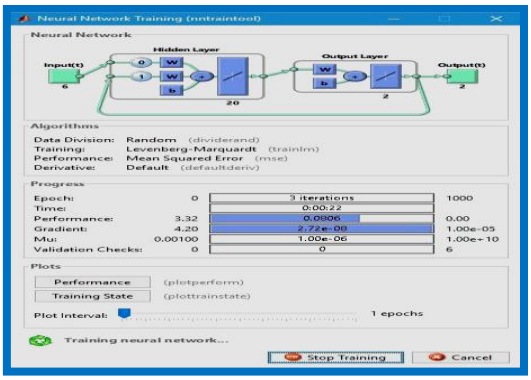

Fig.25: NARX Network Architecture Simulation

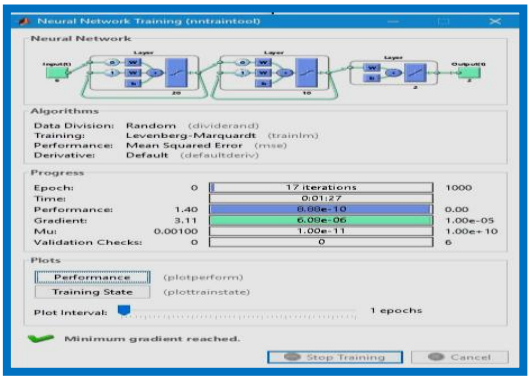

Fig.26: Layer Recurrent Network Architecture Simulation

The confusion plots for each network are obtained next. They depict how well the neural network has fit the data. The confusion matrix shows the percentages of correct and incorrect classifications and is plotted for all samples. The green squares on the matrices diagonal depict correct classifications and the red squares show incorrect classifications. The percentages in the red squares should be very small, indicating few misclassifications if the network has learned to classify properly. If this is not the case then further training or training a network with more hidden neurons is advisable.

Fig. 27, shows the confusion matrix of Feed-Forward Network (FFN) Architecture. The FFN architecture which is one of the architectures giving best accuracy and least time taken has a confusion matrix with 0% mis-classifications and 100% correct classifications.

The confusion matrix of Focused Time-Delay Neural Network (FTDNN) Architecture is depicted by Fig.28. Since FTDNN is the simplest form of dynamic network, the red squares show 33.9% of misclassifications and the green squares have 66.1% of correct classifications.

Fig.27: Confusion Result of Feed Forward Network Architecture

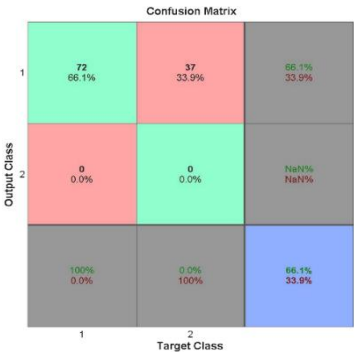

Fig.28: Confusion Result of Focused Time-Delay Neural Network Architecture

8 Inferences and Analysis

From Fig. 29 and Fig. 30 it can be seen that Feed-forward network (FFN) architecture, Cascade-forward back propagation network architecture and Layer recurrent network architecture have the highest percentage accuracy. Elman back propagation network architecture and NARX network also give very high percentage of accuracy. The Distributed time-delay neural network architecture has the least percentage of

accuracy when data is periodically repeating and Focused time-delay neural network architecture has least accuracy for non-repeating data.

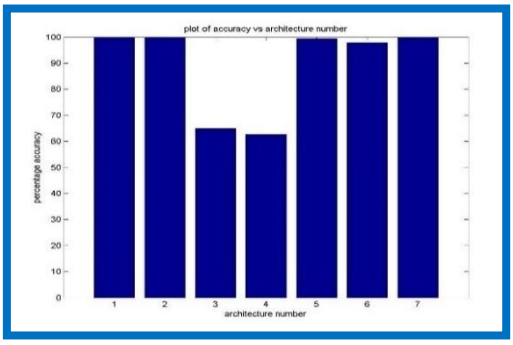

Fig.29: Plot of accuracy v/s architecture (periodically repeating data)

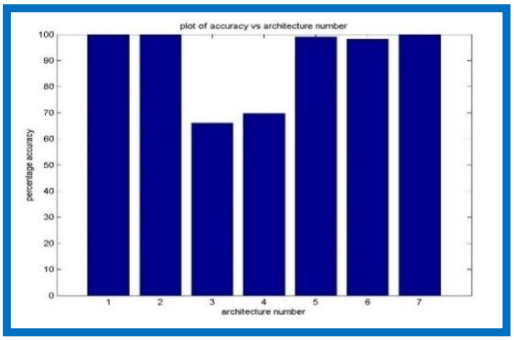

Fig.30: Plot of accuracy v/s architecture (non repeating data)

The Fig. 31 depicts the time taken by each of the architectures in case of periodically repeated data. Feed-Forward network architecture, Focused Time-Delay Neural Network architecture and Elman back propagation network take the least time. NARX network, Distributed Time-delay neural network and Cascade-Forward Back Propagation Network Architecture take more time while Layer Recurrent Network architecture takes the most time. In case of non-repeating data (Fig. 32) Focused Time-Delay Neural Network architecture and Cascade-Forward Back Propagation Network Architecture take least time. Thus, accuracy and time taken plots and Table 4 show that Feed-Forward Network Architecture and Elman Back Propagation Network Architecture are the best suitable for when traffic and channel data repeats periodically while Cascade-Forward Back Propagation Network is the best suitable for when traffic and channel data does not repeat.

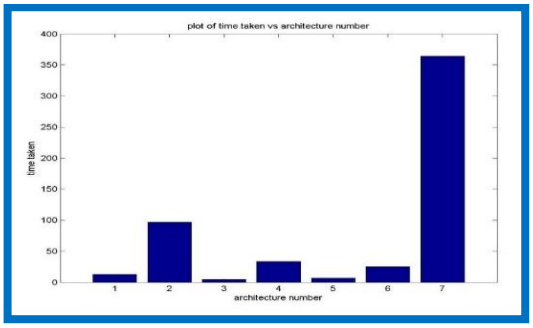

Fig.31: Plot of time taken v/s architecture (periodically repeating data)

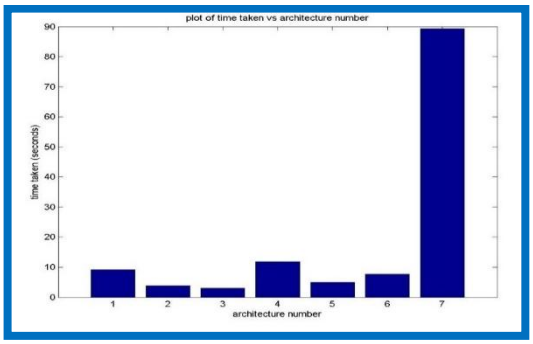

Fig.32: Plot of time taken v/s architecture (non-repeating data)

Table 4: Accuracy and Time taken by different Neural Networks

S. No.	Neural Network Architecture	Periodically Repeating Data		Non-Repeating Data	
		% Optimal	Time Taken (secs)	% Optimal	Time Taken (secs)
1.	Feed-Forward	100%	16.56	100%	9.07
2.	Cascade-Forward	100%	95.79	100%	3.83
3.	Focused Time-Delay	64.93%	3.39	66.05%	3.04
4.	Distributed Time-Delay	62.80%	31.72	69.72%	11.78
5.	Elman Back Propagation	99.39%	5.62	99.08%	4.97
6.	NARX Network	97.86%	17.32	98.16%	7.65
7.	Layer Recurrent	100%	373.29	100%	89.29

References

[1] Bruce A. Fette, "Cognitive Radio Technology", Elsevier Inc, Second Edition, 2009.
[2] Lars Berlemann, Stefan Mangold, "Cognitive Radio and Dynamic Spectrum Access", John Wiley & Sons Ltd , First edition,2009.

[3] L. Lai, H. E. Gamal, H. Jiang, H. V. Poor, "Cognitive medium access: Exploration, exploitation and competition", IEEE Transactions on Mobile Computing,10(2), pp.239-53, Feb. 2011.
[4] Q. C. Zhao, S.Geirhofer, L. Tong, and B. M. Sadler, "Optimal dynamic spectrum access via periodic channel sensing," Wireless Communications and Networking Conference, pp. 33-37, 2007.
[5] Y.-C. Liang, Y. Zeng, E. Peh, and A. T. Hoang, "Sensing-throughput tradeoff for cognitive radio networks," IEEE Transactions on Wireless Communications, 7(4), pp. 1326–1337, April 2008.
[6] P. Wang, L. Xiao, S. Zhou, and J. Wang, "Optimization of detection time for channel efficiency in cognitive radio systems," Wireless Communications and Networking Conference, pp. 111– 115, 2007.
[7] H. Kim and S. K. G., "Efficient discovery of spectrum opportunities with MAC-layer sensing in cognitive radio networks," IEEE Transactions 2008.
[8] N. B. Chang and M. Liu, "Optimal channel probing and transmission scheduling for opportunistic spectrum access," 13th annual ACM International Conference on Mobile Computing and Networking, pp. 27–38, 2007.
[9] Y. Chen, Q. Zhao, and A. Swami, "Joint design and separation principle for opportunistic spectrum access in the presence of sensing errors," IEEE Transactions on Information Theory, 54(5), pp. 2053–2071, 2008.
[10] Q. Zhao and K. Liu, "Detecting, tracking, and exploiting spectrum opportunities in unslotted primary systems," IEEE Radio and Wireless Symposium (RWS), 2008.
[11] S. Huang, X. Liu, Z. Ding, "Short Paper: On Optimal Sensing and Transmission Strategies for Dynamic Spectrum Access," 3rd IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks, pp.1-5, Oct. 2008.
[12] I.K. Aulakh, R. Vig, "Optimization of Secondary User Access in Cognitive Radio Networks" in IEEE Recent Advances in Engineering and Computational Sciences, pp 1-6, March 2014.
[13] I.K. Aulakh, R. Vig, "Optimization of SU's Probability of False Alarm for Dynamic Spectrum Access in Cognitive Radio" in Proceedings of IEEE International Conference on Computing for Sustainable Global Development, pp 710-715, March 2014.
[14] I.K. Aulakh, R. Vig, "Secondary User Aggressiveness Optimization in Sensing-Transmission Scheduling for Cognitive Radio Networks", Journal of Networks, 10(10), pp. 543-550, Jan. 2016.
[15] I.K. Aulakh, R. Vig, "Secondary User Sensing Time Optimization in Sensing-Transmission Scheduling for Dynamic Spectrum Access in Cognitive Radio", Journal of Computer Science, 11(8), pp. 880-891, Aug. 2015.
[16] I.K. Aulakh, R. Vig, "Secondary User Transmission Protection Optimization in Sensing-Transmission Scheduling under varying Channel Noise in Cognitive Radio Networks", International Journal of Systems, Control and Communications, in press**,** March, 2016.
[17] K. Tsagkaris, A. Katidiotis, P. Demestichas, "Neural network-based learning schemes for cognitive radio systems", Computer Communications,31(14), pp 3394–34045 September 2008,
[18] A.He, K.K. Bae, T.R.Newman, J.Gaeddert, K. Kim, R.Menon, L.Morales-Tirado, J. Neel,Y. Zhao, J.H.Reed, W.H.Tranter, "A Survey of Artificial Intelligence for Cognitive Radios",IEEE Transactions on Vehicular Technology, 59(4), pp.1578–1592, May 2010
[19] Y.J. Tang, Q. Y. Zhang, W. Lin, "Artificial Neural Network Based Spectrum Sensing Method for Cognitive Radio", IEEE 6th International Conference on Wireless Communications Networking and Mobile Computing, pp. 1-4, Sept 2010.
[20] M. Bkassiny, Y. Li, S.K.. Jayaweera, "A Survey on Machine-Learning Techniques in Cognitive Radios", IEEE Communications Surveys and Tutorials, 15(3), pp:1136 – 1159, July 2013
[21] M. Malhotra, I.K. Aulakh, R. Vig, "A review on energy-based spectrum sensing in Cognitive Radio Networks" in IEEE International Conference on Futuristic Trends on Computational Analysis and Knowledge Management, pp 561-65, Feb., 2015.
[22] I.K. Aulakh, S. Singh, R. Vig, "Spectrum Sensing Techniques in CRNs: A Review", IEEE 3rd International Conference on Reliability, Infocom Technologies and Optimization, pp.188-193, Oct. 2014.