INTEGRATION OF TWO DIFFERENT SIGNAL PROCESSING TECHNIQUES WITH ARTIFICIAL NEURAL NETWORK FOR STOCK MARKET FORECASTING

Aaron R. Rababaah, University of Maryland Eastern Shore, USA
Dinesh K. Sharma, University of Maryland Eastern Shore, USA

ABSTRACT

In the area of the stock market forecasting, a number of methods have been used. One of the popular methods is Back Propagation Artificial Neural Network (BP-ANN). Exploring the previous work in this area, one can observe that predictive models suffer unreliability due to instability in security markets. We observe that this problem is analogous to a similar problem in signal processing systems where noisy signals resemble the variations and instability in the financial signals. Therefore, we believe that applying filtering techniques of signal processing should improve the quality of the financial signals before using them in predictive models such as BP-ANN. In this work, we will integrate two different pre-processing techniques: Gaussian Zero-Phase filter and Fast Fourier Transform to test their possible contributions to enhance forecasting reliability. To test these two techniques, the past 12 years data for two stock market indices, DOW30, and NASDAQ100, are used in the experimental work to verify and compare the performance of the two techniques.

Keywords: *Back Propagation Artificial Neural Network, Signal Processing, Gaussian Zero-Phase filter, Fast Fourier Transform, Stock Market Forecasting.*

INTRODUCTION

One of an active portfolio management's critical tasks is forecasting stock market variations with time. Although forecasting of the stock market is crucial, many techniques have been explored in this area but did not fully succeed to achieve satisfactory levels of prediction accuracy. Especially, during uncertain local and global circumstances and events, stock market indices behave with a high degree of uncertainty. In the challenging times of the economy, investors are very careful and conservative about their investments and their financial security. According to the Modern Portfolio Theory, non-systemic risk can be reduced by mutual funds and index funds. Whereas, the systematic risk due to the unstable economic and financial situations cannot be controlled. Technical trading techniques claim that by sending buying and selling signals on a regular basis, investors can make better investment decisions even in volatile market conditions. Given that claim, establishing a technical trading mode that can reliably predict stock market indices in very challenging and dynamic economic situations requires refined and advanced solutions.

One of the most popular models which has been investigated and used by many researchers in this area of research is Artificial Neural Networks (ANNs). ANNs are mathematical models that attempt to emulate the human brain neural network and its reasoning process to recognize patterns. The power of ANNs is represented in their learning ability from training on incomplete, imprecise, and partially incorrect examples. Its unique advantage makes it well suited to deal with unstructured problems, inconsistent information, and real-time output (Trippi &Turban, 1996). ANNs have many useful and reliable purposes, including: clustering, classification, and recognition in many fields of research. They are applied to forecast stock and commodity prices, bond ratings, foreign exchange rates, T-bills, bonds, and inflation (Aiken, 1999; Krishnaswamy et al., 2000; Sharma & Alade, 1999).

Several studies have been published in the last two decades that suggested that the ANNs is more reliable than other traditional forecasting techniques (Sharda & Patil, 1990; Tang, 1991; Trippi & Turban, 1993; Atsalakis & Valavanis, 2009). ANNs is a robust predictive model for challenging time series via training to approximate the hidden patterns in a time series. Although ANNs is a reliable predictive model, its effectiveness relies on a number of factors such as: learning algorithm, quality of training data sets, network architecture, etc. (Sharma & Alade, 1999). Further, one of the most important factors is reported to be noise content in the signal that significantly destabilizes the performance of ANNs (Kim, 2006). Therefore, it is of great interest and potential to investigate the benefits of signal processing techniques that can be used for signal de-noising as a preprocessing stage. A very interesting observation made by (Nair et al., 2010) that filtering techniques are widely used in general signals but rarely in financial signals. So, from the preceding discussion, we propose to utilize the reliability of the ANNs and the de-nosing capability of the signal processing techniques in an integrated predictive model for financial signals, especially stock market indices. In this proposed method, the feature vectors are modeled as k-entry days of the stock market close day data. Based on the multi-layer perceptron (MLP) BP-ANN architecture requirements, the classes of feature vectors are taken as the last entry of each vector. Since the stock market data is real data, the vectors are discretized into a suitable number of levels representing the range of classes. The training and testing data was collected from DOW30 and NASDAQ100 over 12 years where the experimental results demonstrated the reliability of the proposed approach with average prediction accuracy of 98.25%.

LITERATURE REVIEW

In the literature, Artificial Neural Networks (ANNs) have been a common choice of investigation for a wide spectrum of applications in business and economics especially in the stock market forecasting problems. White (1988) used ANNs for time series analysis of IBM stock daily returns. Trippi and DeSieno (1992) studied a specific ANNs trading system for S&P 500 index future contracts. Lin and Lin (1993) investigated ANNs' forecasting capability for Dow Jones Industrial Average (DJIA). Kryzanowski et al. (1993) applied the Boltzmann machine for ANNs training to classify stock returns as negative, positive, or neutral. Refenes et al. (1993) applied feed-forward network with multi-layers. Refenes et al. (1994) did a comparative study between regression models with a back-propagation network for stock forecasting. Dropsy (1996) used ANNs to build a nonlinear forecasting technique to predict international equity risk premia. Wang and Leu (1996) introduced an integrated solution of

moving average (ARIMA) and ANNs for forecasting price movement of the Taiwan stock market.

Motiwalla and Wahab (2000) used BPANN in their research work. Lam (2004) studied ANNs' capability of technical analysis for financial performance prediction. Qing et al. (2005) used ANNs to predict stock price movement for companies of Shanghai stock exchange. Constantinou et al. (2006) applied MLP-ANN with two inputs. Zhu et al. (2008) investigated forecasting of stock returns and volumes from various indices by applying the ANNs technique. Atsalakis and Valavanis (2009) presented an extensive survey on the stock market forecasting models and techniques. Manjula et al. (2011) used ANNs models to forecast the daily returns of the Bombay Stock Exchange Sensex. MLP-ANN was used to build the daily return's model, and to provide a better alternative for weight initialization, the MLP network was trained using multiple linear regressions. Qing et al. (2011) conducted a comparative study on the predictive ability of several forecasting models including: a single-factor capital asset pricing model (CAPM), and Fama and French's three-factor model. All these models were compared for the forecasting ability with ANNs. Oliveira et al. (2013) applied ANNs to predict stock price and improve the directional prediction index with two case studies investigated. Park and Shin (2013) presented a complex ANNs architecture that considered many influential factors applied to the global economic index and stock prices of 200 individual companies. Ticknor (2013) introduced a novel model of Bayesian–ANN integrated solution that adds probabilistic nature to the traditional ANN weights which prevent over-fitting and over-training of the ANN model. Recently, Sharma and Rababaah (2014) proposed signal processing technique with ANN to predict stock market forecasting.

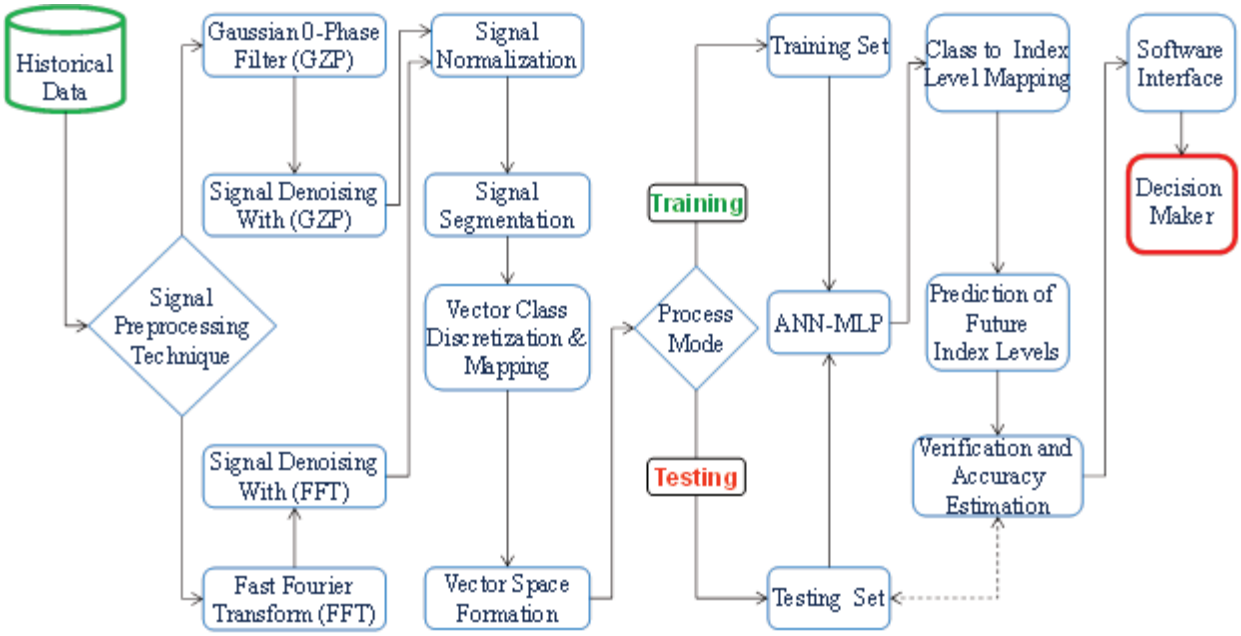
THEORY AND TECHNICAL APPROACH

In this section, we will discuss the technical aspects of the two proposed methods of Fast Fourier Transform (FFT) in the context of signal processing and BP-ANN. Further, the software implementation of both methods will be presented. The integration of the two methods in one solution for Stock Indices Prediction will be discussed as well. Figure 1 depicts the proposed process model that will be described in the following sections.

The first step in our study was to use both GZP and FFT to filter the DOW30 signal. The results of this step are shown in Figure 4 (left). To see the differences in details between the two techniques, we enlarged a snapshot of a section of the signal as illustrated in Figure 4 (right).

Overall, the two techniques produced similar performances with one observation that gave the GZP an advantage over the FFT. This observation was matching the peeks of the input and the filtered signals. The effects of this observation will be demonstrated in the experimental work section.

Figure 1
Flow Diagram of the Proposed Data Processing Models



Historical Data

The online databases of Yahoo Finance (http://finance.yahoo.com/) were utilized to collect the data sets for training and testing the proposed techniques. The collected data covered daily open, close, low, and high indexes of twelve years for NASDAQ100 and DOW30.

Signal Preprocessing Techniques

Noise content in raw data could be a challenging problem for data processing models that use these data sets for training and testing. Therefore, it is important to consider filtering raw data from noise content to make it reliable for the processing model so its forecasting capability is not negatively impacted.

Factors such as natural, political, finical, social, etc., are examples of many that affect stock index data. One of the main objectives in our study is to significantly reduce the noise content of the input signal using the aforementioned signal processing models.

We are interested in this study due to our previous work in Sharma & Rababaah (2014) when we explored this idea of using signal processing techniques in stock indexes. Our previous experience with GZP was very promising and we are interested to compare its performance with another powerful technique of FFT to explore their differences and potentials.

Fast Fourier Transform (FFT)

The fundamental concept of digital-signal filtering is the convolution. Algebraically, convolving two polynomial functions is the operation of multiplying the coefficients of these two functions. Let the coefficients of two functions be denoted as the two vectors: u and v. Further, let the lengths of these two vectors be m and n respectively. Then the output vector of the convolution operation of the two input polynomials is of a length of $m+n-1$. The k^{th} element in this output vector is expressed as:

$$w(k) = \sum_{j} u(j)v(k+1-j) \tag{1}$$

Based on the convolution theory (Quinquis, 2008), the convolution of two sequences can be obtained by the product of their Fourier transform as follows:

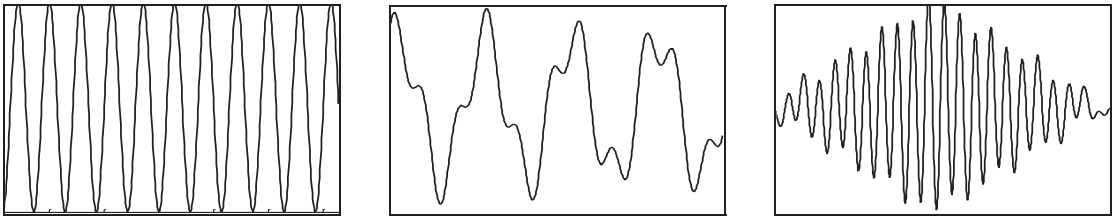
Let x be the first sequence, y be the second sequence, where it assumed that both sequences have the same length; otherwise, they need to be aligned by the zero-padding operation. Further, let X, Y be the discrete Fourier transform (DFT) of the sequences, respectively, then the convolution operation (Figure 2) of these two sequences can be obtained as:

$$w = FFT^{-1}(X \bullet Y) \tag{2}$$

Whereas,

W = the oupt signal of the convolution operation,
FFT^{1} = inverse Fourier transform, and
X•Y = the element-wise product of the two signals.

Figure 2
Illustration of convolution operation of two signals
(Left: 1st input signal; middle: 2nd input signal, and right: the output signal of the convolution operation)



The digital-signal filtering utilizes the convolution theory by designing a filter impulse response (FIR) denoted as h, and convolving it with the input signal x to produce the filtered signal y as follows:

$$y(k) = h(k) * x(k) = \sum_{l=-\infty}^{\infty} h(k-l)x(l) \tag{3}$$

To demonstrate the effect of typical filter, figure 3 illustrates an input noise signal convoluted by a typical averaging filter. It can be observed that the output signal was successfully recovered with high accuracy using convolution-based signal filtering.

Figure 3
Illustration of noise filter using convolving a noisy signal with an arbitrary averaging filter
(Left: noiseless input signal; middle: 2nd input signal with added noise, and right: the output filtered signal)

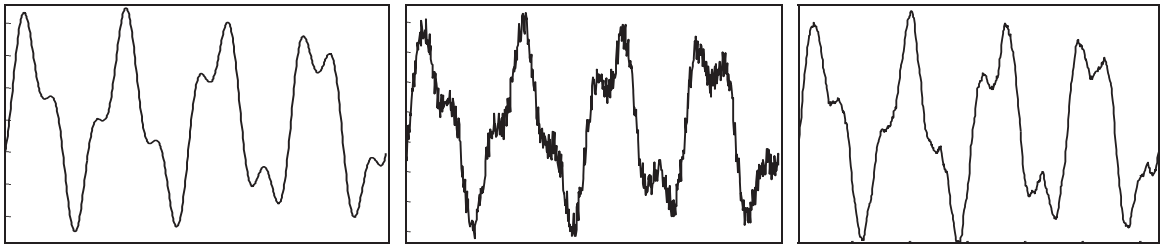
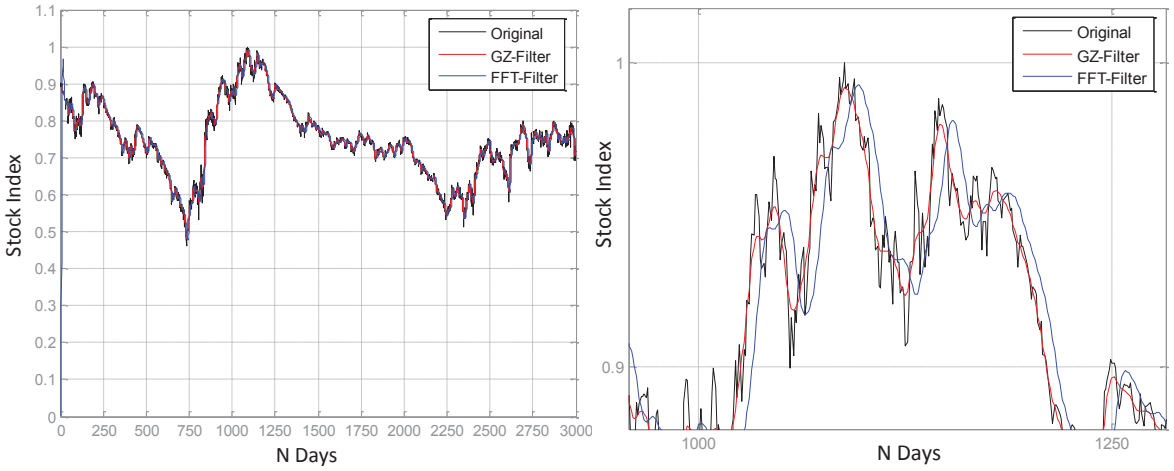


Figure 4 illustrates the preliminary testing results of the signal filtration of DOW30 with GZP vs. FFT. The overall result is depicted in the left chart where in the right chart, a zoomed-in section is chosen to demonstrate a close-up snapshot on the details of the differences between the two filters. It can be observed from the two figures that the two methods have similar filtering effects with one difference that is a forward shift in the case of FFT. The impact of this effect will be elaborated upon in the final testing section of this paper.

Figure 4
Initial Investigation between GZP and FFT de-noising Techniques



Back Propagation Artificial Neural Network (BP-ANN)

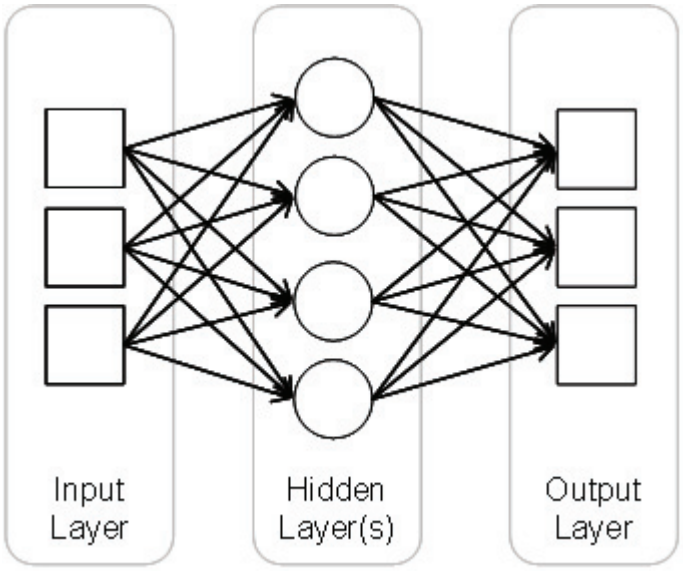
The structure of the Multi-Layer Perceptron (BP)-ANN (Figure 5) consists of three parts:

- Input Layer: the first layer of perceptrons that receives the input vector.
- Hidden Layers: located between the input and the output layer. The output of the input layer is fed into the first hidden layer; the output of the first layer is fed into the next hidden layer, and so on. The number of hidden layers needed varies based on the complexity of the application. Often, the nodes (perceptrons) of the adjacent layers are fully connected.
- Output Layer: The multiple nodes in the output layer typically correspond to multiple classes for multi-class pattern recognition problems.

Multilayer Perceptron Learning Algorithm (Back Propagation)

The most common approach is the gradient descent algorithm, in which a gradient search technique is used to find the network weights that minimize a criterion function. The criterion function to be minimized is the Sum-of-Square-Error. A complete derivation of the algorithm can be found in Haykin (1994).

Figure 5
Multilayer Perceptron Model



The algorithm described in Haykin (1994) is depicted in Figure 6 and can be summarized as follows:
Parameters definitions in the algorithm are defined for a neuron in layer *(l):*

$w^{(l)}$: Synaptic weight vector of a neuron.

$\theta^{(l)}$: Threshold of a neuron.

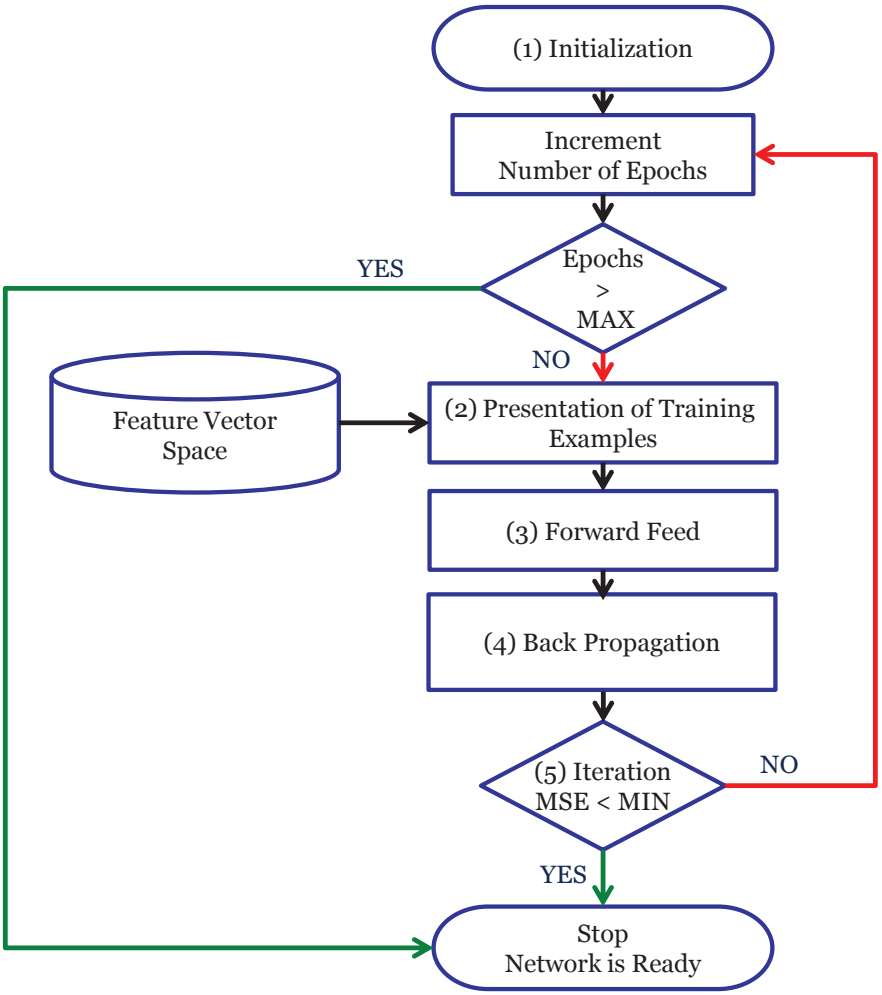
$v^{(l)}$: Vector of net internal activity levels of neurons.

$y^{(l)}$: Vector of function signals of neurons.

$\delta^{(l)}$: Vector of local gradients of neurons.

e_j : The j^{th} error vector where, j = 1, 2, 3, …, n

Figure 6
Block Diagram of the Learning Algorithm of MLP BP-ANN



1. *Initialization*: all synaptic weights and thresholds are set to small random numbers.
2. *Presentations of Training Examp*les: Present the network with an epoch of training examples. For each example in the set, step 3 and 4 are repeated.
3. *Forward Feed:* let a training example in the epoch be [x(n), d(n)], where x(n) is the input vector, and d(n) is the desired output vector on the output layer. The activation potential and function signals of the network are computed, proceeding forward through the network, layer by layer, using the following relation system:

$$v_j^{(l)}(\mathrm{n}) = \sum_{i=0}^{p} w_i^{(l)}(n) y_i^{(l-1)}(n) \tag{4}$$

Where, $w_0^{(l)}(n) = -1, \quad y_0^{(l-1)} = \theta$

$$y_i^{(l)}(\mathrm{n}) = \frac{1}{1+\exp(-v_j^{(l)}(n))} \tag{5}$$

If the neuron is the first hidden layer, then

$$y_i^{(0)}(\mathrm{n}) = x_j(\mathrm{n}) \tag{6}$$

If the neuron is in the output layer ($l = \mathrm{L}$), then

$$y_i^{(L)}(\mathrm{n}) = o_j(n) \tag{7}$$

Hence, the error is computed as

$$e_j(n) = d_j(n) - o_j(n) \tag{8}$$

4. *Back Propagation*: compute the local gradients of the network, proceeding backward, layer by layer.

In the output layer

$$\delta_j^{(L)}(n) = e_j^{(L)}(n) o_j(n)[1 - o_j(n)] \tag{9}$$

In a hidden layer

$$\delta_j^{(l)}(n) = y_j^{(l)}(n)[1 - y_j^{(l)}(n)] \sum_k \delta_j^{(l+1)}(n) w_{kj}^{(l+1)}(n) \tag{10}$$

Hence, adjust the weights:

$$w_{ji}^{(l)}(n+1) = w_{ji}^{(l)}(n) + \alpha[w_{ji}^{(l)}(n) - w_{ji}^{(l)}(n-1)] + \eta \delta_j^{(l)}(n) y_j^{(l-1)}(n) \tag{11}$$

where:

η : is the learning-rate parameter

α : is the momentum constant.

The tradeoff of η is a rough approximation for faster processing, or a better approximation for slower processing. In the case where a high learning-rate is chosen, α is introduced to stabilize the system.

5. *Iteration:* Iterate the computation by presenting new epochs of training examples to the network until the free parameters of the network stabilize their values and the average square error computed over the entire training set is at minimum or acceptable small value. The order of presentation of training examples should be randomized from epoch to epoch. The momentum and the learning-rate parameter are typically adjusted (and usually decreased) as the number of training iterations increases.

Some points need to be considered in the MLP model:

- The weights typically are initialized to small random values, which gives the algorithm a safe start.
- A simple heuristic technique is used to choose learning rates, which is to make the learning rate for each node inversely proportional to the average magnitude of vectors feeding to that particular layer.

- Termination criteria includes the following:
  - A target minimum gradient is reached.
  - The Sum-of-Square-Error falls below a fixed threshold.
  - When all of the training samples have been correctly classified.
  - After a fixed number of iterations have been performed.
  - Cross Validation technique.

  As usual, the available input space is randomly partitioned into a training set and a test set. The training set is further partitioned into two subsets: a subset used for estimation of the model (model training), and a subset used for evaluating the performance of the model (model validation). The validation subset is typically 10 to 20% of the training set. The goal of this technique is to validate the model on a data different from the one used for model estimation.

The best model is chosen after this validation phase, then the chosen model is trained using the full training set (Haykin, 1994).

It is worth mentioning that the last approach (Cross Validation) in contrast to all of the other approaches is not sensitive to the choice of the parameters. It not only avoids premature termination, but can improve the generalization performance. However, it is computationally intensive.

EXPERIMENTAL WORK AND RESULTS ANALYSIS

Three different settings were used in the experimentation stage these are: no signal preprocessing (NPP), Gaussian Zero-Phase Filter (GZP) and Fast Fourier Transform (FFT). The three different settings are presented in this section.

Experiments with Default Parameters for all Techniques

We wanted to test all cases of NPP, GZP and FFT without attempting to investigate the effect of tuning their parameters and set the result as our baseline for later comparisons. We used the two data sets for NASDAQ100 and DOW30 in all experiments. In the next section we will present the experiments of optimization trials for the FFT technique. Thus, Figures 7, 8, and 9 represent the results of the experiments with default parameters. As it can be observed, in the Figures 7-10 and Table 1, NPP's accuracy was the lowest. FFT performed better than NPP and the best accuracy was for GZP in both data sets of NASDAQ100 and DOW30.

Table 1
Summary of Results with Default Parameters

Experiment	Technique	Testing Accuracy	
		DOW30	NASDAQ100
1	NPP	96.86	96.52
2	GZP	98.16	97.88
3	FFT	97.53	96.54

Figure 7
Results of No Signal Preprocessing

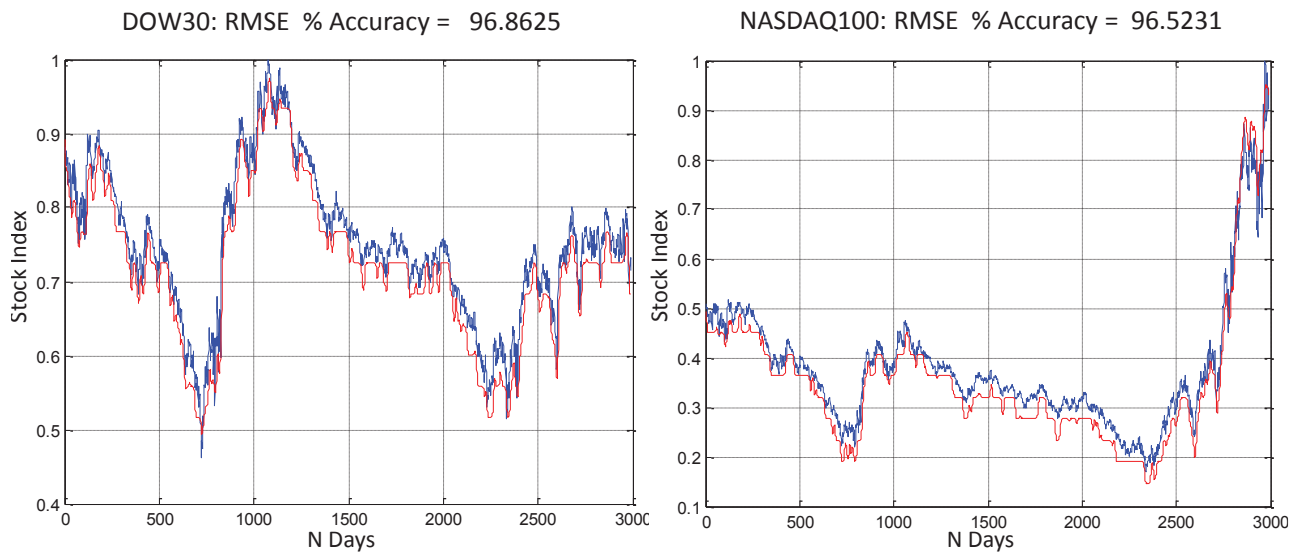


Figure 8
Results of GZP Signal Preprocessing with Default Parameters

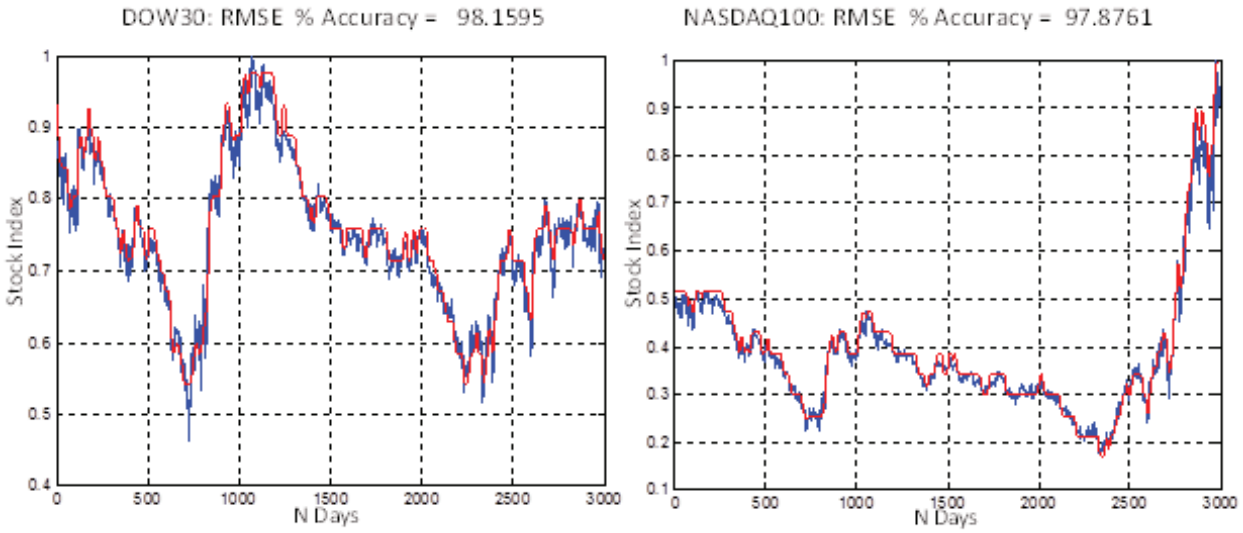


Figure 9
Results of FFT Signal Preprocessing with Default Parameters

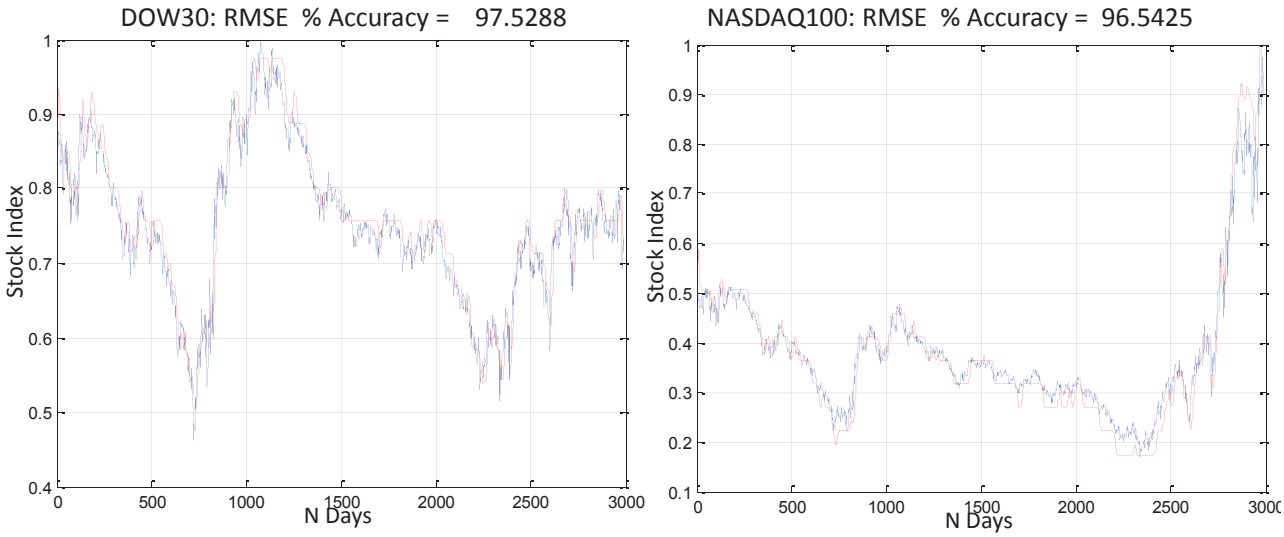
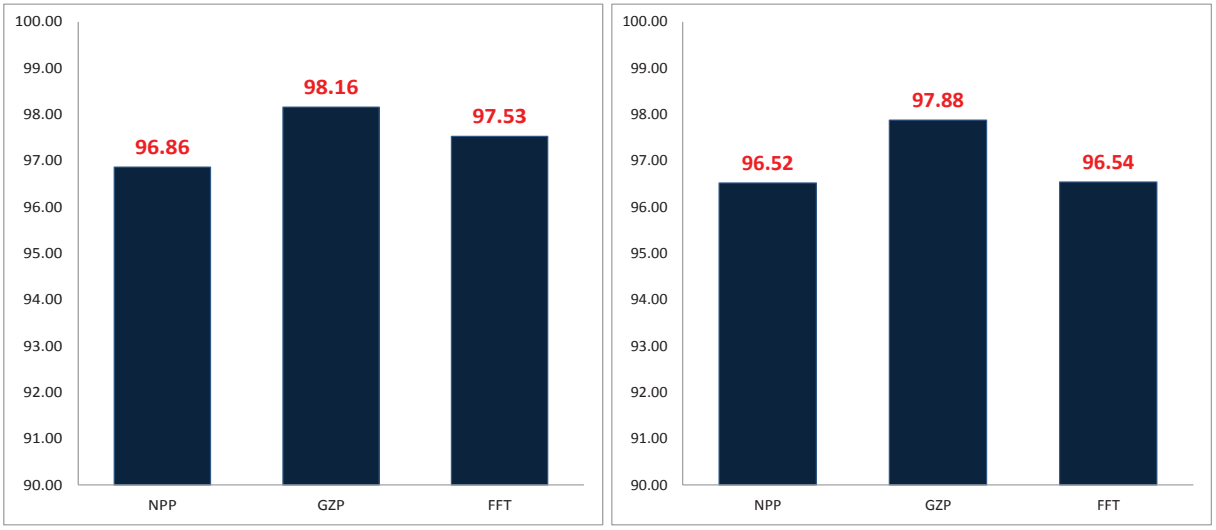


Figure 10
Summary Results with Default Parameters



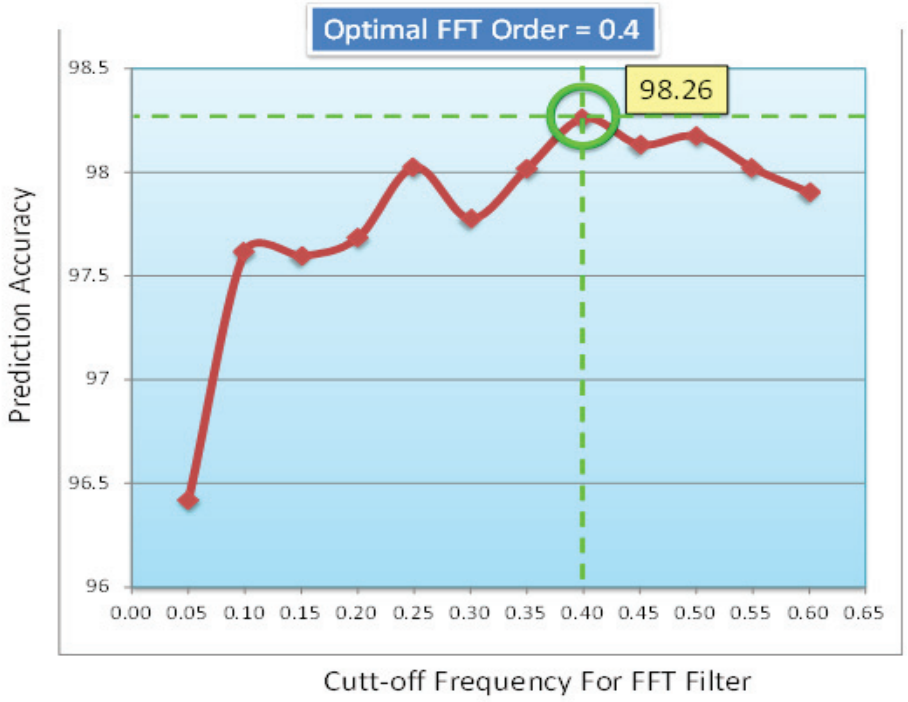
Optimization of FFT

Each model has some parameters that affect its performance and can be fine-tuned to fit a specific application. As we mentioned earlier, we will focus on a new technique, the FFT, that we propose to be compared with other previously tested GZP technique. In stock index prediction, the index signal is very noisy, i.e., it contains significant percentage of high-frequency noise. Therefore, for FFT to be optimized for signal denoising, we need to find the best cut-off frequency for a low-pass FFT-based filter. Table 2 lists all the experimental steps followed to find this best cut-off frequency. Figure 11 plots these experiments to better visualize the relation between the cut-off frequency and the prediction accuracy. As it can be observed, the optimal cut-off frequency for FFT was found to be 0.4 as it is indicated on Figure 11 with an accuracy of 98.26%.

Table 2
FFT Optimization Experiments

Experiment No.	FFT Order	RMSE % Accuracy
1	0.05	96.42
2	0.10	97.62
3	0.15	97.60
4	0.20	97.68
5	0.25	98.02
6	0.30	97.78
7	0.35	98.02
8	0.40	98.26
9	0.45	98.13
10	0.50	98.17
11	0.55	98.02
12	0.60	97.90

Figure 11
FFT Optimization Experiments

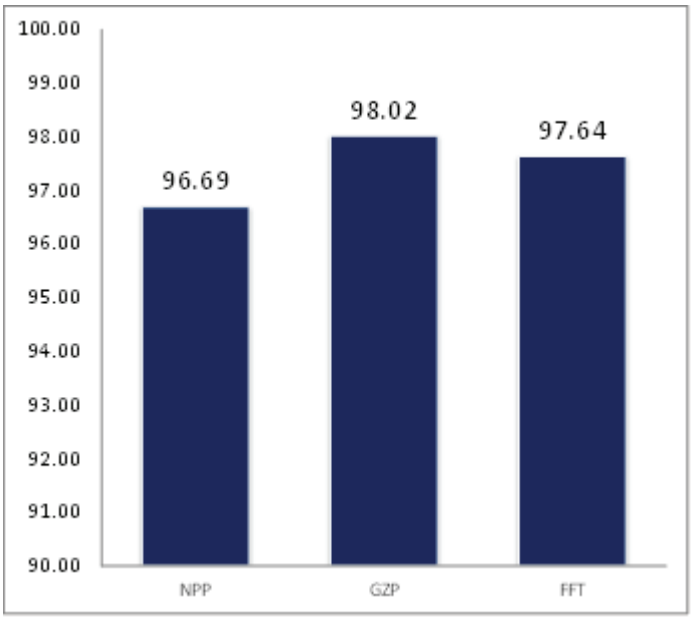


The two filters GZP and FFT were tested after we applied the optimized parameters and there final results are summarized in Table 3. It clearly can be concluded that GZP performed better than FFT in prediction accuracy as shown in Figure 12.

Table 3
Summary of Results after FFT Optimization

Technique	Testing Accuracy		
	DOW30	NASDAQ100	AVERAGE
NPP	96.86	96.52	96.69
GZP	98.16	97.88	98.02
FFT	98.26	97.01	97.64

Figure 12
Summary of Results after FFT Optimization

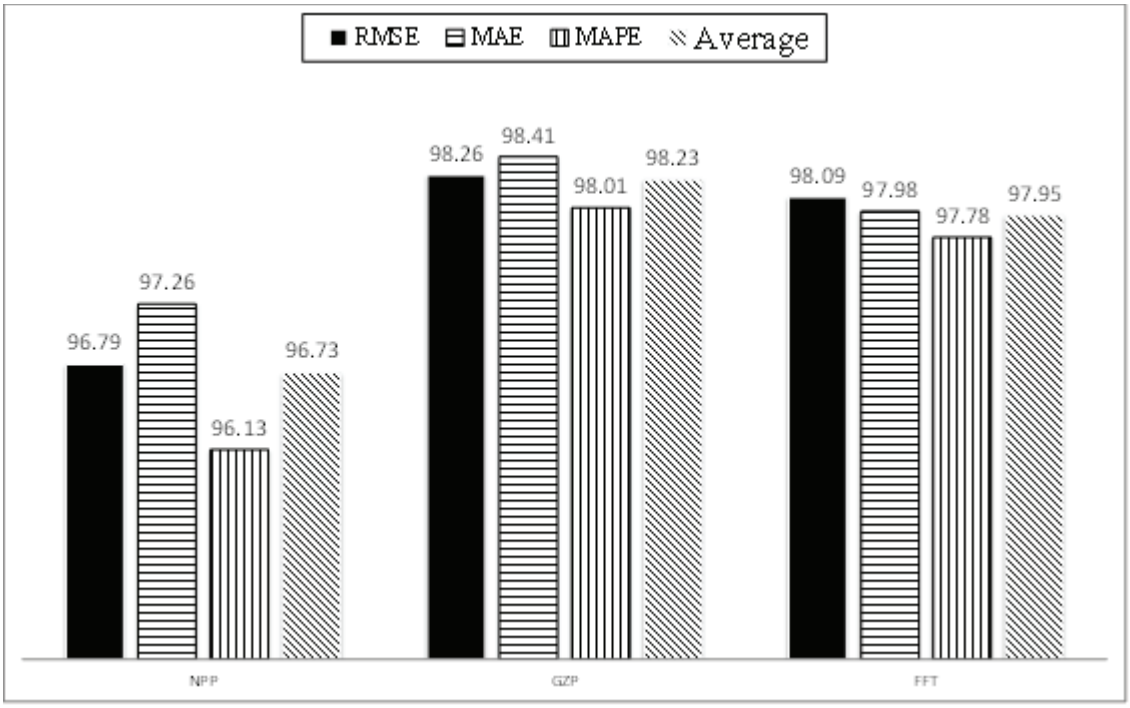


To support our finding using RMSE, we compared it with two other metrics of MAE and MAPE. As it can be seen in summary of this comparison in Table 4 and Figure 13, the average of all three metrics is consistent with our conclusion that GZP performance was better that FFT.

Table 4
Comparison of Three Performance Metrics

Technique	RMSE	MAE	MAPE	Average
NPP	96.79	97.26	96.13	96.73
GZP	98.26	98.41	98.01	98.23
FFT	98.09	97.98	97.78	97.95

Figure 13
Comparison of Three Performance Metrics



CONCLUSIONS

This work aimed at exploring more signal processing techniques in the area of predictive modeling of financial signals. Our previous work (Sharma & Rababaah, 2014) showed that integrating signal processing techniques improve the traditional methods such as Artificial Neural Networks (ANNs). Specifically, we focused on a new signal processing filter of Fast Fourier Transform (FFT) to be compared with Gaussian Zero-Phase (GZP) filter. Three different settings were used in this study as: no signal preprocessing (NPP), Gaussian Zero-Phase Filter (GZP) and Fast Fourier Transform (FFT). DOW30 and NASDAQ100 data sets were used systematically and independently to train and test all proposed techniques integrated with BP-ANN. The three techniques of NPP, GZP and FFT demonstrated the following results: 96.73%, 97.95%, and 98.23% according to RMSE, MAE and MAPE respectively. Our future work include a follow up on signal processing techniques to investigate the potentials of Discrete Wavelet Transform in comparison with all previous models we have studied so far.

Acknowledgement

The first version of this paper was presented at the Allied Academies' International Conference, San Antonio, Texas, October 9-12, 2013.

REFERENCES

Aiken, M. (1999). Using a neural network to forecast inflation. *Industrial Management & Data Systems,* 99(7), 296–301.

Atsalakis, G. S., & K. P. Valavanis (2009). Surveying stock market forecasting techniques – part II: Soft computing methods. *Expert Systems with Applications*, 36(3), 5932–5941.

Constantinou E., R. Georgiades, A. Kazandjian, & G. P. Kouretas (2006). Regime switching and artificial neural network forecasting of the Cyprus Stock Exchange daily returns. *International Journal of Finance and Economics*, 11(4), 371-383.

Dropsy, V. (1996). Do macroeconomic factors help in predicting international equity risk premia? *Journal of Applied Business Research,* 12 (3), 120–132.

Haykin, Simon (1994). *Neural Networks A Comprehensive Foundation.* New Jersey: McMillan Publications.

Hush, D. R. and B. G. Horne (1993). Progress in Supervised Neural Networks. *IEEE Signal Processing Magazine*, January, 8-39.

Kim, K.J. (2006). Artificial neural networks with evolutionary instance selection for financial forecasting. *Expert Systems with Applications*, 30, 519–526.

Krishnaswamy, C. R., E. W. Gilbert, & M. M. Pashley (2000). Neural Network Applications in Finance: A Practical Innovation. *Journal of Financial Practice and Education*, 10(1), 75-84.

Kryzanowski L., M. Galler, & D. Wright (1993). Using artificial neural network to pick stocks, *Financial Analysts Journal,* June/August, 21– 27.

Lam, M. (2004). Neural network techniques for financial performance prediction: integrating fundamental and technical analysis. *Decision Support Systems* 37 (2004) 567– 581.

Lin F.C., & M. Lin (1993), Analysis of financial data using neural nets, *AI Expert* (1993, February) 36– 41.

Manjula, B., S.S.V.N. Sarma, R. Lakshman Naik, & G. Shruthi (2011). Stock Prediction using Neural Network. *International Journal of Advanced Engineering Sciences and Technologies*, 10 (1), 13–18.

Motiwalla L., & M. Wahab (2000). Predictable variation and profitable trading of US equities: A trading simulation using neural networks. *Computer and Operations Research*, 27, 1111–1129.

Nair, B.B., V.P. Mohandas, N.R. Sakthivel, S. Nagendran, A. Nareash, R. Nishanth, S. Ramkumar, D. Manoj Kumar (2010). Application of hybrid adaptive filters for stock market prediction, *Communication and Computational Intelligence (INCOCCI),* 443- 447.

Oliveira, Fagner A. de, Cristiane N. Nobre, & Luis E. Zarate (2013). Applying Artificial Neural Networks to prediction of stock price and improvement of the directional prediction index –Case study of PETR4, Petrobras, Brazil. *Expert Systems with Applications*, 40, 7596–7606.

Park, K. and H. Shin (2013). Stock Price Prediction Based on Hierarchical Structure of Financial Networks. *Neural Information Processing, Lecture Notes in Computer Science*, 8227,456-464

Qing, C., B. Karyl, M. Leggio & J. Schniederjans (2005). A comparison between Fama and French's model and artificial neural networks in predicting the Chinese stock market. *Computers & Operations Research*, 32 (10), 2499–2512.

Qing, C., Mark, E.P., & Karyl B.L. (2011).The three-factor model and artificial neural networks: predicting stock price movement in China. *Annals of Operations Research*, 185(1), 25-44.

Quinquis, A. (2008*). Digital signal processing using MATLAB*. Hoboken, NJ: Wiley.

Rababaah, A. (2009). Energy logic, a novel model for multi-agent multi-modality data fusion, PHD dissertation, College of Engineering, *Technology and Computer Science*, Tennessee State University.

Refenes A.N., M. Azeme-Barac & A.D. Zapranis (1993). Stock ranking: Neural networks vs. multiple linear regressions. *In Proceedings of IEEE ICNN*.

Refenes A.N, A. Zapranis & G. Francies (1994). Stock performance modeling using neural networks: A comparative study with regression models. *Neural Networks*,7(2), 375–388.

Sharda, R. & R.B. Patil (1990). Neural networks as forecasting experts: an empirical test. *In Proceeding of the International Joint Conference on Neural Networks*, 2, 491-494.

Sharma, D.K. & J.A. Alade (1999). Some applications of neural network for business forecasting. *Paradigm*, 2(2), 57-65.

Sharma, Dinesh K. and A. Rababaah (2014). Stock Market Predictive Model Based on Integration of Signal Processing and Artificial Neural Network. *Academy of Information and Management Sciences Journal*, 17(1), 51-70.

Tang, Z. (1991). Time series forecasting using neural networks vs. Box-Jenkins methodology. *Simulation*. 57, 303-310.

Ticknor, J.L. (2013). A Bayesian regularized artificial neural network for stock market forecasting. *Expert Systems with Applications*. 40(14), 5501–5506

Trippi, R. & D. DeSieno (1992). Trading equity index futures with a neural network. *Journal of Portfolio Management*, 27– 33.

Trippi, R. & E. Turban (1996). *Neural Networks in Finance and Investing*. Chicago: Probus Publishing Company.

Wang J.H. & J.Y. Leu (1996). Stock market trend prediction using ARIMA based neural networks. Paper presented at *the IEEE Int. Conf. Neural Networks*, Washington, DC.

White, H. (1998). Economic prediction using neural networks: the case of IBM daily stock returns. *Proceedings of the Second IEEE International Conference on Neural Network*, II451–II458.

Zhu, X., H. Wang, L. Xu & H. Li (2008). Predicting stock index increments by neural networks: The role trading volume under different horizons. *Expert Systems with Applications*, 34, 3043–3054.