Applied Economics and Finance
Vol. 8, No. 3; May 2021
ISSN 2332-7294 E-ISSN 2332-7308
Published by Redfame Publishing
URL: http://aef.redfame.com

# Prediction of Stock Market Index Using a Hybrid Technique of Artificial Neural Networks and Particle Swarm Optimization

Farnaz Ghashami[1], Kamyar Kamyar[2] & S. Ali Riazi[3]

[1] Ph.D. Candidate, LeBow School of Business, Drexel University, Philadelphia, USA

[2] Ph.D. Student, Department of Economics, The Ohio State University, Columbus, USA

[3] M.Sc. Student, Computational Engineering, Ruhr-Universit ät Bochum, Germany

Correspondence: Kamyar Kamyar, Department of Economics, The Ohio State University, Columbus OH 43210, USA

Received: March 17, 2021 Accepted: April 12, 2021 Available online: April 19, 2021

doi:10.11114/aef.v8i3.5195 URL: https://doi.org/10.11114/aef.v8i3.5195

**Abstract**

In this paper we examine the ability of Artificial Neural Network methods (ANN) for predicting the stock market index. We first conduct an ANN analysis and then optimize the ANN model using Particle Swarm Optimization algorithm (PSO) to improve the prediction accuracy. In terms of data, we use NASDAQ index which is one of the most widely followed indices in the United States. Empirical results show that by determining the optimal set of biases and weights using PSO, we can augment the accuracy of the ANN model for this stock market data set.

**Keywords:** artificial neural networks, particle swarm optimization, stock market prediction, hybrid optimization

## 1. Introduction

Stock market is one of the intensive pillars of the economy of each country, as it plays a crucial role in the growth of industries eventually bringing the economy into nourishment. As a result, during the last decades, literature in stock market forecasting has become highly extensive.

Gourav et al. (2020) states how challenging of a task predicting the stock market indices is. The argument is based mainly on the chaotic nature of stock price time series: the data is non-parametric, volatile, highly noisy, non-linear, complex, and dynamic. Additionally, Gholamiangonabadi et al. (2014) discusses how the behavior of stock markets depends on numerous factors such as economic and political and/or businesses' situation, unrevealed preferences of institutional investors, other stock markets, expectation of the investors, and so on. These factors either permanently or temporarily affect the stock market.

For reasons including above, stock market relies on various types of intelligent systems to be forecasted and made trading decisions on. Investors would easily be able to gain additional opportunities for gaining profits by receiving more accurate forecast of the stock index.

Extensive research literature on prediction of stock market indices have found Artificial Neural Network (ANN) to display higher quality in terms of performance and precision of prediction ability as compared with other statistical methods in financial research area. (Moghaddam et. al. 2016) Among the most popular training algorithms for ANN is Backpropagation (BP) algorithm, which has been widely used in the mentioned area due to its preeminent learning ability as well as applicability to many business problems. (Kim 2003) Despite, as the stock market data within the framework of the ANN model (trained by the BP algorithm) is accompanied by downsides of noise and complex dimensionality, the results are prone to convergence to the local minimum. (Qiu et. al. 2020) Thereby, the drawbacks accompanying BP in this area initiate a need for its improvement using other training algorithms. Based on underlying literature, we propose Particle Swarm Optimization algorithm (PSO) to overcome such limitation for nonlinear optimization problems.

This paper employs the PSO algorithm to improve the prediction accuracy of ANN for NASDAQ-100 stock market index and overcome the local convergence problem of the BP algorithm. We first conduct a regular ANN analysis, and then attempt to improve the prediction accuracy by determining the optimal set of weights and biases of the ANN model by using PSO.

In terms of the paper's structure, the second section is dedicated to literature reviews. In the third section the detailed presentation of the used methodology, data description and experiment will be presented. In the fourth section, results and discussion will be presented; and finally, in the fifth section, the conclusion and potential future research questions are discussed.

## 2. Literature Review

There is an extensive compendium of literature seeking to predict stock market indices. We review some prior research employing hybrid techniques. Initial research on the topic was based on statistical approaches with some drawbacks such as poor prediction quality. Then, soft computing tools such as Fuzzy Logic and Neural Networks were employed to develop stock market prediction which, however, also suffer from drawbacks including inconsistent performance on financial data involving noise. Then, some evolutionary computing tools such as PSO and Genetic Algorithm (GA) were proposed to improve the prediction accuracy. The most recent financial research literature is developed using hybrid techniques. (Majhi et. al. 2008)

Moghaddam et al (2016) forecasts the stock market index using neural networks using feed forward artificial neural networks trained by the back propagation algorithm. Mingyue and Song (2016) employed genetic algorithms (GA) to increase the neural network model's accuracy of prediction. Their results show that their proposed method would improve the accuracy in order to predict the direction toward which the stock market goes. Garcia et al. (2018) introduces a hybrid fuzzy neural network to promptly predict the following day direction of stock market indices (German DAX-30 index). The results indicate that applying HyFIS together in addition to factor analysis in order to select the input factors so as to feed the artificial network would serve as a satisfactory alternative to forecasting the following day's market data. Rokhsatyazdi et al. (2020) proposed a model of neural networks that was based on the Long-Short Term Memory (LSTM) and utilizing evolutionary algorithms, called the Differential Evolution (DE), serving to forecast the following day's stock prices. The proposed model achieved better RMSE when compared with statistical forecasting models such as SARIMA.

It is challenging to find a particular training algorithm always and under all conditions serving as the best for every application. (Howard et. al. 2008) In this paper we optimize the ANN model using PSO to improve the prediction accuracy. Although different literature commonly trained the ANN model by using the gradient technique of the BP algorithm, limitation of BP algorithm such as inability to escape local optima is more apparent when ANNs are applied to complex nonlinear optimization problems. (Qiu et. al. 2020) In order to overcome this drawback, global search techniques such as PSO with better computational efficiencies in comparison with GA are proposed further.

## 3. Method

### *3.1 Artificial Neural Network*

Artificial Intelligence (AI) is a reliable area in terms of solving different types of problems such as prediction (forecasting). (Wanto et. al. 2017) AI is an extremely important discipline, which includes a number of well-recognized areas including neural networks. (Chiroma et. al. 2014, Lasisi et. al. 2014) Neural network is an algorithm inspired by the neurons in our brain, which is designed to recognize patterns in complex data. The neural network consists of three layers: an input layer which takes the input variables, a hidden layer which is used for capturing the relationships among variables, and an output layer which predicts the final output. Funahashi an Hornik, et al. have proved for neural networks with sufficient complexity and solely one hidden layer to approximate any unknown function to any degree of desired accuracy. (Qiu et. al. 2016) There exist several single processing elements called neurons, organized in layers. Every neuron in a layer is connected with all the neurons in the previous layer and each connection has different weights. As shown in Figure 1, the idea is for us to insert input data into input layer, sending numbers from one neuron to another through different connections. Once we reach the output layer, we will be able to reach any desired number.

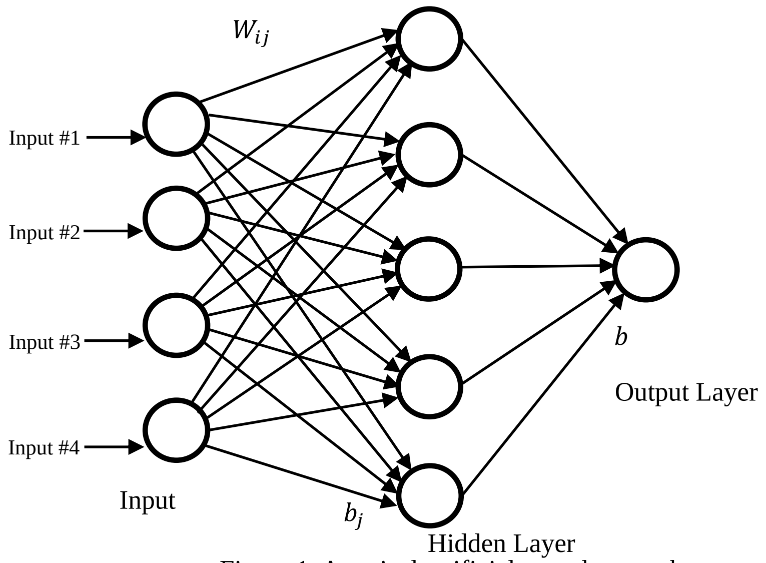


Figure 1. A typical artificial neural network

The input data is denoted by *Input*$_1$, *Input*$_2$, *Input*$_3$, ...*Input*$_i$. Each neuron that is connected to a new neuron has a weight denoted by *w*. Bias, denoted by *b*, approximates where the value of the new neuron starts to be meaningful. Each neuron also has some activation value between 0 and 1 denoted by *a*. Also, there should be an activation function denoted by $\sigma$. Activation helps to measure which weights matters the most. We can calculate a new neuron's value by adding or subtracting a bias to the multiplication of activation and weights. The following equation represents a new neuron value:

$$\sigma(w_1a_1 + w_2a_2 + ... + w_na_n \pm b) \tag{1}$$

To get a matrix of all the activations in the next layer, we need all the activation from the previous layer and all the weights connected to each neuron in the next layer and adding or subtracting a bias to the multiplication of activation and weights then wrapping the whole equation in the sigmoid function:

$$\sigma(\begin{bmatrix} w_{0,0} & w_{0,1} & \dots & w_{0,k} \\ w_{1,0} & w_{1,1} & \dots & w_{1,k} \\ \dots & \dots & \dots & \dots \\ w_{j,0} & w_{j,1} & \dots & w_{j,k} \end{bmatrix} \begin{bmatrix} a_0^0 \\ a_1^0 \\ \dots \\ a_n^0 \end{bmatrix} + \begin{bmatrix} b_0 \\ b_1 \\ \dots \\ b_n \end{bmatrix})$$

In the above formula, $a_0^1$ shows the number two neuron in the first layer; and $w_0^1$ means to the first neuron from neuron two in the previous layer, and so forth. We can write the above formula as:

$$a^{(1)} = \sigma(Wa^0 + b) \tag{3}$$

In a multi-layer structure the process in order is continue, the input neurons, received information, pass it to the neurons in the first hidden layer, the behavior of each neuron is determined. Then the outputs from the first hidden layer are passed to the next layer and the behavior of each neuron in this layer is determined, and so on until it reaches the output layer and produces the desired output.

There exist a number of factors that we can use to see how well a model performs. In order to know the performance of the model we use a standard cost function whose value will later be optimized.

The next step after constructing the structure of the ANN is to train the network. Training the networks is essentially the process of searching for and acquiring the optimal values of different biases and weights of the network. In this paper, we utilize PSO in order to obtain the network's optimum value.

### *3.2 Particle Swarm Optimization Algorithm*

The Particle Swarm Optimization (PSO) is an evolutionary computation technique which is capable of solving highly non-linear problems effectively. Originally developed by Kennedy and Eberhert in 1995, the method seeks to solve an optimization problem by iteratively improving a candidate solution subject to the given constraints. The algorithm moves particles around a search-space to identify solutions locally and globally. While the PSO was originally used for the purpose of analyzing human behavior, it has been used in various contexts since. Theoretically, it defines a cost function in $\mathbb{R}^n \to \mathbb{R}$ to be minimized, for which there exists a candidate solution as an argument. The candidate solution is within the form of a vector of real numbers, and its output is a real number, showing the value of the problem's candidate solution in an objective function form. The algorithm seeks to find a global minimum solution $x$ for which $f(x) \leq f(y)$ for all $y$ in the search-space, while the gradient of $f$ is unknown.

More precisely, if we consider a swarm $i$ in an n-dimensional search space at $t$ iterations the position vector represented by

$$X_i(t) = (x_{i1}, x_{i2}, \ldots, x_{in}), \tag{4}$$

and its velocity is represented by

$$V_i(t) = (v_{i1}, v_{i2}, \ldots, v_{in}). \tag{5}$$

A particle can adjust its position according to the best position found by itself as well as its neighbors. The updated velocity is:

$$V_i\,(t+1) = w.V_i(t) + c1.rand().(P_i - X_i(t)) + c2.rand().(P_g - X_i(t)) \tag{6}$$

The updated position is determined by the following equation:

$$X_i(t+1) = X_i(t) + V(t+1) \tag{7}$$

In the first equation $w$ denotes the weight, *rand()* is a random variable between zero and one, $P_i$ is the best position found by itself, $P_g$ is the best position found by its neighbors, and *c1* and *c2* are the acceleration constants.

The neural network trained by PSO can be detailed as follows:

(1) At time $t$, the positions of all particles are randomly generated, and the dimension of each particle are calculated by the number of weights and biases of the neural network.

(2) At time $t + 1$, the new velocity and new position are calculated. Each position of a particle is considered as one set of potential parameters for the neural network. So, when the particle has a new position, the corresponding weights and the biases of the neural network are changed. The new $P_i$ and $P_g$ are calculated. The output data from the neural network will be evaluated by the mean squared error function.

(3) When the stopping condition was satisfied (e.g., predefined number of iterations is reached), the training of the neural network by PSO is stopped, the final values correspond to the updated weights and biases of the neural network is found.

## 4. Data Description and Empirical Analysis

The data for this analysis is sourced from *Yahoo! Finance*. We considered NASDAQ-100 index which is one of the most widely followed indices in the United States. More precisely, it is a modified capitalization-weighted index that made up of 103 equity securities issued by 100 of the largest non-financial companies listed on the Nasdaq stock market. (Yahoo! Finance) The data set is of daily frequency and extend from 21 October 2020 through 17 March 2021. We considered the short-term historical stock prices (seven prior days) as inputs dataset. So, if the output is $y(k)$ which is the stock price at time $k$, and $n$ is the number of historical data, the equation should be:

$$y(k) = f(y(k-1), y(k-2), y(k-3), \ldots, y(k-n)) \tag{8}$$

In the next step several models using ANN as well as PSO-ANN are developed for both types of aforementioned datasets. The best results based on R-Squared and Mean Squared Error (MSE) were shown in Table 1 and Table 2.

Table 1. Seven Prior Working Days

| **Seven Prior Working Days** | | | | |
|---|---|---|---|---|
| Statistics | ANN | ANN | PSO-ANN | PSO-ANN |
| Neurons | [20,40,20] | [5] | [5] | [10] |
| Particle | —— | —— | 20 | 20 |
| R-Squared | 0.9785 | 0.9763 | 0.9805 | 0.9795 |
| Obj-Function | 28611 | 31617 | 25763 | 27049 |
| No. of Iterations | 1000 | 1000 | 1000 | 200 |
| Running time | 4 Seconds | 2 Seconds | 134.41 Seconds | 32.12 Seconds |
| Test Error for One Dataset | 0.005 | 0.011 | 0.001 | 0.005 |

## 5. Results and Discussion

Table 1 outlines the application of different training algorithms on the seven prior working days data set of NASDAQ-100 stock index. Based on the results, it can be concluded that the proposed method (PSO-ANN) outperforms the typical artificial neural network (ANN). As shown in Table 1, even though the length of time required to perform a computational process for PSO-ANN is above ANN, it can be observed that the proposed method achieves better R-Squared and Mean Squared Error (MSE). Based on the results we can conclude that PSO-NN improve prediction accuracy of ANN model for this stock market data set in this specific time period.

## 6. Conclusion and Future Work

In this project we first tried to train neural network using Backpropagation algorithm, then we adjusted the weights and biases of the neural network model using the PSO algorithm. Finally, we tested the performance of the ANN and PSO-ANN model by applying on NASDAQ-100 stock. Experimental results show that the PSO-ANN architectures proposed in the study improve the prediction accuracy and can be applied to stock market or financial data analysis. This study may be improved further in two ways: (1), including other variables as an input data that may affect the prediction of stock market's performance. And (2), utilizing other optimization methods to adjust the parameters in neural networks.

## References

Alam, M. (2016). *Codes in MATLAB for training artificial neural network using particle swarm optimization.*

Chiroma, H., Abdulkareem, S., Abubakar, A. I., & Herawan, T. (2014). Kernel Functions for the Support Vector Machine: Comparing Performances on Crude Oil Price Data. In: Herawan T., Ghazali R., Deris M. (Eds.), *Recent Advances on Soft Computing and Data Mining. Advances in Intelligent Systems and Computing*, vol. 287. Springer, Cham. https://doi.org/10.1007/978-3-319-07692-8_26

Clerc, M. (2012). *Standard Particle Swarm Optimisation.* HAL open access archive. hal-00764996f

Funahashi, K. (1989). On the approximate realization of continuous mappings by neural networks. *Neural Networks, 2*(3), 183-192. https://doi.org/10.1016/0893-6080(89)90003-8

Garcia, F., Guijarro, F., Oliver, J., & Tamošiūnienė, R. (2018). Hybrid fuzzy neural network to predict price direction in the German DAX-30 index. *Technological and Economic Development of Economy, 24*(6), 2161-2178. https://doi.org/10.3846/tede.2018.6394

Gholamiangonabadi, D., Mohseni, T. S. D., Mohammadi, A., & Menhaj, M. B. (2014). Investigating the performance of technical indicators in electrical industry in Tehran's Stock Exchange using hybrid methods of SRA, PCA and Neural Networks. *2014 5th Conference on Thermal Power Plants (CTPP) IEEE*. https://doi.org/10.1109/CTPP.2014.7040698

Golbon-Haghighi, M., Saeidi-Manesh, H., Zhang, G., & Zhang, Y. (2018). Pattern Synthesis for the Cylindrical Polarimetric Phased Array Radar (CPPAR). *Progress In Electromagnetics Research, 66*, 87-98. https://doi.org/10.2528/PIERM18011016

Hansen, C. "Machine Learning from Scratch". Retrieved from https://mlfromscratch.com/

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural networks, 2*(5), 359-366. https://doi.org/10.1016/0893-6080(89)90020-8

Howard, D., Mark, B., & Martin, H. (2009). *Neural Network Toolbox 6 User's Guide for MATLABfi.*

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN'95 - International Conference on Neural Networks, Perth, WA, Australia, 4.* https://doi.org/10.1109/ICNN.1995.488968

Kim, K. (2003). Financial time series forecasting using support vector machines. *Neurocomputing, 55*(1-2), 307-319. https://doi.org/10.1016/S0925-2312(03)00372-2

Kumar, G., Jain, S., & Singh, U. P. (2020). Stock Market Forecasting Using Computational Intelligence: A Survey. *Computational Methods in Engineering*. https://doi.org/10.1007/s11831-020-09413-5

Lasisi, A., Ghazali, R., & Herawan, T. (2014) Comparative Performance Analysis of Negative Selection Algorithm with Immune and Classification Algorithms. In: Herawan T., Ghazali R., Deris M. (eds) Recent Advances on Soft Computing and Data Mining. *Advances in Intelligent Systems and Computing, 287.* Springer, Cham. https://doi.org/10.1007/978-3-319-07692-8_42

Majhi, R., Panda, G., Sahoo, G., Panda, A., & Choubey, A. (2008). Prediction of S&P 500 and DJIA stock indices using Particle Swarm Optimization technique. *2008 IEEE Congress on Evolutionary Computation* (IEEE World Congress on Computational Intelligence), Hong Kong, China. https://doi.org/10.1109/CEC.2008.4630960

Moghaddam, A. H., Moghaddam, M. H., & Esfandyari, M. (2016). Stock market index prediction using artificial neural network. *Journal of Economics, Finance and Administrative Science, 21*(41), 89-93. https://doi.org/10.1016/j.jefas.2016.07.002

Qiu, M., & Song, Y. (2016). Predicting the direction of stock market index movement using an optimized artificial neural network model. *PloS one, 11*(5), e0155133. https://doi.org/10.1371/journal.pone.0155133

Rokhsatyazdi, E., Rahnamayan, S., Amirinia, H., & Ahmed, S. (2020). Optimizing LSTM Based Network for Forecasting Stock Market. *2020 IEEE Congress on Evolutionary Computation (CEC)*. IEEE. https://doi.org/10.1109/CEC48606.2020.9185545

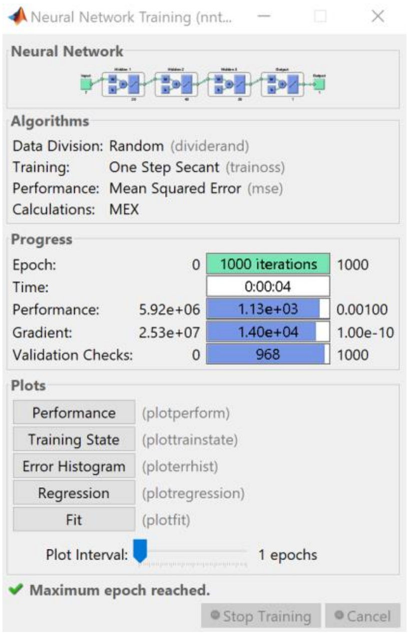
Sho, H. (2019). *Use of Particle Multi-Swarm Optimization for Handling Tracking Problems. Swarm Intelligence - Recent Advances, New Perspectives and Applications.* Javier Del Ser, Esther Villar and Eneko Osaba, IntechOpen. https://doi.org/10.5772/intechopen.85107

Wanto, A., Zarlis, M., & Hartama, D. (2017). Analysis of Artificial Neural Network Backpropagation Using Conjugate Gradient Fletcher Reeves in The Predicting Process. *J. Phys.: Conf. Ser., 930, 012018. https://doi.org/10.1088/1742-6596/930/1/012018*

Yahoo! Finance - Stock Market Live, Quotes, Business & Finance News. Yahoo! Finance. Retrieved from https://finance.yahoo.com/

Zhang, R., Huang. C., & Chen, S. (2018). Futures Trend Strategy Model Based on Recurrent Neural Network. *Applied Economics and Finance, 5*(4), 95-101. https://doi.org/10.11114/aef.v5i4.3306

**Appendix A**: **Analysis for Seven prior working days data set**



Neural Network Training (nnt...

Neural Network

Algorithms

Data Division: Random (dividerand)
Training: One Step Secant (trainoss)
Performance: Mean Squared Error (mse)
Calculations: MEX

Progress

| | | | |
|---|---|---|---|
| Epoch: | 0 | 1000 iterations | 1000 |
| Time: | | 0:00:04 | |
| Performance: | 5.92e+06 | 1.13e+03 | 0.00100 |
| Gradient: | 2.53e+07 | 1.40e+04 | 1.00e-10 |
| Validation Checks: | 0 | 968 | 1000 |

Plots

Performance (plotperform)
Training State (plottrainstate)
Error Histogram (ploterrhist)
Regression (plotregression)
Fit (plotfit)

Plot Interval: 1 epochs

✓ Maximum epoch reached.

Figure 2. Analysis for seven prior working days data set using ANN, (Architecture [20,40,20])

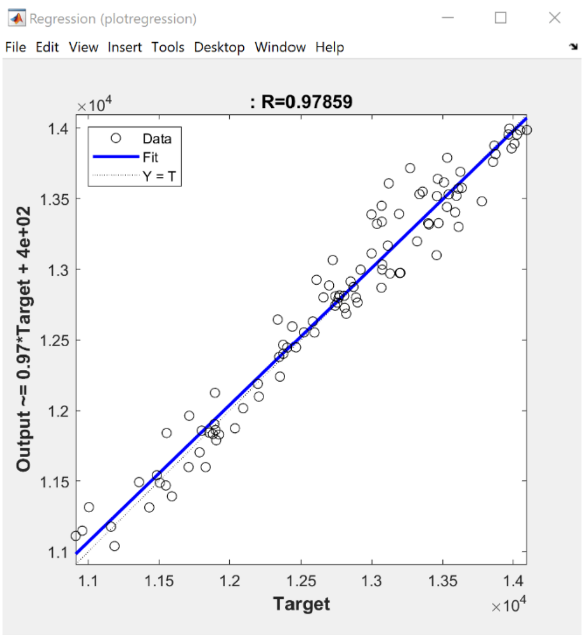
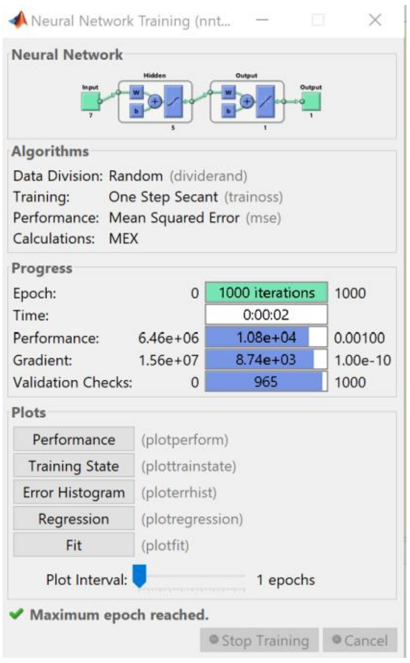


Figure 3. R-Squared for seven prior working days data set using ANN, (Architecture [20,40,20])



Neural Network Training (nnt...

**Neural Network**

**Algorithms**

```
Data Division: Random (dividerand)
Training:      One Step Secant (trainoss)
Performance:   Mean Squared Error (mse)
Calculations:  MEX
```

**Progress**

```
Epoch:                    0    1000 iterations   1000
Time:                          0:00:02
Performance:       6.46e+06    1.08e+04          0.00100
Gradient:          1.56e+07    8.74e+03          1.00e-10
Validation Checks:        0    965               1000
```

**Plots**

```
Performance       (plotperform)
Training State    (plottrainstate)
Error Histogram   (ploterrhist)
Regression        (plotregression)
Fit               (plotfit)
```

Plot Interval: 1 epochs

✓ **Maximum epoch reached.**

Figure 4. Analysis for seven prior working days data set using ANN, (Architecture [5])

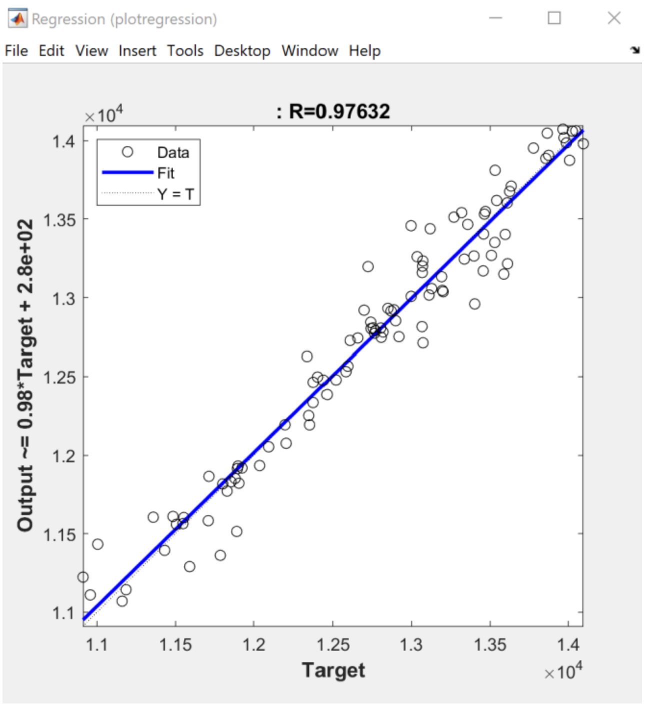


Figure 5. R-Squared for seven prior working days data set using ANN, (Architecture [5])