Predicting financial time series data using artificial immune system-inspired neural networks

Haya Al-askar*, David Lamb,
Abir Jaafar Hussain, Dhiya Al-Jumeily,
Martin Randles and Paul Fergus

School of Computing and Mathematical Sciences,
Liverpool John Moores University,
Liverpool L3 3AF, UK
Email: H.Alaskar@2011.ljmu.ac.uk
Email: d.lamb@ljmu.ac.uk
Email: A.hussain@ljmu.ac.uk
Email: d.aljumaily@ljmu.ac.uk
Email: m.randles@ljmu.ac.uk
Email: p.fergus@ljmu.ac.uk
*Corresponding author

Abstract: This paper investigates a set of approaches for the prediction of noisy time series data; specifically, the prediction of financial signals. A novel dynamic self-organised multilayer neural network based on the immune algorithm for financial time series prediction is presented, combining the properties of both recurrent and self-organised neural networks. In an attempt to overcome inherent stability and convergence problems, the network is derived to ensure that it reaches a unique equilibrium state. The accuracy of the comparative evaluation is enhanced in terms of profit earning; empirical testing used in this work includes normalised mean square error (NMSE) to evaluate forecast fitness and also evaluates predictions against financial metrics to assess profit generation. Extensive simulations for multi-step prediction in stationary and non-stationary time series were performed. The resulting forecast made by the proposed network shows substantial profits on financial historical signals when compared to various solely neural network approaches. These simulations suggest that dynamic immunology-based self-organised neural networks have a better ability to capture the chaotic movement in financial signals.

Keywords: financial signals; artificial immune systems; immunology-inspired computing; higher order neural network; time series data.

Reference to this paper should be made as follows: Al-askar, H., Lamb, D., Hussain, A.J., Al-Jumeily, D., Randles, M. and Fergus, P. (2015) 'Predicting financial time series data using artificial immune system-inspired neural networks', *Int. J. Artificial Intelligence and Soft Computing*, Vol. 5, No. 1, pp.45–68.

Biographical notes: Haya Al-askar is a full time PhD student at the School of Computing and Mathematical Sciences at Liverpool John Moores University. Her research interests include time series prediction and medical data classifications. She completed her MSc in 2010 at Exeter University.

David Lamb is currently a Researcher and Software Engineer in the School of Computing and Mathematical Sciences at Liverpool John Moores University. He spent several years in industry as a software engineer before entering academia. He obtained a first-class BSc (Hons) in Software Engineering from LJMU in 2005; going on to complete a PhD in software observation in large-scale systems at the same institution in 2009. His present research interests are programming support for monitoring and managing large-scale systems, along with software engineering approaches for handling large-scale datasets/Big Data, and pedagogical concerns in teaching and learning for computer programming.

Abir Jaafar Hussain is a Senior Lecturer at the School of Computing and Mathematical Sciences at Liverpool John Moores University, UK. She completed her PhD study at The University of Manchester, UK in 2000 with a thesis title 'Polynomial neural networks for image and signal processing'. She has published numerous refereed research papers in conferences and journal in the research areas of neural networks, signal prediction, telecommunication fraud detection and image compression. She is a PhD Supervisor and an External Examiner for research degrees including PhD and MPhill.

Dhiya Al-Jumeily is an Associated Professor in eSystems Engineering and leads the Applied Computing Research Group at the Faculty of Technology and Environment. He has already developed fully online MSc and BSc programmes and currently heads the enterprise activities at the School of Computing and Mathematical Sciences. He has published numerous refereed research papers in multidisciplinary research areas including: technology enhanced learning, applied artificial intelligence, neural networks, signal prediction, telecommunication fraud detection and image compression. He is a PhD Supervisor and an External Examiner for the degree of PhD.

Martin Randles is currently Researcher, Senior Lecturer and Programme Leader for Software Engineering in the School of Computing and Mathematical Sciences at Liverpool John Moores University. He obtained his BSc (honours) in Mathematics from the University of Manchester in 1983. After a number of years in business and commerce, he received his MSc with distinction in Computing and Information Systems from LJMU in 2003. A PhD in providing self-management for large-scale systems followed this in 2007. His research interests include the application of mathematical techniques to computer science, the control and management of large-scale complex systems and autonomic computing.

Paul Fergus is a Senior Lecturer in the School of Computing and Mathematical Sciences at Liverpool John Moores University. He received his BSc (Hons.) in Artificial Intelligence from Middlesex University in 1997, and a Postgraduate Diploma and MSc in Computing for Commerce and Industry from the Open University in 1999 and 2001, respectively. In 2006, he received his PhD in Self-Adaptive Networked Appliances from Liverpool John Moores University. He is an active researcher in the field of wireless communications, artificial intelligence, bioinformatics, machine learning, medical devices, and technology for health with over 100 refereed publications.

This paper is a revised and expanded version of a paper entitled 'Recurrent neural networks inspired by artificial immune algorithm for time series prediction' presented at Proceedings of International Joint Conference on Neural Networks, Dallas, Texas, USA, 4–9 August 2013.

1 Introduction

Financial time series forecasting is a difficult task due to the intrinsic complexity and numerous affecting variables of the financial system. While many time series may be approximated with a high degree of confidence, financial time series are considered among the most difficult to analyse and predict (Castiglione, 2001). Yao et al. (1996) identifies several factors that complicate the process in predicting change in large financial data sets, such as the stock market. These systems are affected by many interrelated factors ranging from economic and political factors through to psychological factors affecting both powerful decision makers and individuals or consumers. Moreover traditional economic studies indicates that even microeconomic drivers can affect demand consumption patterns and vice versa, thus forming a very complicated highly-interrelated system that is very difficult to model (Yao et al., 1996).

Decisions regarding investments and trading by large companies, financial institutions, and even the economic policy of governments rely on computer modelling forecasts (Hussain et al., 2008). Commercial and financial imperatives ensure that financial time-series prediction receives a large amount of coverage in research literature and this will no doubt continue to be the case.

Various methods and techniques for the prediction of financial time series have been and continue to be developed following some key principles. There are several distinct approaches used for prediction/forecasting, from statistical to artificial intelligence/machine learning techniques. The traditional methods for financial time-series forecasting are based around statistical approaches. Most of the developed prediction methods, however, have very little accompanying scientifically evaluated support or verification: the analysis is unsatisfactory due to the non-linear nature of most of the financial time series data (Dunis and Williams, 2002; Yao and Tan, 2000; HellstrSom and HolmstrSom, 1998).

Accordingly, throughout the last decade, neural networks have gained ground, moving from research experimentation tools, through to production and industrial decision-support systems; assisting managerial and advisory staff in making 'critical financial decisions' (Chen and Leung, 2005). As such, they are regarded as powerful tools for forecasting data, and have significant academic/research interest from fields such as artificial intelligence, machine learning, and statistics. They are non-linear models that can be trained from known datasets, automatically mapping input and output data; effectively mapping past to future values in time series data. Such models are particularly powerful in their ability to extract hidden structures and relationships that govern data mappings (Shachmurove and Witkowska, 2000).

Using neural networks, complex relationships between input and output variables can be automatically identified, by machine, without requiring a human being to specify the nature of the relationship. Neural networks have appeared as a powerful learning technique to perform complex tasks in the highly non-linear dynamic environment of financial time series.

Financial service companies are becoming more and more dependent on computer technologies to establish and maintain competitiveness in the rapidly expanding global economy (Chen and Leung, 2005). Illustrating the point, researchers have identified that several large investment banks (including Goldman Sachs and Morgan Stanley) now have departments dealing solely with the neural network models required for their business investment analysis (Shachmurove and Witkowska, 2000).

There is clearly a business case for such large financial interests to invest in artificial neural networks as a prediction and forecasting method. Neural networks (NN) are considered as a superior method for time-series prediction, as compared to pure statistical methods, owing to their non-linear nature and capacity for data-based training (Yümlü et al., 2005; Ho et al., 2002; Dunis and Williams, 2002). In summary, the ability of a neural network to approximate and produce complex mappings between non-linear data is particularly fitting to this identified problem domain.

Widyanto et al. (2005) introduced a hidden layer inspired by an implementation of a biological immune system algorithm to the back-propagation networks to improve the way in which the NN could generalise their results. The immune system algorithm is based on the controls that regulate the construction of antibody-generating B cells in a body. In their own experiments, the immunity-inspired NN was used to predict temperature-based food quality; it showed an 18% improvement in correct recognition on standard back-propagation NNs (Widyanto et al. 2005).

In this paper, a novel neural network architecture is proposed, based on the immune algorithm and a self-organised neural network. The proposed network is called the 'Dynamic self-organised multilayer neural network inspired by the immune algorithm' (DSMIA). This proposed DSMIA, the multilayer perceptron (MLP) and functional link networks are used for single and multi-step-ahead prediction of financial time series.

Ten financial time series are used to test the performance of the various networks such as the exchange rates and oil price time series. In this set of experiments, the primary interest is to concentrate on the profitable value contained in the predictions for all neural network models. As such, the best model is that which provides the highest percentage of annualised return (AR) on out-of-sample data.

2 The networks

Neural networks are trained to generate functional maps from given input to output data sets. However, differing architectures are better suited to different training dataset types and problem domains. For some problem domains, higher-order input combinations may be more suitable in order that the NN forms an appropriate problem mapping.

This section provides an overview and brief introduction to existing functional link and artificial immune system-inspired neural networks.

2.1 Functional link neural networks

Giles and Maxwell (1987) first introduced FLNN as an addition to feedforward networks. The FLNN architecture extends the traditional feedforward network; aiming to map non-linear relationships between input and output data. Functional links replace the linear weights creating a degree of class separation. FLNNs can use higher-order correlations of input components to perform non-linear mappings with fewer layers. As the architecture is simpler, it is intended to reduce the computational cost incurred in the training stage, whilst demonstrating good approximation performance (Mirea and Marcu, 2002). Despite their otherwise linear nature, providing the input set is suitably descriptive, FLNN-architecture NNs can provide a learning network with greater information capacity and complex learning ability (Cass and Radl, 1996; Giles et al., 1998; Mirea and Marcu, 2002).

FLNNs have created a popular strand of research, with researchers exploring the performance and capability of networks following this architecture. Fei and Yu (1994) showed that FLNN have more powerful approximation capability than conventional back-propagation networks, and form suitable models for system identification (Mirea and Marcu, 2002). Cass and Radl (1996) used FLNN in process optimisation, finding that the training speed is improved significantly on traditional MLP networks, without a sacrifice in computational capacity.

However, FLNN architectures are not without problems. They suffer from weight explosion, with an exponential increase across the number of inputs. As a result, only second or third order networks form practical maps (Kaita et al., 2002; Thimm, 1997). The output of FL-architecture NNs is specified as follows:

$$Y = \sigma\left(W_0 + \sum_{j} W_j X_j + \sum_{jk} W_{jk} X_j X_k + \sum_{jkl} W_{jkl} X_j X_k X_l + \dots\dots\dots \right) \tag{1}$$

where X and Y are the input and the output of the network, respectively. In this case, σ is a non-linear transfer function, and w_o is the adjustable threshold.

2.2 Self-organised network inspired by the immune algorithm

Natural computation, specifically in the form of artificial immune systems, has attracted much research interest; from the augmentation of machine-learning though to descriptive paradigms for self-organising systems (Timmis, 2000).

The self-organised network inspired by immune algorithm (SONIA) network is an enhancement to back-propagation neural networks that introduces a self-organising hidden layer that aims to improve both the generalisation and recognition capacity (Widyanto et al., 2005). The rules governing self-organisation in the hidden layer are loosely modelled on the B-cell construction/mutation in antibody production and antigen recognition.

The concept is based on the relationship between antigens and the B-cell; the latter of which produces antibodies when fully active, and mutate when they encounter known antigens in order that the mutated cells may recognise previously unknown antigens.

In the SONIA network, each hidden unit has a centre representative of the number of connections of attached input vectors. To avoid over-fitting, each centre contains a value representing the strength of connections between input units and their corresponding hidden units. SONIA has been previously applied to financial time-series data prediction, with some success (Hussain and Al-Jumeily, 2007). Two time series were used, modelling the daily exchange rates between the US Dollar and Euro and the IBM common stock closing price. The simulation results showed that the SONIA network produced profit when used for financial time series prediction.

3 Dynamic self-organised multilayer network inspired by the immune algorithm

In this section, recurrent neural network architectures are proposed. The new architectures incorporate recurrent links within the structure, the operation of which creates a self-organising layer; inspired by artificial immune system theory.

Previous work in this field includes the self-organised multilayer NN inspired by the immune algorithm/SMIA (Mahdi et al., 2009), and the dynamic ridge polynomial higher-order neural network/DRPHONN (Ghazali and Hussain 2009).

These works feed into, and are extended by, this work to provide a novel network with many beneficial characteristics. These include the dynamic self-organisation of hidden-layer units, and output-based feedback into the hidden layers. This represents a major improvement compared to feedforward networks, which can only implement a static mapping of the input vectors. In order to model dynamic functions, it is essential to exploit a system capable of storing internal states and implementing complex dynamics. Neural networks with recurrent connections are dynamic systems with temporal/state representations which, because of their dynamic structure, have been successfully used for solving a variety of problems. This work is motivated by the potential of recurrent dynamic systems in solving complex real-world problems. The following subsection provides an overview of the architecture of the dynamic self-organising multilayer network inspired by the immune algorithm (DSMIA).

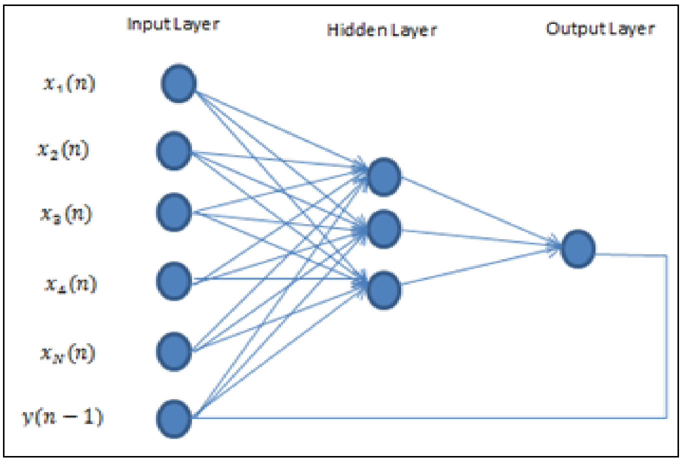
3.1 Properties and network structure of the DSMIA

Applications in forecasting and signal processing require systems that can work with dynamic signals. As discussed previously, traditional feedforward networks can only handle static mappings. The inclusion of previous inputs into a now-augmented set of input values permits dynamic mapping to occur in the network.

The DSMIA architecture differs from this dynamic representation; borrowing from biological networks, in common with the Dynamic Ridge Polynomial Neural Network (DRPNN) architecture, (Ghazali and Hussain, 2009). By incorporating a self-organising internal feedback circuit, DSMIA learns the internal signal properties, providing better detection and prediction of precise signal trends. Therefore, the structure of the DSMIA comprises three layers: the input layer, the self-organising hidden layer, and the output layer, with feedback connection from the output layer to the input layer. The input layer holds copies of the current inputs as well as the previous output of the network, providing the network with memory. As such, previous behaviour of the network is also used as an input affecting current behaviour. Similar to the Jordan recurrent network (Jordan, 1997), the output of the network is fed back to the input through the context units.

The hidden layer of the proposed network is trained in a similar manner to recurrent self-organising maps (Voegtlin, 2002). The feedback connection feeds the activation of the output node to the input context units. This feedback connection, shown in the network structure diagram, Figure 1, gives the network the ability to ‘memorise’ data for use in its mapping, as described earlier. All the connection weights from the input layer to the self-organised layer and from the self-organised layer to the output layer are learnable.

Figure 1 The structure of the proposed DSMIA network (see online version for colours)



3.2 Mathematically specifying the DSMIA

This section will consider the dynamic equations that form the proposed DSMIA network. The following equations assume N is the number of external inputs to the proposed neural network, while O is the number of outputs. As such, the total number of inputs to the network is $N + O$ where:

$$U(n) = \begin{cases} x_i(n), i = 1, \ldots, N \\ y_j(n-1), j = 1, \ldots, O \end{cases} \tag{2}$$

In this equation, x is representative of external inputs, and y the final outputs of the network. The function $U(n)$ describes the n^{th} input of the network.

The function $X_{Hj}(n)$ describes the input to the hidden, self-organising layer, and is determined as follows:

$$X_{Hj}(n) = f_{ht}\left(D_{hj}(n)\right) \tag{3}$$

where f is a non-linear function, further specified as:

$$D_{kj}(n) = V_{hj}(n) + Z_{kj}(n) \tag{4}$$

Comprising:

$$V_{hj}(n) = \alpha \sqrt{\sum_{i=1}^{NI} (W_{hji} - x_i(n))^2} \tag{5}$$

$$Z_{hj}(n) = \beta \sqrt{\sum_{k=1}^{NO} (WZ_{hjk} - y_k(n-1))^2} \tag{6}$$

α, β are experimentally selected parameters with $0<\alpha$ and $0<\beta$.

To summarise; outputs of the self-organised layer form inputs to the nodes in the output layer; which in turn form the final output of the network. Therefore, the (final) k^{th} output is given by:

$$\hat{y}_k = f_y\left(\sum_{j=1}^{N_H} w_{ojk} x_{Hj} + B_{ok}\right), \ldots\ldots k = 1, \ \ldots., \ O \tag{7}$$

The term w_{ojk} represents the strength of the connection from the j^{th} hidden unit to the k^{th} output unit. B_{ok} is a bias associated with the k^{th} output unit, and f_y is a non-linear transfer function.

Taking inspiration from artificial immune systems and recalling that the immune recognition function utilises B cell functionality, where generating antibodies are matched to the various antigens. In this case, the hidden unit is designed as an implementation of that B cell recognition algorithm from the immune system. If we assume that $\hat{y}$ (n) describes a function illustrative of the desired response of the network at time n, and $y(n)$ is the target output at time n. then the error of the network at time n can be specified as:

$$e(n) = y(n) - \hat{y}(n) \tag{8}$$

Therefore, the squared error between the actual and predicted values of the network is given by:

$$E(n) = \frac{1}{2}\sum e(n)^2 \tag{9}$$

The change for any element wo_{jk} of the weights matrix is calculated as follows:

$$w_{o\,jk}(n+1) = w_{o\,jk}(n) + \Delta w_{o\,jk}(n) \tag{10}$$

$$\Delta w_{o\,jk}(n) = -\eta \frac{\partial E(n)}{\partial w_{jk}} \tag{11}$$

$$\frac{\partial E(n)}{\partial w_{jk}} = \frac{\partial E(n)}{\partial \hat{y}_k(n)} \frac{\partial \hat{y}_k(n)}{\partial W_{jk}} \tag{12}$$

$$\frac{\partial \hat{y}_k(n)}{\partial W_{jk}} = X_j \tag{13}$$

Since the f_y *is the sigmoid functio*n the derivative of sigmoid function will be equal to $\hat{y}_k(1-\hat{y}_k)$

$$\frac{\partial E(n)}{\partial \hat{y}_k(n)_k} = (y_k(n) - \hat{y}_k(n))\hat{y}_k(1-\hat{y}_k) \tag{14}$$

where (j = 1…., N_H, k = 1…, O) and η is the learning rate, which is a positive constant.

The change for any particular element B_{oj} of the bias matrix is given as, where (k = 1…,O):

$$\Delta B_{ok}(n) = -\eta \frac{\partial J(n)}{\partial B_{ok}} \tag{15}$$

The self-organised layer is trained in a similar manner to the training algorithm of the Voegtlin recursive self-organising map (Voegtlin, 2002). In this case, the weights of the previous context WZ_{hjk} are updated in the same way as the weight of the external input W_{hji}. The learning rule for updating the weights is based on adaptive weights of neurons that belong to a neighbourhood of best-matching neurons, in the direction of the input vector. This is done by finding D, which is the distance between the input units and centroid of the j^{th} hidden units:

$$D_{hj}(n) = \alpha \sqrt{\sum_{i=1}^{NI} \left(W_{hji} - x_i(n)\right)^2} + \beta \sqrt{\sum_{k=1}^{NO} \left(WZ_{hjk} - y_k(n-1)\right)^2} \tag{16}$$

From $D_{hj}(n)$, the location of the best-matching unit is defined as the unit that minimises $D_{hj}(n)$ and it is determined according to the following equation:#

$$c(n) = \arg\min D_{hj}(n) \tag{17}$$

If the shortest distance $D_c(n)$ is less than the simulation level s_1 (between 0 and 1), then both the weight for the external input vectors and the context vectors are updated according to the following equations:

$$W_{hji}(n+1) = W_{hji}(n) + \gamma D_c(n) \tag{18}$$

$$WZ_{hjk}(n+1) = WZ_{hjk}(n) + \gamma D_c(n) \tag{19}$$

WZ_{hjk} is the weight of the previous output and W_{hji} is the weight for the external inputs, and γ is the learning rate which is updated during the epochs. Note that this is basically the original SOM learning rule utilised to both vectors $W_{hji}(n + 1)$ and $WZ_{hjk}(n + 1)$.

4 Prediction of financial time series

Ten noisy financial time series signals are considered as shown in Table 1. All the signals were obtained from the Time Series Data Library (Datastream International, 2005).

The networks are tested as predictors of one and five steps ahead of financial time. To evaluate this work, two sets of experiments are performed: firstly on non-stationary datasets and secondly on the stationary datasets. For non-stationary signals, all data signals are presented to the networks directly – i.e., without any transformation. Conversely, the stationary set of signals needs a degree of transformation before passing them to the networks, as is generally observed in usual methods of time series prediction. This is because precise daily prices (the non-stationary signals) are not considered as meaningful to trading, the relative magnitude of price changes is seen as more significant (Cao and Tay, 2003).

Table 1 Time series data used

No.	*Time series data*	*Size*
1	US dollar to EURO exchange rate (USD/EUR) 01/07/2002–13/11/2008	1,607
2	US dollar to UK pound exchange rate (USD/UKP) 01/07/2002–13/11/2008	1,607
3	Japanese Yen to US dollar exchange rate (JPY/USD) 01/07/2002–13/11/2008	1,607
4	Dow Jones Ind. Average stock opening price (DJIAO) 01/07/2000–11/11/2008	1,605
5	Dow Jones Industrial Average stock closing price (DJIAC) 01/07/2000–11/11/2008	1,605
6	Dow Jones Utility Average stock opening price (DJUAO) 01/07/2000–11/11/2008	1,605
7	Dow Jones Utility Average stock closing price (DJUAC) 01/07/2000–11/11/2008	1,605
8	NASDAQ composite stock opening price (NASDAQO) 01/07/2000–12/11/2008	1,606
9	NASDAQ composite stock closing price (NASDAQC) 01/07/2000–12/11/2008	1,606
10	Oil price of West Texas Intermediate crude (OIL) 01/01/1985–01/11/2008	389

This sort of pre-processing has been seen to increase the performance of the NN in its forecasting ability (Kaastra and Boyd, 1996). The processing technique adopted is termed relative difference in price (RDP), and is a used by a number of researchers in this field (Thomason, 1999; Cao and Tay, 2003). It creates a five-day measure of the relative difference in price data, smoothing the data and helping to reduce noise. Post-transformation, all input and output variables are scaled between upper and lower bounds of the transfer function, as this has been shown, in previous work, to avoid further computational complexity (Ghazali and Hussain, 2009).

The advantage of using RDP transformation is that the distribution of the transformed data will become more symmetrical and will follow more closely the normal distribution. According to Thomason (1999), this transformation of the signal often enhances the performance of trading systems, when applied in neural network models. The assumption is that the transformation results in the extraction of market characteristics that are more useful to the prediction task than the absolute values alone, and that improved prediction performance translates to improved trading system performance.

The input variables were determined from four lagged RDP values based on five-day periods (RDP-5, RDP-10, RDP-15, and RDP-20) and one transformed signal (EMA15) which is obtained by subtracting a 15-day exponential moving average from the original signal. As mentioned in Thomason (1999), the optimal length of the moving day period, in this case is 15, is not critical, but it should be longer than the forecasting horizon. Since the use of RDP to transform the original time series may remove some useful information embedded in the data, EMA15 was used to retain the information contained in the original data. As argued in Thomason (1999), smoothing both input and output data by using either simple or exponential moving average is a good approach and can generally enhance the prediction performance. Table 2 shows the calculation for the input and output variables.

Table 2 Calculation for input and output variables

	Indicator	*Calculations*
Input variables	EMA15	$P(i) - \overline{EMA_{15}(i)}$
	RDP-5	$(p(i) - p(i-5)) / p(i-5) * 100$
	RDP-10	$(p(i) - p(i-10)) / p(i-10) * 100$
	RDP-15	$(p(i) - p(i-15)) / p(i-15) * 100$
	RDP-20	$(p(i) - p(i-20)) / p(i-20) * 100$
Output variable	RDP+k	$(\overline{p(i+k)}\ \overline{p(i)}) / \overline{p(i)} * 100$
		$\overline{p(i)} = \overline{EMA_3(i)}$

Notes: $EMA_n(i)$ is the n-day exponential moving average of the i^{th} day.
$p(i)$ is the signal of the i^{th} day.
k is forecast horizon; 1 or 5.

Furthermore, the data are scaled to accommodate the limits of the network's transfer function. Manipulation of the data using this process produces a new bounded dataset. The calculation for the standard minimum and maximum normalisation method is as follows:

$$x' = (\max_2 - \min_2) \cdot \left(\frac{x - \min_1}{\max_1 - \min_1} \right) + \min_2 \tag{20}$$

where x' refers to the normalised value, x refers to the observation value (original value), $\min_1$ and $\max_1$ are the respective minimum and maximum values of all observations, $\min_2$ and $\max_2$ refer to the desired minimum and maximum of the new scaled series. One of the desirable effects of minimum and maximum normalisation is the preservation of the relationships of the original series.

5 Training and testing the networks

The performance of the DSMIA network has been benchmarked against the performance of several other NN architectures. As this work extends the SMIA neural networks, the method of benchmarking extends that used in the SMIA work (Mahdi et al., 2009). Performance is compared with MLP, FLNN, and SONIA NN architectures. All the experiments have been performed using Matlab 2011 under Microsoft Windows 7, 64-bit operating system on 3.40 GHz processor.

Table 3 Performance metrics equations

Annualised return (%AR)	*Normalised mean squared error (NMSE)*
$AR = \frac{Profit}{All\ profit} * 100$ $Profit = \frac{252}{n} * CR, \quad CR = \sum_{i=1}^{n} R_i$ $R_i = \begin{cases} +\lvert y_i \rvert & \text{if } (y_i)(\hat{y}_i) \geq 0, \\ -\lvert y_i \rvert & \text{otherwise} \end{cases}$ $All\ profit = \frac{252}{n} * \sum_{i=1}^{n} abs(R_i)$	$NMSE = \frac{1}{\sigma^2 n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$ $\sigma^2 = \frac{1}{n-1} \sum_{i=1}^{n} (y_i - \overline{y})^2$ $\overline{y} = \sum_{i=1}^{n} y_i$
Maximum drawdown (MD)	Signal to noise ratio (SNR)
$MD = \min\left(\sum_{t=1}^{n} \left(CR_t - \max\left(CR_1, \ldots, CR_t \right) \right) \right)$ $CR_t = \sum_{i=1}^{t} R_i, \quad t = 1, \ldots, n$ $R_i = \begin{cases} +\lvert y_i \rvert & \text{if } (y_i)(\hat{y}_i) \geq 0, \\ -\lvert y_i \rvert & \text{otherwise} \end{cases}$	$SNR = 10 * \log_{10}(sigma)$ $sigma = \frac{m^2 * n}{SSE}$ $SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$ $m = \max(y_i)$
Annualised volatility (VOL)	Correct directional change (CDC)
$VOL = \sqrt{252} * \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (R_i - \overline{R})^2}$	$CDC = \frac{1}{n} \sum_{i=1}^{n} d_i$ $d_i = \begin{cases} 1 & \text{if } (y_i - y_{i-1})(\hat{y}_i - \hat{y}_{i-1}) \geq 0, \\ 0 & \text{otherwise} \end{cases}$

Notes: n is the total number of data patterns.
y and $\hat{y}$ represent the actual and predicted output value, respectively.

Several quality measures have been used as shown in Table 3, in which financial measures are used to illustrate how good is the prediction from financial point of view (i.e., is the network can generate profit or not and hence if it can be used as a financial time series predictor). Furthermore, signal processing measures are utilised to illustrate how good the neural networks are performing from signal processing point of view.

The time series data is split into three datasets, forming the training, validation, and out-of-sample sets; representative of 25%, 25%, and 50% of the total data, respectively. The prediction performance of all the networks was evaluated using three financial metrics (Dunis and Williams, 2002); judging the network predictions as fit (or otherwise) for profit generation. Three statistical metrics (Cao and Tay, 2003) are also evaluated, which provide validation of the output signals. The precise algorithms used are as shown in Table 2. The learning rate was experimentally selected between 0.1 and 0.9.

6 Simulation results and discussion

Tables 4, 5, 6 and 7 show average results from 50 simulations, conducted on out-of-sample data. These comprise stationary and non-stationary data signals predicted at one and five steps ahead of time. Non-stationary financial signals are very difficult to predict with any degree of accuracy. Non-stationary signals are highly volatile and possess great instability and noise; there are frequent changes of behaviour, including massive slumps and rises in level.

Therefore, this pattern presents great difficulty in training NNs; they are unable to respond properly to the price values' high-frequency variations. As such, all the networks, the DSMIA included, perform poorly on the annualised return (AR) measure on non-stationary signals. These indicate that when data are passed directly to the neural networks without transformation from non-stationary to stationary, all the neural networks failed to produce good simulations using financial and signal processing measures.

However, for the stationary signals, the networks predicted high percentage profits. NNs perform better on these signals as they are smoothed and undergo translation into the RDP measure, as described in Section 4. Stable predictions, and therefore, a greater profit measure are possible when evaluating against stationary signals.

As it can be seen from Figures 3 and 4, the proposed DSMIA generated profit when used for financial time series prediction. The results indicated that the network generated compatible results with the functional link neural network for one step ahead prediction for stationary data and demonstrated advantaged results when compared with other NN for five step ahead prediction on stationary data. This is an indication that the recurrent links of the proposed networks stored information about the previous values of the signal and hence better prediction was achieved for higher step prediction values in comparison to other neural network architectures. Furthermore, as it can be shown from Figures 5 and 6, the proposed network indicted better results with non-stationary data forecasting for one and five steps ahead prediction.

Price time-series data acquired from three stock opening and closing (six in total) prices has been used in this study: NASDAQ-O and –C (opening and closing), DJIA-O and –C, and DJUA-O and –C. The inclusion of both opening and closing price signals is to investigate the differences between predictions for opening and closing stock results.

Results for the non-stationary signals show that in most cases, the prediction results for one and five steps ahead show a small difference between opening and closing stock prices. It is worth noting these differences are related to the raw data; itself affected by external factors, such as economic climate, blue-chip company financial announcements, and threats of war/civil unrest.

As can be seen in Tables 5 and 6, the DSMIA NN generated profit-based predictions on one and five-steps ahead. For stationary data, five-steps-ahead predictions, DSMIA generated greater profits than the majority of benchmarked NN algorithms.

Table 4 Average results on stationary signals for the prediction of one-step ahead

Performance measures	*Neural networks*	*US/UK*	*JP/US*	*US/EU*	*NASDAQO*	*NASDAQC*	*DJIAO*	*DJIAC*	*DJUAO*	*DJUAC*	*OIL*
AR(%)	MLP	65.14634	64.72067	69.91986	60.5218	62.32483	59.49555	58.85693	52.97216	52.01663	51.19201
	FLNN	78.18972	77.67146	78.58212	67.91466	67.16377	74.08653	73.57757	72.81365	70.84033	73.64221
	SONIA	73.15769	76.19371	71.00268	62.66176	63.35985	63.71905	62.29099	67.01270	68.02663	72.16195
	DSMIA	76.40196	76.28788	72.941440	66.466713	62.866065	64.859029	66.071119	69.986696	70.593948	73.352611
MD	MLP	–1.4304	–1.5806	–1.7923	–4.8447	–4.8762	–4.4906	–2.9165	–6.1459	–5.6157	–19.024
	FLNN	–1.14585	–1.03966	–0.98186	–8.37879	–5.31515	–1.87348	–1.88483	–2.51275	–2.974	–8.70562
	SONIA	–1.51535	–1.24388	–2.69518	–8.84982	–8.32449	–6.60512	–7.99077	–3.55542	–4.17998	–8.32751
	DSMIA	–1.29011	–1.50278	–1.91315	–6.89071	–8.230259	–7.57360	–6.60576	–3.66422	–3.41091	–7.08040
AV	MLP	4.526048	5.717462	4.666936	13.27316	12.58836	11.18889	11.27457	12.48506	12.56166	67.3681
	FLNN	4.204148	5.379993	4.428998	12.92789	12.31514	10.48708	10.56609	11.63671	11.78889	59.43905
	SONIA	4.346859	5.424529	4.63746	13.20117	12.53349	11.00119	11.12426	11.93661	11.94297	60.15546
	DSMIA	4.256646	5.420273	4.590001	12.999914	12.560987	10.951396	10.952866	11.787191	11.802643	59.745353
SR	MLP	14.50009	11.36989	15.00254	4.575426	4.955069	5.322315	5.22938	4.267582	4.169173	0.766643
	FLNN	18.59827	14.43758	17.74275	5.253586	5.453892	7.064812	6.963861	6.257906	6.010538	1.246236
	SONIA	16.83866	14.04827	15.34295	4.742449	5.057593	5.800506	5.610039	5.614729	5.678424	1.20381
	DSMIA	17.95116	14.083690	15.897520	5.117712	5.007122	5.928563	6.038112	5.937937	5.981827	1.228245
SNR(dB)	MLP	20.35	21.75	22.29	23.7	22.83	24.45	23.26	23.84	23.9	21.03
	FLNN	22.95	23.99	24.26	25.07	23.51	24.99	24.99	25.06	25.15	23.13
	SONIA	21.93	23.41	23.04	24.14	22.75	23.61	23.53	25.02	25.03	22.2
	DSMIA	22.43	23.36	23.08	24.49	22.65	23.72	23.67	25.31	25.35	22.69
CDC	MLP	67.97	62.59	62.95	65.87	64.53	63.36	64.88	63.34	64.61	61.36
	FLNN	66.46	63.05	62.14	66.94	63.39	63.51	64.44	59.56	60.08	58.88
	SONIA	66.09	62.35	63.07	66.82	62.39	61.92	62.98	60.48	59.95	62.76
	DSMIA	65.88	59.77	62.80	67.62	62.03	60.38	61.65	60.56	61.11	61.67

Table 5 Average results on stationary signals for the prediction of five-step ahead

Performance measures	*Neural networks*	*US/UK*	*JP/US*	*US/EU*	*NASDAQO*	*NASDAQC*	*DJIAO*	*DJIAC*	*DJUAO*	*DJUAC*	*OIL*
AR(%)	MLP	84.72936	81.19235	91.46321	77.57114	83.09106	74.77388	65.96346	74.07559	68.93831	75.89575
	FLNN	77.68125	86.30544	86.37381	85.91437	85.81948	88.28172	88.31142	87.44099	86.69399	93.69838
	SONIA	88.15049	87.17208	91.84363	85.16167	85.02744	85.3598	84.89345	81.28562	86.66031	91.82156
	DSMIA	89.047661	87.201969	92.028790	86.323692	85.105993	86.629872	86.106254	87.135184	87.517079	92.039333
MD	MLP	–3.15533	–3.54864	–1.78913	–10.1308	–7.34318	–8.24405	–13.0605	–11.1968	–20.4777	–8.03638
	FLNN	–5.87622	–2.72277	–5.14140	–6.84715	–7.08089	–3.82649	–3.72506	–3.25588	–5.36821	–8.75378
	SONIA	–3.2779	–2.72278	–1.39054	–6.19597	–7.36317	–8.41446	–8.37277	–7.78359	–3.70212	–12.60818
	DSMIA	–2.963665	–2.722772	–1.432039	–6.109268	–7.658861	–9.305728	–9.268669	–3.357676	–3.410785	–13.3041 –11.644154
AV	MLP	16.24499	17.56791	15.69251	37.68809	36.53018	33.66308	35.11867	38.27525	39.02029	194.2871
	FLNN	17.04838	16.88852	16.39043	35.7455	35.85963	30.99444	31.03799	35.79927	36.26686	152.054
	SONIA	15.87492	16.76482	15.6362	35.93796	36.06489	31.63171	31.78481	37.03390	36.27336	157.372
	DSMIA	15.757916	16.760712	15.607786	35.637459	36.041823	31.347378	31.426138	35.864341	36.089621	156.745293
SR	MLP	5.246373	4.62544	5.829595	2.063196	2.278035	2.225729	1.888711	1.941499	1.786946	0.390889
	FLNN	4.569979	5.110325	5.289899	2.403571	2.393323	2.848333	2.845357	2.442637	2.390524	0.61651
	SONIA	5.561351	5.199892	5.873841	2.369793	2.357881	2.633558	2.672691	2.564821	2.389255	0.584465
	DSMIA	5.661860	5.202832	5.896499	2.422628	2.361925	2.766082	2.754751	2.429673	2.425167	0.588284
SNR(dB)	MLP	23.14	22.56	26.71	24.46	24.93	23.73	23.09	24.53	24.21	20.24
	FLNN	24.14	22.61	24.37	26.5	25.9	27.1	27.15	27.55	27.39	25.31
	SONIA	24.98	23.02	23.41	25.22	24.33	24.81	24.68	26.28	27.27	25.37
	DSMIA	24.85	22.30	23.38	24.93	24.00	24.72	25.62	27.10	27.36	25.02
CDC	MLP	63.7	62.18	64.65	61.47	59.92	61.92	61.45	60.86	59.66	56.88
	FLNN	63.47	64.44	63.17	61.23	60.72	63.48	63.84	62.61	63.09	56
	SONIA	65.49	63.16	65.09	62.49	59.48	62.91	62.99	61.72	62.97	60.06
	DSMIA	65.93	63.48	65.23	61.85	59.39	63.27	63.54	62.35	62.41	61.21

Table 6 Average results on non-stationary signals for the prediction of one-step ahead

Performance measures	*Neural networks*	*US/UK*	*JP/US*	*US/EU*	*NASDAQO*	*NASDAQC*	*DJIAO*	*DJIAC*	*DJUAO*	*DJUAC*	*OIL*
AR(%)	MLP	6.669613	–0.473902	7.703251	–6.27845	–10.20173	–9.977635	–12.48747	–7.6654061	–6.666097	4.151457
	FLNN	1.993203	–0.905553	9.82592	–3.3082592	–5.49312	–6.546531	–9.72016	–6.254837	–6.376194	–6.630352
	SONIA	2.073759	–8.271959	2.414358	–3.619703	3.531254	–15.34971	–15.34804	–5.068463	–5.201504	24.689167
	DSMIA	5.075575	–4.069113	5.498935	–1.201741	7.886264	–3.891887	4.078142	–1.449810	0.355431	26.762486
MD	MLP	–14.957188	–22.500975	–11.293929	–15.470936	–70.907413	–63.928973	–73.355125	–67.988187	–67160675	–13.563781
	FLNN	–18.341226	–21.509273	–10.26734	–46.467443	–51.820357	–49.59282	–66.6875	–51.65052	–51.482152	–100.89227
	SONIA	–19.83157	–32.49779	–13.49779	–64.55476	–49.69712	–88.91595	–92.87567	40.012786	–45.16187	–50.27416
	DSMIA	–16.272571	–15.434371 –25.576244 –12.737400	–9.23998 –10.787676 –14.032484	–30.243216 –51.476937 –49.915661	–40.470801 –35.417009 –35.623171	–31.948202 –64.331777 –26.912380	–14.144426 –61.632751 –31.030560	–51.271605 –48.281534 –50.142334	–44.666601 –34.473808 –40.291306	–51.651684 –43.942950 –24.269949
AV	MLP	11.403352	12.802219	10.106966	30.732116	30.476655	27.480595	27.88965	29.921184	30.590123	33.256952
	FLNN	11.410718	12.802582	10.095021	30.780716	30.565166	27.535189	23.240831	29.992717	30.660917	131.37748
	SONIA	11.399141	12.77795	10.105815	30.733801	30.472487	27.361664	27.773873	29.9737	30.642176	128.19774
	DSMIA	11.383173	12.794142	10.101788	30.738439	30.495996	27.500631	27.869717	30.643680	29.981559	128.001300
SR	MLP	0.585399	–0.037	–0.762452	–0.20439	–0.3351	–0.36342	–0.44844	–0.25648	–0.21824	–0.000123
	FLNN	0.175035	–0.07084	0.974008	0.107544	–0.17996	–0.23811	–0.41859	–0.20877	–0.20809	–0.05142
	SONIA	0.182488	–0.64743	0.239181	–0.1177	0.115987	–0.56231	–0.55418	–0.16912	–0.16979	0.193607
	DSMIA	0.446797	–0.318106	0.544569	–0.039112	0.259041	–0.141980	0.146780	–0.047426	0.011822	0.209142
SNR(dB)	MLP	16.3	17.68	13.41	17.74	19.38	12.7	13.07	14.19	14.94	19.08
	FLNN	29.26	25.72	30.06	31.13	30.84	29.4	24.76	32.86	32.53	9.04
	SONIA	20.5	27.04	13.97	19.83	16.05	16.97	18.51	16.51	16.88	11.44
	DSMIA	17.05	24.24	11.97	15.45	17.87	14.87	13.07	12.72	12.92	11.26
CDC	MLP	52.78	47.7	52.34	47.6	48.06	48.41	47.66	49.04	48.58	51.34
	FLNN	51.31	46.63	52.13	47.54	48.64	49.22	48.47	48.25	47.28	46.29
	SONIA	50.31	45.45	52.79	47.77	50.55	46.1	45.56	48.16	47.63	57.1
	DSMIA	50.80	47.46	51.40	49.20	52.25	51.67	51.10	50.92	51.48	57.32

Table 7 Average results on non-stationary signals for the prediction of five-step ahead

Performance measures	*Neural networks*	*US/UK*	*JP/US*	*US/EU*	*NASDAQO*	*NASDAQC*	*DJIAO*	*DJIAC*	*DJUAO*	*DJUAC*	*OIL*
AR (%)	MLP	–0.559585	–3.59391	1.117382	–2.40269	1.65726	–0.321544	–0.555403	–3.439273	–3.523605	0.01561
	FLNN	–1.2738	–5.9573	–0.1571	–3.8363	–0.915	–1.3329	–4.0139	–2.9823	–2.9541	–6.3419
	SONIA	–1.48508	2.839703	2.819774	–5.12982	–1.00699	2.440176	1.880855	2.857381	1.735058	–6.13704
	DSMIA	1.824697	4.883264	5.924942	0.102613	–0.645413	2.444301	2.555257	4.642005	4.005996	11.290324
MD	MLP	–14.8441	–20.2364	–11.8548	–37.1925	–34.1537	–37.7277	–40.42709	–49.0705	–51.062871	–88.71052
	FLNN	–16.0545	–20.2988	–12.3168	–38.3696	–40.0468	–39.5673	–47.5507	–50.6108	–15.8044	–94.3542
	SONIA	–17.776	–19.8414	–8.63958	–38.6083	–32.3624	–20.558	–22.1424	–35.7076	–34.5416	–90.0799
	DSMIA	–13.364766	–14.536049	–8.593920	–37.366685	–32.550353	–21.343170	–20.984023	–30.055884	–30.602317	–87.792361
AV	MLP	11.42579	12.8158	10.1309	30.8166	30.5985	27.5912	27.5927	29.9932	30.70504	131.6793
	FLNN	11.4246	12.8066	10.1311	30.8154	30.6204	27.5939	23.288	30.0396	30.7071	131.135
	SONIA	11.41287	12.7995	10.11326	30.77346	30.59728	27.58149	28.00402	29.9849	30.67759	130.3251
	DSMIA	11.417538	12.797590	10.109854	30.767597	30.602293	27.579252	28.001119	29.992023	30.662968	130.448316
SR	MLP	–0.049097	–0.28073	0.11028	–0.07803	0.054188	–0.011719	–0.020069	–0.11477	–0.114946	–0.000611
	FLNN	–0.1116	–0.4654	–0.0155	–0.1245	–0.0299	–0.0483	–0.1725	–0.0993	–0.0962	–0.0484
	SONIA	–0.13091	–0.22183	0.378768	–0.16675	–0.1253	0.088481	0.067178	0.095616	0.056716	–0.04803
	DSMIA	0.159847	0.381776	0.586446	0.003337	–0.021085	0.088642	0.091279	0.154933	0.130994	0.087339
SNR(dB)	MLP	14.7	16.34	13.22	16.54	17	12.39	12.42	14.07	15.49	10.6
	FLNN	15.3	16.51	14.02	23.88	27.7	26.59	22.23	29.39	29.51	9.59
	SONIA	16.83	21.66	13.32	19.94	17.1	16.83	17.14	13.38	14.88	10.92
	DSMIA	15.30	22.04	11.57	16.73	18.99	19.07	12.83	13.48	13.76	10.78
CDC	MLP	52.27	47.95	50.79	48.44	49.79	52.28	51.27	49.57	49.44	49.42
	FLNN	52.52	46.98	50.91	48.04	50.99	52.62	49.44	49.3	49.27	48.58
	SONIA	53.2	50.98	51.89	48.05	48.41	52.54	52.36	50.15	49.83	49.03
	DSMIA	51.52	51.52	52.27	50.45	49.45	52.73	51.60	51.13	50.33	53.00

Figure 2 Size of changes in three data datasets compared with the percentage of profit from the whole dataset available within those changes (see online version for colours)

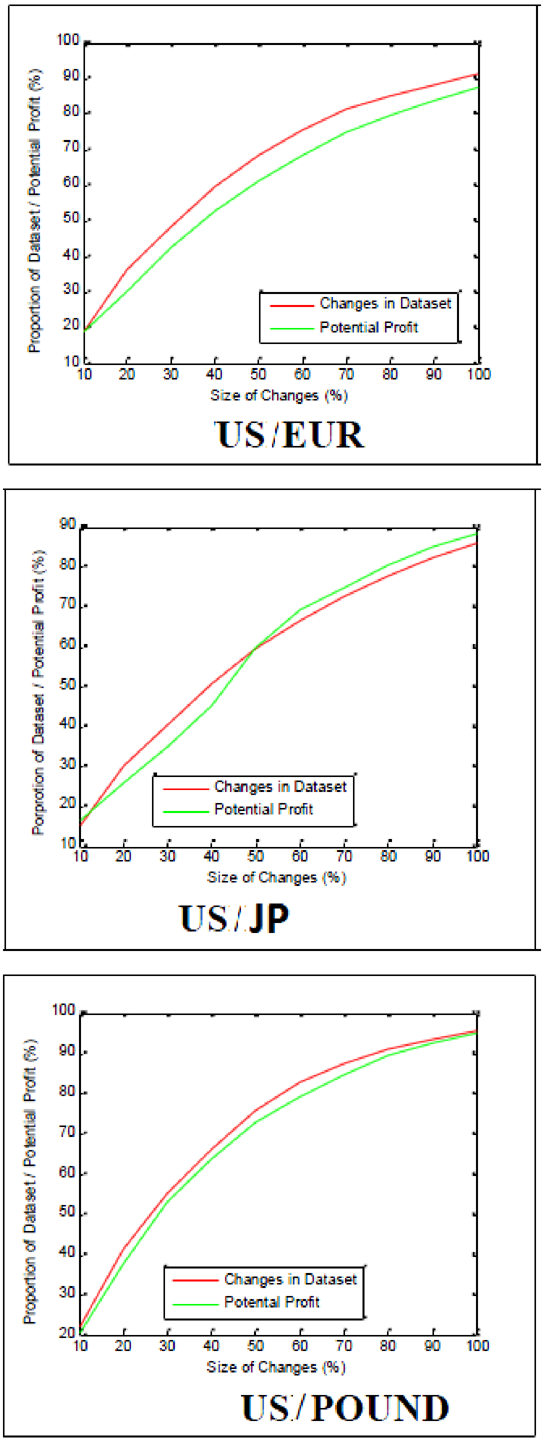


Figure 3 Annualised return on stationary signals for the prediction of one-step ahead (see online version for colours)

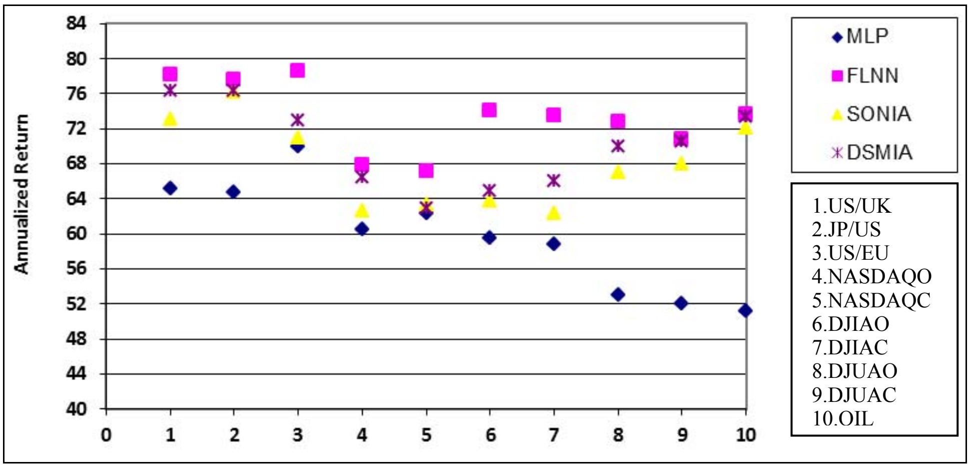


Figure 4 Annualised return on stationary signals for the prediction of five-step ahead (see online version for colours)

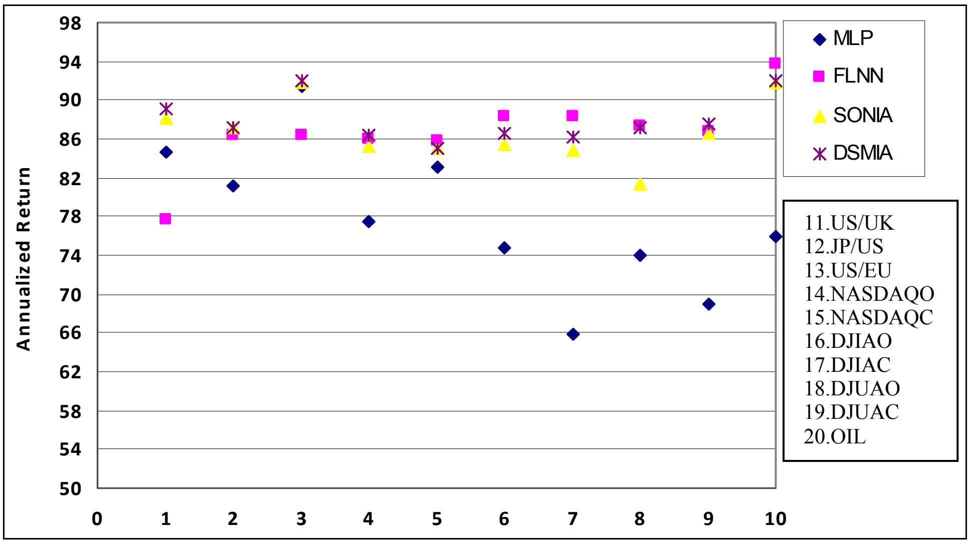


Figure 5 Annualised return on non-stationary signals for the prediction of one-step ahead (see online version for colours)

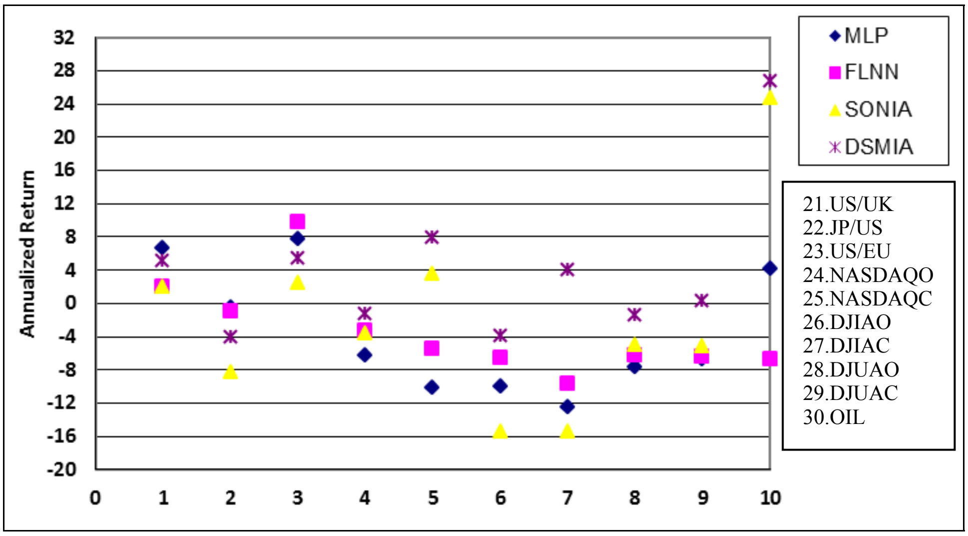
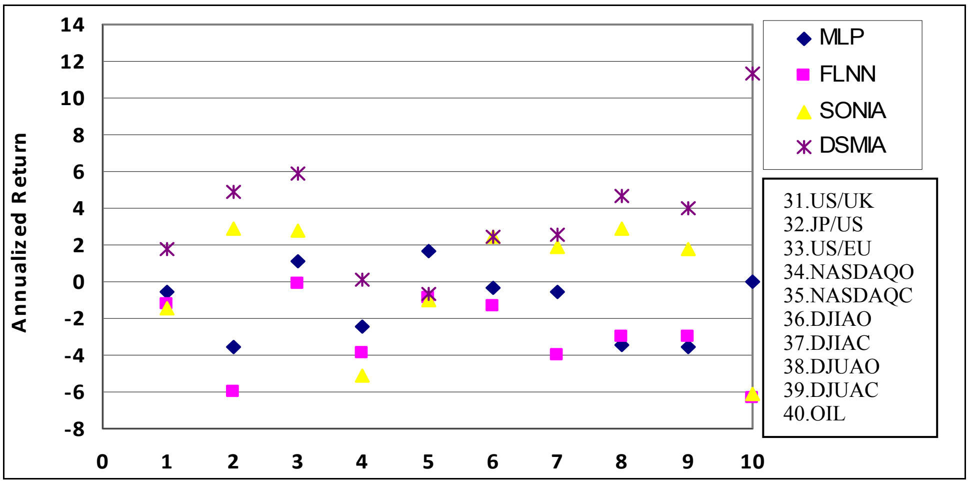


Figure 6 Annualised return on non-stationary signals for the prediction of five-step ahead (see online version for colours)



In the DSMIA network, the learning is centred on the local properties of the signal; the self-organising layer is specifically programmed to adapt to these properties. As such, DSMIA-architecture networks have a more detailed mapping of the underlying data structure, enabling them to respond better to the data changes or structural shifts common in non-stationary data. Furthermore, the proposed recurrent links provide the network with memory enabling it to remember past behaviours and hence the network managed to generate better results for non-stationary and stationary time series prediction in comparison to the benchmarked networks. However, this improvement was achieved on the expenses of longer processing and training time as shown in Table 8.

Table 8 A comparison between the time required to complete the simulation using the SONIA and the proposed DSMIA network

	SONIA (Sec)	*DSMIA (Sec)*
USDUK	*25.2207*	64.0190
JPYUSD	*33.2589*	79.6417
USDEUR	*30.1757*	72.6679
NASDAQO	*28.9414*	42.1859
NASDAQC	*30.7793*	71.9675
DJIAO	*29.6505*	36.7067
DJIAC	*29.3825*	70.2133
DJUAO	34.0757	81.5248
DJUAC	32.6150	70.8630
OIL	4.9141	10.7170

Figure 2 plots the size of change in three selected datasets: the US/EUR, US/GBP, and US/JPY. This is shown against the percentage of profit from the dataset available in those changes. It indicates that given larger dataset changes, the potential profit (on correct predictions) increases. For example, even with a 100% correct prediction rate, the available profit is variable. In US/EUR and US/GBP datasets, potential profit is always less than the size of change. In the US/JPY set, potential profit increases when the size of change is about 50%. Thus, correct predictions are much more valuable on this dataset.

This significant effect on AR has wider ranging implications when considered alongside the different behaviour of the NNs. For several of the benchmarked NN architectures, the behavioural objective of the training is to minimise error over the whole training dataset. However, the training effect on the self-organising layer's learning is localised; the network's concern is primarily with current data values. As such, the network is better placed to predict on these temporally localised events.

7 Summary

This research work underlines an important contribution of a new recurrent self-organising multilayer neural network inspired by artificial immune systems. In this case a recurrent links from the output layer to the input layer was proposed which allows the network to have memory. It is applied to and evaluated against prediction of values in a financial time series; namely its ability to approximate non-linear financial series values. The network has shown its advantages in forecasting both stationary and non-stationary signals, particularly regarding temporally local training behaviour.

The purpose of the proposed network was to look at the improvement achieved when using recurrent links into the structure of the SONIA network. The simulation results indicated that the proposed network achieved improved results when compared to the SONIA network. Furthermore the network generated profits (using the annualised return as a financial measure) for the non-stationary data prediction while most benchmarked networks fails to do so. This is because the proposed network is able to look at the temporal locality of the signal and extract the required information while other network such as the MLP is more capable of predicting the overall trends in the signals.

Early work on applying a regularisation scheme to the DSMIA network is providing encouraging results using a weight-decay between the hidden and output nodes. As such, future work will further investigate and assess an improved, regularised-DSMIA scheme. While weight-decay is not without its performance-related problems, work by others has shown that it can help to avoid over-fitting the network to training data; as such, improving the network (Duda et al., 2001). Another direction of research is to investigate the best choice of network architecture, which includes the number of inputs and the use of higher order terms in the input units similar to the FLNN. This may improve the performance of the proposed network since this can extend the input space into higher dimensional spaces where linear separability is possible.

References

Cao, L. and Tay, F. (2003) 'Support vector machine with adaptive parameters in financial time series forecasting', *Neural Networks, IEEE Transactions on*, Vol. 14, No. 6, pp.1506–1518 [online] http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1257413 (accessed 12 July 2013).

Cass, R. and Radl, B. (1996) 'Adaptive process optimization using functional-link networks and evolutionary algorithm', *Control Engineering Practice*, Elsevier, Vol. 4, No. 11, pp.1579–1584.

Castiglione, F. (2001) 'Forecasting price increments using an artificial neural network', *Advances in Complex Systems*, Vol. 1, pp.1–12 [online] http://www.worldscientific.com/doi/abs/10.1142/S0219525901000097 (accessed 12 July 2013).

Chen, A. and Leung, M. (2005) 'Performance evaluation of neural network architectures: the case of predicting foreign exchange correlations', *Journal of Forecasting*, Vol. 24, No. 6, pp.403–420 [online] http://onlinelibrary.wiley.com/doi/10.1002/for.967/abstract (accessed 12 July 2013).

Datastream International (2005) *Financial Time Series Data*.

Duda, R., Hart, P. and Stork, D. (2001) *Pattern Classification*, 2nd ed., John Wiley & Sons, New York [online] http://books.google.co.uk/books?hl=en&lr=&id=Br33IRC3PkQC&oi=fnd&pg=PR3&dq=DUDA+2001+pattern+classification&ots=2vDROsd6Gl&sig=kAqSIixWfD-GtqwHVafANejuyss (accessed 12 July 2013).

Dunis, C. and Williams, M. (2002) 'Modelling and trading the EUR/USD exchange rate: do neural network models perform better?', *Derivatives Use, Trading and Regulation*, Vol. 8, No. 3, pp.1–24 [online] https://www.jmu.ac.uk/AFE/AFE_docs/cibef0202.pdf (accessed 12 July 2013).

Fei, G. and Yu, Y.L. (1994) 'A modified Sigma-Pi BP Network with self-feedback and its application in time series analysis', *Proc. SPIE 2243, Applications of Artificial Neural Networks V*, 2243, pp.508–515.

Ghazali, R. and Hussain, A.J. (2009) 'Non-stationary and stationary prediction of financial time series using dynamic ridge polynomial neural network', *Neurocomputing*, Vol. 72, Nos. 10/12, pp.2359–2367 [online] http://www.sciencedirect.com/science/article/pii/S0925231208005596 (accessed 12 July 2013).

Giles, C. and Maxwell, T. (1987) 'Learning, invariance, and generalization in high-order neural networks', *Applied Optics*, Vol. 26, No. 23, pp.4972–4978 [online] http://www.opticsinfobase.org/abstract.cfm?&id=30898 (accessed 12 July 2013).

Giles, C., Griffin, R. and Maxwell, T. (1998) 'Encoding geometric in variances in HONN A. I. of physics, ed. American Institute of Physics, USA, pp.310–309 [online] http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Encoding+geometric+invariances+in+HONN#0 (accessed 12 July 2013).

HellstrSom, T. and HolmstrSom, K. (1998) *Predicting the Stock Market* [online] http://gillesdaniel.com/papers/1998.Hellstrom.Predicting the stock market.pdf (accessed 12 July 2013).

Ho, S., Xie, M. and Goh, T.N. (2002) 'A comparative study of neural network and Box-Jenkins ARIMA modeling in time series prediction', *Computers & Industrial Engineering*, Vol. 42, Nos. 2/4, pp.371–375 [online] http://www.sciencedirect.com/science/article/pii/S0360835202000360 (accessed 12 July 2013).

Hussain, A. and Al-Jumeily, D. (2007) 'How good is the backpropogation neural network using a self-organised network inspired by immune algorithm (SONIA) when used for multi-step financial time series prediction? Advances in neural networks – ISNN 2007, *Lecture Notes in Computer Science*, Springer, 4492, pp.921–930 [online] http://link.springer.com/chapter/10.1007/978-3-540-72393-6_110 (accessed 12 July 2013).

Hussain, A. et al. (2008) 'Financial time series prediction using polynomial pipelined neural networks', *Expert Systems with Applications*, Elsevier, Vol. 35, No. 3, pp.1186–1199 [online] http://www.sciencedirect.com/science/article/pii/S0957417407003442 (accessed 12 July 2013).

Jordan, M.I. (1997) 'Serial order: a parallel distributed processing approach', *Advances in Psychology: Neural-Network Models of Cognition-Biobehavioral Foundations*, Vol. 121, pp.471–495 [online] http://www.sciencedirect.com/science/article/pii/S0166411597801112 (accessed 12 July 2013).

Kaastra, I. and Boyd, M. (1996) 'Designing a neural network for forecasting financial and economic time series', *Neurocomputing*, Vol. 10, No. 3, pp.215–236 [online] http://dx.doi.org/10.1016/0925-2312(95)00039-9 (accessed 12 July 2013).

Kaita, T., Tomita, S. and Yamanaka, J. (2002) 'On a higher-order neural network for distortion invariant pattern recognition', *Pattern Recognition Letters*, Vol. 23, No. 8, pp.977–984. [online] http://www.sciencedirect.com/science/article/pii/S0167865502000284 (accessed 12 July 2013).

Mahdi, A. et al. (2009) 'The application of the neural network model inspired by the immune system in financial time series prediction', *In 2009 Second International Conference on Developments in eSystems Engineering*, Abu Dhabi: IEEE, pp.370–376 [online] http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5395148 (accessed 18 June 2013).

Mirea, L. and Marcu, T. (2002) 'System identification using functional-link neural networks with dynamic structure', in L. Basañez (Ed.): *15th Triennial World Congress*, Barcelona, Spain, pp.443–443 [online] http://www.nt.ntnu.no/users/skoge/prost/proceedings/ifac2002/data/content/02111/2111.pdf (accessed 12 July 2013).

Shachmurove, Y. and Witkowska, D. (2000) *Utilizing Artificial Neural Network Model to Predict Stock Markets* [online] http://www.researchgate.net/publication/4821332_Utilizing_Artificial_Neural_Network_Model_to_Predict_Stock_Markets/file/e0b49514ca0a568aee.pdf.

Thimm, G. (1997) *Optimization of High Order Perceptrons*, Swiss Institute of Technology (EPFL) [online] http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.48.225&rep=rep1&type=pdf (accessed 12 July 2013).

Thomason, M. (1999) 'The practitioner method and tools', *Journal of Computational Intelligence in Finance*, Vol. 7, No. 3, pp.36–45.

Timmis, J. (2000) *Artificial Immune Systems: A Novel Data Analysis Technique Inspired by the Immune Network Theory*, University of Wales [online] http://ethos.bl.uk/OrderDetails.do?uin=uk.bl.ethos.436207 (accessed 12 July 2013).

Voegtlin, T. (2002) 'Recursive self-organizing maps', *Neural Networks*, Elsevier, Vol. 15, Nos. 8/9, pp.979–991 [online] http://www.sciencedirect.com/science/article/pii/S0893608002000722 (accessed 12 July 2013).

Widyanto, M.R. et al. (2005) 'Improving recognition and generalization capability of back-propagation NN using a self-organized network inspired by immune algorithm (SONIA)', *Applied Soft Computing*, Vol. 6, No. 1, pp.72–84 [online] http://www.sciencedirect.com/science/article/pii/S1568494604001127 (accessed 12 July 2013).

Yao, J. and Tan, C.L. (2000) 'A case study on using neural networks to perform technical forecasting of forex', *Neurocomputing*, Elsevier, Vol. 34, Nos. 1–4, pp.79–98 [online] http://linkinghub.elsevier.com/retrieve/pii/S0925231200003003 (accessed 12 July 2013).

Yao, J., Poh, H. and Jasic, T. (1996) 'Foreign exchange rates forecasting with neural networks', *International Conference on Neural Information Processing*, Springer-Verlag, Hong Kong, pp.754–759 [online] http://www.smartquant.com/references/NeuralNetworks/neural31.pdf.

Yümlü, S., Gürgen, F.S. and Okay, N. (2005) 'A comparison of global, recurrent and smoothed-piecewise neural models for Istanbul stock exchange (ISE) prediction', *Pattern Recognition Letters*, Elsevier, Vol. 26, No. 13, pp.2093–2103 [online] http://linkinghub.elsevier.com/retrieve/pii/S0167865505000875 (accessed 18 June 2013).