Algorithm for Determining the Probability of Arrival of an SU to a Base Station in Cognitive Radio Networks

Danilo Alfonso López [1*], Leyla Carolina Sarmiento[2], Edwin Rivas Trujillo[3]
[1]Department of Electronic Engineering,
University Distrital Francisco José de Caldas, Bogotá, Colombia (South America)
danilo.lopez.sarmiento@gmail.com
dalopezs@udistrital.edu.co.
[2]Department of Psychology,University La Sabana, Bogotá, Colombia (South America)
leylacls@gmail.com
[3]Department of Electrical Engineering, University Distrital Francisco José de Caldas,
Bogotá, Colombia (South America)
erivas@udistrital.edu.co

Abstract - This document presents the development of an algorithm that predicts the arrival of a secondary user (SU) to a base station (BS) in a cognitive network based on infrastructure, requesting a Best Effort (BE) or Real Time (RT) type of service with a determined bandwidth (BW) implementing neural networks. The algorithm dynamically uses a novel neural network construction technique using the geometric pyramid topology and trains a Multilayer Perceptron Neural Networks (MLPNN) based on the historical arrival of an SU to estimate future applications; This will allow to manage more quickly the information in the BS for the selection of the best channel in CRN as it is placed before the arrival of the SUs. As a final result the software application determines the probability of arrival at a future time point and calculates the performance metrics to measure the effectiveness of the predictions made.

Keywords: Cognitive radio, MLPNN, base station, prediction, best effort, real time.

1. Introduction

Studies by the Federal Communications Commission (FCC) show that the saturation and scarcity of the electromagnetic spectrum is due to its mismanagement and not to the scarcity of this resource, which proves that the policies implemented to avoid interference between networks and Operators have led to a low actual use of the assigned spectrum for some channels and a high use for others[1, 2]. In this sense, the concept of cognitive radio (CR)[3] has been proposed as a means of aiding the establishment and implementation of technical solutions aimed at benefiting the spectral efficiency in present and future wireless networks in a dynamic way. The RC is based on 4 main functions, which are described in[4], and the decision-making stage consists of the characterization, channel selection and reconfiguration sub-stages of the cognitive nodes. The development of this article focuses on the characterization phase, which will focus on the modeling of SU behavior and subsequent prediction to determine the probability of arrival of a cognitive node to a central station at a future time point; This in order to give an indication to the BS of the type of users and requirements that will have to be processed (assign channels). This is intended to reduce the time it takes the BS to assign channels, optimizing the system[4]. Based on this premise, the application of artificial intelligence techniques allows for adapting the changes in the arrival behavior of SUs in a BS based on autonomous learning. Taking advantage of this property, MLPNN is used and evaluated in order to determine its response to this type of future estimates.

2. Artificial Neural Networks

Artificial Neural Networks (ANNs) are computational models that emerged as an attempt to achieve mathematical formalizations about the structure of the brain. These imitate the structure of the nervous system, focusing on the functioning of the human brain, based on learning through experience, with the consequent extraction of knowledge from it. An ANN can be considered a mathematical model of mental and brain activity theories, based on the exploitation of parallel local processing and the properties of distributed representation [5].

3. MLPNN Model for Estimating SU Arrival

In the following subsections, the process of creation of the algorithm is outlined from a machine learning perspective. In the first part, the form of representation of knowledge is presented, in which the form of representation of the input and output variables is defined; Later the construction of the architecture of the neural network from a dynamic point of view is defined, the neural network changes its topology according to the size of the channel occupancy history. Finally, the process of training and validation of the created neural network and the metrics to measure the performance of the same is explained.

3.1 Representation of an SU history

$\{x(i), y(i)\}$ is defined as a pair of coordinates in R^{n*3}, where x(i)is the binary representation of a time unit in a Space R^n; n is the number of digits in the binary representation and y(i)in a space R^3, where the first component corresponds to the request or not of a Best Effort type service; The second, the request for a Real Time type service and the third the bandwidth required in KHz. An example of this representation corresponds to the one shown in the following representation (Equation 1):

$$\{x(1), y(1)\} = \{[0\ \ 0\ \ 0], [0\ \ 0\ \ 0]\}$$

$$\left\{\begin{array}{c|c} \begin{matrix}[0 & 0 & 0]\\ [0 & 0 & 1]\\ [0 & 1 & 0]\\ [0 & 1 & 1]\\ [1 & 0 & 0]\\ [1 & 0 & 1]\\ [1 & 1 & 0]\\ [1 & 1 & 1]\end{matrix} & \begin{matrix}[0 & 0 & 0]\\ [0 & 1 & 20]\\ [0 & 1 & 20]\\ [1 & 0 & 43]\\ [1 & 0 & 44]\\ [1 & 0 & 45]\\ [0 & 1 & 48]\\ [0 & 1 & 45]\end{matrix} \\ \underbrace{\qquad}_{Inputs\ x(i)} & \underbrace{\qquad}_{Outputs\ y(i)} \end{array}\right\} \tag{1}$$

This first approximation of representation of SUs with their respective characteristics considers the neural network topology without taking into consideration the nature of the data that are intended to be characterized. Because the transfer functions between each layer of the neural network are given by a sigmoid function, the range of the data is 0 to 1. This is not considered a problem for the data domain that is intended to characterize except for the case of R^3whose third component has domain in the natural numbers (and which corresponds to the bandwidth). In this sense, it is proposed to separate the data set (shown above) into two groups and to use two neural networks. The first network specializes in the characterization of data set $y(i)^1$, represented as described in Equation 2, and follows the design criteria:

- The number of neurons in the input layer corresponds toR^nde $x(i)$.
- The number of neurons in the output layer corresponds to the dimension R^2 of $y(i)$, each of the neurons will be specialized in modeling a secondary user characteristic.
- The number of neurons in the hidden layers is obtained following the geometric pyramid topology.

$$\left\{\begin{array}{c|c} \begin{matrix}[0 & 0 & 0]\\ [0 & 0 & 1]\\ [0 & 1 & 0]\\ [0 & 1 & 1]\\ [1 & 0 & 0]\\ [1 & 0 & 1]\\ [1 & 1 & 0]\\ [1 & 1 & 1]\end{matrix} & \begin{matrix}[0 & 0]\\ [0 & 1]\\ [0 & 1]\\ [1 & 0]\\ [1 & 0]\\ [1 & 0]\\ [0 & 1]\\ [0 & 1]\end{matrix} \\ \underbrace{\qquad}_{Inputs\ x(i)} & \underbrace{\qquad}_{Outputs\ y(i)^1} \end{array}\right\} \tag{2}$$

The second neural network specializes in the characterization of the data set $y(i)^2$, represented in Equation 3, with the following criteria:

- The number of neurons in the input layer corresponds to R^nde $x(i)$.
- The number of neurons in the output layer corresponds to the dimension R^1 de $y(i)^2$, each of the neurons will be specialized in modeling a secondary user characteristic.
- The number of neurons in the hidden layers is obtained following the geometric pyramid topology.

$$\left\{\begin{array}{c|c} \begin{matrix}[0 & 0 & 0]\\ [0 & 0 & 1]\\ [0 & 1 & 0]\\ [0 & 1 & 1]\\ [1 & 0 & 0]\\ [1 & 0 & 1]\\ [1 & 1 & 0]\\ [1 & 1 & 1]\end{matrix} & \begin{matrix}[0]\\ [20]\\ [20]\\ [43]\\ [44]\\ [45]\\ [48]\\ [45]\end{matrix} \\ \underbrace{\qquad}_{Inputs\ x(i)} & \underbrace{\qquad}_{Outputs\ y(i)^2} \end{array}\right\} \tag{3}$$

3.2 Mathematical model of the neuronal system

For the development of the operation of the neural network, the set of examples shown in equation 2 is considered. Following the guidelines proposed for the construction of the proposed neural network, we obtain a three-layer system with 3 neurons in the input layer, 2 in the hidden layer and 2 in the output layer, a graphical representation of this neural network is shown in Figure 1.

In addition, the following variables are defined:

m: Number of layers of the neural network; θ: Control weight matrix, which maps (generates an association) from one layer i to one layer i + 1; A: activation unit in layer i.

The procedure for calculating the output of the neural network is defined as shown in Equation 4, called forward propagation algorithm.

$$A^{(i)} = g\left(\theta^{(i-1)T} A^{(i-1)}\right) \qquad (4)$$

where, T is the transposed operation; $A^{(i)}$is the layer output to be calculated; $A^{(i-1)}$ corresponds to the previous layer output; $i = 1, 2, 3, \ldots., m$; $A^{(i)} = X$; $g$the sigmoid function.

Considering the control weight matrix$\theta^{(1)}$, we proceed to calculate the transition from the input layer to the hidden layer (Equations 5 and 6).

$$\theta^{(1)} = \begin{bmatrix} \theta_{11}^{(1)} & \theta_{12}^{(1)} \\ \theta_{21}^{(1)} & \theta_{22}^{(1)} \\ \theta_{31}^{(1)} & \theta_{32}^{(1)} \end{bmatrix} (5)$$

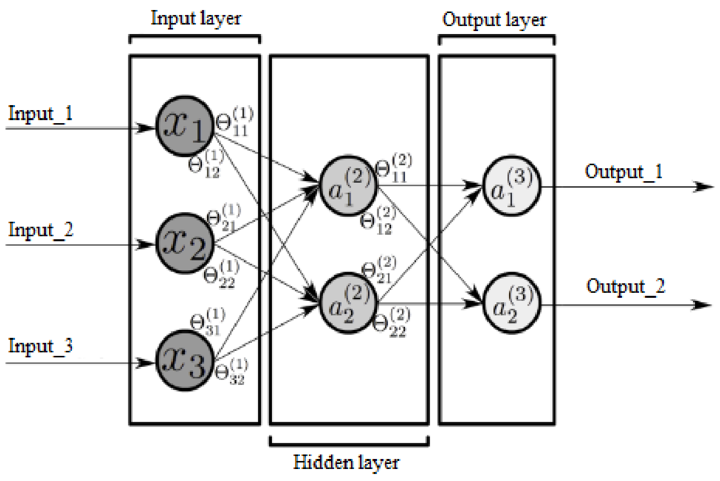


Figure 1. Representation of the MLB for the data set of equation 2.

$$A^{(1)} = X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad (6)$$

Thus, the transition from the input layer to the output layer would be given as described in Equation 7.

$$A^{(2)} = \begin{bmatrix} g\left(x_1\theta_{11}^{(1)} + x_2\theta_{21}^{(1)} + x_3\theta_{31}^{(1)}\right) \\ g\left(x_1\theta_{12}^{(1)} + x_2\theta_{22}^{(1)} + x_3\theta_{32}^{(1)}\right) \end{bmatrix} \qquad (7)$$

For simplicity, the following variables are defined for the matrix $A^{(2)}$ (Equation 8).

$$A^{(2)} = \begin{bmatrix} a_1^{(2)} \\ a_2^{(2)} \end{bmatrix} \qquad (8)$$

When calculating the transition from the hidden layer to the output layer by reference to the control weight matrix $\theta^{(2)}$ (Equation 9 y 10):

$$\theta^{(2)} = \begin{bmatrix} \theta_{11}^{(2)} & \theta_{12}^{(2)} \\ \theta_{21}^{(2)} & \theta_{22}^{(2)} \end{bmatrix} \qquad (9)$$

$$A^{(3)} = \begin{bmatrix} g\left(a\theta_{11}^{(2)} + a_2\theta_{21}^{(2)}\right) \\ g\left(a_1\theta_{12}^{(2)} + a_2\theta_{22}^{(2)}\right) \end{bmatrix} \qquad (10)$$

3.3 Flowchart for the Learning Algorithm

MLPNN Training Diagram is show in Figure 2.A fragment of the MLPNN code for neural network optimization is shown in Figure 3. It should be noted that the sequence shown implies the existence of two Theta1 and Theta2 arrays corresponding to the average weight matrices of the neural network.

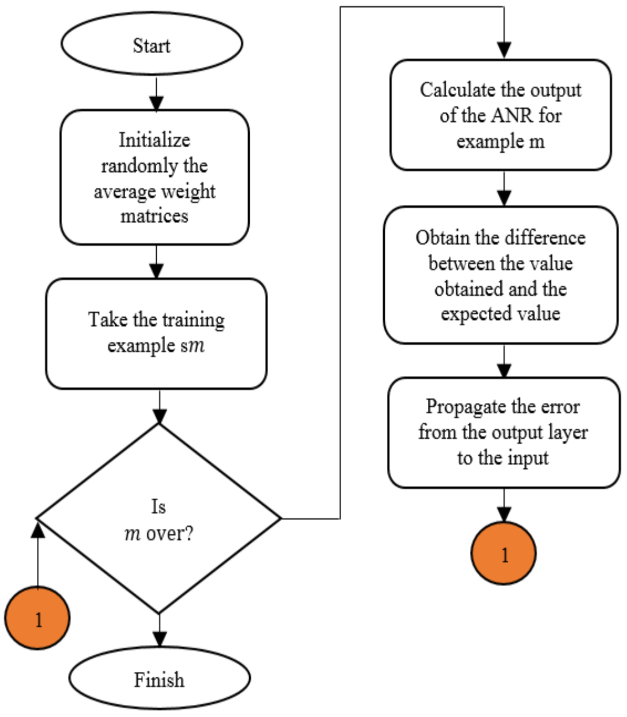

Start
Initialize randomly the average weight matrices
Take the training example sm
Is m over?
1
Finish
Calculate the output of the ANR for example m
Obtain the difference between the value obtained and the expected value
Propagate the error from the output layer to the input
1

Figure 2. MLPNN Training Diagram.

The algorithm takes the training examples to find the optimal values of Theta1 and Theta2 that minimize the error obtained.

MLPNN Algorithm

```
$delta\_accum\_1 = zeros\ (size\ (Theta1));$
$delta\_accum\_2 = zeros\ (size\ (Theta2));$
for $t = 1{:}m$ do
    $a\_1 = X(t,:\ );$
    $z\_2 = a\_1\ *\ Theta1';$
    $a\_2 = [1\ sigmoid\ (z\_2)];$
    $z\_3 = a\_2\ *\ Theta2';$
    $a\_3 =\ sigmoid\ (z\_3);$
    $y\_i = zeros\ (1, K);$
    $y\_i(y(t)) = 1;$
    $delta\_3 = a\_3 - y\_i;$
    $delta\_2 = delta\_3\ *\ Theta2\ .$
            $*\ sigmoidGradient\ ([1\ z\_2)];$
    $delta\_accum\_1 = delta\_accum\_1 +$
            $delta\_2\ (2{:}end)'\ *\ a\_1;$
    $delta\_accum\_2 = delta\_accum\_2 + delta\_3'\ *$
    $a\_2;$
end;
$Theta1\_grand =\ delta\_accum\_1\ /\ m$
$Theta2\_grand =\ delta\_accum\_2\ /\ m$
```

Figure 3. MLPNN algorithm.

3.4 Neural Network Training

During the training process of the neural network, the value of the control weight matrices is determined using the back propagation algorithm which includes the following guidelines within its algorithm:

- Randomly initialize the weights of matrices with numbers between -1 and 1.
- Implement the forward propagation algorithm to obtain $A^{m}$for any x x(i).
- Calculating the cost $J(\theta)$ from equation 11, in order to obtain the difference between the expected values and the values obtained, the objective is to have its value approach as close to 0.

$$J(\theta) = -\frac{1}{m} * \sum_{x=0}^{m} \sum_{x=0}^{n} \Big(y(x) * \log\big((A^{(x)}\big)_m + \big(1 - y(x)\big) * \log\big((A^{(x)}\big)_n \Big) \tag{11}$$

- Calculate the partial derivatives of $\frac{dJ(\theta)}{d\theta_{ij}^{k}}$ in order to minimize the error to the maximum (equation 12).

$$\theta_{ij}^{k} = \theta_{ij}^{k} - \alpha \frac{dJ(\theta)}{d\theta_{ij}^{k}} \tag{12}$$

4. Software Implementation

To determine the ability and precision of the MLPNN algorithm to calculate the probability of arrival of the next SU (with BE or RT and BW type QoS criteria) to the BS, a software application was developed in C#.Figure 4 shows the creation phase of the request history for BE, RT and BW (for the figure the past behavior is displayed requesting BE and RT only).

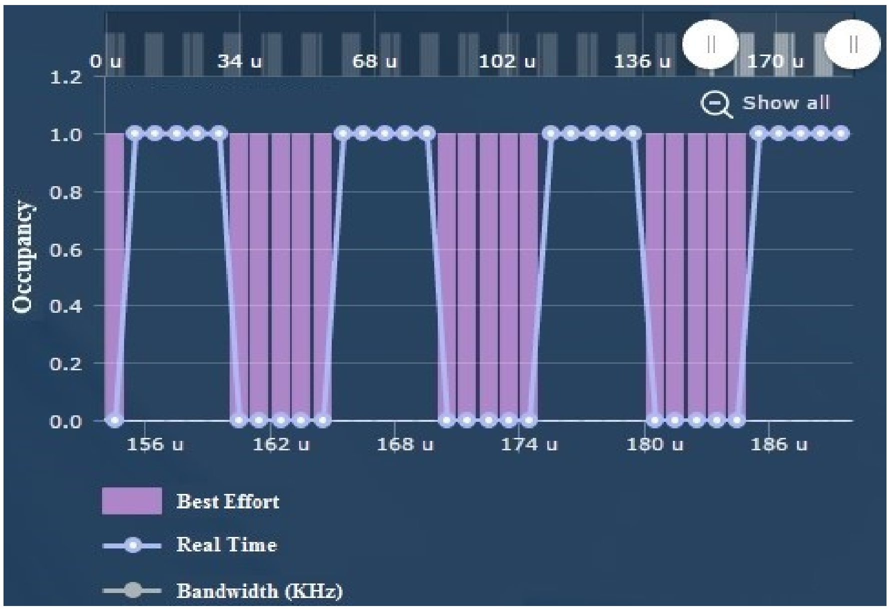
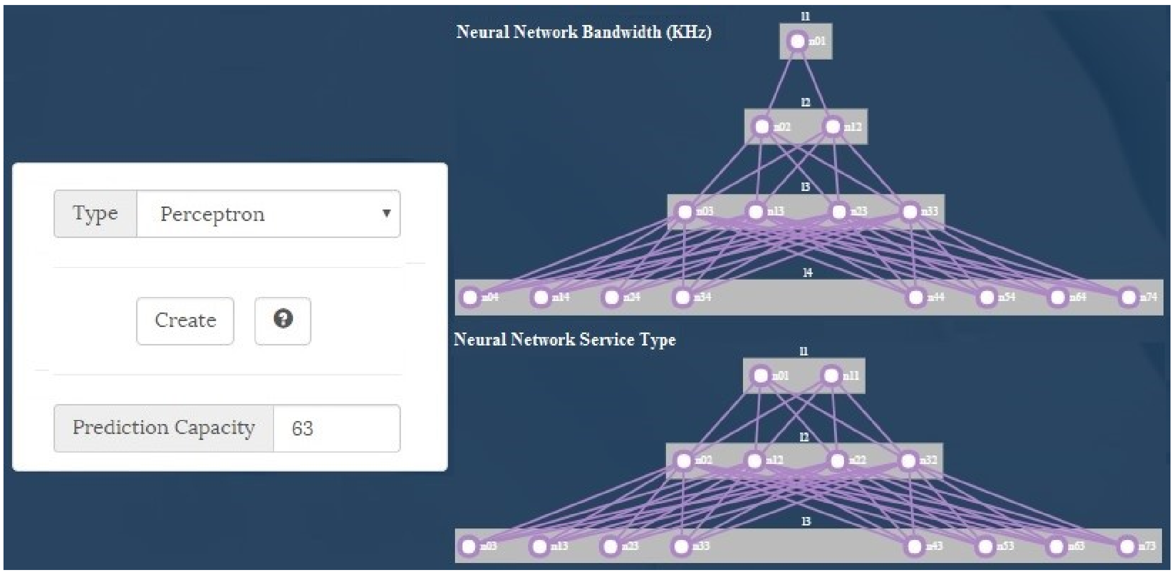


Figure 4. Software for predicting arrival of SUs (history creation).

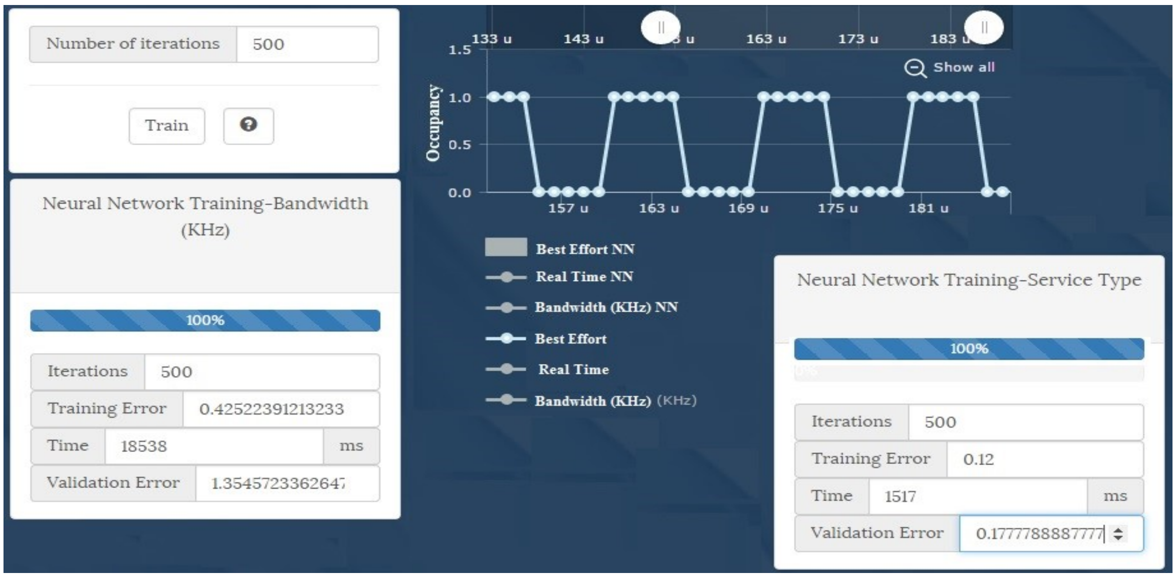
Figure 5, shows an on-screen capture of the second phase of the software, where the two MLPNN neural networks are created (the first one specialized in estimating the BW that is likely to request the SU, and the second trained to predict the probability of requesting a BE or RT service).



Type	Perceptron
Prediction Capacity	63

Figure 5. Creation stage of the specialized MLPNNs.

Figure 6 graphically represents the training or learning stage of neural networks. Only the modeling for the historical behavior in the BE type requests is shown, where it is clear that the MLPNN manages to establish the (past) pattern requested by the SU.



Number of iterations	500

Neural Network Training-Bandwidth (KHz)

100%

Iterations	500	
Training Error	0.42522391213233	
Time	18538	ms
Validation Error	1.3545723362647	

Neural Network Training-Service Type

100%

Iterations	500	
Training Error	0.12	
Time	1517	ms
Validation Error	0.1777788887777	

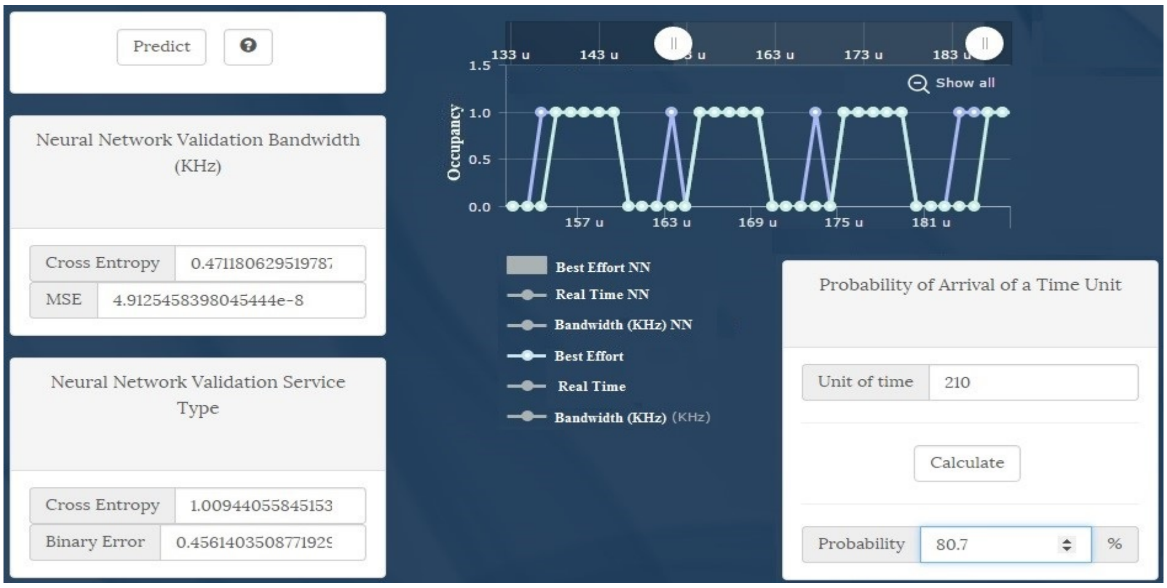
Figure 6. Training phase of the neural network.

The last phase of the algorithm corresponds to the prediction, which will estimate the future 30% of the historical data and compare them with the actual behavior (Figure 7).

5. Results evaluation

In order to test the developed proposal, three test cases (from MS Excel) were generated using uniform, Poisson and exponential distributions.

The quantitative results during the training phase for 200 examples are shown in Tables 1, 3, 5; and the responses in the estimation of the RT, BE and AB requests are observed in Tables 2, 4, 6.



Neural Network Validation Bandwidth (KHz)

Cross Entropy	0.47118062951978
MSE	4.9125458398045444e-8

Neural Network Validation Service Type

Cross Entropy	1.00944055845153
Binary Error	0.456140350877192

Probability of Arrival of a Time Unit

Unit of time	210	
Probability	80.7	%

Figure 7. Prediction phase (probability calculation of a QoS request) of the neural network.

Table 1. Training results for the "exponential distribution" test case.

Metrics	1° MLPNN (BW)	2° MLPNN (BE y RT)
Iterations	500000	500000
Training Error	0,13705	0.04018
Time (msec)	350856	261833
Validation Error	0.00027	0.00027
Success Rate (%)	62	99

Table 2. Prediction results for the "exponential distribution" test case.

Metrics	1° MLPNN (BW)	2° MLPNN (BE y RT)
Cross Entropy	0.43791	4,70093
MSE	0.05005	N/A
Binary error	N/A	0,47761
Success Rate (%)	48	72

Table 3. Training results for the "Poisson distribution" test case.

Metrics	1° MLPNN (BW)	2° MLPNN (BE y RT)
Iterations	500000	500000
Training Error	0.37262	0.17537
Time (msec)	333243	307682
Validation Error	0.00205	0.00205
Success Rate (%)	11	95

Table 4. Prediction results for the "Poisson distribution" test case.

Metrics	1° MLPNN (BW)	2° MLPNN (BE y RT)
Cross Entropy	0.44064	0.47060
MSE	0.00767	N/A
Binary error	N/A	0.1875
Success Rate (%)	5	91

Table 5. Training results for the "uniform distribution" test case.

Metrics	1° MLPNN (BW)	2° MLPNN (BE y RT)
Iterations	500000	500000
Training Error	0.54275	0.29872
Time (msec)	354977	357718
Validation Error	0.00205	0.02485
Success Rate (%)	3	93

Table 6. Prediction results for the "uniform distribution" test case.

Metrics	1° MLPNN (BW)	2° MLPNN (BE y RT)
Cross Entropy	0.90017	4,70093
MSE	0.10927	N/A
Binary error	N/A	0,89655
Success Rate (%)	2	55

The results found in the prediction suggest that the success percentage is low when predicting the BW to be requested by the SU. It should be noted that this metric evaluates that at any time point the expected value is equal to the value obtained without any margin of error. In this sense, for example, for the exponential distribution (figure 8), it is observed that the neural network identified the pattern, which is why the MSE (which in this case shows the difference between the expected and minimum values) is very small and in the order of hundredths.

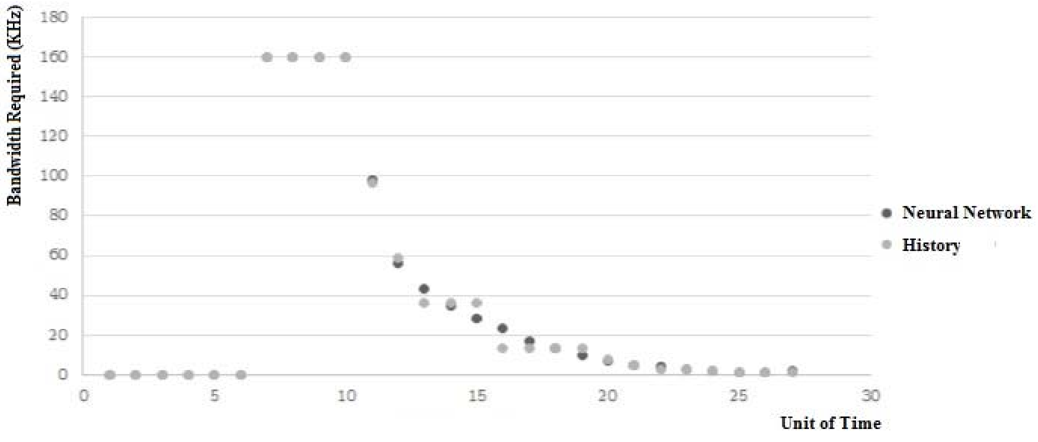


Figure 8. Predicted exponential distribution for variable BW.

Another characteristic that can be drawn from the system behavior from the response given to the test cases is that it was possible to identify patterns for the Exponential and Poisson distributions; However, the "Uniform" case, as it did not present a pattern in its historical data, it was not possible to model or predict its behavior.

6. Conclusions.

The present paper proposes the development of an algorithm to estimate the probability of arrival of SUs to the central station of a cognitive radio network requesting a BE or RT type service, with a certain BW. The results show that the system is more efficient when the MLPNN can establish a pattern in the historical sequence; Otherwise, the success percentage in the estimation of the next request by an SU may be very low, rendering its implementation unviable because the channels reserved by the base station may not meet the characteristics that cognitive users will actually require.

7. References

[1] Galvis A. Accesodinámico alespectro: Estado actual, tendencias y retos. Journal entre Ciencia e Ingeniería.2008;2(4): 38-57.
[2] Rodríguez D, Paz H, Bohórquez M. Cognitive radio technology in the UHF band. Journal Tecnura. 2012; 18(39): 138-155.
[3] Mitola J. Software radios - survey, critical evaluation and future directions, in Proceedings of the National Telesystems Conference (NTC 1992). Washington D.C, EE.UU. 1992 May 19-20.
[4] López D, Trujillo E, Gualdron O. Elementos fundamentales que componen la radio cognitiva y asignación de bandas espectrales. Journal Información tecnológica. 2015; 26(1): 23-40.
[5] 5.López R, Fernández J. Las redes neuronales artificiales: Fundamentos teóricos y aplicaciones prácticas. Ed Netbiblo, 2008, ISBN: 978-84-9745-246-5.