Expert Systems With Applications 59 (2016) 195–207

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

An adaptive stock index trading decision support system

Wen-Chyuan Chiang[a], David Enke[b,*], Tong Wu[c], Renzhong Wang[d]

[a] Collins College of Business, The University of Tulsa, 800 South Tucker Drive, Helmerich Hall 118B, Tulsa, OK, 74104, United States
[b] Department of Engineering Management and Systems Engineering, Laboratory for Investment and Financial Engineering, Intelligent Systems Center, Missouri University of Science and Technology, 221 Engineering Management, 600 W. 14th Street, Rolla, MO, 65409-0370, United States
[c] SphereXX.com, 9142 S. Sheridan, Tulsa, OK, 74133, United States
[d] Microsoft Corporation, 205 108th Ave NE #400, Bellevue, WA, 98004, United States

ARTICLE INFO

Article history:
Received 11 February 2016
Revised 19 April 2016
Accepted 20 April 2016
Available online 25 April 2016

Keywords:
Decision support system
Neural networks
Particle swarm optimization
Denoising
Adaptive stock selection
Direction prediction

ABSTRACT

Predicting the direction and movement of stock index prices is difficult, often leading to excessive trading, transaction costs, and missed opportunities. Often traders need a systematic method to not only spot trading opportunities, but to also provide a consistent approach, thereby minimizing trading errors and costs. While mechanical trading systems exist, they are usually designed for a specific stock, stock index, or other financial asset, and are often highly dependent on preselected inputs and model parameters that are expected to continue providing trading information well after the initial training or back-tested model development period. The following research leads to a detailed trading model that provides a more effective and intelligent way for recognizing trading signals and assisting investors with trading decisions by utilizing a system that adapts both the inputs and the prediction model based on the desired output. To illustrate the adaptive approach, multiple inputs and modeling techniques are utilized, including neural networks, particle swarm optimization, and denoising. Simulations with stock indexes illustrate how traders can generate higher returns using the developed adaptive decision support system model. The benefits of adding adaptive and intelligent decision making to forecasts are also discussed.

1. Introduction

Computer technology is becoming increasingly involved in the area of finance. The high computation capability of computers enables one to do complicated analysis utilizing large amounts of historical data. As previously studied (Balvers, Wu, & Gilliland, 2000; Chaudhuri & Wu, 2003; Fliess & Join, 2009; Lee, Lee, & Lee, 2010; Lo & MacKinlay, 1988; Lo & MacKinlay, 2011; Rahman & Saadi, 2008; Semenov, 2008), stock prices do not completely move in a random walk, but are nonlinearly related to historical data, as well as various other fundamental, technical, and macroeconomic factors. In studying these nonlinear relationships, nonlinear regression and neural networks have often been utilized for prediction and clustering (Atsalakis & Valavanis, 2009a; Atsalakis & Valavanis, 2009b; Chavarnakul & Enke, 2009; Enke & Mehdiyev, 2013; Hadavandi, Shavandi, & Ghanbari, 2010; Huang and Tsai, 2009). Systems that use technical analysis, along with expert systems and other forms of computational intelligence for chart pattern recognition, have also been developed (Bogullu, Enke, & Dagli, 2002; Cervelló-Royo, Guijarro, & Michniuk, 2015; Gorgulho, Neves, & Horta, 2011; Lee & Jo, 1999; Leigh, Modani, Purvis, & Roberts, 2002; Lin, Yang, & Yixu, 2011; Thawornwong, Enke, & Dagli, 2001; Yamaguchi, 1989). Nonetheless, such systems are often sensitive to the selection of input variables and parameter settings, and typically do not offer the ability to adapt once trained or modeled for a specific stock, stock index, or other financial asset.

Decision support systems (DSS) have been recognized as a useful tool for helping users plan and make decisions. DSS hides all theories, modeling, and algorithms behind a friendly user interface. Research has been conducted to determine the best DSS framework within various industries (Berner, 2007; Hoogenboom et al., 2004; Kawamoto, Houlihan, Balas, & Lobach, 2005; Klein & Methlie, 2009; Kuo, Chen, & Hwang, 2001; Martinez, Ruan, & Herrera, 2010). However, less work has been done in the financial industry, in particular, with the stock market. Most application of decision support systems for the stock market focus on a simple screening based on multiple criteria (Huang, Hung, & Yen, 2005; Quah, 2008; Sevastjanov & Dymova, 2009), while few models have tried to provide intelligent trading assistance through an investigation and analysis of historical and current data (Albadvi, Chaharsooghi, & Esfahanipour, 2007; Atsalakis & Valavanis, 2009b; Kuo et al., 2001; Lai, Fan, Huang, & Chang, 2009). Before the popularity of neural

* Corresponding author. Tel.: +573-341-4749.
E-mail addresses: wen-chyuan-chiang@utulsa.edu (W.-C. Chiang), enke@mst.edu (D. Enke), wutongquick@gmail.com (T. Wu), rzwumr@gmail.com (R. Wang).

http://dx.doi.org/10.1016/j.eswa.2016.04.025
0957-4174/

networks, most research focused on time series analysis, which is used to extract meaningful statistics and forecast future events based on historical data. Models developed from time series theory that are widely used in the financial area include ARIMA, ARCH, and GARCH, among others (Bollerslev, 1986; Box, Jenkins, & Reinsel, 2013; Engle, 1982; Vejendla & Enke, 2013a, 2013b). However, interest continues to increase for using various forms of computational intelligence, such as artificial neural networks, to forecast individual stock and stock market prices.

Kuo (1998) proposed a DSS for the stock market through integration of fuzzy neural networks and fuzzy Delphi, in which both quantitative and qualitative data were studied. The neural network used develops a relationship between independent values and the dependent variable, such as the tendency of the stock market to rise or fall. The authors used a fuzzy neural network to initialize the weights to overcome the backpropagation learning algorithms inability to respond efficiently in a reasonable time. There are several other approaches developed that utilize a hybrid model of neural networks and fuzzy logic. For example, Atsalakis and Valavanis (2009a) developed a neuro-fuzzy adaptive control system that uses an adaptive neuro-fuzzy inference system controller to direct the stock market process model. Cheng, Chen, Teoh, and Chiang (2008) proposed a fuzzy time-series model that improves the adaptive expectation model technique, allowing modification of forecasts based on recent prices. Hadavandi et al. (2010) combine genetic fuzzy systems (GFS) and self-organizing map (SOM) neural networks for building a stock price forecasting expert system. The model involves three stages: (1) stepwise regression analysis to choose the key variables that are to be considered in the model; (2) categorize the data set into k clusters using SOM; and (3) feed all clusters into independent GFS models with the ability of rule base extraction and database tuning.

Samaras and Matsatsinis (2004) describe a multi-criteria decision support system that aims at presenting an evaluation of stocks on the Athens Stock Exchange on the basis of fundamental analysis. Tsaih, Hsu, and Lai (1998) developed a hybrid AI system for forecasting the S&P 500 stock index. They first applied a rule-based system for providing training samples. A reasoning neural network was then employed to generate trading triggers. Tsai and Hsiao (2010) combine multiple feature selection methods to find more representative variables for better prediction. In their research, principle component analysis, genetic algorithms, and a decision tree (CART) were used. Using representative variables that had been filtered, a backpropagation neural network was then developed to predict the stock price. Amornwattana, Enke, and Dagli (2007) illustrate the use of a hybrid options pricing model that uses multiple neural network models, along with a decision support system structure, to identify the correct stock options trading strategy based on direction and volatility forecast, as well as return forecasting (Enke & Amornwattana, 2008). The use of technical analysis, along with employing a neural network as a decision maker, has also been used for stock trading (Chavarnakul & Enke, 2009; Thawornwong, Enke, & Dagli, 2003)

Previous research illustrated that neural networks can be an effective tool for stock market prediction. Additional studies have also been done to consider the variables that are selected as inputs to the neural network. Chang, Liu, Lin, Fan, and Ng (2009) developed a three-stage system, CBDWNN, for stock trading prediction. These stages include: (1) screening out the potential stocks and the important influential factors; (2) using a backpropagation neural network to predict the buy/sell points of stock price; and (3) adopting case-based dynamic windows to further improve the forecasting results. Thawornwong and Enke (2003) performed an adaptive selection of financial and economic variables for artificial neural networks, utilizing methods for finding input data with the most information content.

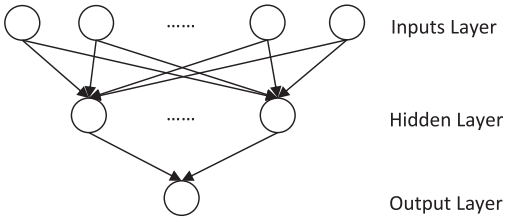

Fig. 1. Artificial neural network structure.

However, most of the previous research tried to find one model that could be applied to all stocks. When it comes to the entire stock market, it is questionable whether we can find a general model to fit every stock or index. A model fitting one security well may perform poorly for another security. Such a finding will be confirmed later in this paper. To solve this problem, an adaptive stock index trading support system has been developed which utilizes an artificial neural network for predicting the future stock index price movement direction, adaptively creating a new model for every single stock index, rather than applying a generalized model. This information is then used to help users make better trading decisions. A particle swarm algorithm has been integrated into the system to overcome the disadvantages of the backpropagation algorithm, a common learning algorithm used for training neural networks. Furthermore, most researchers have focused on predicting the stock index price. However, it is believed that if one can predict the future direction of stock index price movement, the prediction result could be more helpful for making profit. Therefore, a system has been developed for forecasting the price movement direction instead of price level itself. Research has shown that such a forecast can often result in more accurate trading results (Enke & Thawornwong, 2005; Thawornwong & Enke, 2003).

The remainder of the paper begins with a review of previous research in the areas of neural networks and particle swarm optimization. This is followed by an overview of the architecture and methodology of the proposed trading system. Finally, numerous simulation results will be demonstrated, both illustrating how the system works and evaluating how well the system performs.

2. Background

2.1. Artificial neural network

Artificial neural networks (ANN) are inspired by the functional and structural aspects of biological neural networks. ANNs have become an important tool in statistical data modeling, especially for discovering the non-linear relationship between an input and output dataset. An ANN consists of groups of neurons that are interconnected. The architecture usually includes one input layer, one or more hidden layers, and an output layer. The ANN updates its interconnection weights toward optimization by processing the inflow of data and the computation of an output. The structure of the ANN is illustrated in Fig. 1.

The backpropagation training algorithm is a widely used algorithm in artificial neural network modeling. The backpropagation algorithm is a generalization of least mean-squared error measurement that modifies network weights to minimize the mean-squared error between the desired and actual output of the network. Backpropagation learning was first introduced by Bryson and Ho (1969) and further developed by Paul Werbos, David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams, which later helped this algorithm gain better recognition (Russell & Norvig, 2003). Backpropagation learning requires a neural network activation function that is differentiable. However, backpropagation does have some limitations, in particular, the convergence is slow and

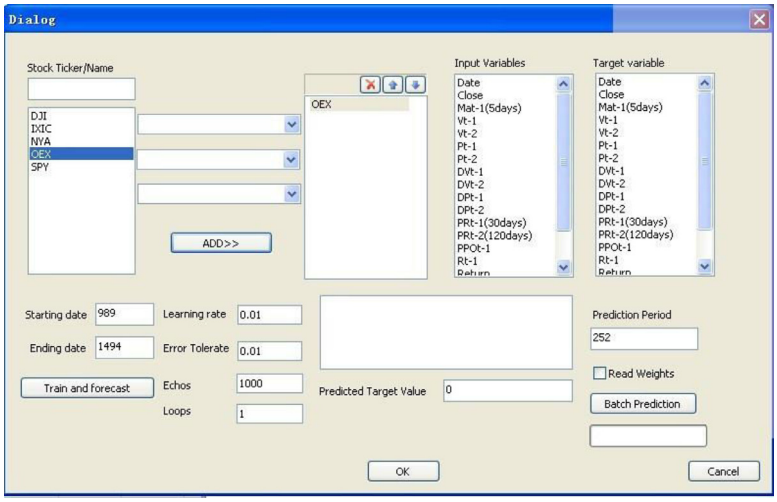

Fig. 2. DSS user interface.

not guaranteed during the learning phase, which may result in converge to a local optimal.

2.2. Particle swarm optimization

Particle swarm optimization (PSO) is a computation method that optimizes a problem by improving a candidate solution iteratively based on a fitness measurement function. Each particle's movement is influenced by a local best position in the search space, eventually moving toward the global best position. Each particle has its own position and velocity, with both updated throughout the searching progress. The optimization process will be stopped when the optimization goal is achieved, or the maximum number of iterations is reached. Satisfaction is not always guaranteed.

PSO can be used together with artificial neural networks for initializing the weights (Nikelshpur & Tappert, 2013; Shi, 2001) and training the network (Gudise & Venayagamoorthy, 2003; Zhang, Zhang, Lok, & Lyu, 2007). In the former case, PSO is used to pretrain a neural network to arrive at an optimum initial set of network weights. In the latter case, each particle represents a weight vector, and the PSO velocity and position update equations are used to adjust weights and biases, instead of gradient information. The training set is then used to calculate the fitness (mean square error) of a particle in the feed forward passes for all training patterns.

3. System overview

An adaptive stock direction prediction system has been developed in this paper. The system utilizes an artificial neural network to forecast the movement of future stock prices. Direction information is then used as trading signal in order to assist users in making trading decisions. The output from the neural network will be standardized between zero and one: output from 0.5 to 1 will be categorized to one, indicating that the stock index price is going up; otherwise, output from 0 to less than 0.5 will be categorized to zero, indicating that the stock index price is going down. The system will give a trading suggestion to buy or hold the stock index when the output is one, or to sell when the output is zero. Different from other DSS that are used for trading stocks or stock indexes, the proposed system will adaptively adjust the model to optimally fit each stock index. This is different from traditional stock trading systems that require building a generalized model that often suffers in performance as one begins to move away from the initially defined input-output training pairs. Alternatively, the proposed system also provides an interface for users to manually specify models and parameters to conduct the prediction process. Fig. 2 provides an illustration of the stock index selection user interface.

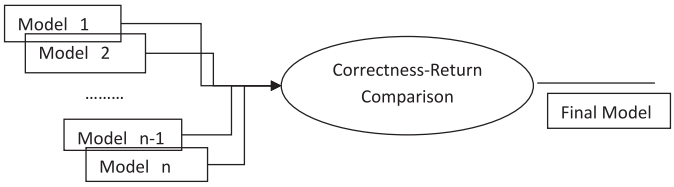

Fig. 3. Adaptive model selection.

4. Forecasting methodology

4.1. Adaptive methodology overview

The accuracy of the prediction and network performance are greatly impacted by the selection of the inputs for the neural network. Initially, a general model was developed to describe the relationship among stock index variables. Principal component analysis, genetic algorithms, and decision trees are widely used for filtering representative variables. However, for this research it will be very difficult to determine the best fit model when considering multiple stock indices, such that there will not be enough evidence to prove that the selected model will fit other stocks indices well, even with numerous experiments and theoretical analysis. After a number of experiments, we believe that optimal results are better achieved when we adopt different models specifically for each stock index. This is unique compared to other traditional research and trading systems where modelers attempt to optimize system performance based on a specific timeframe and desired output. For our modeling, an adaptive prediction system has been designed to determine the model heuristically. With the adaptive approach, it is no longer necessary to determine the best representative variables and best theory models for stock indices. The system will attempt to fit a best model automatically. This is a more empirical process. Therefore, in the proposed system, the final prediction model will vary for each individual stock index. Inputs will be selected from a group of potential variables provided by experts. Some network parameters may also vary in different cases, such as the learning rate, maximum epochs, or the learning time frame. The software determines the selection of variables as shown in Fig. 3.

Since stocks have different characteristics, it is believed that the proposed system will lead to a better fitting model. However, the adaptive variables selection strategy introduces a problem as well. With N potential variables, the different possible combination of inputs for the neural network will reach 2^N-1, which can be a huge number with increasing N. This will lead to large time consumption and computation resource, which might make this approach infeasible for real world use. To solve this problem, one possible method is to decrease the size of the neural network used. In particular, reducing the number of input neurons will dramatically reduce the time consumption. Some research has indicated that less complicated networks are good enough and often lead to better performance (Blum & Rivest, 1992). Bebis and Georgiopoulos (1994) have discussed that small and simple networks are better. Therefore, we believe that the network size can be reduced without losing prediction capability. Based on a number of experiments we conducted in our research, we find that most of the best performing models have five variables or less. As a result, we set the maximum number of the inputs for the neural network at five, such that the number of combination for inputs can be reduce to the following,

$$C = C_N^1 + C_N^2 + C_N^3 + C_N^4 + C_N^5 \tag{1}$$

where N is the number of potential variables. For example, when N is equal to 10, the total number of possible models will be 637. While still a large number, this becomes feasible with high-speed computers.

The system is designed to use two years data to train the neural network, and one year of data to evaluate the model. Such a timeframe should consider most market cycles. Different models will run iteratively. Direction prediction correctness (DPC) and returns from simulations in the evaluating period will be calculated and used as tools for determining the final model for the next process. DPC is obtained by calculating the percentage of correct predictions during the testing period. Usually the model with a higher DPC will give a higher return, but this is not guaranteed. As we try to obtain a higher return during real market transactions, the model with the highest return will be used as the final adoption model for the next prediction.

In the learning process we adopt a moving window approach for selecting training data, which helps reflect the most recent relationship between the input and output data. As the predicting date moves forward, the starting and ending date of training data increases at the same rate. The neural network model was developed using the C++ programming language. This software was built with a friendly user-interface such that users can easily modify various parameters to satisfy specific needs. The backpropagation learning algorithm is used for the neural network modeling in order to minimize the mean square error between the estimated and real output.

4.2. System architecture

As illustrated in Fig. 4, the system consists of four basic modules: the data pre-processing module, the neural network initialization module, the neural networking training module, and the output module.

4.2.1. Data preprocessor

The first module in the system architecture is the data preprocessor, which normalizes the input data between the range of 0 and 1 (discussed further in Section 5). Data normalization is required by the characteristic of the neural network. There are various methods to normalize data, such as standard score, student's t-statistic, and student residual. In our research, the normalization is processed as:

$$V_n = \frac{V_i - V_{min}}{V_{max} - V_{min}} \tag{2}$$

where V_i is the raw data, while V_{min} and V_{max} are the minimum value and maximum value among the data, respectively. V_n represents the value after normalization.

4.2.2. Particle swarm in neural network initialization module

The neural network initialization module is designed to overcome the disadvantage of the backpropagation algorithm, which lacks the ability to respond in a limited time and may possibly converge to a local optimal (Lippmann, 1987). In order to improve the efficiency, the particle swarm algorithm has been adopted in this module. The particle swarm algorithm has a larger searching space, which will help the neural network avoid the local optimal. The output of the particle swarm optimization will be used as the initial weights for the neural network. Since our proposed system has a large amount of computation involved, this module will be helpful in reducing time consumption.

4.2.3. Neural network training module

The neural network training module consists of a 3-layer feedforward network with one hidden layer. There has been extensive research for determining the size of the hidden layer. A common rule of thumb is that the number of nodes required in the hidden layer should be less than twice the number of inputs (Berry & Linoff, 1997; Swingler, 1996). In the proposed system, the number of nodes in the hidden layer is set equal to 2N–1, where N is the number of inputs. The number of nodes in the output layer is 1, the stock movement direction. As mentioned earlier, the backpropagation algorithm has been adopted as the learning algorithm. This module is designed to conduct a training process and prediction.

4.2.4. Output module

The trading signal will be presented in the output module. There are three different scenarios: buy, hold, and sell. If traders are not holding any stock indexes when the direction prediction is greater than 0.5 (buy signal), they will buy the stock index, otherwise, the trader will do nothing. If the trader is holding the stock index when the direction prediction is greater than 0.5, they will hold the stock index, otherwise, when the direction prediction is less than 0.5 (sell signal), the stock index will be sold.

5. Independent variables

Analysis starts from the selection of potential variables, which will be used in the model as inputs. Economic variables are usually released monthly and quarterly and typically stay the same for a specific period. As a result, economic variables are excluded from our daily forecast. For daily prediction, the most important parameters are the closing price and volume of the stock index for the last trading day. These two pieces of information contain the most recent stock index trading information. In addition to the basic trading data of stock indices, more technical measurements are also studied, as discussed below.

Moving averages are a frequently used measurement in technical analysis, indicating short and long-term direction trends based on the period in which the average is calculated. The moving average of the previous five trading days' closing price is considered in the proposed system, providing a short-term trend analysis. This shorter period is better suited for daily trading. For the analysis, a simple Moving Average (MA), which is the non-weighted mean of the previous n data points, is calculated as:

$$MA_{t-1} = \frac{\sum_1^n P_{t-n}}{n} \tag{3}$$

where P_{t-n} is the stock price at time $t - n$.

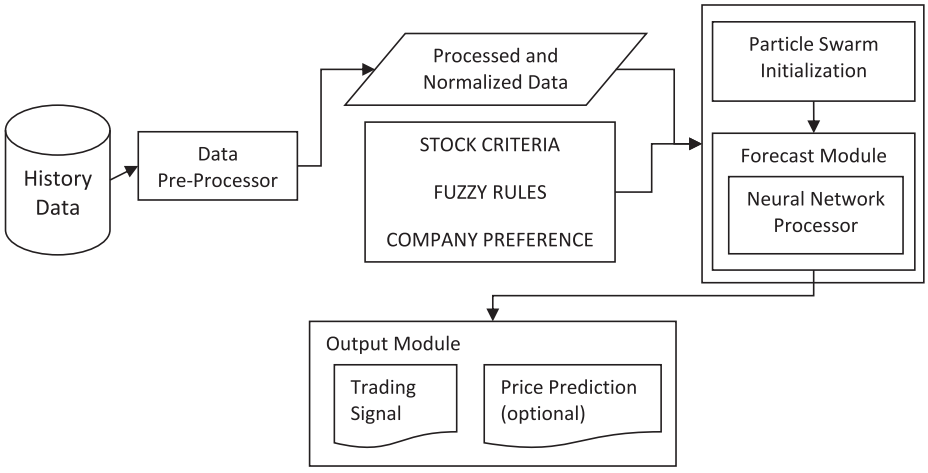

Fig. 4. System architecture.

Support and resistance indicators are other important parameters that are introduced into the model for this research. Support and resistance are concepts in technical analysis that measure the tendency of stocks and indexes to have support from going lower in price, or resistance from going higher in price. For this research, a Moving Price level Percentage (MPP) measure is used as an indicator for support and resistance. To obtain the MPP for the last K days, we initially calculate the increment between the current price and the minimum price for the last K days. Dividing the increment by the difference of the maximum and minimum stock index price over the K trading days gives the MPP. A larger MPP suggests a higher level of price, and potential bigger level of resistance (smaller value otherwise).

$$MPP = \frac{P - P_{min}}{P_{max} - P_{min}} \tag{4}$$

The Percentage Price Oscillator (PPO) is a technical momentum indicator showing the relationship between the short-term and long-term averages. In a traditional view, a buy signal is issued when PPO measures above 0, while a sell signal is issued when PPO is below 0. To calculate PPO, both a fast and slow exponential moving average is required. For the modeling, the PPO is calculated as:

$$PPO = \frac{9\ Day\ EMA - 26\ Day\ EMA}{26\ Day\ EMA} \tag{5}$$

where *EMA* stands for the Exponential Moving Average. Exponential moving averages provide additional weight to more recent data, with the price impact on the moving average declining exponentially as prices are older relative to each other and to the current price.

Closing prices and volumes were collected from finance.yahoo.com. The time series used was from January 29, 2004 to June 24, 2011. The various price and volume changes, as well as the chosen technical indicators, were calculated in Excel based on the collected data. In total, 12 variables are selected to form a potential parameter pool as shown in Table 1. Different combinations of inputs are adopted during the modeling trials, with the final model based on the prediction performance.

6. Forecast evaluation

The proposed system trained the neural network with two years of daily data from January 2008 through December 2009. A twelve-month sample from January 2010 through December 2010 was retained to access the prediction performance of the neural network.

Table 1
Potential variables.

Variable name	Description
$\mathbf{DP_{t-1}}$	The change of price from time t–1 and t–2
$\mathbf{DP_{t-2}}$	The change of price from time t–2 and t–3
$\mathbf{DV_{t-1}}$	The change of volume from time t–1 and t–2
$\mathbf{DV_{t-2}}$	The change of volume from time t–2 and t–3
$\mathbf{MA_{t-1}}$ **(5 days)**	The moving average of the last 5 days at time t–1
$\mathbf{MA_{t-1}}$ **(10 days)**	The moving average of the last 10 days at time t–1
$\mathbf{MA_{t-1}}$ **(30 days)**	The moving average of the last 30 days at time t–1
$\mathbf{MPP_{t-1}}$ **(30 days)**	The moving price level in the last 30 days at time t–1
$\mathbf{MPP_{t-1}}$ **(120 days)**	The moving price level in the last 120 days at time t–1
$\mathbf{PPO_{t-1}}$	The percentage price oscillator at time t–1

6.1. Evaluation description

In order to evaluate the performance of the neural network for generating trading signals, a Direction Correctness Percentage (DCP) was calculated. The DCP measures what percentage of the direction predictions are inline with the real direction changes in the historical data. If the DCP is greater than 50%, it may be possible to profit from the system.

Simulations were conducted to study how much potential benefit there might be from using the proposed system. Hence, the returns from transactions that follow the trading signal output from the system are calculated. In the simulation, the software runs once for each trading day from January 2010 to December 2010, yielding 252 returns. These returns are then compared with those obtained using a buy-and-hold strategy. Returns from the trading support system were observed to be significant higher than their buy-and-hold counterpart. This suggests that the trading system has successfully helped in making trading decisions (example details and results are provided next).

6.2. Model selection simulation

To illustrate how the proposed system adaptively develops the final model, the simulation results from testing the SPY (S&P 500 Index ETF) and IXIC (NASDAQ Composite Index) are used as examples. Tables 2 and 3 present part of the individual simulation results. The system tried numerous different models for each index. For each scenario, direction predictions were conducted over the 252 trading days in year 2010 (typically 365 days minus weekends and holidays). In those simulations, trading was conducted in accordance with the signals suggested by the proposed system. An accumulative return over the one-year period was then calculated.

Table 2
Selected simulation return for SPY.

SPY	Training data Jan 02, 2008 to Dec 31, 2009		Training data Mar 02, 2008 to Dec 31, 2009	
Neural network inputs	DCP	Return	DCP	Return
DPt–1, DPt–2, MPPt–1(30days), MPPt–2(120days), PPOt–1	56.75%	–0.36%	55.56%	–4.83%
Vt–1, Vt–2, Pt–1, Pt–2, MPPt–1(30days)	54.37%	9.81%	55.56%	2.39%
DVt–1, DVt–2, DPt–1, DPt–2, MPPt–1(30days)	53.97%	–0.62%	56.75%	2.25%
DVt–1, DPt–1, MPPt–1(30days)	55.56%	7.20%	53.97%	–8.76%
Vt–1, Pt–1, MPPt–1(30days)	53.17%	8.55%	54.37%	6.74%
Pt–1, Pt–2, MPPt–1(30days)	57.14%	13.67%	57.94%	11.00%
Pt–1, Pt–2, MPPt–1(30days), PPOt–1	54.76%	1.26%	56.75%	6.52%
MAt–1(5days), Pt–1, Pt–2, MPPt–1(30days)	57.94%	11.76%	55.56%	3.80%
MAt–1(5days), DPt–1, DPt–2, MPPt–1(30days)	57.54%	**14.70%**	57.14%	7.22%
MAt–1(5days), DPt–1, DPt–2	56.75%	12.04%	58.73%	11.05%
MAt–1(5days), DPt–1, DPt–2, PPOt–1	57.54%	7.41%	57.54%	5.49%
MAt–1(5days), DPt–1, DPt–2, MPPt–1(30days), MPPt–2(120days)	57.14%	–4.35%	58.33%	0.58%

Table 3
Selected simulation return for IXIC.

IXIC	Training data Jan 02, 2008 to Dec 31, 2009		Training data Mar 02, 2008 to Dec 31, 2009	
Neural network inputs	DCP	Return	DCP	Return
DPt–1, DPt–2, MPPt–1(30days), MPPt–2(120days), PPOt–1	55.16%	–6.21%	53.57%	–2.48%
Vt–1, Vt–2, Pt–1, Pt–2, MPPt–1(30days)	53.87%	7.25%	50.00%	–0.36%
DVt–1, DVt–2, DPt–1, DPt–2, MPPt–1(30days)	55.95%	8.72%	53.97%	1.74%
DVt-1, DPt–1, MPPt–1(30days)	57.54%	18.55%	58.33%	13.08%
Vt–1, Pt–1, MPPt–1(30days)	54.37%	8.64%	54.37%	8.83%
Pt–1, Pt–2, MPPt–1(30days)	55.56%	11.62%	52.78%	5.97%
Pt–1, Pt–2, MPPt–1(30days), PPOt–1	53.57%	1.31%	54.37%	11.77%
MAt–1(5days), Pt–1, Pt–2, MPPt–1(30days)	58.33%	18.21%	53.17%	12.31%
MAt–1(5days), DPt–1, DPt–2, MPPt–1(30days)	59.13%	16.67%	55.56%	7.54%
MAt–1(5days), DPt–1, DPt–2	59.52%	**20.24%**	56.35%	5.72%
MAt–1(5days), DPt–1, DPt–2, PPOt–1	60.32%	17.04%	57.14%	3.45%
MAt–1(5days), DPt–1, DPt–2, MPPt–1(30days), MPPt–2(120days)	56.35%	4.84%	55.16%	1.39%

Table 4
List of indices.

Index	Description	**Index**	Description
IXIC	NASDAQ Composite	**XLF**	Select Sector Financial
SPY	SP500 Index (large cap index)	**XLY**	Select Sector Consumer Discretionary
QQQ	Nasdaq 100 (large / medium tech stocks)	**XLE**	Select Sector Energy
MDY	S&P Midcap 400 (mid cap index)	**XLV**	Select Sector Health Care
IWM	iShares Russell 2000 index (small cap)	**XLK**	Select Sector Technology
FXI	iShares China	**SMH**	Semiconductor Holders
EFA	iShares EAFE Index, Europe	**OIH**	Oil Service Holders
EEM	iShares Emerging Markets Index	**FTSE**	FTSE 100 Index (UK)
HSI	Heng Seng Index (Hong Kong)	**ATX**	Austrian Traded Index in EUR
TWII	TSEC weighted index (Taiwan)	**FCHI**	CAC 40 (French)
BSESN	Bombay Stock Exchange (India)	**AEX**	Amsterdam Exchange index

It can be clearly observed that the combination of parameters significantly affects the performance of prediction. One can also see that the best prediction result for each index was achieved by different models.

As shown in Tables 2 and 3, the system is able to determine the best fitting model based on the highest return. For the S&P 500 Index (traded using the SPY exchanged traded fund), the inputs selected for the final model are MA_{t-1}(5 days), DP_{t-1}, DP_{t-2}, PR_{t-1}(30 days), and the training period was for the two years starting from January 2, 2008. The direction prediction correctness reaches 57.54%, with return of 14.70%. For index IXIC, the inputs selected for the final model are MA_{t-1} (5 days), DP_{t-1}, and DP_{t-2}, and training period is the two years starting from January 02, 2008. The DCP and return are 59.52% and 20.24%, respectively. Such results illustrate that a single model may not fit every stock index. Another interesting observation is that higher prediction accuracy does not always guarantee higher return, further highlighting why return is used as the criteria for determining the final predictive model.

6.3. Performance evaluating

More simulations were conducted using 20 indices (or ETFs representing various indices). These indices/ETFs are a good mix of large, mid, and small-cap stocks, along with a selection of international and domestic indices. The selected indices are listed in the Table 4.

As described in Section 6.1, the simulations were conducted for year 2010, which has 252 trading days. In order to access how well the system helps to predict stock index price movement (measured

Table 5
Simulation result for U.S. indices.

U.S. indices	DCP	Return (Signal)	Return (buy-hold)	ΔReturn	Improvement in %
IXIC	62.20%	**24.75%**	14.92%	9.83%	65.88%
SPY	61.11%	**20.34%**	13.69%	6.65%	48.57%
QQQ	58.33%	**29.09%**	19.29%	9.80%	50.80%
MDY	56.35%	**35.14%**	24.52%	10.62%	43.31%
IWM	53.97%	**31.48%**	25.60%	5.88%	22.96%

Table 6
Simulation result for selected sectors.

Selected sectors	DCP	Return (Signal)	Return (buy-hold)	ΔReturn	Improvement in %
XLE	53.17%	**34.01%**	20.75%	13.26%	63.90%
XLF	50.00%	**22.41%**	13.45%	8.96%	66.61%
XLK	59.13%	**20.26%**	11.15%	9.11%	81.70%
XLV	54.76%	**16.48%**	2.49%	13.99%	562%
XLY	54.37%	**27.50%**	24.98%	2.52%	10.08%

Table 7
Simulation result for iShares indices.

iShares indices	DCP	Return (Signal)	Return (buy-hold)	ΔReturn	Improvement in %
EEM	55.56%	**36.48%**	17.07%	19.41%	114%
EFA	54.76%	**30.09%**	8.13%	21.96%	270%
FXI	52.78%	**28.07%**	4.54%	23.53%	518%

Table 8
Simulation result for commodity indices.

Commodity	DCP	Return (Signal)	Return (buy-hold)	ΔReturn	Improvement in %
OIH	53.57%	**24.17%**	21.71%	2.46%	11.33%
SMH	52.78%	**29.66%**	18.74%	10.92%	58.27%

Table 9
Simulation result for international indices.

International indices	DCP	Return (Signal)	Return (buy-hold)	ΔReturn	Improvement in %
FTSE	51.98%	**20.15%**	12.57%	7.58%	60.30%
ATX	52.62%	**25.98%**	21.75%	4.23%	19.44%
FCHI	54.37%	**23.11%**	4.35%	18.76%	431.2%
AEX	54.37%	**28.64%**	10.53%	18.12%	172.1%
HSI	55.95%	**20.69%**	6.85%	13.84%	202.0%
TWII	57.94%	**29.13%**	15.31%	13.82%	90.26%
BSESN	55.95%	**27.58%**	13.24%	14.34%	108.3%

by prediction accuracy), as well as how much profit can be obtained (measured by simulated returns), a comparison study was conducted. The return of each security trade following the trading signals outputted from the system was compared with those obtained by following a buy-and-hold strategy (where you simply buy the security and hold it for the duration of the holding period). The simulation results are shown in Tables 5–9.

The improvement of the proposed system on U.S. indices looks quite consistent. The average prediction accuracy and improvement were 58.39% and 46.30%, respectively.

The performance of these selected sectors is similar to those of U.S. indices. The proposed system increased the returns by about 10%.

For the iShares indices, the proposed system performed very well. The average return reached 31.55%, up from about 10%.

The improvement on return for the OIH oil index was mild. Compared to other stocks indexes, the proposed system provided less improvement in this area, possibly due to the fact that commodity and macroeconomic indicators/values were not used in the proposed model.

Seven international indexes were also simulated. As shown in Table 9, the average prediction accuracy was 54.74%. The average improvement on return was 154.80%.

Compared to the simulation results from other research, one may observe that the prediction accuracy as shown in the simulation results above is relatively low. Two main reasons account for the relative low prediction accuracy. First, unlike monthly or quarterly forecasting, the daily movements are usually small while carrying more noise. It is hard for the system to catch the small changes. Nonetheless, even direction predictions near 55% are considered good enough to trade and generate profit. Secondly, the return, instead of the prediction accuracy, is used as the criteria in selecting the final model. The highest prediction accuracy does not guarantee the highest return.

The returns from the trades that follow the signal outputs from the proposed system outperform those obtained from buy-and-hold strategy by 136.53%, on average. Particularly, when the buy-and-hold return is low, significant higher returns can be obtained by the proposed system. For example, the XLV return is improved from 2.49% to 16.48%; the FXI return is improved from 4.54% to

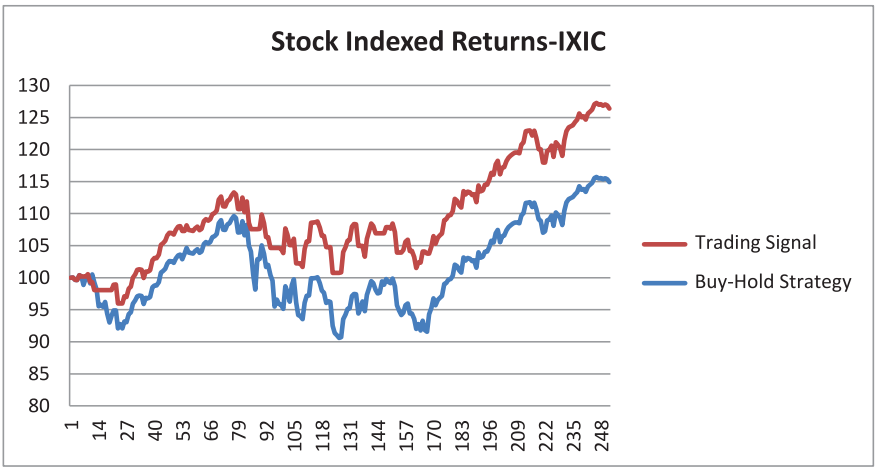

Fig. 5. Trading simulation result over time for IXIC.

28.07%. Investigation of the simulation results reveals that the system sometimes fails to recognize small price changes, but predicts the major changes in price movement. Based on the simulation results, the proposed system appears to be able to provide significantly higher return compared to the buy-and-hold strategy.

A trading example using IXIC index is shown in Fig. 5. In the figure, the returns that follow two strategies, i.e., the traditional buy-and-hold strategy and the trading signals outputted from the proposed system, are compared over the 252 trading days. As shown in the figure, the two lines started from the same point, 100, and then diverged over time. It can be observed that the system helped avoid several sharp downturns during the simulation, which occurred at the 12th, 78th, 89th, 122nd, 150th, and 160th days. The system failed to response to several smaller decreases. However, one benefit of not trading on smaller decreases is that the number of transactions is reduced, resulting in less transaction and trading fees. At the end of the 252 trading days, a significantly higher return (indexed at 126) was achieved with the assistance of the proposed system than could be achieved utilizing the buy-and-hold strategy (indexed at 114). Nonetheless, for this simulation there were still 50 transactions that occurred during the 252 trading days, including 25 buy and 25 sell signals. This suggest that even without small movement trades, excessive trading transition cost may still eliminate the benefits of the trading system in those cases where the additional return generated by the model was not significant.

7. Denoising approach

Given that excessive trading could eliminate potential profit opportunities, it is worth considering better ways to deal with stock market volatility. While the prices of stock indices fluctuate daily, the price movements do not always reflect the value of the indices or their associated long-term trends. This is due in part to various noises in the stock market, such as those caused by speculation and program trading, among others. In addition, selling stocks at every downturn will result in numerous transactions, generating higher transaction fees. If overall transaction fees become too high, the investment may no longer be profitable.

As a result of the noise in the stock market, there exist meaningless movements during a short-term period. From the previous simulations, it was found that the neural network has difficulty catching-up with the small changes, such that the small fluctuations may confuse the neural network when it is trying to discover the underlying relationship between the variables. Therefore, as an additional test, the data was smooth using a wavelet transformation before providing it to the neural network. Wavelets are considered a very useful tool for extracting information from various types of data. In our experiment, wavelet decomposition was used to preprocess stock prices and volume before providing the data to the neural network (using the system described earlier). The model for a noisy signal can be described as:

$$s(n) = f(n) + \sigma e(n) \tag{6}$$

where s(n)=noisy signal, f(n)=principal content, σ =volatility, and e(n)=the noise. The denoising process is designed to filter out the noise and recover signal *f*. Signal *f* is considered as useful data with less noise. Fig. 6 illustrates the transformation of the original data into de-noised data for the SPY ETF.

From Fig. 6, one can observe that the de-noised price and volume movement are smoothed. The trends of these data are thus clearer. Using the same system that was developed before, we conducted a simulation of trading on the SPY index using the de-noised data. The result is shown in Fig. 7.

From Fig. 7, one can observe that a significant higher return can be achieved by following the trading signals generated by the proposed trading system with denoising. The return that follows the trading signals reaches 50% in 252 trading days, improving from around 20% without denoising. The return achieved by the buy-and-hold strategy was around 13%. From Fig. 7, one can also observe that the trading system helped one avoid several big downturns that occurred at days 12, 78, 144, and 210. Another important finding is that denoising data helped to dramatically reduce the number of transactions. The system can be improved by ignoring small fluctuations but remain effective in detecting larger trends. As a result, the transaction fees can be reduced, thus increasing the returns. While a return improvement from 13% to 20% may get washed out with numerous trades and transition fees, a return improvement from 13% to 50% with less trading and lower overall transaction fees has a much better chance of generating profit for the trader.

Another example is shown in Fig. 8, where similar results are observed. The system successfully avoided the big downturns at days 30 and 210. The number of transactions was 14 over 252 trading days. The return following the trading signal reaches 42%, which is an improvement from 27.58% without denoising data. In contrast, the return using the buy-and-hold strategy is only around 13%.

Additional simulations were conducted on the indices as summarized in Table 10. The denoising approach further improved the system performance by 87.80% from the previous approach that did not apply denoising. The number of transactions was on average

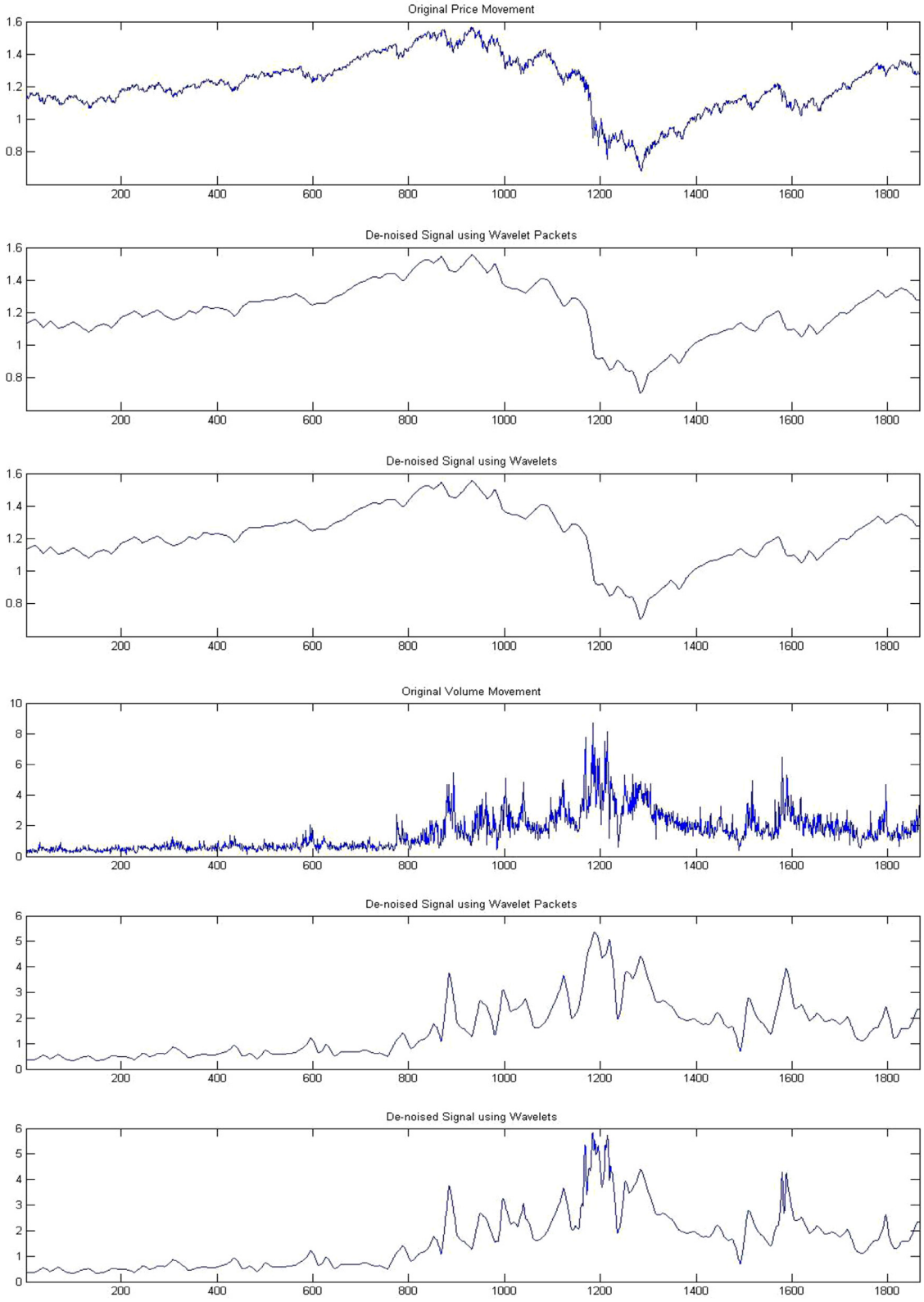

Fig. 6. Transformation of the original data into denoised data for the SPY ETF.

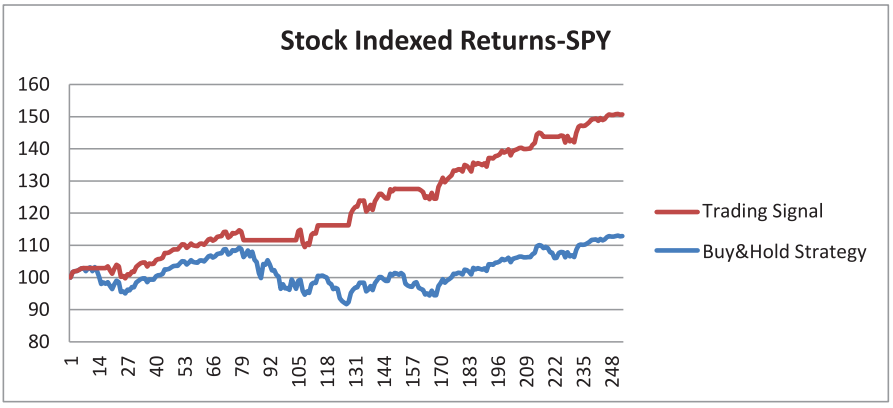

Fig. 7. Simulation trading result for SPY using denoised data.

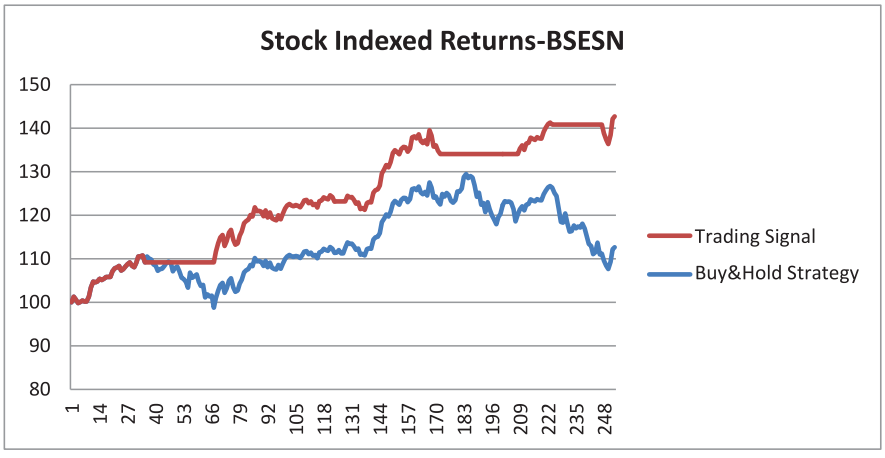

Fig. 8. Simulation trading result for BSESN using denoised data.

Table 10
Simulation result with the denoising approach.

Indices	Return (buy-hold)	Return (Signal) denoised	Return (Signal)	Number of Transactions (denoised based)
BSESN	13.24%	27.58%	**41.54%**	**10**
SPY	13.69%	20.34%	**41.89%**	**14**
EEM	17.06%	36.48%	**54.20%**	**18**
EFA	8.13%	30.09%	**50.42%**	**16**
IWM	25.60%	31.48%	**61.84%**	**12**
MDY	24.52%	35.14%	**54.29%**	**16**
QQQ	19.29%	29.09%	**50.78%**	**18**
XLF	13.45%	22.41%	**48.28%**	**20**
XLY	24.98%	27.50%	**59.24%**	**20**
AEX	10.53%	28.64%	**30.81%**	**12**
ATX	21.75%	25.98%	**55.91%**	**10**
FCHI	4.35%	23.11%	**42.88%**	**16**
FTSE	12.57%	20.15%	**34.61%**	**12**
HSI	6.85%	20.69%	**45.30%**	**12**
OIH	21.71%	24.17%	**71.08%**	**14**
SMH	18.74%	24.17%	**58.80%**	**16**
TWII	15.31%	29.13%	**39.05%**	**8**
XLE	20.75%	34.01%	**59.31%**	**10**
XLK	11.15%	20.26%	**46.67%**	**18**
XLV	2.49%	16.48%	**24.05%**	**12**
FXI	4.54%	28.07%	**54.40%**	**10**

Table 11
Simulation result with transaction costs.

	Buy-and-hold	Trading following signals	Denoising approach
Investment	$100,000	$100,000	$100,000
Ending Balance	$113,000	$128,000	$142,000
Transaction Cost	$20	$500	$140
Profit	$12,980	$27,500	$41,860
Adjusted Return	12.98%	27.50%	41.86%

Table 12
Simulation result with short selling.

Indices	Return (Signal) denoised	Return (with Short)
BSESN	41.54%	61.14%
SPY	41.89%	81.77%
EEM	54.20%	82.81%
EFA	50.42%	78.62%
IWM	61.84%	85.32%
MDY	54.29%	83.64%
QQQ	50.78%	73.58%
XLF	48.28%	65.88%
XLY	59.24%	86.28%
AEX	30.81%	45.95%
ATX	55.91%	73.98%
FCHI	42.88%	73.00%
FTSE	34.61%	55.77%
HSI	45.30%	76.40%
OIH	71.08%	87.59%
SMH	58.80%	69.32%
TWII	39.05%	49.57%
XLE	59.31%	90.33%
XLK	46.67%	65.73%
XLV	24.05%	45.08%
FXI	54.40%	103.79%

14 (including both buying and selling), down from an average of 50 without denoising.

As can be seen from Table 10, the denoising approach helped the trading system in two aspects. First, the returns are greatly improved. This is important given that each network structure has been simplified in order to provide adaptability, unlike other trading systems that attempt to improve performance through the use of more complicated network architectures, which often result in increased training time and lower generalization ability. Second, the number of transactions is dramatically lower, and hence the transaction costs are reduced. Often, when prices are noisy, there is an increase in the number of false signals. The use of denoising has the benefit of not only reducing the amount of unprofitable trades, but it does so by helping to better isolate the underlying trend, even in periods with noisy price action. This can have a positive impact in the areas of financial risk management and general trend forecasting.

8. Impact of transaction costs

As demonstrated above, the average number of transactions is 14 and 50 for the denoising approach and the model without denoising, respectively. The average returns are 42%, 28%, and 13% for denoising approach, trading following signals, and buy-and-hold strategy, respectively. Assuming that $100 K is invested through a typical online discount broker, that the broker is charging $9.99 for each transaction, and that an investible ETF is available for each index, the results in Table 11 can be obtained.

As shown in Table 11, even with transaction costs taken into account, the model is still producing significant higher returns on trading following signals both with and without the denoising approach, as compared to the buy-and-hold strategy. Nonetheless, further highlighting the trend by reducing the noise that results from short-term volatility has a significant impact on lowering false trading signals, trading activity, and transaction costs, thereby increasing profitability.

9. Short selling

The previous study has shown that the system helps significantly increase returns. To further improve the performance, short selling is considered. In general, short sellers hope to profit from a decline in the price of stocks - they still use a buy low and sell high approach, but initially sell at a higher price after borrowing a security, hoping to buy the security back later at a lower price before returning the borrowed security. By predicting the movement direction of a stock's price, it may be possible to suggest a short sell when a down signal is provided from the system. To illustrate how this works, a summary of simulations is listed in Table 12. Again, the simulation is conducted over the same period as described in Section 7, i.e., simulating 252 trading days in year 2010.

As it can been seen from Table 12, the introduction of short selling into the system further increases the returns during the test period. With the help the system's direction forecasts, profit can now also be made during the downturn of a stock index, as well as during an increase. This is unlike many trading systems that combine technical analysis and computational intelligence for "long-only" trading (Bogullu et al., 2002; Cervelló-Royo et al., 2015; Lee & Jo, 1999; Leigh et al., 2002; Lin et al., 2011, Gorgulho et al., 2011; Thawornwong et al., 2001; Yamaguchi, 1989). The implication is that such a system can now be used in situations where is it more natural for a market participant to take both a long and short position, such as the futures or foreign exchange markets. This also opens up the system to be used for hedging and financial risk management, where the ability to successfully short a security and take an arbitrage position is critical.

10. Conclusions

For this research, an adaptive intelligent stock trading decision support system has been proposed that utilizes particle swarm optimization and an artificial neural network to predict a stock index's future movement direction. While technical analysis has been used in other trading systems, researchers have often focused on price sequences and patterns, with some form of computational intelligence being used to help identify such patterns (Cervelló-Royo et al., 2015; Lee & Jo, 1999; Leigh et al., 2002; Yamaguchi, 1989). Other researchers have used technical indicators for generating trading signals, including the use of multiple indicators without optimization (Thawornwong et al., 2001), or the optimization of a single indicator (Bogullu et al., 2002). When multiple technical indicators have been used, the developed systems typically use each indicator, regardless of the analysis period or the stock or index being considered (Gorgulho et al., 2011 , Lin et al., 2011). The proposed adaptive model is unique in that it optimizes network performance for each individual stock index, and only uses those technical indictors that facilitate the best network performance.

This system also overcomes a major weakness of a traditional artificial neural network approach, which is often highly sensitive to the selection of inputs and parameter settings, by adaptively adjusting the model to fit different stock indices. Different from other research approaches regarding stock and stock index selection, the adaptability and flexibility of the proposed system makes it applicable to real life situations, without the disadvantage of trying to develop a new system each time the input data and/or desired output change. Furthermore, rather than predicting the stock index price, a strength of the proposed system is that it generates trading signals by predicting the direction of the stock index price. Unlike many other traditional systems that attempt to forecast the level of an index or stock price, training on the direction of the stock index price results in a network that is less complex, allowing for faster system training and greater testing accuracy. The introduction of the particle swarm module into the system also helps to further reduce time consumption in computation, making the enumeration strategy feasible for real-time application, unlike other more complicated systems. Simulation and evaluation on sample stock indices shows that a measurable investment improvement is achieved by conducting trading that follows the trading signals provided by the proposed system, compared to that achieved by

following the buy-and-hold strategy. The uses of denoising and short selling were also shown to increase trading returns by reducing transaction cost and increasing trading opportunities, respectively.

Nonetheless, the system is not without its weaknesses. The current system is limited to the use of technical indicators and patterns as inputs. While this may offer an advantage for the stock market indexes that were tested, additional studies can be performed to further highlight the robustness of the model. This might also be true for the time period under study. Some models, especially those that utilize technical inputs, inherently perform better in trending markets (bull or bear markets), as opposed to trading markets that might display volatility, but are otherwise trendless over the examination period. In our case model performance was only considered for one calendar year, although the year in question (2010) was not in a single confirmed bull or bear market trend for the entire year.

Regarding future directions, a number of areas for improvement can be considered. First, given its adaptive nature, it is possible to use the proposed system for other asset classes and types of securities. Individual stocks, commodities, and futures contracts can be considered. In addition, foreign exchange contracts are a good candidate for future analysis given the importance of using technical analysis when trading this asset class, as well as having the ability to receive a short position trading signal. Second, other forms of inputs in addition to common technical analysis indicators and patterns can be considered, such as macroeconomic data, especially if longer testing periods (other than daily) are considered. Third, other classification models, beyond a simple feed-forward neural network can be considered, including those that often perform well with financial data, such as probabilistic neural networks, generalized regression neural networks, and hybrid fuzzy-genetic-neural network models. Finally, additional methods for preprocessing the input data, such as using principal component analysis, information gain theory, or clustering can be used to further highlight the most important and useful system inputs, thereby further increasing network classification speed and accuracy.

References

Albadvi, A., Chaharsooghi, S. K., & Esfahanipour, A. (2007). Decision making in stock trading: an application of PROMETHEE. *European Journal of Operational Research, 177*(2), 673–683. doi:10.1016/j.ejor.2005.11.022.

Amornwattana, S., Enke, D., & Dagli, C. (2007). A hybrid options pricing model using a neural network for estimating volatility. *International Journal of General Systems, 36*(5), 558–573.

Atsalakis, G. S., & Valavanis, K. P. (2009). Forecasting stock market short-term trends using a neuro-fuzzy based methodology. *Expert Systems with Applications, 36*(7), 10696–10707.

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

Balvers, R., Wu, Y., & Gilliland, E. (2000). Mean reversion across national stock markets and parametric contrarian investment strategies. *The Journal of Finance, 55*(2), 745–772. doi:10.1111/0022-1082.00225.

Bebis, G., & Georgiopoulos, M. (1994). Optimal feed-forward neural network architectures. *IEEE Potentials, 13*(4), 27–31.

Berner, E. S. (2007). *Clinical decision support systems*. New York, NY: Springer Science and Business Media, LLC. http://www.arteimi.info/site/aibooks/Clinical.Decision.Support.Systems.2nd.ed.2007.Springer.3HAXAP.pdf.

Berry, M. J. A., & Linoff, G. S. (1997). *Data mining techniques*. NY: John Wiley & Sons.

Blum, A. L., & Rivest, R. L. (1992). Training a 3-node neural network is NP-Complete. *Neural Networks, 5*, 117–127.

Bogullu, V. K., Enke, D., & Dagli, C. (2002). Using neural networks and technical indicators for generating stock trading signals. *Intelligent Engineering Systems through Artificial Neural Networks, 2*, 721–726.

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics, 31*(3), 307–327.

Box, G. E. P., Jenkins, G. M., & Reinsel, G. C. (2013). *Time series analysis: Forecasting and control*. Hoboken, NJ: John Wiley & Sons.

Bryson, E., & Ho, Y. (1969). *Applied optimal control: Optimization, estimation, and control* (p. 481). New York, NY: Blaisdell Publishing Company or Xerox College Publishing.

Cervelló-Royo, R., Guijarro, F., & Michniuk, K. (2015). Stock market trading rule based on pattern recognition and technical analysis: Forecasting the DJIA index with intraday data. *Expert Systems with Applications, 42*(14), 5963–5975.

Chang, P. C., Liu, C. H., Lin, J. L., Fan, C. Y., & Ng, C. S. P. (2009). A neural network with a case based dynamic window for stock trading prediction. *Expert Systems with Applications, 36*(3), 6889–6898.

Chaudhuri, K., & Wu, Y. (2003). Mean reversion in stock prices: evidence from emerging markets. *Managerial Finance, 29*(10), 22–37. doi:10.1108/03074350310768490.

Chavarnakul, T., & Enke, D. (2009). A hybrid stock trading system for intelligent technical analysis-based equivolume charting. *Neurocomputing, 72*(16-18), 3517–3528.

Cheng, C. H., Chen, T. L., Teoh, H. J., & Chiang, C. H. (2008). Fuzzy time-series based on adaptive expectation model for TAIEX forecasting. *Expert Systems with Applications, 34*(2), 1126–1132.

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica: Journal of the Econometric Society, 50*(4), 987–1007.

Enke, D., & Amornwattana, S. (2008). A hybrid derivative trading system based on volatility and return forecasting. *The Engineering Economist, 53*(3), 259–292.

Enke, D., & Mehdiyev, N. (2013). Stock market prediction using a combination of stepwise regression analysis, differential evolution-based fuzzy clustering, and a fuzzy inference neural network. *Intelligent Automation and Soft Computing, 19*(4), 636–648.

Enke, D., & Thawornwong, S. (2005). The use of data mining and neural networks for forecasting stock market returns. *Expert Systems with Applications, 29*, 927–940.

Fliess, M., & Join, C.. *A mathematical proof of the existence of trends in financial time series arXiv:0901.1945 [cs, Math, Q-Fin, Stat]* (January 14) http://arxiv.org/abs/0901.1945 .

Gorgulho, A., Neves, R., & Horta, N. (2011). Applying a GA Kernel on optimizing technical analysis rules for stock pricing and portfolio composition. *Expert Systems with Applications, 38*, 14072–14085.

Gudise, V. G., & Venayagamoorthy, G. K. (2003). Comparison of particle swarm optimization and backpropagation as training algorithms for neural networks. In *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE* (pp. 110–117). http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1202255.

Hadavandi, E., Shavandi, H., & Ghanbari, A. (2010). Integration of genetic fuzzy systems and artificial neural networks for stock price forecasting. *Knowledge-Based Systems, 23*(8), 800–808.

Hoogenboom, G., Jones, J. W., Wilkens, P. W., Porter, C. H., Batchelor, W. D., Hunt, L. A., Boote, K. J., Singh, U., Uryasev, O., & Bowen, W. T. (2004). *Decision support system for agrotechnology transfer version 4.0*. Honolulu, HI (CD-ROM): University of Hawaii.

Huang, C. L., & Tsai, C. Y. (2009). A hybrid SOFM-SVR with a filter-based feature selection for stock market forecasting. *Expert Systems with Applications, 36*(2), 1529–1539.

Huang, S. M., Hung, Y. C., & Yen, D. C. (2005). A study on decision factors in adopting an online stock trading system by brokers in Taiwan. *Decision Support Systems, 40*(2), 315–328.

Kawamoto, K., Houlihan, C. A., Balas, E. A., & Lobach, D. F. (2005). Improving clinical practice using clinical decision support systems: a systematic review of trials to identify features critical to success. *BMJ, 330*(7494), 765–772.

Klein, M., & Methlie, L. B.. *Knowledge-based decision support systems with applications in business: A decision support approach* http://www.citeulike.org/group/8357/article/4172889.

Kuo, R. J. (1998). A decision support system for the stock market through integration of fuzzy neural networks and fuzzy Delphi. *Applied Artificial Intelligence, 12*(6), 501–520.

Kuo, R. J., Chen, C. H., & Hwang, Y. C. (2001). An intelligent stock trading decision support system through integration of genetic algorithm based fuzzy neural network and artificial neural network. *Fuzzy Sets and Systems, 118*(1), 21–45.

Lai, R. K., Fan, C. Y., Huang, W. H., & Chang, P. C. (2009). Evolving and clustering fuzzy decision tree for financial time series data forecasting. *Expert Systems with Applications, 36*(2), 3761–3773.

Lee, C. C., Lee, J. D., & Lee, C. C. (2010). Stock prices and the efficient market hypothesis: Evidence from a panel stationary test with structural breaks. *Japan and the World Economy, 22*(1), 49–58. doi:10.1016/j.japwor.2009.04.002.

Lee, K. H., & Jo, G. S. (1999). Expert system for predicting stock market timing using a candlestick chart. *Expert Systems with Applications, 16*(4), 357–364.

Leigh, W., Modani, N., Purvis, R., & Roberts, T. (2002). Stock market trading rule discovery using technical charting heuristics. *Expert Systems with Applications, 23*(2), 155–159.

Lin, X., Yang, Z., & Yixu, S. (2011). Intelligent stock trading system based on improved technical analysis and echo state network. *Expert Systems with Applications, 38*(9), 11347–11354.

Lippmann, R. P. (1987). An introduction to computing with neural nets. *IEEE ASSP Magazine, 4*(2), 4–22.

Lo, A. W., & MacKinlay, A. C. (1988). Stock market prices do not follow random walks: evidence from a simple specification test. *Review of Financial Studies, 1*(1), 41–66. doi:10.1093/rfs/1.1.41.

Lo, A. W., & MacKinlay, A. C. (2011). *A non-random walk down wall street*. Princeton, New Jersey: Princeton University Press.

Martinez, L., Ruan, D., & Herrera, F. (2010). Computing with words in decision support systems: An overview on models and applications. *International Journal of Computational Intelligence Systems, 3*(4), 382–395.

Nikelshpur, D., & Tappert, C.. *Using particle swarm optimization to pre-train artificial neural networks: selecting initial training weights for feed-forward back-propagation neural networks* http://csis.pace.edu/~ctappert/srd2013/c5.pdf.

Quah, T. S. (2008). DJIA stock selection assisted by neural network. *Expert Systems with Applications, 35*(1–2), 50–58. doi:10.1016/j.eswa.2007.06.039.

Rahman, A., & Saadi, S. (2008). Random walk and breaking trend in financial series: An econometric critique of unit root tests. *Review of Financial Economics, 17*(3), 204–212. doi:10.1016/j.rfe.2007.05.002.

Russell, S., & Norvig, P. (2003). *Artificial intelligence: A modern approach*. Englewood Cliffs, New Jersey: Prentice Hall.

Samaras, G. D., & Matsatsinis, N. F. (2004). An intelligent decision support system for portfolio management. *Operational Research, 4*(3), 357–371.

Semenov, A. (2008). Testing the random walk hypothesis through robust estimation of correlation. *Computational Statistics & Data Analysis, 52*(5), 2504–2513.

Sevastjanov, P., & Dymova, L. (2009). Stock screening with use of multiple criteria decision making and optimization. *Omega, 37*(3), 659–671. doi:10.1016/j.omega.2008.04.002.

Shi, Y. (2001). Particle swarm optimization: developments, applications and resources. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on: 1* (pp. 81–86). http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=934374.

Swingler, K. (1996). *Applying Neural Networks: A Practical Guide*. San Francisco, California: Academic Press.

Thawornwong, S., & Enke, D. (2003). The adaptive selection of financial and economic variables for use with artificial neural networks. *Neurocomputing, 56*, 205–232.

Thawornwong, S., Enke, D., & Dagli, C. (2001). Using neural networks and technical analysis indicators for predicting stock trends. *Intelligent Engineering Systems through Artificial Neural Networks, 11*, 739–744.

Thawornwong, S., Enke, D., & Dagli, C. (2003). Neural networks as a decision maker for stock trading: A technical analysis approach. *International Journal of Smart Engineering Systems Design, 5*(4), 313–325.

Tsai, C., & Hsiao, Y. (2010). Combining multiple feature selection methods for stock prediction: Union, intersection, and multi-intersection approaches. *Decision Support Systems, 50*(1), 258–269.

Tsaih, R., Hsu, Y., & Lai, C. C. (1998). Forecasting S&P 500 stock index futures with a hybrid AI system. *Decision Support Systems, 23*(2), 161–174.

Vejendla, A., & Enke, D. (2013). Performance evaluation of neural networks and GARCH models for forecasting volatility and option strike prices in a bull call spread strategy. *Journal of Economic Policy and Research, 8*(2), 1–19.

Vejendla, A., & Enke, D. (2013). Evaluation of GARCH, RNN, and FNN models for forecasting volatility in the financial markets. *IUP Journal of Financial Risk Management, X*(1), 41–49.

Yamaguchi, T. (1989). A technical analysis expert system in the stock market. *Future Generation Computer Systems, 5*(1), 21–27.

Zhang, J. R., Zhang, J., Lok, T. M., & Lyu, M. R. (2007). A hybrid particle swarm optimization back-propagation algorithm for feedforward neural network training. *Applied Mathematics and Computation, 185*(2), 1026–1037.