International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 22 (2018) pp. 15836-15842

Bayesian Regularization in Multi-Layer Perceptron Artificial Neural Network Model to Predict Signal Power Loss Using Measurement Points

Virginia Chika Ebhota[1],*, Joseph Isabona[2], Viranjay M. Srivastava[3]

[1,2,3]Department of Electronic Engineering, Howard College, University of KwaZulu-Natal, Durban, South Africa.

Correspondence Author

Abstract

Alternatively, artificial neural networks (ANNs) have been applied for signal power loss prediction. This work studies the performance of five different first and second order approximation learning algorithms. The learning algorithms have been used to train measured data from a Long Term Evolution (LTE) network. The measured data serves as inputs to a Multi-Layer Perceptron (MLP) artificial neural network model and the learning algorithms are used to predict the pattern of the signal power loss. Its focus is to find out the algorithm with the least measurement errors using first order statistical performance indicators.

Keywords: Signal power loss, Multi-layer Perceptron neural network, learning algorithms.

1. INTRODUCTION

In determining loss in signal power during electromagnetic wave propagation from one point to another or to multiple points in wireless communication environment as seen in most cases, there is need to comprehend the pattern of radio frequency (RF) propagation in that environment. Signal power loss prediction during propagation in outdoor environment is both a difficult and complex task as a result of different physical mechanisms such as reflection, refraction, scattering and multipath phenomenon [1, 2].

Apart from the traditional empirical and deterministic prediction models, artificial neural networks models have successfully been used as better prediction alternatives. Artificial neural networks can be defined as adaptive statistical tools which models almost the same way as the biological nervous system. Like human, ANNs are capable of learning by examples such as representation of a given process by mimicking related examples. As a result of its accurateness, simplicity and litheness in adapting to different environments, with distinguished characteristics such as ability to learn from data, generalization of patterns in data and ability to model non-linear functions, they have been employed in solving difficult problems such as pattern recognition and regression problems in different fields [3, 4]. Employing artificial neural network algorithms for problem solving is on the basis of their stochastic and evolutionary methods in finding out the relationship among physical parameters of problems [5, 6]. They fundamentally find out the impact of different mechanisms of the propagation phenomenon on the signal power loss without any necessary and complex mathematical calculations. The prediction of signal propagation loss using the technique of ANNs has been introduced in various literatures. [1] Used artificial neural network models for the prediction of Path loss in urban area while a hybrid model that combines a traditional model, COST Walfisch-Ikegami model with adaptive neural component was used in [6]. The influence of training set selection in ANN based path loss propagation predictions was presented in [7] and a generalized regression neural network model for the prediction of path loss at 900MHz for Jos city in Nigeria was presented in [8]. A novel ANN model for path loss prediction in different environment of propagation medium was presented in [4] while [9] used artificial neural network for macro cell prediction.

In this work, different learning algorithms were used in ANN model to predict signal power loss using functional points. The synthesized ANN was built by means of MLP network. A well designed MLP network, trained with measured data from a built-up micro cellular LTE communication network environment was used to find the prediction pattern of the different learning ANN algorithms. The number of the nodes at the input layer node depends on the type of data given to the MLP network. Choosing of the hidden layers and the nodes are important as there is need for accurate prediction and at the same time MLP network has to converge without being over trained. For this study, these requirements were harmonized using one hidden layer with 20 nodes. The output is a single node layer that carries out the error between the measured data and the predicted values. In the training of the MLP network sets using the different learning algorithms, input-output pattern was created, first order statistical error indicators were used to measure the difference between the measured data and the predicted values. The generated results show the effectiveness of each of the learning algorithm in signal power loss prediction. Bayesian regularization learning algorithm shows the least prediction errors.

2. MATERIALS AND METHODS

2.1 Study Area and method of data collection

The research work was carried out in a micro cellular environment: Waterline, Port Harcourt, Nigeria. Waterline is a sub-urban area in Port Harcourt. Port Harcourt is a major city in River State with a population of over 2,000,000 according to 2006 national census, making it one of the biggest cities in Nigeria [10]. It is a built-up terrain with clusters of residential, heavy industrial and moderate commercial buildings. It has a tropical wet climate characterize by heavy and lengthy rainy seasons and short dry season and a land mass of 360 Km^2 and water 9 Km^2. The city is located to the north of Owerri, Imo state, south of Atlantic Ocean, east of Uyo, Akwa Ibom state and west of Yenegoa, Bayelsa State. The geographical coordinate lies within latitude of $4^0$49'27''north and longitude of $7^0$2'1'' east. Drive test was employed to capture the electromagnetic signal power at various points from the chosen base station transmitting at frequency of 1900MHz. Electromagnetic signal power computed at different distances were used as an input in training a MLP neural network using the five (5) different training algorithms to ascertain their performance.

3. MULTI-LAYER PERCEPTRON (MLP) NETWORK

Multi-layer perceptron network is a feed forward artificial neural network created by Rosenblatt in 1958 [11]. It is built using the perceptrons as the building blocks for a bigger and more practical network structure to cater for the limitations of the mapping ability of the single perceptron network. A standard MPL network structure is made up of source nodes which form the input layer, one or more hidden layers which are the computational nodes and an output layer. The inputs signal propagates layer by layer through the network. Each of the nodes in the network contains a differentiable non-linear activation function. A perceptron is an algorithm used for supervised learning of functions that is capable of determining if inputs which are represented by vector numbers fit in some particular class or not. Apart from the input nodes, every other node is a neuron or processing element that has a non-linear activation function [12, 13]. The node activation function simply defines the output of the node when given an input or sets of inputs.

Multi-layer perceptron networks uses different learning techniques, the most common being supervised learning back propagation technique. Using supervised learning back propagation technique, the output values are compared with the expected results by calculating the value of some error function that is predefined. The error is fed back through the network while the weight of each connection is adjusted by the algorithm so as to reduce error function value by certain amount. After sufficient training cycles, the network converges at a certain state with small calculation error. By this, the network has learned a clear target function. Therefore learning occurs by varying of connection weights after processing of each data based on the output error in comparison to the expected result. This is referred to as back propagation of error. Generally, a MLP network is made up of two physical components: processing units known as neurons and directed weighted connections between those neurons.

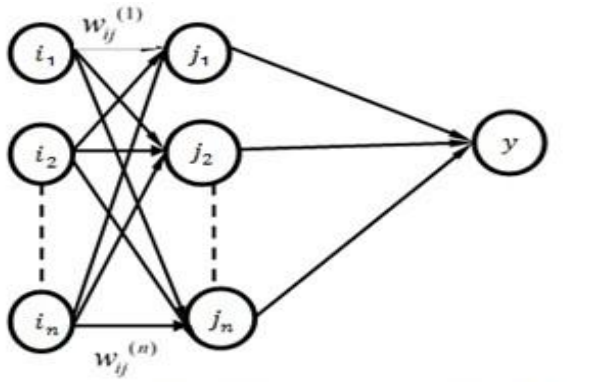

Figure 1. A simple architecture of a Multi-layer perceptron network with one hidden layer.

If i and j denotes two neurons, the strength of the connection known as the connecting weight between the two neurons is express w^{ij} [14]. Data are transmitted between neurons through connections with the connecting weight [15]. Considering i neurons connected to j neurons in figure 1 above, the propagation function of the j neurons receives an output $0_{i1}, \ldots \ldots \ldots .. 0_{in}$ of neurons $i_1, i_2, \ldots \ldots .. i_n$ that are connected to j. These are transformed based on the connecting weight w_{ij} into the network input net_i which can then be processed by the activation function.

If $I = \{ i_1, i_2, \ldots \ldots .. i_n \}$ elements, there exists then the j network input known as $net_i = f_{prop}(0_{i1}, \ldots \ldots .. 0_{in}, w_{ij1}, \ldots \ldots \ldots \ldots w_{ijn})$

Multiplying the output of each neuron i by w_{ij}, the summation of the results give:

$$net_j \sum_{i \in 1} (0_i . w_{ij}) \quad \ldots\ldots\ldots\ldots\ldots (1)$$

The first layer comprises of a linear combination of i of the n - dimensional inputs:

$$a_j = \sum_{i=0}^{n} w_{ij}^{(1)} 0_i \quad \ldots\ldots\ldots\ldots\ldots (2)$$

i_0 = 1 with the leading out weights from the corresponding biases. a_j a_j is the activation of the $i$input w_{ij} and is the weight.

j_n is interpreted as the output of the of the hidden layer.

The outputs of the hidden layer are linearly combined in the second layer to give the activation of the y output layer:

$$a_y = \sum_{j=0}^{n} w_{jy}^{(2)} j \quad \ldots\ldots\ldots\ldots\ldots (3)$$

J_0= 1and corresponds to the bias.

The transformation in the second layer of the MLP is parameterized by $w_{jy}^{(2)}$. Using a sigmoid activation function, the output unit is transformed as:

$$y_n = f(s) = \frac{1}{1+exp^{(-ay)}} \quad \ldots\ldots\ldots\ldots\ldots (4)$$

Equation 2 and 3 are combined to give the overall equation which describes the forward propagation through the MLP network. It describes the computation of the output vector from the input vector given the weight matrices:

$$y_n = g\ (\sum_{j=0}^{n} w_{jy}^{(2)}\ j\ (\sum_{i=0}^{n} w_{ij}^{(1)} 0_i)) \quad ....(5)$$

Like every neural network, training using MLP network comprise of data training using learning algorithms which gives an output [16].

4. LEARNING ALGORITHMS

The key attribute of artificial neural network is their ability to get accustomed to problems by means of training and thereafter be able to solve an unknown problem of similar class. Learning rules are known as algorithms. They are used in finding appropriate weights or other required network parameters. A neural network learns from different phenomenon and a learning system changes so as to adapt to the situation e.g. environmental changes [17]. This paper used five different learning algorithms summarized in table 1.

The algorithms investigated comprise of both first order and second order approximation algorithms. The basic objective of training pattern is for global error reduction by weight and bias adjustments. The learning algorithms used are:

Resilient back propagation algorithm (trainrp) - This is a first order propagation algorithm that eliminates the negative effects of the magnitude of the partial derivative [18]. The learning rate is given as:

$$\Delta_{ij}^{t} = \begin{cases} \eta^{+} * \Delta_{ij}^{t-1}, & if\ \frac{\delta E^t}{\delta w_{ij}} * \frac{\delta E^t}{\delta w_{ij}} > 0 \\ \eta^{-} * \Delta_{ij}^{t-1}, & if\ \frac{\delta E^t}{\delta w_{ij}} * \frac{\delta E^t}{\delta w_{ij}} < 0 \\ \Delta_{ij}^{t-1}, & \end{cases} \quad ....(6)$$

Where $0< \eta^- <1< \eta^+$. Δ_{ij} = the learning rate, η = learning rate factor, E is the partial derivative of the error function, w_{ij} = the weight.

Quasi-Newton algorithm (trainbfg) - Quasi-Newton method approximates the inverse Hessian by a different matrix G, applying the first partial derivatives of loss function [19, 20]. The Quasi-Newton is defined as:

$$W_{i+1} = W_i - (G_i . g_i).\eta_i \quad i = 0, 1, \ldots \quad .....(7)$$

η = the training rate G = inverse Hessian approximation

4.1. Levenberg- Marquardt algorithm (trainlm)

Lavenberg-Marquardt method was designed to enhance the second order training speed without the need of calculating or approximating the Hessian matrix as in Newton algorithm or quasi-Newton algorithm [21]. The parameter improvement using Levenberg-Marquardt algorithm is updated as:

$$w_{i+1} + w_i - (J_i^T \cdot J_i + \lambda_i I)^{-1} \cdot (2 J_i^T \cdot e_i), i = 0, 1, \ldots \quad (8)$$

λ = damping factor, I = the identity matrix, e = the vector of error terms, J = Jacobian matrix

Scaled conjugate gradient algorithm (trainscg)

Scaled conjugate gradient algorithm eliminates the line search at every learning iteration using step size scaling mechanism [22, 23]. The direction of new search is described as:

$$E_K'' = \frac{E'(w_k + \sigma_k \rho_k) - E'(w_k)}{\sigma_K} + \lambda_k \rho_k \quad \ldots\ldots\ldots\ldots\ldots (9)$$

$E(w)$ Error function, E^{11} = Hexian matrix, E' = the gradient of E, λ= scaling factor. σ Is introduced in order to approximate the Hexian matrix.

Bayesian regularization algorithm (trainbr)

Bayesian regularization minimizes squared errors and weight combination and determines the accurate combination in order to ensure a generalize network [24]. The modified performance function is defined as:

$$F_{mp} = \beta SSE + \alpha SSW \quad \ldots\ldots\ldots\ldots\ldots\ldots (10)$$

$$SSE = \sum_{q=1}^{N} e_q^2\ (x) \quad \text{and} \quad SSW = \sum_{j=1}^{n} w_j^2$$

η = the total number of weights and biases $w_{j=}$ in the network. ∝ = the decay rate, β = training rate.

5. TRAINING AND PREDICTION

The most essential step during the training process involves a proper characterization of the points of measurements in the training routes in accordance to their dominant path type. The choice of training points was a planned process in order to get adequate and balanced supply of measured points numbers that belongs to different propagation conditions. A total of 151 measurement points were recorded, each has a different received signal power. The MLP neural network was then trained with the measured data using the different algorithms. 100% of the measured data were used as training data.

Training of the MLP neural network was carried out using neural network training tool box *(nntraintool)* in MATLAB 2013a.The training process was made more efficient and the speed increased by normalizing the input and desired output values to lie in the neighborhood of zero mean and unity standard deviation [6]. For instance, the measured and the predicted data were normalized using excel spread sheet with the expression:

$$V_n = \frac{(V_i - V_{min})}{V_{max} - V_{min})} \quad \ldots\ldots\ldots\ldots\ldots (11)$$

Where V_n= normalize value, V_i= initial parameter value, V_{min}= minimum parameter value, V_{max}= maximum parameter value.

Prediction results of the different algorithms were analyzedusing first order statistical performance indicators defined by equations below.

$$\text{RMSE} = \sqrt{\frac{1}{N_{exp}} \sum_{p=1}^{N} [l(p) - y_o(p)]^2} \quad \ldots\ldots\ldots\ldots\ldots (12)$$

$$MAE = \frac{1}{N_{exp}} \sum_{p=1}^{N} l(p) - y_0(p) \quad \ldots\ldots\ldots\ldots\ldots (13)$$

$$\text{SD} = \sqrt{\left(\frac{1}{N_{exp}} \sum_{i=1}^{N_{exp}} |l_i - y_i| - MAE\right)^2} \quad \ldots\ldots\ldots\ldots\ldots (14)$$

Where *Nexp*= number of the experimental pattern, *l(p)* is the already estimated signal power loss loss value (using traditional signal power loss methods) of the input pattern p^{th}, *yo(p)*= the neural network output when p^{th} input pattern is given to its input.

Correlation coeffient (*r*): This ranges from -1.0 to +1.0, the prediction value and the actual value are closely related with correlation coefficient closer to +1 or -1 and 0 indicates no relationship between the variables.

6. RESULTS AND DISCUSSION

It has been a demanding task to correctly specify the most suitable algorithm for the training of neural network for signal power loss prediction. This is as a result of many factors such as the complexity of the area under consideration, the number of the training data, the quality of the training data, weights and bias in the network etc. This work measured the effectivenes of five different neural network learning algorithms using measured data from Waterline in PortHarcourt, Nigeria. Inputs and objectives were normalized before the neural network training started so they are scaled in certain range. The five algorithms used were assesed using first order performance error measurements stated above. The results are shown in table 1.

Table I. Performance results of the five trained algorithms using error measurement

Error Measurement	Training algorithms				
	trainbfg	**trainrp**	**trainlm**	**trainscg**	**trainbr**
RMSE Run 1 Run 2 Run3	3.0885 3.0537 3.0847	3.0929 3.2578 3.3375	2.7424 2.7542 2.7630	3.2706 3.1921 3.1664	2.5004 2.6319 2.5669
Average	3.07556	3.2294	2.7532	3.2097	2.5664
MAE Run 1 Run 2 Run 3	2.4492 2.3539 2.4651	2.5040 2.6015 2.6296	2.0892 2.0582 2.1389	2.5978 2.5543 2.5159	1.8940 1.9852 1.9288
Average	2.4227	2.5784	2.0954	2.5560	1.9360
SD Run 1 Run2 Run 3	1.8814 1.9453 1.8545	1.8156 1.9610 2.0552	1.7765 1.8300 1.7490	1.9871 1.9144 1.9226	1.6324 1.7280 1.6398
Average	1.8937	1.9439	1.7851	1.9414	1.6667
r Run 1 Run 2 Run 3	0.9119 0.9138 0.9117	0.9110 0.9017 0.8976	0.9307 0.9303 0.9301	0.9005 0.9056 0.9066	0.9429 0.9369 0.9398
Average	0.9126	0.9034	0.9303	0.9042	0.9398

The overall result demonstrates that training with Bayesian Regularization algorithm, *(trainbr)* gave the best result with the least RMSE of 2.5664, the best MAE of 1.9360, the least SD of 1.6667, and the highest *r* of 0.9398. This was followed by the Levenberg-Marquardt algorithm (*trainlm)* which gave the RMSE of 2.7532, MAE of 2.0954, SD of 1.7851 and r of 0.9303. Prediction patterns of the algorithms with the measured data are shown in figures 1 and 2. The best two algorithms that gave the least errors were shown.

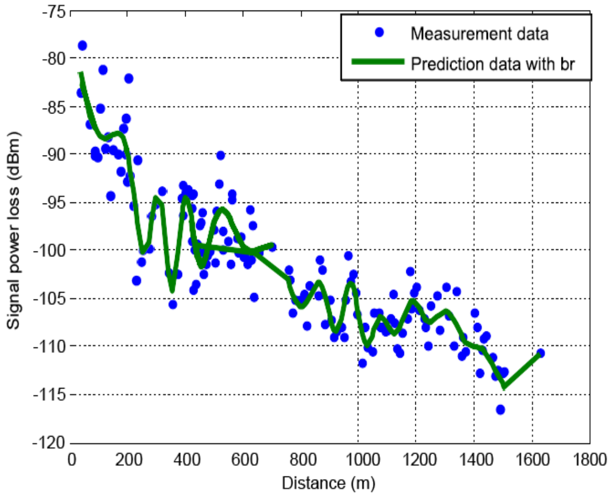

Figure 2. Signal Power loss prediction with Bayesian Regularization algorithm

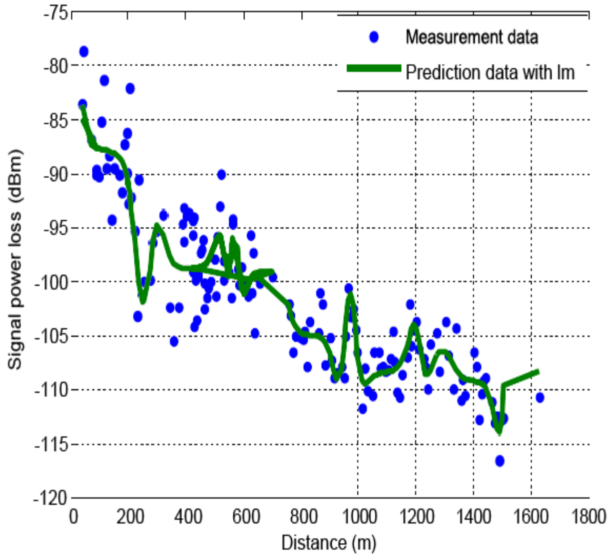

Figure 3. Signal Power loss prediction with Levenberg-Marquardt algorithm

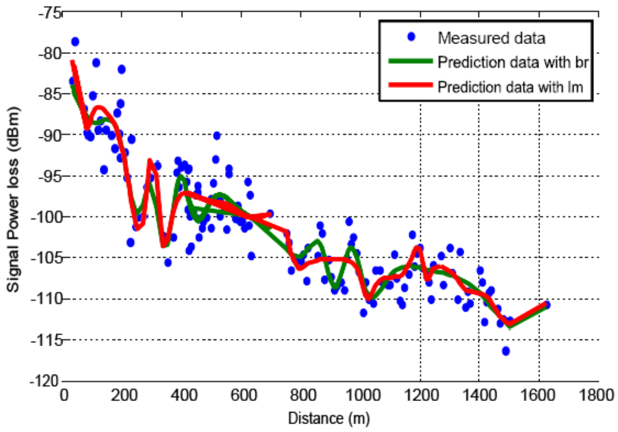

Figure 4. Combined predictions with Bayesian Regularization and Levenberg- Marquardt algorithms.

Bayesian Regularization algorithm *(trainbr)* gave the least error results, but with a price. It took longer training time than other algorithms. The training time was 22seconds for all the 1000 iterations. However, the accuracy is worth the time, as Levenberg-Marquardt algorithm *(trainlm)* though showed a faster training time of 4seconds, but for only 42 iterations. The same pattern was seen with the other algorithms (*trainrp, trainbfg and trainscg)* where most of the iterations were not fully trained. They also gave higher measurement errors. A combined plot of Bayesian Regularization and Levenberg-Marquardt algorithms showing their learning ability with the measured data is shown in figure 4. Bayesian Regularization was seen as the best training algorithm with the best training performance as 6.2573 at epoch 1000 for 1000 iterations. However, Levenberg-Marquardt algorithm demonstrated the best validation algorithm with the best validation performance as 13.5645 at epoch 36 for 42 iterations.

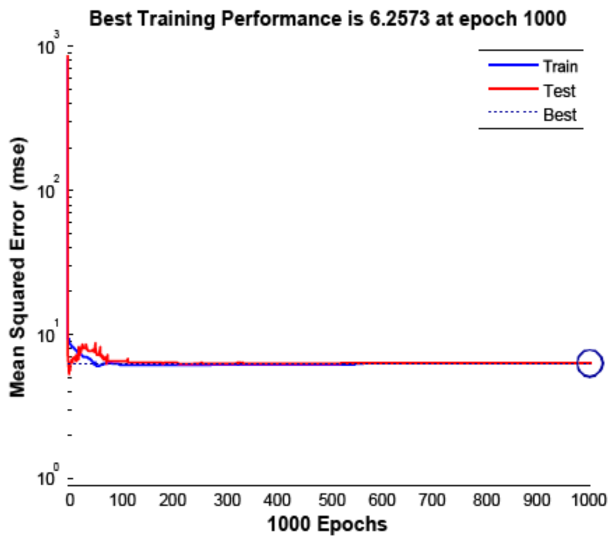

Figure 5. Performance training using Bayesian Regularization algorithm

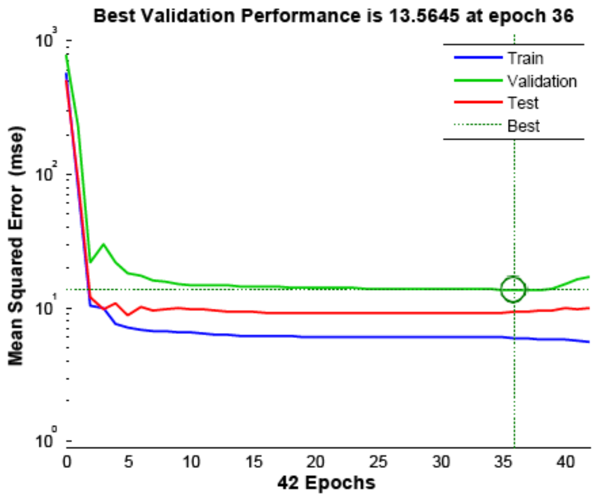

Figure 6. Performance training using Levenberg-Marquardt algorithm

7. CONCLUSION

This paper briefly looked at five different Multi-Layer Perceptron learning algorithms. From the conducted training using measured data from LTE micro cellular environment, Bayesian Regularization shows better prediction accuracy over other learning algorithms. It gave the least measurement errors but required longer training time. Levenberg-Marquardt algorithm demonstrated the best validation algorithm with the least training time. Nevertheless, the training time aspect of the different algorithms used is not the subject of this paper; hence it was not discussed in this paper.

REFERENCES

[1] P. S. Sotirios and K. Siakavara, "Mobile radio propagation path loss prediction using Artificial Neural Networks with optimal input information for urban environments," International Journal of Electronics and Communications (AEÜ), pp. 1453-1463, 2015.

[2] N. Aleksandar and N. Natasa, "Micro Cell Electric Field Strength Prediction Model Based Upon Artificial Neural Networks " Internation Journal of Electronics and Communications (AEU), vol. 64, pp. 733–738, 2010.

[3] M. Piacentini and F. Rinaldi, "Path Loss Prediction in Urban Environment Using Learning Machines and Dimensionality Reduction Techniques," Springer-Verlag 2010.

[4] P. Sridhar, "Novel Artificial Neural Network Path Loss Propagation Models for Wireless Communications," Advances in Wireless and Mobile Communications, vol. 10, pp. 233-237, 2017.

[5] O. Ali Riza, A. Mustafa, K. Mehmet, G. Mehmet, and H. S. Murat, "The Prediction of Propagation Loss of FM Radio Station Using Artificial Neural Network," Journal of Electromagnetic Analysis and Applications, vol. 6, pp. 358-365, 2014.

[6] J. Isabona and V. M. Srivastava, "A Neural Network based Model for Signal Coverage Propagation Loss Prediction in Urban Radio Communication Environment," International Journal of Applied Engineering Research, vol. 11, pp. 11002-11008, 2016.

[7] F. a. A. Ignacio, A. R. A. Juan, and P. e. F. a. Fernado, "Influence of Training Set Selection in Artificial Neural Network-Based Propagation Path Loss Predictions," International Journal of Antennas and Propagation, pp. 351-487, 2012.

[8] C. A. Deme, "A Generalized Regression Neural Network Model for Path Loss Prediction at 900 MHz for Jos City, Nigeria" American Journal of Engineering Research (AJER), vol. 5, pp. 01-07, 2016.

[9] E. Östlin, H.-J. e. Zepernick, and H. S. Uzuki, "Macrocell Path-Loss Prediction Using Artificial Neural Networks," IEEE Transactions on Vehicular Technology, vol. 59, pp. 2735-2746, 2010.

[10] O. N. Njoku, Eastern Nigeria under British Rule. Department of History, University of Nigeria, 2008.

[11] H. Simon, Neural Networks - A Comprehensive Foundation, 2nd ed. Englewood Cliffs Prentice- Hall, 1998.

[12] R. Collobert and S. Bengio, "Links between Perceptrons, MLPs and SVMs," Proc. Int'l Conf. on Machine Learning (ICML), 2004.

[13] [13] P. D. Wasserman and T. Schwartz, "Neural networks. II. What are they and why is everybody so interested in them now?," IEEE Expert, vol. 3, pp. 10-15, 1988.

[14] M. Hassoun, Fundamentals of Artificial Neural Networks. Massachusetts: MITPress, 1995.

[15] K. David. (2005, 05/04). A Brief Introduction to Neural Networks. Available: http://www.dkriesel.com/en/science/neural_networks

[16] T. Szandała, "Comparison of Different Learning Algorithms for Pattern Recognition with Hopfield's Neural Network," Elsevier 2015 Annual International Conference on Biologically Inspired Cognitive Architectures, vol. 71, pp. 68-75, 2015.

[17] H. White, "Learning in neural networks: a statistical perspective," Neural Computation, vol. 1(4), pp. 425-464, 1989.

[18] C. Igel and M. Hüsken, "Empirical Evaluation of the Improved Rprop Learning Algorithm,"

Neurocomputing, vol. 50, pp. 105-123, 2003.

[19] M. Remzi and B. Djavan, "Artificial neural networks for decision making in urologic oncology," Ann Urol (Paris), pp. 110-115, 2013.

[20] R. Haelterman, "Analytical study of the least squares quasi-Newton method for interaction problems," PhD, Ghent University, 2009.

[21] C. Kanzow, N. Yamashita, and M. Fukushima, "Levenberg-Marquardt methods with strong local convergence properties for solving nonlinear equations with convex constraints," JCAM, vol. 172, pp. 375-97, 2004.

[22] Moller, "A Scaled Conjugate Gradient Algorithm for Fast Supervised Learning Neural Networks," vol. 6, pp. 525-533, 1993.

[23] F. LiMin, "Neural Networks in Computer Intelligence," Ed-McGraw-Hill, 1994.

[24] D. J. MacKay, "Bayesian Interpolation," Neural Computation, vol. 4, pp. 415-447, 1992.