Phone Numbers Classification (PNC) with Feed-forward Neural Networks

Safdar Hayat

Department of Computing and Technology, Iqra University Islamabad Campus Plot No. 5, Sector H-9, Islamabad-44000, Pakistan.

(Tech. Support Engineer at Prime Minister Secretariat AJK)
Email: Wellcom.to.safdar@gmail.com.

***Abstract*____ A neural network (NN)-based method for phone number classification or recognition is provided in this paper. The used network is a one-hidden-layer multilayer perceptron (MLP) classifier. Its training is based on backpropagation learning. I present the results of a Feed Forward Neural Network trained to classify phone numbers into four categories: Different training data were pre-processed and then tested to distinguish between four classes/patterns of phone numbers in order to train the FFNN. My goal is to provide a coalescence of the published research in this field and to arouse further research interest in and efforts to research the identified topics.**

***Keywords*— classification, multilayer perceptron, tansig, feed- forward backpropagation, phone numbers classification (PNC).**

I. Introduction

One of the most involved fields of neural network research and implementation is classification. There is a wide and growing volume of literature on the subject. Recent advances in neural networks (NNs) have provided potential alternatives to conventional classification and pattern recognition techniques [5],[6]. These nets, which are inspired from research into biological nervous systems, are made up of a large number of basic nonlinear computational elements (neurons or nodes) linked by links with varying weights. These networks' innate parallelism enables them to pursue several hypotheses at once, leading to high computation speeds. Furthermore, because of the multiple processing nodes, each of which is responsible for a small portion of the job, they have a higher level of robustness or fault tolerance than traditional computers. As a result, damage to a few nodes or ties has little impact on overall results.

This paper provides a systematic and competitive analysis of supervised learning neural network classification. Network is trained by giving a set of examples and then test the trained network to check its performance whether it gives optimal result or not. This report is all about that how data is collected and presented for training and testing, at the end the network is checked by giving some unknown data as testing. The multilayer perceptron, a sort of artificial neural network with one hidden layer and 25 hidden neurons trained using a backpropagation algorithm, has been successfully used for number classification. The statistical method has been the most extensively studied and used in practice among the different frameworks in which pattern recognition or classification has typically been formulated. Neural network strategies and methods developed from statistical learning theory have recently gotten a lot of coverage. In a wide range of fields, pattern classification has been used to solve problems. [8]

Pattern classification is basically a science in which we developed machines that can categorize data (patterns) grounded on pre-existing information or statistical data obtained from the patterns. [7]

The classification stage is the main focus of this study. I'll focus at the efficiency of a multilayer perceptron (MLP), classifier of neural network with one hidden layer, as well as the effects on performance by increasing or decreasing the hidden layer neurons.

The following is a breakdown of the paper's structure. Introduce data sets that were used in neural networks in the following section (NN). In Section III, I discuss the architecture of the network which shows the decision boundaries. In Section IV, a training algorithm is developed for classification of phone numbers by feedforward neural network, based the error backpropagation algorithm and also cited the function and parameters which were used to train, learn and test the net. The training and testing of the network are included in section V and VI. Comparison of Simulation results are presented in Section VII. In this section the performance is compared on the basis of increasing or decreasing the hidden layer neurons and number of epochs and the simulation results are presented. Finally, Section IX presents the conclusion

II. Specification Of Data Sets

In this section, the pre-processing, classification of numbers according to their pattern and data sets are specified.

A. Pre-processing

Pre-processing involves data collection, data normalization and processed the data to achieve well generalization goal. For this purpose firstly removed the first digit i.e. 0 form every number which is not required because every number starts with 0 and secondly also removed the last digit which is insignificant, and hence create an array of nine digits along every phone number.

B. Classes

For classification divide the data into four classes;

- Class1=International numbers
- Class2=Cellular numbers
- Class3=Nationwide numbers
- Class4=Local numbers

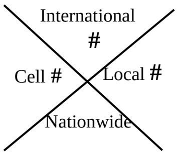


Fig 1.Illustrate 4 classes

C. Data Sets

The datasets for this study is described as follow.

1) Training Set

Total 387 set of different examples were used to input and train the network. Two examples of international numbers, 203 examples of cell numbers, 23 examples of nationwide numbers and 156 examples of local numbers.

2) Target Set

Total four set of different targets were used to learn the network. Every target specifies a unique class, so mentioned a specified target along every example belong to a specific class.

3) Testing Set

Total 100 set of different examples were used to test the network. Three examples from different international numbers, 50 examples from different cell numbers, 8 examples from different nationwide numbers and 37 examples from different local numbers are chosen.

III. Network Architecture

As network architecture, I used a general feed forward neural network (FNN) with one hidden layer. An oriented acyclic graph is generated by the neurons in such a network. Since there are no global recursive links, the overall network structure is static (memory less). As a consequence, neurons can be organized such that each neuron's inputs come exclusively from previous neurons' outputs.

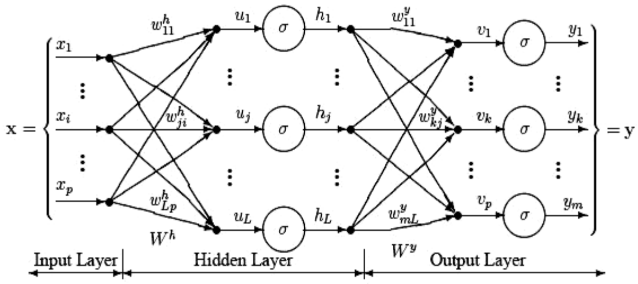


Fig 2. Illustrate the network architecture with one hidden layer

This is a generalization of a multilayer feed forward neural network (MFNN), in which neurons are organized in layers and only neurons in the same layer are linked.

To calculate a neuron's activation value ("net" valuation), add the output values yji(t) of the incoming synapses firstly.

$$a_j(t) = \sum_{i \epsilon IN_i} y_{ji}(t).$$

INi is a group of neurons that are linked to neuron j's input. Second, on a_j (t), a nonlinear function, fj(.), is applied, yielding the following output value:

$$Z_j(t) = f_j(a_j(t)).$$

Unit bias is caused by a "bias synapse," which gives a constant value to a neuron's summation node.

$$y_{ji}(t) = b_{ji}$$

IV. Functions And Parameters

This section describes the algorithm on which the network is worked, and the functions and parameters used in networks.

A. Algorithm

Step First: The 'dotprod' weight function, 'netsum' net input function, and 'tansig' transfer functions are used to build one hidden layer with 25 hidden neurons and four output neurons in feed-forward networks.

Step Two: Weights from the input appear in the first layer. The previous layer's weight is carried over into subsequent layers. Biases exist in each layer. The network output is the last layer.

The weights and biases of every layer are established with ‘initnw,' while the network is fixed with default weights and biases in this case.

Step Three: Adaptation is performed using the ‘adaptwb' , which updates weights using the ‘learngdm' learning function; in this network adoption procedure is also default as ANN Matlab Toolbox.

Step Four: Train the network with the ‘traingdx’training function.

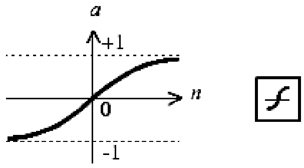
Step Five: Test the trained network by giving the examples from the test data set.

Step six: The 'mse' performance function is used to evaluate performance.

B. Transfer Function

The default transfer function for backpropagation neural networks is the sigmoid function 'tansig.' It is the most commonly used function. The mathematical relationship represents the sigmoid function.

$$\varphi(v) = \frac{1}{1+\exp(-av)}$$



$a = tansig(n)$

Fig 3:Tan-Sigmoid Transfer Function

The sigmoid function is a two-state output gate that can be opened (1) or closed (zero). The gate should be partly opened since the function is continuous (i.e. somewhere between 0 and 1). Models with sigmoid transfer functions are often more precise and help with generalized learning traits. The use of sigmoid transfer functions likewise result in prolonged training times.

C. Function for Training

The net is trained by the 'trainlm' Levenberg Marquardt training function. The 'trainlm' function is a net training function that utilizes Levenberg Marquardt optimization to update bias and weight values. 'trainlm' is the toolbox's default backpropagation algorithm, and it's every time the fastest. It's extremely suggested as a first-choice supervised algorithm, while it does need extra memory than other algorithms.

D. Learning Function

For the sake of learning of the network, the ‘learngdm' Gradient descent along with momentum weight & bias learning function is utilized. The learning process is directed by learngdm's learning parameters, which are shown here with their default values.

LP.lr – 0.01	Learning-rate
LP.mc – 0.9	Momentum-constant

“Learngdm” uses gradient descent along-with momentum to figure out the weight change dW for a given neuron based on the input P and error E, the weight (or bias) W, learning rate LR, and momentum constant MC:

```
[dW = mc*dWprev + (1-mc)*lr*gW]
```

The learning state LS stores and reads the earlier weight change dWprev.

E. Performance function

The ‘mse' is used as a network performance function. A network efficiency function is called mse. The mean of squared errors is used to assess the network's performance. The MSE is equal to the summation of the variance and the estimator's squared bias.

$$\mathrm{MSE}(\hat{\theta}) = \mathrm{Var}(\hat{\theta}) + (\mathrm{Bias}(\hat{\theta},\theta))^2$$

Where θ is estimator

James Berger, a decision theorist, has criticized the practice of mean squared error without question. The predictable value of one specific utility function, the quadratic utility function, is the adverse of the mean squared error, which may or may not be the suitable utility function to utilize in a given set of situations. In some cases, though the mean squared error can be a worthy approximation to a loss function that falls naturally in an application.[1]

V. Training Network

A. Input Data Set

For training the network 387 set of examples of phone numbers from different classes were used. Two examples of International numbers, 203 examples of cell numbers, 23 examples of nationwide numbers and 156 examples of local numbers.

B. Target Data Set

Total four set of different targets were used to learn the network. Every target specifies a unique class, so mentioned a specified target along every example belong to a specific class.

C. Stopping criteria

When any of the following conditions arise, training must come to a halt:

- The number of epochs (repetitions) has been attained.
- The time limit has been reached.
- Performance is kept to a bare minimum in order to achieve the goal.
- The performance gradient is lower than the minimum value set by min grad.
- (momentum unit)mu exceeds mu maxmu.

- Since the last time it was reduced, validation performance has improved more than max fail times. (when using validation).

But here the validation sets were not used so the network has been achieved the targets with 25 hidden layer neurons at 2500 epochs hence it was the stopping criteria.

VI. TESTING NETWORK

To test the network Total 100 set of different examples were used. Three examples from different international numbers, 50 examples from different cell numbers, 8 examples from different nationwide numbers and 37 examples from different local numbers are chosen. The test result shows the 100 % accuracy of the network by obtaining the targets given to the network.

VII. COMPARISON OF SIMULATION RESULTS

After testing the network, simulation results were calculated according to the following states.

A. Regression

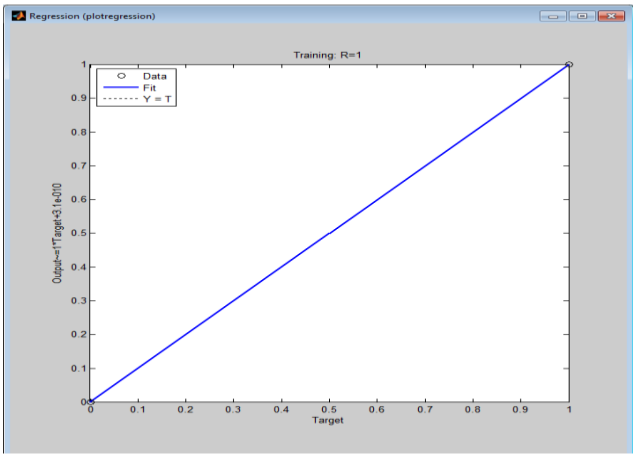


Fig 9. Plot Results same regression with 2000/2500 Epochs and 22/25 hiddenlayer neurons

B. Performance

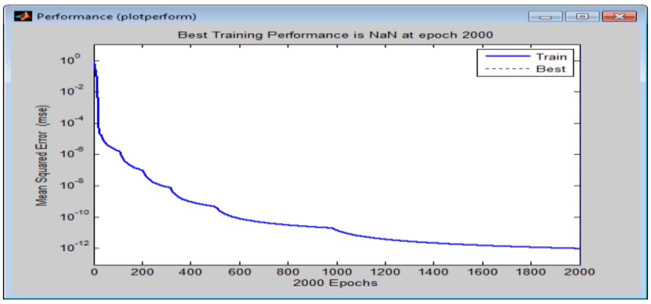


Fig 5. Plot Results of performance with 2000 Epochs and 22 hidden layerneurons

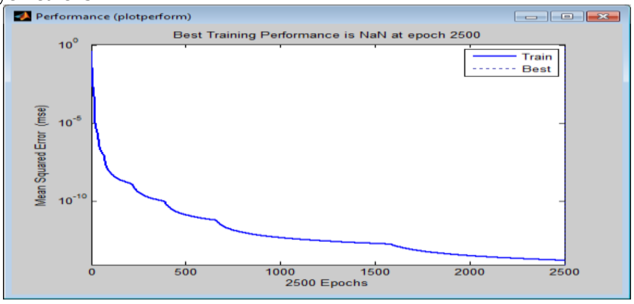


Fig 6. Plot Results of performance with 2500 Epochs and 25 hidden layerneurons

C. Training State

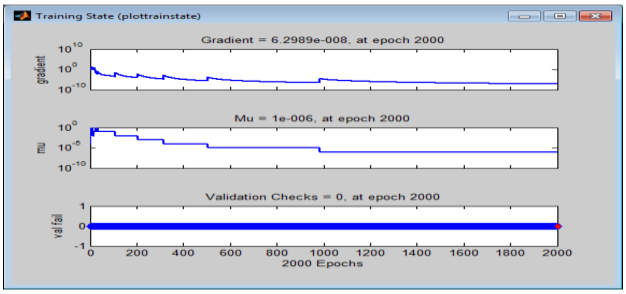


Fig 7. Plot Results of training state with 2000 Epochs and 22 hidden layerneurons

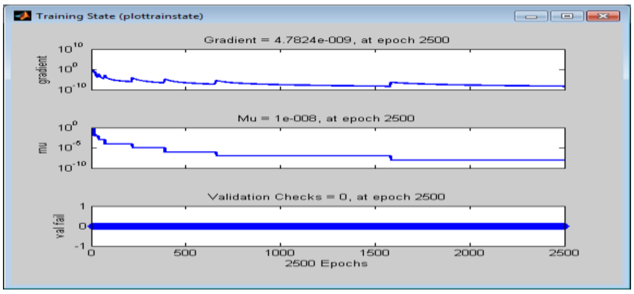


Fig 8. Plot Results of training state with 2500 Epochs and 25 hidden layer neurons

VIII. THE BACK PROPAGATION PROCEDURE

The Widrow-Hoff learning law was generalized to multiple-layer networks and non-linear differentiable transfer functions to create back propagation (BP). Input vectors and target vectors are utilized to train a network till it can estimate a function, adjust the input vectors with specific output vectors, or categorize input vectors in the way that the analyst specifies [2]. Biases, a sigmoid function, and a linear output layer that can estimate any function with a limited number of breaks, see [3] ,[4].

So errors are calculated by the propagation procedure in feed-forward neural networks.

IX. CONCLUSIONS

In this work, I developed a Phone number classification technique and compared the performance by different parameters.

A classifier of Feed Forward Neural Network having one hidden layer was utilized in this study.

The experimental results illustrate that the proposed method is classifying the phone number data sets efficiently and effectively among the proposed classes of phone numbers.

The obtained results are 100% using the normal parameters of the FFNN network.

From all kinds of results and tests the system shows

- Higher classification rate
- Smaller training time

ACKNOWLEDGMENT

First and foremost, I'd like to express my gratitude to ALLAH Almighty, who is the Most Merciful, Compassionate, and Kind. I was able to complete this task thanks to His generosity and blessings. Many devoted people contributed to the success of this study. I would like to thank those who shared their ideas & materials. This research would not have been possible without the unrelenting support and encouragement of my supervisor, special thanks to my Mother Khadija Anjum, father Khizar Hayat and my wife Saima Waheed for their commendable support.

REFERENCES

[1]. Berger, James O. (1985). "2.4.2 Certain Standard Loss Functions". Statistical decision theory and Bayesian Analysis (2nd ed.). New York: Springer-Verlag. p. 60.ISBN 0-387-96098-8.MR0804611.

[2]. S. M. Aqil Burney, Tahseen A. Jilani, "Time Series forecasting using artificial neural network methods for Karachi Stock Exchange". A Project at Department of Computer Science, University of Karachi. 2002.

[3]. R. Hecht-Nielsen. "Theory of the backpropagation neural network".International Joint Conference on Neural Networks, IEEE, New York,pp. 593-605, 1989.

[4].N.Schraudolph, Nicol, and T. J. Sejnowski,"Tempering backpropagation networks: not all weights are created equal". Appear in Advances in Neural Information Processing Systems 8, MIT Press, Cambridge, 1996.

[5].Ashish Dehariya, et al. "An Effective Approach for Medical Diagnosis Preceded by Artificial Neural Network Ensemble", 978-1- 4244-8679-3 /11/$26.00 © 2011 IEEE.

[6].Omar Al-Jarrah, et al. "Network Intrusion Detection System Using Neural Network Classification of Attack Behavior", Journal of Advances in Information Technology Vol. 6, No. 1, February 2015.

[7].David J. Miller, et al. "Adversarial Learning Targeting Deep Neural Network Classification: A Comprehensive Review of Defenses Against Attacks", Proceedings of the IEEE (Volume: 108, Issue: 3, March 2020).

[8].Rüdiger W. Brause, "Medical Analysis and Diagnosis by Neural Networks", J.W. Goethe-University, Computer Science Dept., Frankfurt a. M., Germany, May 21, 2001.