Flexible Neural Trees Ensemble for Stock Index Modeling

Yuehui Chen[1], Bo Yang[1,3] and Ajith Abraham[1,2]

[1]School of Information Science and Engineering
Jinan University, Jinan 250022, P.R.China
Email: yhchen@ujn.edu.cn
[2]School of Computer Science and Engineering
Chung-Ang University, Seoul, Republic of Korea
Email: ajith.abraham@ieee.org
[3]State Key Lab. of Advanced Technology
for Materials Synthesis and Processing,
Wuhan University of Science and Technology, Wuhan, P.R. China
Email: yangbo@ujn.edu.cn

Abstract. The use of intelligent systems for stock market predictions has been widely established. In this paper, we investigate how the seemingly chaotic behavior of stock markets could be well represented using Flexible Neural Tree (FNT) ensemble technique. We considered the Nasdaq-100 index of Nasdaq Stock Market^{SM} and the S&P CNX NIFTY stock index. We analyzed 7-year Nasdaq-100 main index values and 4-year NIFTY index values. This paper investigates the development of novel reliable and efficient techniques to model the seemingly chaotic behavior of stock markets. The structure and parameters of FNT are optimized using Genetic Programming (GP) like tree structure based evolutionary algorithm and Particle Swarm Optimization (PSO) algorithms, repectively. A good ensemble model is formulated by the Local Weighted Polynomial Regression (LWPR). This paper investigates whether the proposed method can provide the required level of performance, which is sufficiently good and robust so as to provide a reliable forecast model for stock market indices. Experimental results show that the model considered could represent the stock indices behavior very accurately.
Key Words: Flexible neural tree, GP-like tree structure based evolutionary algorithm, particle swarm optimization, ensemble learning, stock index

1 Introduction

Prediction of stocks is generally believed to be a very difficult task - it behaves like a random walk process and time varying. The obvious complexity of the problem paves the way for the importance of intelligent prediction paradigms. During the last decade, stocks and futures traders have come to rely upon various types of intelligent systems to make trading decisions [1], [2], [14], [15], [16],

[3], [19], [20]. Several intelligent systems have in recent years been developed for modelling expertise, decision support and complicated automation tasks [3], [4], [17], [18]. In this paper, we analyzed the seemingly chaotic behavior of two well-known stock indices namely the Nasdaq-100 index of Nasdaq^{SM} [5] and the S&P CNX NIFTY stock index [6]. The Nasdaq-100 index reflects Nasdaq's largest companies across major industry groups, including computer hardware and software, telecommunications, retail/wholesale trade and biotechnology [5]. The Nasdaq-100 index is a modified capitalization-weighted index, which is designed to limit domination of the Index by a few large stocks while generally retaining the capitalization ranking of companies. Through an investment in the Nasdaq-100 index tracking stock, investors can participate in the collective performance of many of the Nasdaq stocks that are often in the news or have become household names. Similarly, S&P CNX NIFTY is a well-diversified 50 stock index accounting for 25 sectors of the economy [6]. It is used for a variety of purposes such as benchmarking fund portfolios, index based derivatives and index funds. The CNX Indices are computed using market capitalization weighted method, wherein the level of the Index reflects the total market value of all the stocks in the index relative to a particular base period. The method also takes into account constituent changes in the index and importantly corporate actions such as stock splits, rights, etc. without affecting the index value.

Our research is to investigate the performance analysis of FNT [9], [10], [11] ensemble for modelling the Nasdaq-100 and the NIFTY stock market indices. The hierarchical structure of FNT is evolved using GP with specific instructions. The parameters of the FNT model are optimized by PSO algorithm [7]. We analyzed the Nasdaq-100 index value from 11 January 1995 to 11 January 2002 [5] and the NIFTY index from 01 January 1998 to 03 December 2001 [6]. For both the indices, we divided the entire data into almost two equal parts. No special rules were used to select the training set other than ensuring a reasonable representation of the parameter space of the problem domain [2].

2 The Flexible Neural Tree Model

The function set F and terminal instruction set T used for generating a FNT model are described as $S = F \bigcup T = \{+_2, +_3, \ldots, +_N\} \bigcup \{x_1, \ldots, x_n\}$, where $+_i (i = 2, 3, \ldots, N)$ denote non-leaf nodes' instructions and taking i arguments. x_1,x_2,...,x_n are leaf nodes' instructions and taking no other arguments. The output of a non-leaf node is calculated as a flexible neuron model (see Fig.1). From this point of view, the instruction $+_i$ is also called a flexible neuron operator with i inputs.

In the creation process of neural tree, if a nonterminal instruction, i.e., $+_i (i = 2, 3, 4, \ldots, N)$ is selected, i real values are randomly generated and used for representing the connection strength between the node $+_i$ and its children. In addition, two adjustable parameters a_i and b_i are randomly created as flexible activation function parameters. For developing the FNT, the flexible activation function $f(a_i, b_i, x) = e^{-(\frac{x-a_i}{b_i})^2}$ is used. The total excitation of $+_n$ is $net_n =$

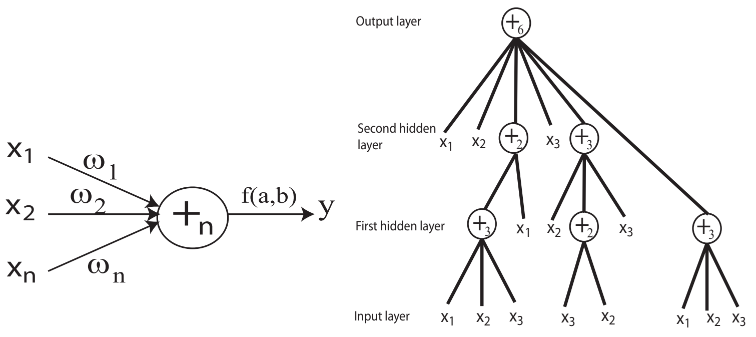

Fig. 1. A flexible neuron operator (left), and a typical representation of the FNT with function instruction set $F = \{+_2, +_3, +_4, +_5, +_6\}$, and terminal instruction set $T = \{x_1, x_2, x_3\}$ (right)

$\sum_{j=1}^{n} w_j * x_j$, where $x_j (j = 1, 2, \ldots, n)$ are the inputs to node $+_n$. The output of the node $+_n$ is then calculated by $out_n = f(a_n, b_n, net_n) = e^{-(\frac{net_n - a_n}{b_n})^2}$. The overall output of flexible neural tree can be computed from left to right by depth-first method, recursively.

2.1 Tree Structure Optimization

Finding an optimal or near-optimal neural tree is formulated as a product of evolution. In our previous work, the probabilistic incremental program evolution (PIPE) and ant programming (AP) algorithm have been employed to find a near-optimal neural tree [9], [12]. In this study, the crossover and selection operators used are same as those of standard GP. A number of neural tree mutation operators are developed as follows:

(1) Changing one terminal node: randomly select one terminal node in the neural tree and replace it with another terminal node;
(2) Changing all the terminal nodes: select each and every terminal node in the neural tree and replace it with another terminal node;
(3) Growing: select a random leaf in hidden layer of the neural tree and replace it with a newly generated subtree.
(4) Pruning: randomly select a function node in the neural tree and replace it with a terminal node.

2.2 Parameter Optimization with PSO

The Particle Swarm Optimization (PSO) conducts searches using a population of particles which correspond to individuals in evolutionary algorithm (EA). A population of particles is randomly generated initially. Each particle represents a potential solution and has a position represented by a position vector $\mathbf{x_i}$. A swarm of particles moves through the problem space, with the moving velocity of each particle represented by a velocity vector $\mathbf{v_i}$. At each time step, a function f_i representing a quality measure is calculated by using $\mathbf{x_i}$ as input. Each particle keeps track of its own best position, which is associated with the best fitness it has achieved so far in a vector $\mathbf{p_i}$. Furthermore, the best position among all the particles obtained so far in the population is kept track of as $\mathbf{p_g}$. In addition to this global version, another version of PSO keeps track of the best position among all the topological neighbors of a particle. At each time step t, by using the individual best position, $\mathbf{p_i}$, and the global best position, $\mathbf{p_g(t)}$, a new velocity for particle i is updated by

$$\mathbf{v_i(t+1)} = \mathbf{v_i(t)} + c_1\phi_1(\mathbf{p_i(t)} - \mathbf{x_i(t)}) + c_2\phi_2(\mathbf{p_g(t)} - \mathbf{x_i(t)}) \tag{1}$$

where c_1 and c_2 are positive constant and ϕ_1 and ϕ_2 are uniformly distributed random number in [0,1]. The term $\mathbf{v_i}$ is limited to the range of $\pm\mathbf{v_{max}}$. If the velocity violates this limit, it is set to its proper limit. Changing velocity this way enables the particle i to search around its individual best position, $\mathbf{p_i}$, and global best position, $\mathbf{p_g}$. Based on the updated velocities, each particle changes its position according to the following equation:

$$\mathbf{x_i(t+1)} = \mathbf{x_i(t)} + \mathbf{v_i(t+1)}. \tag{2}$$

2.3 Procedure of the general learning algorithm

The general learning procedure for constructing the FNT model can be described as follows.

1) Create an initial population randomly (FNT trees and its corresponding parameters);
2) Structure optimization is achieved by the neural tree variation operators as described in subsection 2.
3) If a better structure is found, then go to step 4), otherwise go to step 2);
4) Parameter optimization is achieved by the PSO algorithm as described in subsection 2. In this stage, the architecture of FNT model is fixed, and it is the best tree developed during the end of run of the structure search. The parameters (weights and flexible activation function parameters) encoded in the best tree formulate a particle.
5) If the maximum number of local search is reached, or no better parameter vector is found for a significantly long time then go to step 6); otherwise go to step 4);
6) If satisfactory solution is found, then the algorithm is stopped; otherwise go to step 2).

3 The FNT Ensemble

For most regression and classification problems, combining the outputs of several predictors improves on the performance of a single generic one [22]. Formal support to this property is provided by the so-called bias/variance dilemma [21], based on a suitable decomposition of the prediction error. According to these ideas, good ensemble members must be both accurate and diverse, which poses the problem of generating a set of predictors with reasonably good individual performances and independently distributed predictions for the test points. Diverse individual predictors can be obtained in several ways. These include: (i) using different algorithms to learn from the data (classification and regression trees, artificial neural networks, support vector machines, etc.), (ii) changing the internal structure of a given algorithm (for instance, number of nodes/depth in trees or architecture in neural networks), and (iii) learning from different adequately-chosen subsets of the data set. The probability of success in strategy (iii), the most frequently used, is directly tied to the instability of the learning algorithm [2]. That is, the method must be very sensitive to small changes in the structure of the data and/or in the parameters defining the learning process. Again, classical examples in this sense are classification and regression trees and artificial neural networks (ANNs). In particular, in the case of ANNs the instability comes naturally from the inherent data and training process randomness, and also from the intrinsic non-identifiability of the model. In what follows, three ensemble methods are employed for the stock index forecasting problems.

3.1 The Basic Ensemble Method

A simple approach to combining network outputs is to simply average them together. The basic ensemble method (BEM) output is defined:

$$f_{BEM} = \frac{1}{n}\sum_{i=1}^{n} f_i(x) \tag{3}$$

This approach by itself can lead to improved performance, but doesn't take into account the fact that some FNTs may be more accurate than others. It has the advantage that it is easy to understand and implement and can be shown not to increase the expected error.

3.2 The Generalized Ensemble Method

A generalization to the BEM method is to find weights for each output that minimize the positive and negative classification rates of the ensemble. The general ensemble method (GEM) is defined:

$$f_{BEM} = \sum_{i=1}^{n} \alpha_i f_i(x) \tag{4}$$

where the $\alpha_i' s$ are chosen to minimize the root mean square error between the FNT outputs and the desired values. For comparison purpose, the optimal weights of the ensemble predictor are optimized by using PSO algorithm.

3.3 The LWPR Method

To investigate more efficient ensemble method, a LWPR approximation approach is employed in this work[13]. In this framework, the final output of FNT ensemble is approximated by a local polynomial model, i.e.,

$$f_{LWPR} = \sum_{i=1}^{M} \beta_i t_i(x) \tag{5}$$

where t_i is a function that produces the ith term in the polynomial. For example, with two inputs and a quadratic local model we would have $t_1(x) = 1$, $t_2(x) = x_1$, $t_3(x) = x_2$, $t_4(x) = x_1^2$, $t_5(x) = x_1 x_2$, $t_6(x) = x_2^2$. Equation (5) can be written more compactly as

$$f_{LWPR} = \beta^T t(x) \tag{6}$$

where $t(x)$ is the vector of polynomial terms of the input x and β is the vector of weight terms. The weight of the ith datapoint is computed as a decaying function of Euclidean distance between x_k and x_{query}. β is chosen to minimize

$$\sum_{i=1}^{N} \omega_i^2 (f_{LWPR} - \beta^T t(x)) \tag{7}$$

where ω_i is a Gaussian weight function with kernel width K:

$$\omega_i = exp(-Distance^2(x_i, x_{query})/2K^2). \tag{8}$$

For this problem, an algorithm based on a multiresolution search of a quickly constructible augmented kdtree without needing to rebuild the tree, has been proposed for fast predictions with arbitrary local weighting functions [13].

4 Experiments

We considered 7-year stock data for the Nasdaq-100 Index and 4-year for the NIFTY index. Our target is to develop efficient forecast models that could predict the index value of the following trade day based on the opening, closing and maximum values of the same on a given day. The assessment of the prediction performance of the different ensemble paradigms were done by quantifying the prediction obtained on an independent data set. The Root Mean Squared Error (RMSE), Maximum Absolute Percentage Error (MAP) and Mean Absolute Percentage Error (MAPE) and Correlation Coefficient (CC) were used to study the

Table 1. Empirical comparison of RMSE results for four learning methods

	Best-FNT	BEM	GEM	LWPR
Nasdaq-100	0.01854	0.01824	0.01635	4.41×10^{-5}
NIFTY	0.01315	0.01258	0.01222	1.96×10^{-7}

Table 2. Statistical analysis of four learning methods (test data)

	Best-FNT	BEM	GEM	LWPR
		Nasdaq-100		
CC	0.997542	0.997610	0.997757	0.999999
MAP	98.1298	98.3320	97.3347	0.4709
MAPE	6.1090	6.3370	5.7830	0.0040
		NIFTY		
CC	0.996908	0.997001	0.0997109	0.999999
MAP	28.0064	34.3687	26.8188	7.65×10^{-4}
MAPE	3.2049	2.9303	2.6570	1.92×10^{-5}

performance of the trained forecasting model for the test data. MAP is defined as follows:

$$MAP = max(\frac{|P_{actual,i} - P_{predicted,i}|}{P_{predicted,i}} \times 100) \tag{9}$$

where $P_{actual,i}$ is the actual index value on day i and $P_{predicted,i}$is the forecast value of the index on that day. Similarly $MAPE$ is given as

$$MAPE = \frac{1}{N} \sum_{i=1}^{N} (\frac{|P_{actual,i} - P_{predicted,i}|}{P_{predicted,i}}) \times 100 \tag{10}$$

where N represents the total number of days.

We used instruction set $I = \{+_2, +_3, \ldots, +_6, x_0, x_1, x_2\}$ for modeling the Nasdaq-100 index and instruction set $I = \{+_2, +_3, \ldots, +_8, x_0, x_1, x_2, x_3, x_4\}$ for modeling the NIFTY index. We have conducted 10 FNT models for predicting the Nasdaq-100 index and the NIFTY index, respectively. And then three ensemble methods discussed in Section 3 are employed to predict the both index.

Table 1 summarizes the test results achieved for the two stock indices using the four different approaches. Performance analysis of the trained forecasting models for the test data was shown in Table 2. Figures 2 and 3 depict the test results for the one day ahead prediction of the Nasdaq$-$100 index and the NIFTY index respectively.

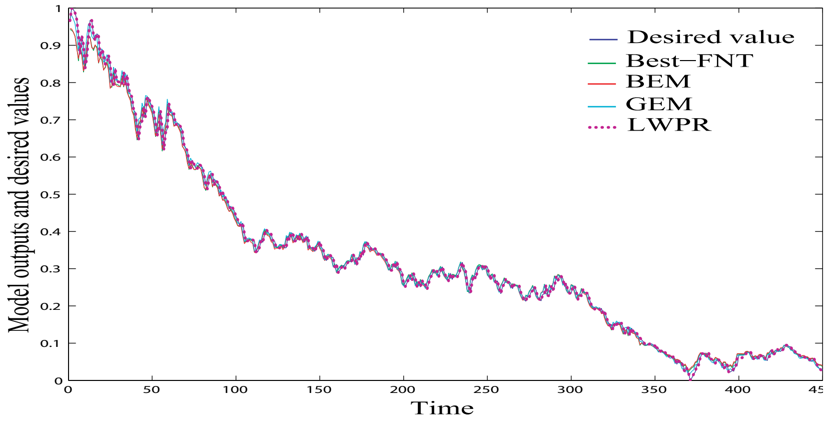

Fig. 2. Test results showing the performance of the different methods for modeling the Nasdaq-100 index

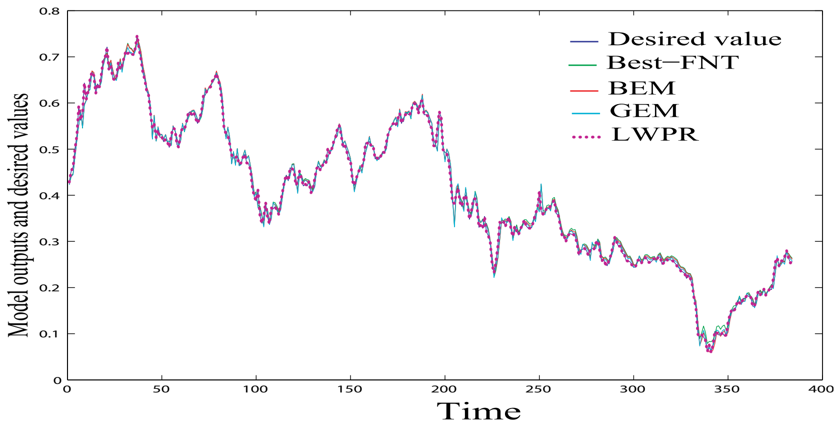

Fig. 3. Test results showing the performance of the different methods for modeling the NIFTY index

5 Conclusions

In this paper, we have demonstrated how the chaotic behavior of stock indices could be well represented by FNT ensemble learning paradigm. Empirical results on the two data sets using FNT ensemble models clearly reveal the efficiency of the proposed techniques. In terms of RMSE values, for the Nasdaq-100 index and the NIFTY index, LWPR performed marginally better than other models. For both index (test data), LWPR also has the highest correlation coefficient and the lowest value of MAPE and MAP values. A low MAP value is a crucial indicator for evaluating the stability of a market under unforeseen fluctuations. In the present example, the predictability assures the fact that the decrease in trade is only a temporary cyclic variation that is perfectly under control. Our research was to predict the share price for the following trade day based on the opening, closing and maximum values of the same on a given day. Our experiment results indicate that the most prominent parameters that affect share prices are their immediate opening and closing values. The fluctuations in the share market are chaotic in the sense that they heavily depend on the values of their immediate forerunning fluctuations. Long-term trends exist, but are slow variations and this information is useful for long-term investment strategies. Our study focus on short term, on floor trades, in which the risk is higher. However, the results of our study show that even in the seemingly random fluctuations, there is an underlying deterministic feature that is directly enciphered in the opening, closing and maximum values of the index of any day making predictability possible. Empirical results also show that LWPR is a distinguished candidate for the FNT ensemble or neural networks ensemble.

Acknowledgments

This research was partially supported the National High Technology Development Program of China (863 Program) under contract number 2002AA4Z3240, and The Provincial Science and Technology Development Program of Shandong under contract number SDSP2004-0720-03.

References

1. Abraham A., Nath B. and Mahanti P.K.: Hybrid Intelligent Systems for Stock Market Analysis, Computational Science, Springer-Verlag Germany, Vassil N Alexandrov et al (Editors), USA, 337-345, 2001.
2. Abraham A., Philip N.S., and Saratchandran P.: Modeling Chaotic Behavior of Stock Indices Using Intelligent Paradigms, International Journal of Neural, Parallel and Scientific Computations, USA, Volume 11, Issue (1,2): 143-160, 2003.
3. Leigh W., Modani N., Purvis R. and Roberts T.: Stock market trading rule discovery using technical charting heuristics, Expert Systems with Applications, 23(2): 155-159, 2002.
4. Leigh W., Purvis R. and Ragusa J.M.: Forecasting the NYSE composite index with technical analysis, pattern recognizer, neural network, and genetic algorithm: a case study in romantic decision support, Decision Support Systems, 32(4): 361-377, 2002.
5. Nasdaq Stock Market^{SM}: http://www.nasdaq.com
6. National Stock Exchange of India Limited: http://www.nse-india.com
7. Kennedy, J. and Eberhart, R.C.: Particle Swarm Optimization. In Proc. of IEEE International Conference on Neural Networks, 1942-1948, 1995.
8. Maqsood I., Khan M.R. and Abraham A.: Neural Network Ensemble Method for Weather Forecasting, Neural Computing and Applications, Springer Verlag London Ltd., 13(2): 112-122, 2004.
9. Chen, Y., Yang, B., Dong, J.: Nonlinear System Modeling via Optimal Design of Neural Trees, International Journal of Neural Systems, 14(2): 125-137, 2004.
10. Chen, Y., Yang, B., Dong, J., Abraham A., Time-series forcasting using flexible neural tree model, Information Science, 2005. (In press)
11. Chen Y., Abraham A., "Feature Selection and Intrusion Detection using Hybrid Flexible Neural Tree", ISNN'05, LNCS 3498, 439-444, 2005

12. Chen Y., Yang B. and Dong J., Evolving Flexible Neural Networks using Ant Programming and PSO algorithm, International Symposium on Neural Networks (ISNN'04), LNCS 3173, 211-216, 2004.
13. Moore A. and Schneider J. and Deng K.: Efficient Locally Weighted Polynomial Regression Predictions, Proceedings of the Fourteenth International Conference on Machine Learning, 236-244, 1997.
14. Abraham A., Philip N.S., Nath B. and Saratchandran P, Performance, Analysis of Connectionist Paradigms for Modeling Chaotic Behavior of Stock Indices, Second International Workshop on Intelligent Systems Design and Applications, Computational Intelligence and Applications, Dynamic Publishers Inc., USA, 181-186, 2002.
15. Chan W.S. and Liu W.N., Diagnosing shocks in stock markets of southeast Asia, Australia, and New Zealand, Mathematics and Computers in Simulation 59(1-3): 223-232, 2002.
16. Francis E.H. Tay and L.J. Cao, Modified support vector machines in financial time series forecasting, Neurocomputing, 48(1-4):847-861, 2002.
17. Quah T.S. and Srinivasan B., Improving returns on stock investment through neural network selection, Expert Systems with Applications 17(4): 295-301, 1999.
18. Wang Y.F., Mining stock price using fuzzy rough set system, Expert Systems with Applications 24(1): 13-23, 2002.
19. Oh K.J. and Kim K.J., Analyzing stock market tick data using piecewise nonlinear model, Expert Systems with Applications 22(3): 249-255, 2002.
20. Kim K.J. and Han I., Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index, Expert Systems with Applications 19(2): 125-132, 2000.
21. Geman S., Bienenstock E., R. Doursat, Neural networks and the bias/variance dilemma, Neural Computation, 4:1C58, 1992.
22. Sharkey, A.J.C. (Ed.), Combining Artificial Neural Nets, Springer, London, 1999.
23. Breiman, L., Bagging predictors, Machine Learning 24:123C140, 1996.