Hybrid Methods for Stock Index Modeling

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

[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

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 neural network, TS fuzzy system and hierarchical TS fuzzy techniques. To demonstrate the different techniques, we considered Nasdaq−100 index of Nasdaq Stock Market^{SM} and the S&P CNX NIFTY stock index. We analyzed 7 year's Nasdaq 100 main index values and 4 year's NIFTY index values. This paper investigates the development of novel reliable and efficient techniques to model the seemingly chaotic behavior of stock markets. The parameters of the different techniques are optimized by the particle swarm optimization algorithm. This paper briefly explains how the different learning paradigms could be formulated using various methods and then investigates whether they can provide the required level of performance, which are sufficiently good and robust so as to provide a reliable forecast model for stock market indices. Experiment results reveal that all the models considered could represent the stock indices behavior very accurately.

1 Introduction

Prediction of stocks is generally believed to be a very difficult task. The process behaves more like a random walk process and time varying. The obvious complexity of the problem paves 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]. Several intelligent systems have in recent years been developed for modelling expertise, decision support and complicated automation tasks etc[3][4]. In this paper, we analyzed the seemingly chaotic behavior of two well−known stock indices namely Nasdaq−100 index of Nasdaq^{SM} [5] and the S&P CNX NIFTY stock index [6]. 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 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 hybrid soft computing paradigms for modelling the Nasdaq−100 and NIFTY stock market indices. We considered neural network, TS Fuzzy system [7] and hierarchical fuzzy system [8][9][10][11][12][13][14][15]. The hierarchical structure is evolved using tree-structure-based evolutionary algorithm with specific instructions. The parameters of the different techniques are optimized by the particle swarm optimization algorithm [16]. 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 Takagi-Sugeno Fuzzy Systems (TS-FS)

Fuzzy inference systems are composed of a set of if-then rules. A Sugeno-Takagi fuzzy model has the following form of fuzzy rules [7]:

$$R_j : \text{if } x_1 \text{ is } A_{1j} \text{ and } x_2 \text{ is } A_{2j} \text{ and } \ldots \text{ and } x_n \text{ is } A_{nj}$$
$$\text{Then } y = g_j(x_1, x_2, \ldots, x_n),\ (j = 1, 2, \ldots, N)$$

where $g_j(\cdot)$ is a crisp function of x_i. Usually, $g_j(x_1, x_2, \ldots, x_n) = \omega_0 + \omega_1 x_1 + \omega_2 x_2 + \ldots + \omega_n x_n$. The overall output of the fuzzy model can be obtained by:

$$y = \frac{\sum_{j=1}^{N} g_j(\cdot) T_{i=1}^{m_j} \mu_{ij}(x_i)}{\sum_{j=1}^{N} T_{i=1}^{m_j} \mu_{ij}(x_i)} \tag{1}$$

where $1 \leq m_j \leq n$ is the number of input variables that appear in the rule premise, N is the number of fuzzy rules, n is the number of inputs, μ_{ij} is the membership function for fuzzy set A_{ij} and T is a t-norm for fuzzy conjunction.

A number of researches have been devoted to identify the TS-FS model. The parameters to be determined are the division of input space, the shapes

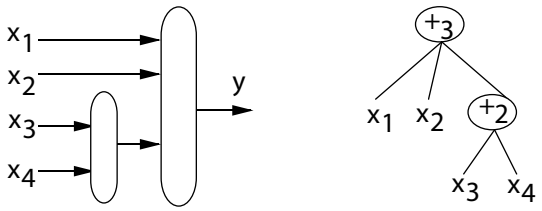

Fig. 1. Left: An example of possible hierarchical TS-FS model with 4 inputs and 3 hierarchical ayers, Right: the tree structural representation of the corresponding hierarchical TS-FS model, where the used instruction set is $I = \{+_2, +_3, x_1, x_2, x_3, x_4\}$.

of membership functions in the antecedent parts and the linear weights in the consequent parts. In this research, the parameters of the TS fuzzy model are optimized by using PSO algorithm.

3 Hierarchical TS-FS : Encoding and Evaluation

A Hierarchical TS Fuzzy Systems (H-TS-FS) not only provide a more complex and flexible architecture for modelling nonlinear systems, but can also reduce the size of rule base to some extend. But there is no systematic method for designing of the hierarchical TS-FS model yet. The problems in designing of hierarchical fuzzy logic system include:(1) Selecting a proper hierarchical structure; (2) Selecting the inputs for each TS fuzzy submodel; (3) Determining the rule base for each TS fuzzy submodel; (4) Optimizing the parameters in the antecedent parts and the linear weights in the consequent parts of the TS fuzzy submodel.

In this sense, finding a proper hierarchical TS-FS model can be posed as a search problem in the structure and parameter space. Fig.1(left) shows an example of possible hierarchical TS-FS models with 4 input variables and 3 hierarchical layers. A hybrid automatic approach has been proposed to optimize the hierarchical TS-FS with a Ant Programming (AP) and PSO algorithms [15]. In this research, a modified tree-structure based GP-like evolutionary algorithm was employed to find a optimal architecture of the H-TS-FS.

3.1 Encoding and Calculation

A tree-structural based encoding method with specific instruction set is selected for representing a hierarchical TS-FS in this research. The reasons for choosing this representation are that (1)the trees have a natural and typical hierarchical layer; (2)with pre-defined instruction sets, the tree can be created and evolved using the existing tree-structure-based approaches, i.e., AP, Genetic Programming (GP) and PIPE algorithms.

Assume that the used instruction set is $I = \{+_2, +_3, x_1, x_2, x_3, x_4\}$, where $+_2$ and $+_3$ denote non-leaf nodes' instructions and taking 2 and 3 arguments, respectively. x_1,x_2,x_3,x_4 are leaf nodes' instructions and taking zero argument

each. In addition, the output of each non-leaf node is calculated as a single TS fuzzy sub-model (Section 2). For this reason the non-leaf node $+_2$ is also called a two-inputs TS fuzzy instruction/operator. Fig.1(right) shows the corresponding tree structural representation of the hierarchical TS-FS model.

It should be noted that in order to calculate the output of each TS-FS sub-model (non-leaf node), parameters in the antecedent parts and consequent parts of the TS-FS submodel should be encoded into the tree. The output of a hierarchical TS-FS tree can be calculated in a recursive way.

3.2 Objective function

In this work, the fitness function used for GP-like evolutionary algorithm and PSO is given by Root Mean Square Error (RMSE):

$$Fit(i) = \sqrt{\frac{1}{P}\sum_{j=1}^{P}(y_1^j - y_2^j)^2} \tag{2}$$

where P is the total number of training samples, y_1^j and y_2^j are the actual and model outputs of j-th sample. $Fit(i)$ denotes the fitness value of i-th individual.

4 An Approach for evolving the Hierarchical TS-FS

4.1 Architecture Optimization of the hierarchical TS-FS

Finding an optimal or near-optimal neural tree is formulated as a product of evolution. 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.

4.2 Parameter optimization with PSO

For the parameters optimization of the hierarchical TS-FS, a number of global and local search algorithms, i.e., GA, EP, gradient based learning method can be employed. The basic PSO algorithm is selected for parameter optimization due to its fast convergence and ease to implementation.

The PSO [16] 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 (Eqn.(5) in this study) 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, $\mathbf{p_i(t)}$, and the global best position, $\mathbf{p_g(t)}$, a new velocity for particle i is updated by

$$\mathbf{v_i(t+1)} = \mathbf{v_i(t)} + c_1\phi_1(\mathbf{p_i(t)} - \mathbf{x_i(t)}) + c_2\phi_2(\mathbf{p_g(t)} - \mathbf{x_i(t)}) \tag{3}$$

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:

$$\mathbf{x_i(t+1)} = \mathbf{x_i(t)} + \mathbf{v_i(t+1)}. \tag{4}$$

4.3 The proposed learning algorithm

The general learning procedure for the optimal design of the hierarchical TS-FS model can be described as follows.

1) Create the initial population randomly (hierarchical TS-FS and their corresponding parameters);
2) Structure optimization by GP-like algorithm.
3) If the better structure is found, then go to step 4), otherwise go to step 2);
4) Parameter optimization by PSO algorithm. In this stage, the tree structure is fixed, and it is the best tree taken from the end of run of the structure search. All of the parameters encoded in the best tree formulated a parameter vector to be optimized by PSO;
5) If the maximum number of PSO search is reached, or no better parameter vector is found for a significantly long time (say 100 steps for maximum 2000 steps) then go to step 6); otherwise go to step 4);
6) If satisfied solution is found, then stop; otherwise go to step 2).

5 Experiments

We considered 7 year's stock data for Nasdaq-100 Index and 4 year's for 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. We used the same training and test data sets to evaluate the fuzzy TS and hierarchical TS fuzzy models. For comparison purpose a neural network model trained by PSO algorithm is also implemented. Experiments were carried out on a Pentium IV, 2.8 GHz Machine with 512 MB RAM and the codes were executed using C/C++. Test data was presented to the trained soft computing models and the output from the network was compared with the actual index values in the time series. The assessment of the prediction performance of the different soft computing 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 performance of the trained forecasting model for the test data. MAP is defined as follows:

$$MAP = max(\frac{|P_{actual,i} - P_{predicted,i}|}{P_{predicted,i}} \times 100) \tag{5}$$

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

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}(\frac{|P_{actual,i} - P_{predicted,i}|}{P_{actual,i}}) \times 100 \tag{6}$$

where N represents the total number of days.

We used instruction set $I = \{+_2, +_3, +_4, x_0, x_1, x_2\}$ for modeling Nasdaq-100 index and instruction set $I = \{+_2, +_3, \ldots, +_6, x_0, x_1, x_2, x_3, x_4\}$ for modeling NIFTY index. We used a NN model with network architecture {3-12-1} for modeling Nasdaq-100 index and another NN model with network structure {5-12-1} for modeling NIFTY index.

Table 1 summarizes the training and test results achieved for the two stock indices using the three different approaches. Figures 2 and 3 depict the test results for the one day ahead prediction of Nasdaq−100 index and NIFTY index respectively.

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

	NN-PSO	Fuzzy-TS	H-TS-FS
Training results (RMSE)			
Nasdaq−100	0.02573	0.02634	0.02498
NIFTY	0.01729	0.01895	0.01702
Testing results (RMSE)			
Nasdaq−100	0.01864	0.01924	0.01782
NIFTY	0.01326	0.01468	0.01328

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

	NN-PSO	Fuzzy-TS	H-TS-FS
		Nasdaq-100	
Correlation coefficient	0.997704	0.997538	0.997698
MAP	141.363	156.464	138.736
MAPE	6.528	6.543	6.205
		NIFTY	
Correlation coefficient	0.997079	0.997581	0.0997685
MAP	27.257	30.432	27.087
MAPE	3.092	3.328	3.046

6 Conclusions

In this paper, we have demonstrated how the chaotic behavior of stock indices could be well represented by different hybrid learning paradigms. Empirical results on the two data sets using three different learning models clearly reveal the efficiency of the proposed techniques. In terms of RMSE values, for Nasdaq-100 index, H-TS-FS performed marginally better than other models and for NIFTY index, NN approach gave the lowest generalization RMSE values. For both data sets, H-TS-FS has the lowest training error. For Nasdaq-100 index (test data), Fuzzy TS has the highest correlation coefficient but the lowest value of MAPE and MAP value was achieved by using the H-TS-FS model. Highest correlation coefficient, and the best MAPE/MAP values for NIFTY index were achieved using the H-TS-FS trained using GP-like evolutionary algorithm and the PSO model. The number of fuzzy rules obtained by direct fuzzy method are 27 for Nasdaq-100 data and 243 for NIFTY data. The number of fuzzy rules for obtained by H-TS-FS are 18 for Nasqad-100 data and 99 for NIFTY data. 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 there are various advantages and disadvantages for the different techniques considered. There is little reason to expect that one can find a uniformly best learning algorithm for

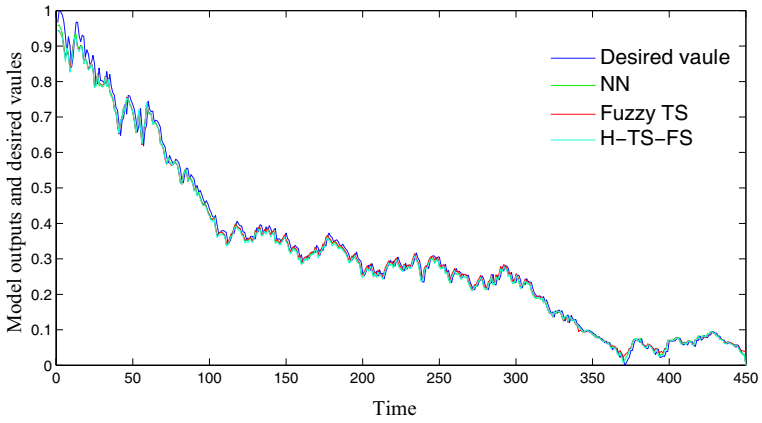

Fig. 2. Test results showing the performance of the different methods for modeling Nasdaq-100 index

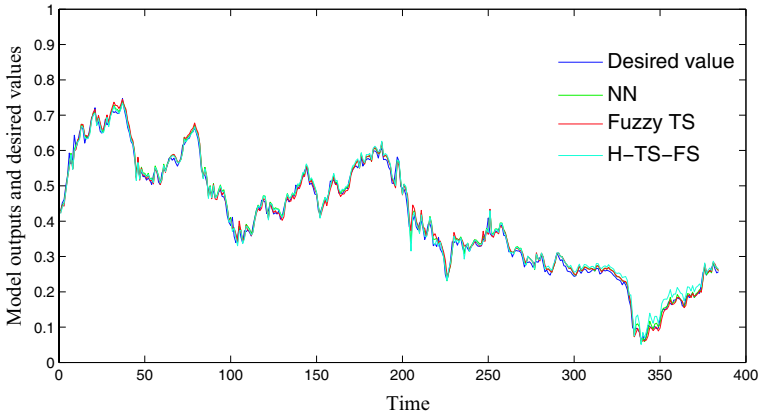

Fig. 3. Test results showing the performance of the different methods for modeling NIFTY index

optimization of the performance for different stock indices. This is in accordance with the no free lunch theorem, which explains that for any algorithm, any elevated performance over one class of problems is exactly paid for in performance over another class [17]. Our future research will be oriented towards determining the optimal way to combine the different learning paradigms using an ensemble approach [18] so as to compliment the advantages and disadvantages of the different methods considered.

References

1. Abraham A., Nath B. and Mahanti P.K.: Hybrid Intelligent Systems for Stock Market Analysis, Computational Science, Springer-Verlag Germany, Vassil N Alexan-

drov et al (Editors), USA, (2001)337-345.
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), (2003)143-160.
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), (2002)155-159.
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),(2002)361-377.
5. Nasdaq Stock Market^{SM}: http://www.nasdaq.com
6. National Stock Exchange of India Limited: http://www.nse-india.com
7. Takagi, T. and Sugeno, M. : Fuzzy identification of systems and its application to modeling and control. IEEE Trans. Syst. Man, Cybern., **15**, (1985) 116-132
8. Brown, M., Bossley K.M., Mills D.J. and Harris C.J. : High dimensional neurofuzzy systems: overcoming the curse of dimensionality. Proc. 4th Int. Conf. on Fuzzy Systems. (1995) 2139-2146
9. Rainer, H. : Rule generation for hierarchical fuzzy systems. Proc. of the annual conf. of the North America Fuzzy Information Processing. (1997) 444-449
10. Wang, L.-X. : Analysis and design of hierarchical fuzzy systems. IEEE Trans. Fuzzy Systems, **7**, (1999) 617-624
11. Wang, L.-X. : Universal approximation by hierarchical fuzzy systems. Fuzzy Sets and Systems, **93** (1998) 223-230
12. Wei, C. and Wang, L.-X. : A note on universal approximation by hierarchical fuzzy systems. Information Science, **123**, (2000) 241-248
13. Lin, L. C. and Lee, G.-Y. : Hierarchical fuzzy control for C-axis of CNC tuning centers using genetic algorithms. Journal of Intelligent and Robotic Systems, **25** (1999) 255-275
14. Duan, J.-C. and Chung, F.-L. : Multilevel fuzzy relational systems: structure and identification. Soft Computing **6** (2002) 71-86
15. Chen Y., Yang B. and Dong J.: Automatic Design of Hierarchical TS-FS Models using Ant Programming and PSO algorithm, The Eleventh International Conference on Artificial Intelligence: Methodology, Systems, Applications, LNCS, 3192, (2004)285-294.
16. Kennedy, J. and Eberhart, R.C.: Particle Swarm Optimization. In Proc. of IEEE International Conference on Neural Networks, (1995) 1942-1948.
17. Macready W.G. and Wolpert D.H.: The No Free Lunch theorems, IEEE Trans. on Evolutionary Computing, 1(1), (1997)67-82.
18. 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),(2004)112-122.