Neural Networks for Event Detection from Time Series: A BP Algorithm Approach

Dayong Gao¹, Y. Kinouchi¹, and K. Ito²

¹ Faculty of Engineering

² Faculty of the Integrated Arts and Sciences
The University of Tokushima, Japan

Abstract. In this paper, a relatively new event detection method using neural networks is developed for financial time series. Such method can capture homeostatic dynamics of the system under the influence of exogenous event. The results show that financial time series include both predictable deterministic and unpredictable random components. Neural networks can identify the properties of homeostatic dynamics and model the dynamic relation between endogenous and exogenous variables in financial time series input-output system. We also investigate the impact of the number of model inputs and the number of hidden layer neurons on forecasting. . . .

1 Introduction

Neural networks have been applied in time series to improve multivariate prediction ability [1–3]. Neural networks map an input space (present and past values of the time series) onto an output space (future value). Future values of variables often depend on their past values, the past values of other correlated variables, and other uncertain factors. For example, the future price of a stock depends on political and international events as well as various economic indicators.

Most studies on time series analysis using neural networks address the use of neural networks as a forecasting tool, but limited research focuses on the application of neural networks as a data mining technique to extract event patterns from time series. These forecasts rely on historical trends or past experiences to predict the future. The measurement of past trends may be interrupted by changes in policy or data availability. When the past cannot be used to predict the future due to major unexpected shifts in economic, political, or social conditions, the future can be more uncertain due to the occurrence of major events. Evaluating the impact of these events (exogenous variables) on stock markets are interesting to economists, because the markets are affected by less measurable but important forces in economic analysis and financial prediction.

In this paper, a systematic approach for time series using neural networks is developed to generate an event sequence and to model the dynamic relation between stock market index (endogenous variables) and macroeconomic environment (exogenous variables). Because time series related to the input-output

Fig. 1. Neural Network Configuration and Nonlinear Neuron Model (*Inset*). The $input(x) - output(y)$ relation of nonlinear neurons is given by a conventional sigmoid function such as $y = \tanh(x)$. The $input(x)$ is expressed by a form $\Sigma(w_i x_i)$, where w_i is a weight for an input component x_i .

system of financial markets may include non-linearity, back propagation (BP) neural networks are used to clarify basic predictive properties. This model can acquire automatically the dynamics included in the data by use of an error back propagation (BP) algorithm [4].

2 Methodological Consideration

2.1 Identification by Neural Networks

Identification by neural networks for time series data $x_1, x_2, \dots, x_k, \dots, x_N$ requires predicting or estimating a datum x_k based on preceding observed n data $x_{k-n}, \dots, x_{k-2}, x_{k-1}$, *i. e.*, prediction by a n th order MA model, where $x_1, x_2, \dots, x_k, \dots, x_N$ are discrete time sequences. In order to implement the prediction, a multilayer neural network model as shown in *Fig. 1* is used. The network has one input layer $i_1 = x_{k-n}, \dots, i_n = x_{k-1}$ and one output layer x_k .

Each weight w_i is adjusted to coincide with output o_k and target x_k by a gradient method. For error function F defined by

$$F = \sum_{k \in S_t} \varepsilon_k^2, \quad \varepsilon_k = o_k - x_k \quad (1)$$

where S_t is a set of training patterns, a correction δw_i of w_i to minimize error function is given by

$$\delta w_i = -\eta \frac{\partial F}{\partial w_i} \quad (2)$$

where η is the learning rate parameter, a small positive constant. To calculate δw_i for neurons in hidden layers, an error back propagation (BP) algorithm is used to train the network [4]. After training, the network is tested by the use of the rest patterns to verify whether the network acquires source dynamics, *i. e.*, generalization. If the network output for each test pattern is almost close to

Table 1. Data Acquisition and Normalized Results. N is the maximum number of trading days for TOPIX during 1995–2000; m is the normalized factor. Min, Mean, Max and MaxDev are the minimum, mean, maximum and maximum deviation of acquired and normalized data respectively.

Data Acquisition				
N	Min	Mean	Max	MaxDev
1481	980.00	1415.89	1754.78	435.89
Normalized Results				
m	Min	Mean	Max	MaxDev
500	-0.87178	-4.2E-06	0.67778	0.87178

the respective target, the network is considered to have acquired the underlying dynamics from the training patterns.

2.2 Data Acquisition

The most popular treatment of input data is to feed neural networks with either the data at each observation, or the data from several successive observations. In this paper, we consider daily data of TOPIX (Tokyo Stock Price Index) from Japanese stock market in the period 1995–2000.

Succeeding data of TOPIX, denoted by t_k , are converted to the deviation from their mean value and scaled by a factor m to normalize their values in the range of -1 to 1 by equation:

$$x_k = \frac{t_k - \bar{t}}{m}, \quad k = 1 \sim N \tag{3}$$

where k is the number of trading days, corresponding to 6 years from 1995 to 2000 and N is the maximum number. $\bar{t}$ is the average of t_k .

In Table 1, historical data of TOPIX are divided into training phases and test phases, with each phase containing half of the collected data. The data of the earlier period (1995-1997) are used for training; the data of the latter period (1998-2000) are used for test.

Real world data contain varying degree of noise. To demonstrate our method and to compare with real data from stock markets, two types of the data except TOPIX are generated as follows: 1) random data with uniform distribution between -1 to 1, and 2) the data from a sinusoidal wave, *i. e.*, deterministic data, with unit amplitude at an interval of $T/60.1$ (T : period). These two types of generated data represent a sequence of true random numbers where no model has a chance and simple noise-free data, respectively.

2.3 Model Identification

Various measurement methods such as approximate entropy, auto-correlation, average mutual information, and number of false nearest neighbors can be used

to evaluate time series data [5]. These methods can quantify data regularity and randomness if there are large amounts of the data. However, stock market data are relatively limited and this study uses even more limited data containing various market environments at fixed periods. If the data periods are too long, market environments change much; financial patterns and the neural network models also change much.

The following “ R -method” of quantifying data regularity and randomness can be applied to a relatively small amounts of the data. This R -method can also be used to measure multi-valued property of the data and determine neural network configuration. To calculate values of R , let x denote the input vector $(i_1, i_2, \dots, i_{N-n})$ and y denote the output o . The data produce input vectors $x_1, x_2, \dots, x_{N-n}$ and corresponding outputs $y_1, y_2, \dots, y_{N-n}$. If x_i and x_j are similar, then y_i and y_j are also similar for deterministic data, but y_i and y_j may be quite different for random data and multi-valued function. The change of $|y_j - y_i|$ for y_i corresponds to all x_j in the neighborhood of each x_i as a measurement of data randomness. Values of R are defined by

$$R = \frac{1}{\sum_i |\Delta_i|} \sum_{i=1}^{N-n} \sum_{j \in \Delta_i} f(z_{ji}) \quad (4)$$

where

$$\begin{aligned} \Delta_i &= \{j | |x_i - x_j| / \sqrt{n} x_{rms} \leq \beta, j \neq i\}, \quad \text{if } x_j \text{ exists} \\ &= \{j | \min |x_i - x_j| / \sqrt{n} x_{rms}, j \neq i\}, \quad \text{otherwise} \end{aligned} \quad (5)$$

Here, $|\Delta_i|$ represents the number of x_j in the neighborhood Δ_i . The x_{rms} is a root mean square value for the data x_k (Eq.1), and the y_{rms} is a root mean square value for the data y_i . The $\sqrt{n} x_{rms}$ is the mean length of the vector x_i taken in the n dimensional space. The z_{ji} is given by

$$z_{ji} = (|y_j - y_i|) / (y_{rms} / \sqrt{n} x_{rms}) \quad (6)$$

The function $f(z_{ji})$ is a logistic function as following:

$$\begin{aligned} f(z_{ji}) &= 0, \quad \text{if } |x_j - x_i| / \sqrt{n} x_{rms} \leq \beta \text{ and } |y_j - y_i| / y_{rms} \leq \beta \\ &= 1 - (1 + \alpha z_{ji}) \exp(-\alpha z_{ji}), \quad \text{otherwise} \end{aligned} \quad (7)$$

where α and β are positive constants. If the data $x_1, x_2, \dots, x_N$ are obtained by sampling a deterministic single-valued time function, using Eq.7, $f(z_{ji}) = 0$, because $y_i \approx y_j$ for all x_j existing in the neighborhood Δ_i of any x_i . Values of R are zero when the data number N is relatively large. If the data come from a random function, $f(z_{ji}) = 1$ because y_j is independent of y_i , therefore $R=1$ for the relatively large data numbers. R for any data will have a value between zero and one, indicating the degree of randomness in the data. High values of R

Fig. 2. Optimum Neural Networks. Values of R for TOPIX start low at $n=2$, but increase at $n=3$, and then start to decrease from $n=4$. This process characterizes the deterministic nature of the data.

indicate high randomness and high statistical dependence between the data; low values of R indicate more recognizable patterns and high statistical independence of the data.

3 Empirical Results

3.1 Model Development

Network size is determined by the input number n , the number of hidden layers (1 or 2) and the number of neurons (m_1 and m_2) in the hidden layers. If different network sizes have similar values for the error function, the smallest network size is optimum [6]. In this paper, we use R -method (see Sect. 2.3) to characterize the deterministic and random properties of the data in relation to the input data so as to determine optimum neural networks.

Samples from TOPIX are used to measure the degree of randomness. Values of R are determined for the input sizes $n = 2, 3, 4, 5, 6, 7, 8$. Each sample is grouped into an overlapping cluster with 3 years of the data, enough to characterize the dynamic properties of the system. 6 month graduations are used to characterize moving group samples MGS(1), MGS(2), ..., MGS(7) such as

$$MGS_k : x_k, x_{k+1}, \dots, x_{k+n}, k = 1 \sim 7 \quad (8)$$

where n corresponds to 3 years of the data.

After calculating values of R for both TOPIX and two types of generated data (see Sect. 2.2), mean values of R corresponding to moving group samples for TOPIX are used to evaluate deterministic and random properties of the data. Values of R for two other types of generated data are calculated as comparison. Here, the data number for these two types of the data is 750 because training data number for the stock market indexes is about 750. Values of R for random data are mean values of five generated data sets. For values of R , we choose $\&A = 0.4$ and $\&B = 0.01$.

Table 2. Forecasting Performance for *NN341* (iteration=3000). $\&G$ learning rate parameter; RMS: root mean square error.

Phase	Data Period	$\&G$	RMS
Training	1995–1997	0.005	0.001926
Test	1998–2000	0.005	0.036991

Fig. 2 shows that the degree of randomness for TOPIX lays between values of R for random data and sinusoidal data. Therefore, the data from TOPIX might have both deterministic and random properties.

Values of R can also be used to determine the minimum input number of dynamic variables needed to train the networks. At input number $n=3$, there is the maximum statistical independence between distance observations of data series, so input number $n=3$ maximize the network ability to learn from the data and should be the optimum number of nodes in the input layer. To verify the optimum input number, RMS error corresponding to various input number of nodes is calculated. It is found that neural networks corresponding to input number $n=3$ give minimum RMS error.

To finally determine the optimum sizes of neural networks, depending on the determined input number, the networks with 1 hidden layer and 2 hidden layers are used for training and test by changing neuron sizes in the hidden layer(s). It is found that the networks with 1 hidden layer and 4 neurons can make RMS error converge to lower values. Optimum network size *NN341* shows the number of neurons in each layer for TOPIX. Therefore, the neural network forecasters are constructed based on determined input pattern and the number of neurons in the hidden layer(s).

3.2 Estimation of Deterministic Component

To verify whether the optimum neural network acquires underlying dynamics of the system from the data, both training RMS error and test RMS error should be estimated.

All the neurons in the multilayer perceptron should desirably learn at the same rate. To a good degree of approximation, the misadjustment of back propagation (BP) algorithm is directly proportional to the learning rate parameter [7]. Therefore, if the learning rate parameter is increased so as to accelerate the learning process, then the misadjustment is increased. Careful attention has therefore to be given to the choice of the learning rate parameter in the design of the network in order to produce a satisfactory overall performance.

Table 2 shows the simulation results for *NN341*. During convergence, weight updates not only include the change dictated by the learning rate, but also include a portion of the last weight change. The training process will terminate when the RMS error is less than 0.001 or the learning times are over 3,000 iterations. Since the RMS errors for both training and test have small values,

Fig. 3. Convergence Process of NN341. The solid line shows the training process. The dotted line describes corresponding test process.

Fig. 4. Comparison of Neural Network Outputs (predicated values) and Target Data (x_k) after training of 3000 iterations.

smaller than the fluctuation of the data, the network obtains good generalization and acquires the source dynamics.

Fig. 3 shows that before training, synaptic connections w_{ij} have random volumes and the network yields large RMS error. After training of 3000 iterations, where one iteration corresponds to one round of training using all training data, the average RMS error becomes as small as 0.019626. RMS error in test phases is somewhat higher than that in training phases. However, RMS errors in both phases show a similar trend, so neural networks acquire underlying dynamics of the system from the data.

It is found from *Fig. 4* that the network outputs follow the data change in both early training and later test phases. Clearly, the network has acquired the regularity in both phases.

3.3 Estimation of Random Component

To evaluate the predictability of neural networks and to analyze the random components of the data, training errors (the residuals between the outputs of neural networks and corresponding target data) are used as input data, and neural networks with the same optimum sizes are retrained.

Fig. 5. Prediction by *NN341* Using Training Error as Input Data. When the Training errors are used as the input data, the outputs of neural networks are close to zero.

Fig. 6. Degree of Randomness for Random Data, TOPIX, and Sinusoidal Data. Maximum number of TOPIX is 741. Maximum number of random and sinusoidal data is 5000. Values of R for random data are the mean of ten data series. The x -axis graduation varies.

It is found from *Fig. 5* that the outputs of neural networks are close to zero. This may mean that training errors are white noise, so we can deduce that TOPIX data are composed of a regular (deterministic) part and a random part.

To verify this further, both TOPIX and two types of generated data (see Sect. 2.2) are trained similarly using neural networks. After training, the network outputs for random data are close to zero and the network outputs for deterministic data close to the target data. The network outputs for TOPIX fall between outputs for random data and outputs for deterministic data.

It is found from *Fig. 6* that values of R gradually approach one (1) for random data and zero (0) for sinusoidal data as the data number increases. Values of R for random data reduce with decreasing the data number because randomness may be difficult to find using relatively small amounts of the data. As in the case of neural network outputs, values of R for TOPIX are between values of R for random data and sinusoidal data. This evidence gives a convincing picture that TOPIX are considered to include both deterministic and random components. Calculating values of R gives us important insights into regularity and noise, showing how neural networks can forecast performance in financial markets.

Fig. 7. Change of Internal Representation. Input data are half-year TOPIX daily data, and 1-month shift. MPG_1 = the period from Jan. 1995 to Jun. 1995, ..., MPG_{67} = the period from Jul. 2000 to Dec. 2000.

3.4 Network Adaptability and Environment Influence

Since the neural networks acquire dynamic properties from the data, stock market mechanism can be understood well by examining how external forces influence the network adaptability. The properties of trained networks are represented by weights w_i between neurons, a kind of internal representation of the networks. The optimum network $NN341$ has 21 weights denoted by vector w . The dynamic properties of the system vary with time, and this adaptive change may be evaluated by the change of weight vector w . The networks are trained successively by moving pattern groups $MPG_1, MPG_2, \dots, MPG_{67}$ such as

$$MPG_k : P_k, P_{k+1}, \dots, P_{k+n}, k = 1 \sim 67 \quad (9)$$

where n corresponds to 6 months of the data. Each group has 6 months of the data, enough to acquire dynamics of the system. Convergence of training by using each pattern group produces weight vectors, $w_1, w_2, \dots, w_{67}$. Dynamics of the system is characterized by the matrix of transition probabilities between network groups. The adaptive change of acquired dynamics may be evaluated by the direction change of weight vectors from w_1 . The direction change is measured by $\cos \theta_k$ that is

$$\cos \theta_k = \frac{w_k \cdot w_1}{|w_k| \cdot |w_1|}, k = 1 \sim 67 \quad (10)$$

where θ_k is an angle between two weight vectors.

Inspection of the change of $\cos \theta_k$ in *Fig. 7* provides insight into how environment influences the network properties. Plotting the weight vectors as a function of time reveals that the environment influences during the corresponding periods. Since $\cos \theta_k$ is more than about 0.6 as a whole, the properties of the network is considered to be maintained almost the same. Generally, $\cos \theta_k$ is kept relatively constant for a period of time, but there are gradual changes in some periods. It is therefore considered that homeostatic dynamics of the system may be adjusted occasionally due to environment changes and then stay in steady state for some periods. At some points, $\cos \theta_k$ changes, indicating that the networks obtain different weight vectors, such as in MPG_6 , corresponding to June 1995–Nov. 1995; MPG_{40} , corresponding to Apr. 1998–Sep. 1998; and

MPG_{53} , corresponding to May 1999–Oct. 1999, the networks obtain slightly different weight vectors.

4 Conclusion

This paper demonstrated the usefulness of neural networks as an event detection technique for financial time series. Empirical results show that dynamic properties of the regularity components were captured by neural network model. The results can be attributed to the high ability of the network to capture nonlinear market patterns. Calculation of data randomness shows that randomness combines with determinism to create statistical order. Prediction using training errors as inputs of networks and calculating degree of randomness prove that financial time series include both predictable components and unpredictable components. The property that weights in neural networks remain relatively stable in some periods characterizes the structure of financial markets. Such property shows that stock market has memory, maybe because investors often make market decisions based on past situations. Evaluating the change of weights in neural networks shows that homeostatic dynamics change occasionally, perhaps because of adaptation to environmental factors, which may induce the fluctuations. Neural networks can acquire exogenous regularities and endogenous dynamics.

In our research, we have systematically examined effects of various factors such as the number of model inputs, the number of hidden layer neurons and influence of the event. The question that still remains is under what conditions are they better than traditional statistical methods. Future experiments could focus on comparative study of neural network approach with other statistical time series approach to understand the nature of financial markets well.

References

1. Apostolos-Paul N. Refenes, A. Neil Burgess, and Yves Bentz: Neural Networks in Financial Engineering: A Study in Methodology. *IEEE Transactions on Neural Networks*, **8-6** (1997) 1222–1267.
2. Weigend, A., Huberman, B. and Rumelhart, D.: Predicting the future: a connectionist approach. *International Journal of Neural Systems*, **1-3** (1990) 193–209.
3. Jingtao Yao, Chew Lim Tan and Hean-lee Poh.: Neural network for technical analysis: a study on KLCI. *International Journal of Theoretical and Applied Finance*, **2-2** (1999) 221–241.
4. Rumelhart, D. E., Hinton, G. E., Williams, R. J.: Learning internal representation by error propagation. In Rumelhart, D. E., McClelland, J. L., and the PDP research group (Eds), *parallel distributed processing*, MIT press, Cambridge, MA (1986) 318–362.
5. Perambur S. Neelakanta: *Information-Theoretic Aspects of Neural Networks*. CRC Press LLC, (1999).
6. Matsuba, I., Masui, H., Hebishima, S.: Optimizing multiplayer neural networks using fractal dimensions of time series data. *Proc IEEE Trans Ind Elec*, **44** (1997) 707–716.
7. Simon Haykin: *Neural Network: A Comprehensive Foundation*. Macmillan College Publishing Company (1994).