Classification with multilayer perceptron by back-propagation learning for double moon

Manjunath Dharshan Shanthigrama Rangaswamy
Ira A Fulton Engineering
Arizona State University
Tempe, USA
mdshanth@asu.edu

Abstract—**This paper presents the use of multilayer perceptron by back-propagation to make classification determinations for double moon problem. Specifically, a Levenberg–Marquardt Back Propagation from MATLAB's Neural Network toolbox was used.**

Keywords—Multilayer perceptron, Levenberg-Marquardt algorithm, classification, back-propagation, MATLAB

I. Introduction

In the field of non – linear classification, a typical multilayer perceptron with BP have been intensively studied over the past several decades. Artificial Neural Networks (ANN) aim to construct a activation function with a goal of fitting or classifying a dataset that minimizes the error between the datapoints and the function. The selection of network size is a critical issue. If there are too few hidden nodes the network may not be able to approximate the given function, and if there are too many, the network may exhibit poor generalization performance because of over fitting [1].

Multilayer perceptron chooses random hidden node parameters and calculate the output weights with the least squares algorithm. This method can achieve a fast training speed, as well as good classifying accuracy. ANN may be used to compute more complex, non-linear hypothesis equation. ANN naturally discovers new features of the system being analyzed as a product of its internal mechanisms. These features cause ANN to be more capable of fitting or classifying complicated data sets and being less limited by poor initial feature choices. While there are many ways to implement Multilayer perceptron, the archetypal implementation is comprised of an input layer, any number of hidden layers and an output layer as seen in Fig. 1.

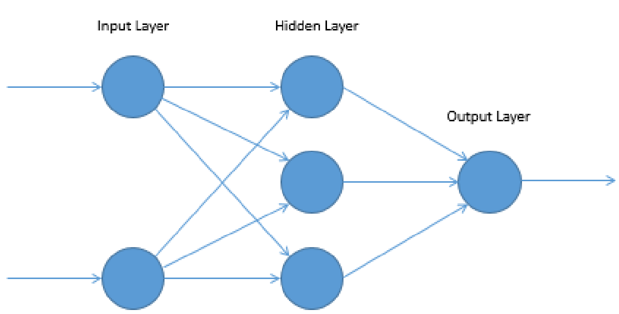

Figure 1: Neural Network Architecture Example [2]

II. Methodology

Levenberg-Marquardt backpropagation is a network training function that updates weight and bias values according to Levenberg-Marquardt optimization. trainlm is often the fastest backpropagation algorithm in the toolbox and is highly recommended as a first-choice supervised algorithm, although it does require more memory than other algorithms [3]. The Levenberg-Marquardt algorithm uses this approximation to the Hessian matrix in the following Newton-like update show in (1)

$$x_{k+1}=x_k-[J^TJ+\mu I]^{-1}J^Te \tag{1}$$

The gradient can be computed as show in (2)

$$g = J^Te \tag{2}$$

trainlm can train any network as long as its weight, net input, and transfer functions have derivative functions. Backpropagation is used to calculate the Jacobian jX of performance perf with respect to the weight and bias variables X [3].

III. Implementation

A. Creating the dataset

Levenberg-Marquardt backpropagation were applied to the created dataset double moon to make classification determinations. Specifically, 1000 data pairs 500 each in Region A and B respectively were used for training purpose and 400 pairs from the training samples were used for verification of the trained neural network. The radius of the inner and the outer ring is 10 and 16 respectively with both rings separated at a distance of -6. The data points were created using MATLAB inbuilt function rand within the specified limits. The created data samples is as seen in Fig. 3.

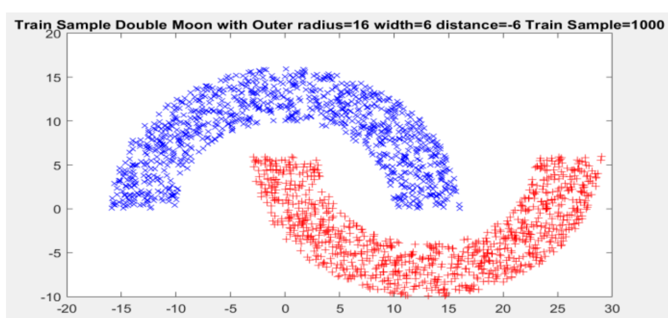

Figure 3: Training set with 1000 pairs of training samples

B. *Training a multilayer perceptron*

The back-propagation training algorithm uses gradient descent to attempt to locate the local (or global) minimum of the error surface. The training involves initializing the weights, presenting the first input vector from training data to network, propagate through hidden layer and activation function to get output. Calculating the error signal by comparing the actual output to desired output, propagate error signal back and adjust the weights. Repeat until error is satisfactorily small [4].

IV. RESULTS

From the observation after 30 epoch the error remains the same. To achieve same result stopping criteria is 40 epoch, learning rate = 0.01, hidden layer size = 3, weights and bias initialization between [-0.1 0.1]

A. *Tables and Figures*

Table 1 shows the root mean squared error and accuracy for various combinations of hidden layers and epoch for learning rate = 0.01, width = 6, distance = -6 and radius = 16 used by the Levenberg-Marquardt backpropagation multilayer perceptron.

TABLE I. RMSE OF TEST DATA

Inputs		RMSE			Accuracy
Hidden Layer	Epoch	Test Error	Train Error	Validation Error	(%)
1	7	0.75	0.08	0.1	89.5
3	40	0.02	0.02	0.015	99
10	100	0.0076	0.0077	0.008	100

The confusion matrix for 3 hidden layers with 40 Epoch at 0.01 learning rate is seen in Fig. 4.1.

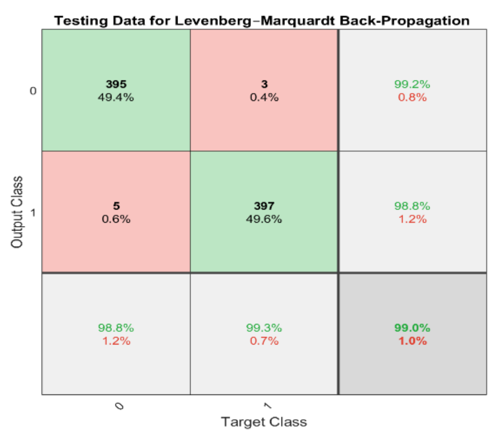

Figure 4.1: Confusion Matrix with 40 Epoch

Figures (4.2) and (4.3) shows the Decision boundary of the test data and the Performance plot for the trained neural network with 3 hidden layer, Epoch 40 and learning rate 0.01.

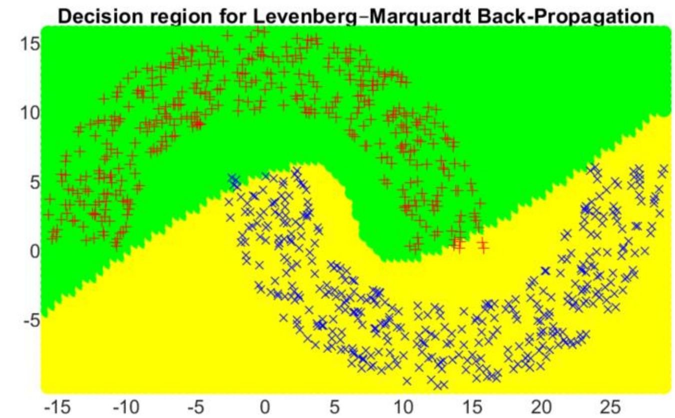

Figure 4.2: Decision Boundary for the Test sample

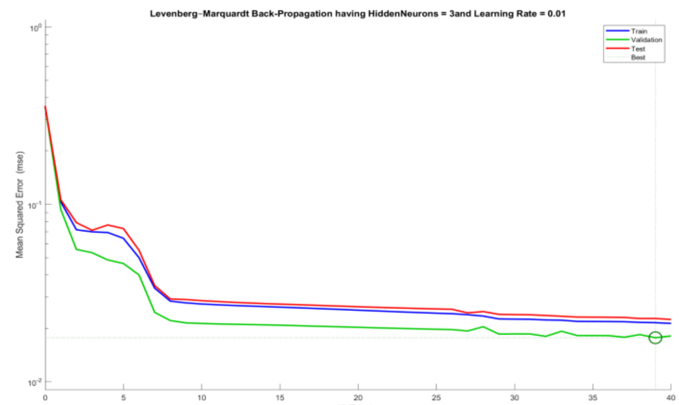

Figure 4.3: Performance plot of the trained model

V. CONCLUSION

The multilayer perceptron has been shown to be a useful tool for classification . Based on the complexity of the problem choosing appropriate learning rate, hidden layer size and number of iterations is very crucial in order to avoid over fitting and underfitting. On the above observation double moon problem is of low complexity. So, fewer hidden layers with lower learning rate will achieve local (global) minimum. The numerous difficulties in implementing, training and interpreting the multilayer perceptron must be balanced against the performance benefits when compared to more traditional, and often inappropriate, techniques[4].

REFERENCES

[1] Hui Wen, Weixin Xie, Jihong Pei, “A Structure-Adaptive Hybrid RBF-BP Classifier with an Optimized Learning Strategy” ATR Key Lab of National Defense, shenzhen University, shenzhen 518060, China

[2] https://blogs.msdn.microsoft.com/smallbasic/2014/11/06/neural-network-extension-small-basic-featured-thread/

[3] “Trainlm”,Levenberg-Marquardt backpropagation MATLAB, [Online] https://www.mathworks.com/help/deeplearning/ref/trainlm.html [Accessed : 30-Jan-2019]

[4] M. W. GARDNER and S. R. DORLING, “ARTIFICIAL NEURAL NETWORKS (THE MULTILAYER PERCEPTRON)”, School of Environmental Sciences, University of East Anglia, Norwich, Norfolk NR4 7TJ, UK, Published June 1998 .

[5] krishnayash-T,https://github.com/krishnayash-T/Non-Linear-Classification-Matlab, [Accessed : 30-Jan-2019]