Financial Data Mining Using Time delay Neural Networks

Hyun-jung Kim [a], Kyung-shik Shin [b]

Ewha Womans University, College of Business Administration
11-1 Daehyun-Dong, Seodaemun-Gu , Seoul 120-750, Korea
[a] *Tel: +82-2-3277-3767, Fax: +82-2-3277-2776, E-mail: charitas@empal.com*
[b] *Tel: +82-2-3277-2799, Fax: +82-2-3277-2776, E-mail: ksshin@ewha.ac.kr*

Abstract

This study investigates the effectiveness of time delay neural networks (TDNN) for the time dependent prediction domain. Although it is well-known fact that the back-propagation neural network (BPN) performs well in pattern recognition tasks, the method has some limitations in that it can only learn an input-output mapping of static (or spatial) patterns that are independent of time or sequences.
The preliminary results show that the accuracy of TDNN is higher than the standard BPN with time lag. Our proposed approaches are demonstrated by the stock market prediction domain.

Keywords:

Time-delay neural networks, Time series prediction, Stock market prediction

Introduction

Forecasting price movements in stock markets is a major challenge confronting investors, speculator and businesses. In their quest to forecast the markets, they assume that future occurrences are based at least in part on present and past events and data. However, financial time series are among the 'noisiest' and most difficult signals to forecast.

The early studies of stock market prediction have tended to use statistical techniques. However studies using only classical statistical techniques for prediction reach their limitations in applications with nonlinearities in the data set. Efforts to improve global linear autoregression (AR) models include systematically increasing the order of interaction (bilinear models; [12]), splitting the input space across one variable and allowing for two AR models (threshold AR models; [35]) and using the nonlinearities of a Volterra expansion [41].

The control of nonlinear systems is an area of active research; approaches to this problem include both explicit embedding models [4,18,31] and implicit connectionist strategies [29,43]. Compared with statistical method, one advantage for artificial neural network (ANN) is to handle the nonlinear problems by using the hidden layer. Researchers work for reassuring proofs that ANN with hidden layer can essentially fit any well-behaved function and its derivative [3,6,8,11,17,2519,42,45] to results on the ability to generalize [41].

Among the ANN algorithms, the back-propagation neural network (BPN) is the most popular method in many applications such as classification, forecasting and pattern recognition. However, a major limitation of the BPN is that it can only learn an input-output mapping of static (or spatial) patterns that are independent of time. To overcome this limitation, two methods applying the time property are proposed; the first is the use of recurrent links. And the second is the use of time-delayed links.

This study investigates the effectiveness Time Delay Neural Network (TDNN) in detecting temporal pattern for the stock market prediction tasks. TDNN is a multilayer feedforward network whose hidden neurons and output neurons are replicated across time. Since TDNN has a number of control variables such as the period of time delay, the number of hidden nodes, the network architecture, the choice of activation function and so on, we apply genetic algorithms (GAs) to support optimization of architectural factors and network topologies.

The rest of the paper is organized as follows: the next section provides a description of the previous research on stock market application. Section 3 proposes ANN for time series property. Section 4 describes the research data and experiments. In section 5, empirical results are summarized and analyzed. In the final section, conclusions and the limitations of this study are presented.

Stock Market Prediction

The early days of these studies focused on estimating the level of return on stock price index. Lee *et al.* [27] developed the intelligent stock portfolio management system (ISPMS). They attempted to build expert systems by integrating machine learning and expert's opinion. One of the earliest

studies for stock market prediction using AI, Kimoto *et al.* [23] used several learning algorithms and prediction methods for developing the Tokyo stock exchange prices index (TOPIX) prediction system. They used the modular neural network to learn the relationships among various market factors. Kamijo and Tanikawa [20] employed the recurrent neural network (RNN) for analyzing candlestick charts and Ahmadi [1] used the back-propagation neural network with the generalized delta rule to predict the stock market. Kim [22] carried out the prediction of Korean stock market index using RNN, TDNN, time delay recurrent neural network (TDRNN) and adaptive time delay neural network (ATNN).

Some researchers investigated the issue of predicting the stock index futures market. Trippi and DeSieno [36] and Choi *et al.* [7] predicted the daily direction of change in the S&P index futures using ANN. Duke and Long [9] executed daily predictions of the German government bond futures using back-propagation neural network. The above studies are mainly focused on applications of ANN to the stock market prediction.

Recent research tends to include novel factors and to hybridize several AI techniques. Hiemstra [16] proposed fuzzy expert systems to predict stock market returns. He suggested ANN and fuzzy logic to predict stock market returns. A more recent study of Kohara *et al.* [24] incorporated prior knowledge to improve the performance of stock market prediction. Tsaih *et al.* [37] integrated the rule based technique and ANN to predict the direction of change of the S&P 500 stock index futures on a daily basis. Some researchers tend to include novel factors to the learning process:

Some of them did not produce outstanding prediction accuracy partly because of the tremendous noise and non-stationary characteristics in stock market data. Lawrence *et al.* [26] pointed to be difficult for high noise data then the networks fall into a naive solution such as always predicting the most common output. Another reason may be the local convergence of the gradient descent algorithms. Most ANN used in prior research employed the gradient descent algorithm to train the network [21,33].

ANN for Time Series Property

Previous studies of the time-series properties of various indicators have shown nonlinear dynamics [15]. Among them, most forecasting methods are only capable of picking up general trends and have difficulty in modeling cycles. Two methods may be applied to add a memory to a feed-forward neural network. The first is the use of recurrent links [10,44], while the second is the use of time-delayed links [5,14, 30].

Recurrent Neural Network

Unlike BPN, recurrent neural network (RNN) is permitted to have feedback connections among the neurons. The operation of the RNN involves the combined use of two network structures, the original recurrent network and the adjoin network, which work together. The original network is characterized by the forward propagation equation; for a given input-output, this network computes the error vector. The adjoin network is characterized by the backward propagation equation; the adjoin network takes the error vector form the original network and uses it to compute the adjustments to the synaptic weights of the original network [15].

The standard BPN algorithm suffers from the inability to fill in patterns, because of complete reliance on feed-forward connections. The use of RNN algorithm overcomes the pattern-completion problem by virtue of its inherent feedback connections. Almeida [2] has confirmed experimentally that feedback structures are much better suited to this kind of problem.

The use of feedback connections in the RNN makes it less sensitive to noise and lack of synchronization, and also permits it to learn faster, compared to the BPN. However, the BPN is more robust than the RNN with respect to the choice of a high learning rate parameter [34].

Time delay Neural Network

The time delay neural network (TDNN) is much more complex than BPN as it requires to explicitly manage the delays storing the activations and the back-propagated error signals for each unit and for all the time delays [40]. The TDNN tries to learn time based as well as functional relationships that correlate the input data to projected neural outputs.

At first, TDNN proposed by Waibel *et al.* [38] for dealing with speech recognition. In fact, TDNN has been successfully applied to speech recognition [13,39].

TDNN is a multi-layer feed-forward network whose hidden neurons and output neurons are replicated across time. It was devised to capture explicitly the concept of time symmetry as encountered in the recognition of an isolated word using spectrogram [15, 28]. A spectrogram is a two-dimensional image in which the vertical dimension corresponds to frequency and the horizontal dimension corresponds to time; the intensity of the image corresponds to signal energy [32]. TDNN are made up by units that get as input, at the generic time instant t, the outputs of the previous level units summed with the outputs at time t-1, t-2,..., t-n all suitably weighted. These delayed inputs let the unit know, at time t, part of the history of the signal, allowing the creation of richer and more complex decision surfaces. The training of TDNN takes place through a temporal expansion of the time delays all over the input sequence [5].

We consider a neural network with L levels, containing N_l units in each level l. Let us define the delayed input vector x for unit i pertaining to level l at discrete time t

$$x_i(t) = [\, x_i(t), x_i(t-1),\ldots, x_i(t-T_l)]^T, \quad i = 1, \ldots, N_l$$

and the vector of weights with which each input is weighted and put as output to unit j of level l+1

$$w_{ji} = [\, w_{ji}(0), w_{ji}(1),\ldots, w_{ji}(T_l)]^T, \quad j = 1, \ldots, N_{l+1}.$$

The contribution to the output field to unit j from unit i will be given by

$$s_{ji}(t) = w_{ji}^{T} x_i(t) \quad (1)$$

while the output field for unit j will be

$$s_j(t)=\sum s_{ji}(t) \text{ over the index } i=1,...,N_l. \quad (2)$$

Named f the units transfer function, one has

$$x_j(t) = f(s_j(t)) \quad (3)$$

The network architecture can be exemplified as in Figure 1, where we have chosen L=3, T_2=3, T_1=2.

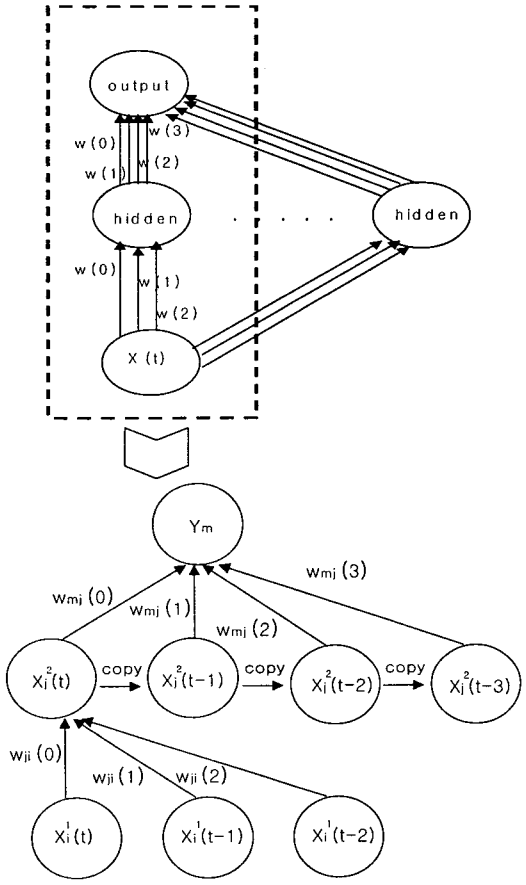

Figure 1. TDNN Algorithm

(Adapted from Cancelliere, R. and Gemello, R. [5], Revised by Kim, H.J. and Shin, K.S.)

The hidden units activations are copied a number of times equal to the number of delays; in Figure 1, for the sake of simplicity, only one unit have been shown for each level.

Defined $d_i(t)$ the i th element of the target vector, we have instantaneous error

$$e_i(t) = d_i(t) - x_i(t) \quad (4)$$

total instantaneous squared error

$$e^2(t)=\sum e^2_i(t) \text{ over the index } i=1, ..., N_l \quad (5)$$

total squared error

$$e^2 =\sum e^2(t) \text{ over the index } t=0, ..., T. \quad (6)$$

The target vector $d(t)$ changes at each time t, synchronously with the input, training is supervised at each time t and the net output depends, because of delays, on inputs at times t, t-1,..., t-T_1.

As the output unit has no delay, the formulation of output unit error signal is equal to the error signal of the standard BPN algorithm. The TDNN algorithm differs in the formulation of the error signal of hidden units.

Model Development

Research data used in this study comes from the daily Korea Stock Price Index 200 (KOSPI 200) form January 1997 to December 1999. The total number of samples includes 833 trading days.

We investigated two methods to add a memory to ANN above: the use of recurrent and time-delayed links. In this study, we only consider the TDNN that is the ANN with time-delayed links.

To compare with TDNN, we also design BPN with the following time series using lagging inputs. Actually, we can get the similar effect of a TDNN using a standard BPN by modifying the input data. The number of historical units controls the length of the patterns that the network can recognize. This leads to the fact that the major difference with TDNN is that the BPN with time lag has no time delay back with hidden neuron output.

We use the value of stock price index as an input variable and output variables. At the generic time instant t, we selected KOSPI 200 at time t as input variables, KOSPI at time t+1 as output variables. Figure 2 represents the BPN architecture, where we have chosen 3 days time lag.

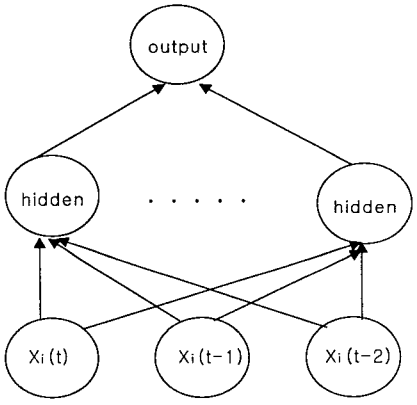

Figure 2.BPN with Time Lag (3 days)

We use moving window approach for train, test and validation set extraction method that is the efficient way to assess the accuracy of ANN model with input combination over time. This method start to train on the defined quantity of training records, test on the subsequent quantity of test records and validate on the subsequent quantity of validation

records; that is the first fold. This process move all records forward by size of the validation set, retrain, retest and revalidate and continue through all folds until the end of the data set is reached. Then average mean square error (AMSE) yielding the averaged value of the mean square error (MSE) of each fold is computed. In this study, the training set size is 400 records, test and validation set size is 15 records each. Therefore, resulting folds are totally 28 folds through this process until the end of the data and validation records in last fold are 13 records. Figure 3 illustrates the moving window approach.

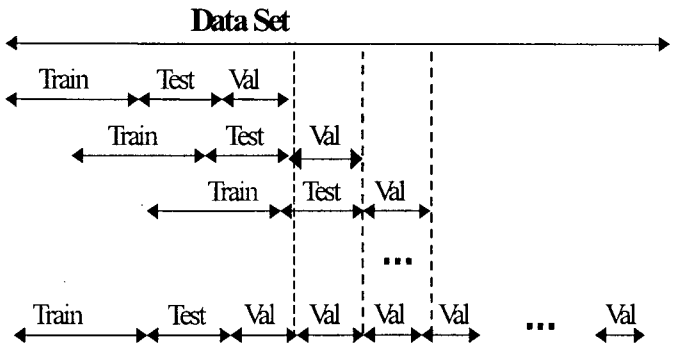

Figure 3. Moving Window Approach

In building neural networks models, GAs are applied to find an optimal or near optimal network topology and time lag. Fitness function we use is to minimize the error of AMSE. We use population size of 50, selection rate of 0.5, and mutation rate of 0.25 for the experiment. After selection, the dropped poor chromosomes are replaced with new ones by the cloning method. The crossover is performed by picking a cut point and by exchanging between the cut point and the end of the string of the parents. As a stopping condition, we use 10 generations. The neuro-genetic algorithms software NeuroGenetic Optimizer (NGO) version 2.5 executes these processes.

Result & Analysis

We set GAs to search the number of input neurons within maximum 4 time delays, the number of hidden neurons in BPN with time lag, and the number of hidden node, time delay such as connection of hidden and output neurons in TDNN for optimization tasks. We also control the following variables for efficient search. The transfer function of hidden neuron is set to the logistic function and the numbers of hidden neurons are limited to 8. The number of hidden layer is given one. The derived results by genetic search are summarized in Table 1.

To reduce the impact of random variation in GAs search process, we replicate the experiment several times. We suggest the best networks found in each model. As a result, TDNN can have the three types as the connection of neurons, TDNN1 is the general TDNN architecture with time delay of both hidden and output neurons. TDNN2 and TDNN3 only have time delay of either hidden or output neurons respectively.

Table 1. The optimized architectural factors using GA

		Number of input neurons	Number of hidden neurons	Connection of hidden neurons	Connection of output neurons	Time delay
TDNN1	T1_1	1	5	4	2	4
	T1_2	1	5	3	5	6
TDNN2	T2_1	1	5	1	5	4
	T2_2	1	8	1	5	4
TDNN3	T3_1	1	3	7	1	6
	T3_2	1	2	7	1	6
BPN	B_1	3 (t, t-1, t-4)	6	1	1	0
	B_2	3 (t, t-3, t-4)	8	1	1	0

This research compares models in experiment for the prediction of stock price index. As mentioned above section, the metric used in this experiment, which is the average mean square error (AMSE), is calculated between the expected and actual neural outputs and is averaged across all output neurons. AMSE is calculated using the following formula:

$$\text{AMSE} = \frac{\sum (\text{Y actual} - \text{Y predicted})^2}{n} \tag{7}$$

For comparison with the enhancement of each technique, we compute the mean square error (MSE) of the searched model.

Table 2. The measure of accuracy

		AMSE		MSE	
		Value	Average	Value	Average
TDNN1	T1_1	2.96	3.50	3.49	4.02
	T1_2	4.04		4.55	
TDNN2	T2_1	3.66	3.98	4.23	4.48
	T2_2	4.30		4.73	
TDNN3	T3_1	3.63	3.68	3.96	4.15
	T3_2	3.73		4.33	
BPN	B_1	6.42	6.45	7.40	7.44
	B_2	6.48		7.47	
Bench Mark		2.84	2.84	3.43	3.43

We use benchmark for wishing to gauge results from BPN and TDNN. The MSE of the benchmark is also the measure of difference between the actual value and the predicted value; we consider the predicted value as the actual value of one day after from the actual value time. This is represented as benchmark to verify the application of the proposed model

to the domain. Test results are summarized in Table 2.

F-Test results for the comparison of each model are summarized in Table 3. If F-value is significantly large or small, we can understand that the performance difference between the models exists.

Table 3. F-test result for the comparison of the difference between models

[a]F-values

	TDNN1	TDNN2	TDNN3	BPN
Bench Mark	1.17[a] **	1.31 ***	1.21 ***	2.17 ***
TDNN1		1.11 *	1.03	1.85 ***
TDNN2			0.93	1.66 ***
TDNN3				1.79 ***

* significant at 10%
** significant at 5%
*** significant at 1%

Overall, the AMSE and MSE of TDNN are smaller than those of BPN. As the F-tests results, it shows the prediction ability of BPN, TDNN model is significantly different. And the measure of benchmark is the lowest among the other values. These results show that TDNN doesn't beat the benchmark but outperforms comparing with BPN. Specifically, the average error values of the TDNN1 that has time delay of both hidden and output neurons, is the smallest among other networks. The results of F-tests support that TDNN1 performs better than others. And it also appears that the other types of TDNN such as TDNN2, TDNN3 is significantly indifferent.

From the results of experiments, we can conclude that TDNN is the better approach to reflect the temporal patterns than ordinary BPN.

Conclusion

This study investigates the effectiveness of Time Delay Neural Networks for the time dependent prediction domain. The preliminary results show that the accuracy of TDNN is higher than the standard BPN with time lag.

Although, BPN is well known for its powerful non-parametric pattern recognition capabilities, the method has neither feedbacks nor delays, and consequently no memory of the past inputs: the output result is strictly a function of the instantaneous input to the network. That aspect may constitute a limitation of standard BPN, particularly in the case of sequential patterns generated in time. Meanwhile TDNN is more adequate method to reflect the temporal patterns since it requires managing the delays storing the activations and the back-propagated error signals for each unit and for all the time delays.

This study has limitations. Although GAs intends to optimize the network topologies and to select the feature subsets, we controlled some variables such as the transfer function and the number of hidden layer for searching efficiency. However, there are an infinite number of possible combinations with these control variables. In setting up the GAs optimization problem, we must select the several parameters such as stopping condition, the population size, crossover rate, mutation rate and so on. The values of these parameters can greatly influence the performance of the algorithm. The varying parameters also generate a lot of groups for our general result.

TDNN we applied also has the limitation of its inability to learn or adapt the values of the time delays. Time delays are fixed initially and remain the same throughout training. As a result, TDNN may have poor performance due to the inflexibility of time delays and due to mismatch between the choice of time-delay values and the temporal location of the important information in the input patterns. In addition, the performance may vary depending on the range of the time delay values.

To overcome this limitation, our future work will be an adaptive time-delay neural network model. This network adapts its time-delay values as well as its weights during training to better accommodate to changing temporal patterns and to provide more flexibility for optimization tasks.

References

[1] Ahmadi, H. 1990. Testability of the arbitrage pricing theory by neural networks. Proceedings of the International Conference on Neural Networks, San Diego, California.

[2] Almeida, L. B. 1987. A learning rule for asynchronous perceptrons with feedback in a combinational environment. 1st IEEE International Conference on Neural Networks 2, San Diego, CA.

[3] Barron, A.R. 1992. Universal Approximation Bounds for Superpositions of a Sigmoidal Function. IEEE Trans. Info. Theory 38.

[4] Bradley, E. 1992. Taming Chaotic Circuits. Ph.D. Thesis, Massachusetts Institute of Technology.

[5] Cancelliere, R., and Gemello, R. 1996. Efficient training of time delay neural networks for sequential patterns. Neurocomputing 10.

[6] Chatfield, C. 1993. Neural networks: Forecasting breakthrough or passing fad. International Journal of Forecasting 9.

[7] Choi, J. H., Lee, M. K., and Rhee, M. W. 1995. Trading S&P 500 stock index futures using a neural network. Proceedings of the Third Annual International Conference on Artificial Intelligence Applications on Wall Street, New York.

[8] Cybenko, G. 1989. Approximations by superposition of a sigmoidal function. Math. Control, Signals Systems 2.

[9] Duke, L. S., and Long, J. S. 1993. Neural network futures trading- A feasibility study. Society for Worldwide Interbank Financial Telecomm. Adaptive intelligent systems,

Amsterdam: Elsevier Science Publishers.

[10] Elman, J. L. 1998. Finding structure in time. CRL Technical Report (8801), La Jolla University of California, San Diego.

[11] Funahashi, K. I. 1989. On the Approximate Realization of Continuous Mapping by Neural Networks. Neural Networks 2.

[12] Granger, C.W., and Anderson, A.P. 1978. An Introduction to Bilinear Time Series Models. Gottingen: Vandenhoek and Ruprecht.

[13] Haffner, P., and Waibel, A. 1991. Time-Delay Neural Networks embedding time alignment: a performance analysis. Proceedings of the Eurospeech, Genova.

[14] Haffner, P., and Waibel, A. 1992. Multi-state time delay neural networks for continuous speech recognition. Advances in neural information processing systems 4.

[15] Haykin, S. 1994. Neural Networks: A comprehensive foundation. Macmillan College Publication, New York.

[16] Hiemstra, Y. 1995. Modeling structured nonlinear knowledge to predict stock market returns, In R.R. Trippi, Chaos and nonlinear dynamics in the financial markets: theory, evidence and applications. Chicago, Illinois: Irwin.

[17] Hinton, G..E., and Sejnowski, T.J. 1986. Learning and Relearning in Boltzmann Machines. Parallel Distributed Processing, Cambridge, M.A: MIT Press.

[18] Hubler, A. 1989. Adaptive Control of Chaotic Systems. Helv. Phys. Acta 62.

[19] Irie, B., and Miyake, S. 1998. Capabilities of Three-Layered Perceptrons. Proceedings of the IEEE Second International Conference on Neural Networks 2, San Diego, CA.

[20] Kamijo K., and Tanikawa, T. 1990. Stock price pattern recognition: A recurrent neural network approach. Proceedings of the International Joint Conference on Neural Networks, San Diego, California.

[21] Kim, K. J., and Han, I. G.. 2000. Using Genetic Algorithms to Support Artificial Neural Networks for the Prediction of the Korea Stock Price Index. Conference on Intelligent information systems, Korea.

[22] Kim, S. S. 1998. Time-delay recurrent neural network for temporal correlations and prediction. Neuroconputing 20.

[23] Kimoto, T., Asakawa, K., Yoda, M., and Takeoka. M. 1990. Stock market prediction system with modular neural network. Proceedings of the International Joint Conference on Neural Networks, San Diego, California.

[24] Kohara, K., Ishikawa, T., Fukuhara, Y., and Nakamura, Y. 1997. Stock price prediction using prior knowledge and neural networks. International Journal of Intelligent Systems in Accounting, Finance and Management 6 (1).

[25] Laird, P., and Saul. R. 1993. Discrete Sequence Prediction and Its Application. Machine Learning.

[26] Lawrence, S., Tsoi, A.C., and Giles, C.L. 1996. Noisy time series prediction using symbolic representation and recurrent neural network grammatical inference. In Institute for Advanced computer Studies, Technical report UMIACS-TR-96-27 and CS-TR-3625. University of Maryland.

[27] Lee, J. K., Kim, H. S., and Chu, S. C. 1989. Intelligent stock portfolio management system. Expert systems 6 (2).

[28] Lin, D. T., Dayhoff, J. E., and Ligomenides, P. A. 1992. Trajectory recognition with a time delay neural network. International Joint Conference on Neural Networks, Baltimore.

[29] Miller, W.T., Stutton, R.S., and Werbos, P.J. 1990. Neural Networks for Control. Cambridge, MA: MIT Press.

[30] Miyatake, M., Sawai, H., Minami, Y., and Shikano, K. 1990. Integrated training for spotting Japanese phonemes using large phonemic time?delay neural networks. Proceeding ICASSP.

[31] Ott, E., Grebogi, C., and Yorke, J.A. 1992. Controlling Chaos. Phys. Rev. Lett 64.

[32] Rabiner, L. R., and Schafer, R. W. 1978. Digital processing of Speech Signals. Prentice Hall, New Jersey.

[33] Shin, K. S., Kim, K. J., and Han, I. G.. Financial Data Mining Using Genetic Algorithms Technique: Application to Korean Stock Market. Intelligent Systems in Accounting, Finance and Management.

[34] Simard, P. Y., Ottaway, M. B., and Ballard, D. H. 1989. Fixed point analysis for recurrent networks. Advances in Neural Information Processing Systems 1, San Mateo, CA: Morgan Kaufmann.

[35] Tong, H., and Lim, K.S. 1980. Threshold Autoregression, Limit Cycles and Cyclical Data. J. Roy. Stat. Soc 42.

[36] Trippi, R. R., and DeSieno, D. 1992. Trading equity index futures with a neural network. The Journal of Portfolio Management 19.

[37] Tsaih, R., Hsu, Y., and Lai, C. C. 1998. Forecasting S&P 500 stock index futures with a hybrid AI system. Decision Support Systems 23.

[38] Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., and Lang, K. 1988. Phoneme recognition: Neural networks versus hidden markov models. Proceedings of IEEE International Conference on Acoust, Speech, Signal Processing.

[39] Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., and Lang, K. 1989. Phoneme recognition using time delay neural networks. IEEE Transaction on Acoustics, Speech, Signal Processing, 37 (3).

[40] Wan, E. A. 1990. Temporal backpropagation for FIR neural networks. International Joint Conference Neural networks, San Diego, CA.

[41] Weigend, A. S., and Gershenfeld, N. A. eds. 1992. Time series prediction: Forecasting the future and understanding the past. Addison-Wesley publication.

[42] White, D.A. 1990. Connectionist Nonparametric Regression: Multilayer Feedforward Networks Can Learn Arbitrary Mappings. Neural networks 3.

[43] White, D.A., and Sofge, D.A, eds. 1992. Handbook of Intelligent Control. Van Nostrand Reinhold.

[44] Williams, R.J., and Zipser, D. 1989. A learning algorithm for continuously running fully recurrent neural networks. Neural Computations 1.

[45] Yoon, Y., and Swales, G.. 1991. Predicting stock price performance: A neural network approach. Proceedings of the IEEE 24th Annual Conference on System Science.