Time series forecasting with nonlinear autoregressive neural network

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 forecast future data points from a small, oscillatory data set Specifically, a nonlinear, autoregressive neural network is used. Broydon - Fletcher - Goldfarb - Shanno (BFGS) optimization algorithm is used to train the NARNN. The number of hidden layers, size of the hidden layers, random seed, and delay (number of previous samples to consider) are altered to get the optimized model with the least root mean square error. Finally, the chosen model is programmed to predict the data points beyond the given data set.**

Keywords - Multilayer perceptron, artificial neural networks, time series, nonlinear, NARNN, BFGS algorithm, 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. Like most predictive machine learning approaches, artificial neural networks (ANN) aim to construct a function with the goal of fitting a particular set of data that minimizes the amount of error between the data points and the function. Artificial neural networks have two distinct advantages over simpler methods such as linear regression. First, ANN may be used to compute more complex, non-linear hypothesis equations [1] if desired. Second, 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 complicated data sets and being less limited by poor initial feature choices. 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 prediction accuracy. ANN may be used to compute more complex, non-linear hypothesis equation. 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.

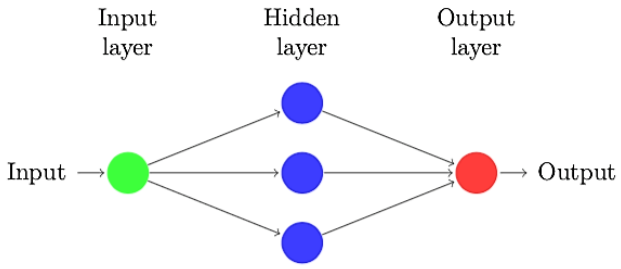

Figure 1: Neural Network Architecture Example [1]

An activation function defined as g(z) and shown in (1). Note, (1) specifically corresponds to a sigmoid (logistic) activation function. Activation functions are not limited to this form.

$$g(z) = \frac{1}{1+e^{-z}} \tag{1}$$

II. METHODOLOGY

A non-linear, autoregressive neural network (NARNN) is used to forecast 20 future values of a 124-point data set. The NARNN is based off of the non-linear, autoregressive exogenous model, which is an approach commonly used for modeling timeseries [3]. In this approach, the model uses not only the values of the training series provided, but also uses values previously predicted by the network. The NARNN was implemented using a built in function in MATLAB called narnet [4]. The default NARNN implementation in MATLAB utilizes ten hidden layers. The NARNN was trained using Bayesian regularization, which utilizes Levenberg-Marquardt optimization to iteratively update the weight and bias values for back propagation [5]. Firstly, the time series data is prepared using preparets and train the network as shown in the Fig. 2.1

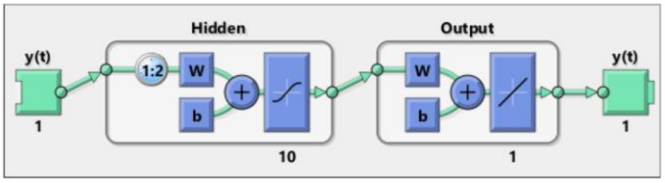

Figure 2.1: Network Model [4]

To predict the output for the next 20-time steps, simulate the network in closed loop form [4] as shown in the Fig. 2.2. The network only has one input. In closed loop mode, this input is joined to the output.

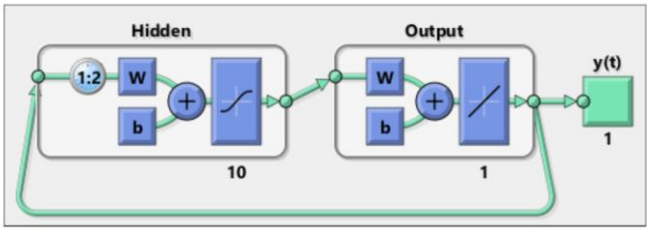

Figure 2.2. Network in closed loop form [4]

III. Implementation

The given 124 time series data features are Normalized. Then these normalized features are prepared using preparets before it is fed into the neural network model. Then the network is trained. Broydon - Fletcher - Goldfarb - Shanno (BFGS) optimization algorithm is used to train the NARNN. Next, simulating the network in closed loop form and next 20-time steps is predicted. One of the main topics from the lectures given by Andrew Ng on artificial neural networks pertained to how many hidden layers are required in the network architecture. In the lectures, Dr. Ng indicates that one hidden layer is typically sufficient, but that, in general, more is better. With this idea in mind, the number of hidden layers and the delay is altered in NARNN to get a better idea of how many hidden layers would lead to an optimal tradeoff between accuracy and required training time. Finally, the combination of hidden layer size and max input delay with least RMSE value is selected to predict the final 20-time steps value. The given training data is plotted to see the time series pattern with time steps in X-axis and Values in Y-axis as shown in the Fig. 3.1.

Figure 3.1: Training set with 124 data time steps

IV. Results

From the observation the maximum gradient is reached after 11 epoch. To achieve same result maximum input delay is 120, learning rate = 0.01, hidden layer size = 22, weights and bias initialization between [-0.7 0.7]. The RMSE result for the test data is found least for the above criteria.

A. *Tables and Figures*

Table 1 shows the root mean squared error for various combinations of hidden layers and max input delay for learning rate = 0.01.

TABLE I. RMSE OF TEST DATA

Model	**RMSE – Test data**
max input delay = 2, hidden layer size = 10	88.2429
max input delay = 100, hidden layer size = 15	33.0683
max input delay = 120, hidden layer size = 22	29.0488
max input delay = 25, hidden layer size = 30	42.1514

The predicted values for the next 20 time steps from 125 to 144 is shown in Fig. 4.1.

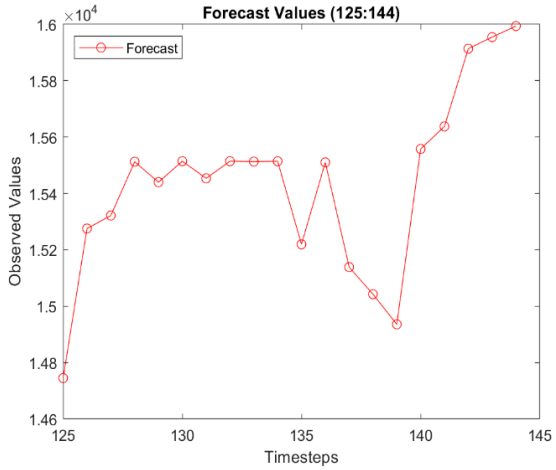

Figure 4.1: Forecast values

The forecasted values of 20-time step with the training time step is shown in the Fig. 4.2.

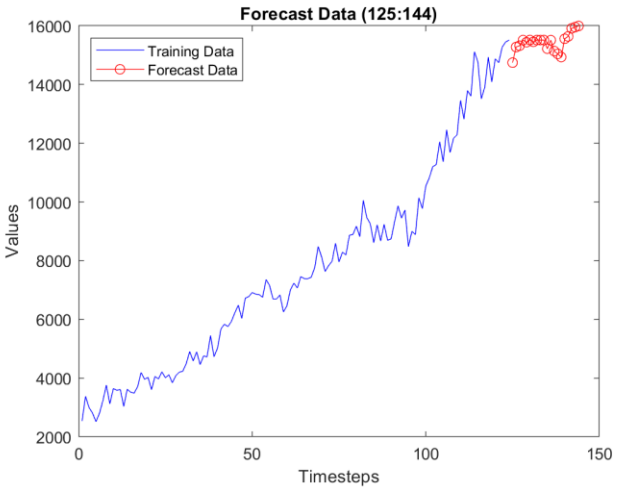

Figure 4.2: Forecast values with the training data

Response of the time series with the Validation data values of the network along with error is given in the Fig 4.3

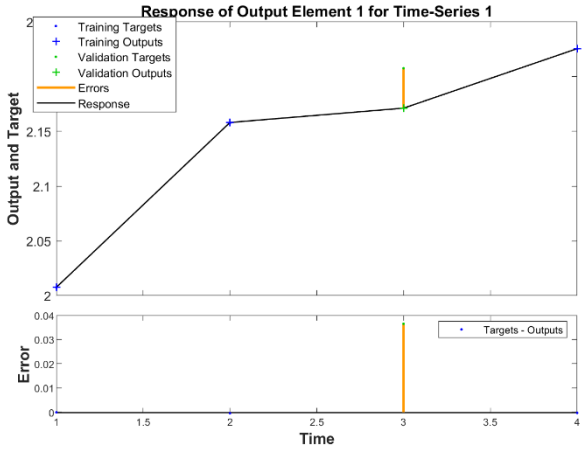

Figure 4.3: Time series response with Validation data values of network

V. Conclusion

By using an artificial neural network, we have been able to fit a small data set with a relatively high degree of accuracy, while still maintaining a healthy bias vs variance tradeoff. Furthermore, utilizing an advanced NAR (nonlinear autoregressive) neural networks allowed for a greater degree of control over how strongly biased the model ultimately was, as seen in Table I. We observed that maximum input delays and hidden layer size is important to achieve good accuracy by avoiding high variance and keeping the computational cost as low as possible.

References

[1] J. W. Tukey, Exploratory data analysis. Reading: Addison-Wesley, 1977.

[2] 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

[3] A. Dertat, "Applied Deep Learning - Part 1: Artificial Neural Networks," Towards Data Science, 08-Aug-2017. [Online]. Available: https://towardsdatascience.com/applied-deep-learning-part-1-artificial-neural-networks-d7834f67a4f6. [Accessed: 28-Feb-2019].

[4] "narnet," Nonlinear autoregressive neural network - MATLAB. [Online]. Available: https://www.mathworks.com/help/deeplearning/ref/narnet.html. [Accessed: 28-Feb-2019].

[5] K. Madsen, H. B. Nielsen, and O. Tingleff, "METHODS FOR NON-LINEAR LEAST SQUARES PROBLEMS Second Edition," Apr-2004. [Online].Available:http://www2.imm.dtu.dk/pubdb/views/edoc_download.php/3215/pdf/imm3215.pdf.