Global Journal of Pure and Applied Mathematics
ISSN 0973-1768 Volume 5, Number 3 (2009), pp. 189–200

http://www.ripublication.com/gjpam.htm

Strategies for Indian Share Market Investment through ANFIS

P. Thirunavukarasu

Assistant Professor, P.G. and Research Department of Mathematics, Periyar E.V.R. College (Autonomous), Tiruchirappalli,Tamil Nadu, India

Abstract

The stock market is one of the most popular investing places because of its expected high profit. Traditionally, technical analysis approach that predicts stock prices based on historical prices and volume, basic concepts of trends, price patterns and oscillators, is commonly used by stock investors to aid investment decisions. Advanced intelligent techniques, ranging from pure mathematical models and expert systems to fuzzy logic have also been used in many financial trading systems for investing and predicting stock prices. In recent years, most of the researchers concentrate their research work towards the future prediction of share market prices by using Artificial Neural Networks (ANN). In this paper we newly propose a methodology that neural network is applied to the investor's financial decision making to invest in the restricted share (sub-sectors) in a continuous time frame work and further it is extended to establish the fuzzy design rule to design an optimum investment rule that one can earn more return on investment in a share market. Again, the FIS trained through ANFIS and the proposed ANFIS (Artificial Neural Fuzzy Inference System) has been tested with stock data obtained from the Indian Share Market Index BSE. In this paper, the design, implementation and performance of the proposed neural network and ANFIS are described.

Keywords: Indian Stock Market, Neural Networks, Fuzzy inference rule and share market index.

Section 1: Introduction

From the beginning of time it has been man's common goal to make his life easier. The prevailing notion in society is that wealth brings comfort and luxury, so it is not surprising that there has been so much work done on ways to predict the markets. Various technical, fundamental, and statistical indicators have been proposed and

used with varying results. However, no one techniques or combination of techniques have been successful enough to consistently "beat the market". Traditionally, technical analysis approach specified in Maier, H.R. and Dandy, G.C (1996), that predicts stock prices based on historical prices and volume, the Dow Theory, basic concepts of trends, price patterns and oscillators, is commonly used by stock investors to aid investment decisions. Advanced intelligent techniques ranging from pure mathematical models and expert systems [Stokelj, T, Paravan, D and R. Golob (2002), Garson, G.D.,(1991) and Milne, L.K.,(1995)] to neural networks [Gedeon, T.D,(1997), Kim, C.Y., Bae, G.J., Hong, S.W., Park, C.H., Moon, H.K. and Shin, H.S.(2001), D.E. Rumelhan, J.I.. McClelland, el. al.,(1986) and D.E. Rumelhart. B. Widrow and M.A. Lehr,(1994)] have also been used in many financial trading systems for stock prediction. Ultimately, in most of the researchers have been derived the various methodology for predicting future share market prices using ANN. But in this paper, the ANFIS is used to identify the proper section of sub-sectors of investment that one can earn the optimum return on investment.

Indian Share Market Index BSE is increased or decreased depending on the performance of the various sub indexes (sectors) namely BSEIT, BSEED, BSEFMCG, BSEHC, BSECG, TECH, BSEPSU, BANKEX, AUTO, METAL and OILGAS. During data collection (Figure-1 represents the quarter average index value of last 42 slots), we have observed that the fluctuations and variations can happened rapidly for all sub-sectors from time to time. Therefore, the area of investment in the sub-sectors and its identification is very difficult task and ambiguity in nature of investors. The decision making of the identification of the optimum sub-sectors investment for one or more scripts are vital role. The combinations are some portion of the shares are more profit oriented but the risk is more and some of the shares are very less risk but the profit is some what good, but not that much of first one. In this paper, we proposed a new methodology for optimizing share profit by the way of ANFIS. The proposed network has been tested with BSE data.

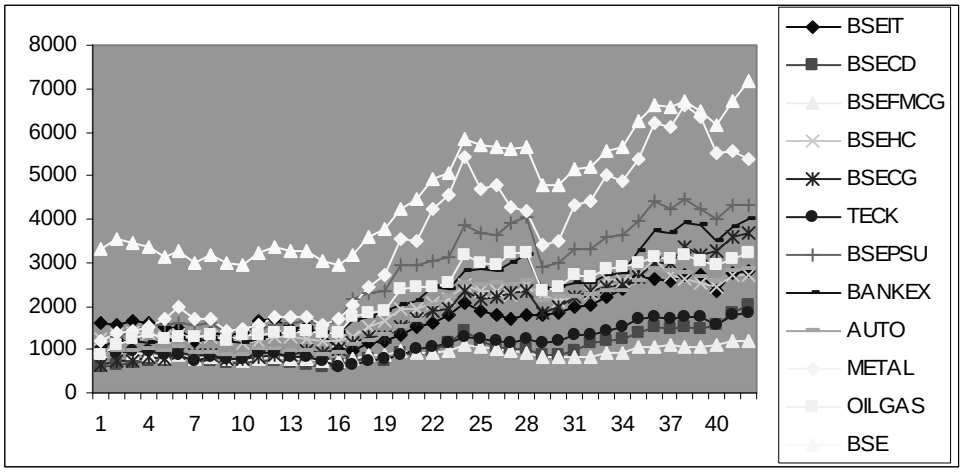

Figure 1: BSE Index Vs Average of 42 quarter.

The following ways the paper was arranged. Section 2 provides the information related to neural network and its learning rule (back propagation) for predicting the minimum number of iterations required to sub-sectors investment in stock a market. Section 3 covers the introduction of fuzzy design rule and the evaluation procedure for ANFIS. This will help how ANFIS is useful for predicting the script (shares) selection of the sub-sectors investment. In both the sections details how ANFIS has been designed to outperform current techniques. The conclusion is given in section 4.

Section 2 : Artificial Neural Network

Before the age of computers, people traded stocks and commodities primarily on intuition. As the level of investing and trading grew, people searched for tools and methods that would increase their gains while minimizing their risk. Statistics, technical analysis, fundamental analysis, time series analysis, Chaos theory and linear regression are all used to attempt to predict and benefit from the markets direction. None of these techniques has proven to be the consistently correct prediction tool that is desired, and many analysis argue about the usefulness of many of the approaches. However, these methods are presented as they are commonly used in practice and represent a base-level standard for which neural networks should outperform. Also, many of these techniques are used to preprocess raw data inputs, and their results are fed into neural networks as input.

NN concepts and its terminology

To model complex process in many physical systems, the use of Artificial Neural Network (ANN) has been extensively in use in recent times. As a branch of artificial intelligence, this robust and versatile tool is being modeled after the human neurological system, consisting of a series of neurons (the basic computing elements), interconnected together to allow recognition of incidents that have had a similar pattern to the current input. Especially for pattern recognition and function approximation, ANN, equipped with parallel distributed processing architecture, is well recognized as a very powerful computational tool, having the ability to learn and to generalize from examples to produce meaningful solutions to problems even in case of erroneous or incomplete data.

Neural networks have widely been used in water resources and environmental modeling for forecasting water resources and environmental variables, as well as for time series modeling (Maier, H.R. and Dandy, G.C (1996)). Most often feed-forward networks, which employ a sliding window over a sequence of data (i.e., to induce the function in ANN architecture, using a set of older records as inputs and a single output as the target value of the network), are used for time series modeling. Although, in general, non-linear, auto-regressive time series modeling is difficult than linear models, yet with the ANN approach such a restriction does not apply (Garson, G.D.,(1991)). Similarly, in contrast to the auto-regressive and moving average methods, ANNs are nonparametric data driven approaches that can capture nonlinear data structures without prior assumption about the underlying relationship in a particular problem. Besides, ANNs are more general and flexible modeling and

analysis tools for forecasting applications, capable of finding nonlinear structures, as well as linear ones. In fact, linear autoregressive (AR) models are special cases of ANNS without hidden nodes(Milne, L.K.,(1995)).

For an explanatory or casual forecasting problem, the inputs to an ANN are usually the independent or predictor variables (Gedeon, T.D, (1997)). The functional relationship estimated by the ANN can be written as:

$$Y = F(x_1, x_{2,} x_{3, \ldots} x_n) \quad (1)$$

Where $x_1, x_{2,} x_{3, \ldots} x_n$ are n independent variables and y is a dependent variable. In this sense, the neural network is functionally equivalent to a nonlinear regression model. For an extrapolative or time series problem, on the other hand, inputs are typically the past observations of the series and the output is the future value. The function mapping performed by the ANN is as follows:

$$Y_{t+1} = F(y_t, y_{t-1,} y_{t-2, \ldots,} y_{t-n}) \quad (2)$$

Where y_t is the observation at time t. Thus the ANN is equivalent to the nonlinear autoregressive model along the series.

For a time series problem, a training pattern consists of a fixed number of lagged observations of the series. There are N observations $y_1, y_{2,} y_{3, \ldots,} y_N$ in the training set and if one-step-ahead forecasting is required, then using an ANN with n input nodes, we have N-n training patterns. The first training pattern will be composed of $y_1, y_{2,} y_{3, \ldots,} y_n$ as inputs and Y_{n+1} as the target output. The second training pattern will contain $y_1, y_{2,} y_{3, \ldots,} y_{n+1}$ as inputs and Y_{n+2} as the desired output. Finally, the last training pattern will be $y_{N-n}, y_{N-n+1, \ldots,} y_{N-1}$ for inputs and y_N for the target output. Typically, a least- squares based objective function or cost function to be minimized during the training process is (Gedeon, T.D, (1997)):

$$E = \frac{1}{2} \sum_{I=n+1}^{N} (y_i - a_i)^2 \quad (3)$$

where a_i is the output of the network and ½ is included to simplify the expression of derivatives computed in the training algorithm.

Most of the environmental and water resources applications of ANN have used feed-forward networks for function approximation. While a majority of them used the back-propagation training algorithm, a few of them attempted other algorithms (Maier, H.R. and Dandy, G.C (1996)). The general structure of a feed-forward neural network is shown in Fig.2. The nodes in an input layer receive the inputs of the model and they flow through the network and produce outputs at nodes in the output layer. The working principle of feed-forward neural network is available elsewhere (Masters, T (1993)). Mathematically, a three-layer neural network with *I* input nodes, *J* hidden nodes in a single hidden layer, and *K* output nodes, can be expressed as:

$$O_{pk} = f_1\left(\sum_{j=1}^{J} w_{jk}^{o} f_2\left(\sum_{i=1}^{I} w_{ij}^{h} x_{pi} + b_1^j \right) + b_2^k \right), \quad \forall k \in 1,2,...,K \quad (4)$$

where O_{Pk} is the output from the k^{th} node of the output layer of the network for the P^{th} vector (data point); X_{Pi} is the input at the i^{th} node of input layer from p^{th} vector

(data point); w_{jk}^{o} is the connection weight *between* j^{th} node of the hidden layer and k^{th} node of the output layer (Fig. 2); w_{ij}^{h} is the connection weight between i^{th} node of the input layer and j^{th} node of the hidden layer; and b_1^j *and* b_2^k are bias terms; and f1(.) and f2(.) are activation functions. The logistic sigmoid function, a commonly used activation function has the form of JJ.Vidal (1991):

$$f(x) = \frac{1}{1+e^{-x}} \tag{5}$$

The linear activation function has the form

$$f(x) = x \tag{6}$$

In this study sigmoid function is used for f_2 and the linear function is applied for f_1. The sigmoid functions (which plots like curves) normally have a tendency to push the value *of f(x)* to the extremes (binary in the case of logistic sigmoid; bipolar in the case of tanh function). Thus the sigmoid functions are more suitable for classification problems. When continuous outputs are expected, as in the case of time series modeling, sigmoid functions are not a good choice. There are several other activation functions used in many other studies, however, this work did not analyze the suitability of activation functions for reservoir inflow generation.

Linear autoregressive models assume the prediction equation to be a linear combination of a fixed number of previous data in the time series. Including a noise term, it can be written as (D.E.Rumelhart, B.Widrow and M.A.Lehr (1994)):

$$x(t) = \sum_{i=1}^{p} \alpha_i x(t-i) + \xi(t)$$

$$= F^{L}(x(t-1),x(t-2),\ldots,x(t-p)) + \xi(t) \tag{7}$$

If p previous sequence elements are taken, one speaks of an AR(p) model of time series. Finding an appropriate AR(p) model means choosing an appropriate and estimating the coefficients α_i through techniques like least squares optimization procedures. This techniques, although rather powerful, is naturally limited, since it assumes a linear relationship among sequence elements (P.D.Wassennan (1989)). It becomes clear that a feed forward neural network can replace the linear function F^{L} in equation (7) by an arbitrary non-linear function F^{NN} as in equation (8) (K.G.Schwcller and A.L.Plagman (1989)).

$$x(t) = F^{NN}(x(t-1),x(t-2),\ldots,x(t-p)) + \xi(t) \tag{8}$$

This non-linear function can be estimated based on samples from the series, using one of the well-known learning or optimization techniques like back propagation or conjugate gradient (B.Widrow, D.E.RumelhaJrl and M.A.Lchr (1994)). Making F^{NN} dependent on p previous elements is identical to using p input units being fed with p adjacent sequence elements. This input is usually referred to as a time window, since it provides a limited view on part of the series. Non-linear autoregressive models are potentially more powerful than linear ones in that they can model much more complex underlying characteristics of the series.

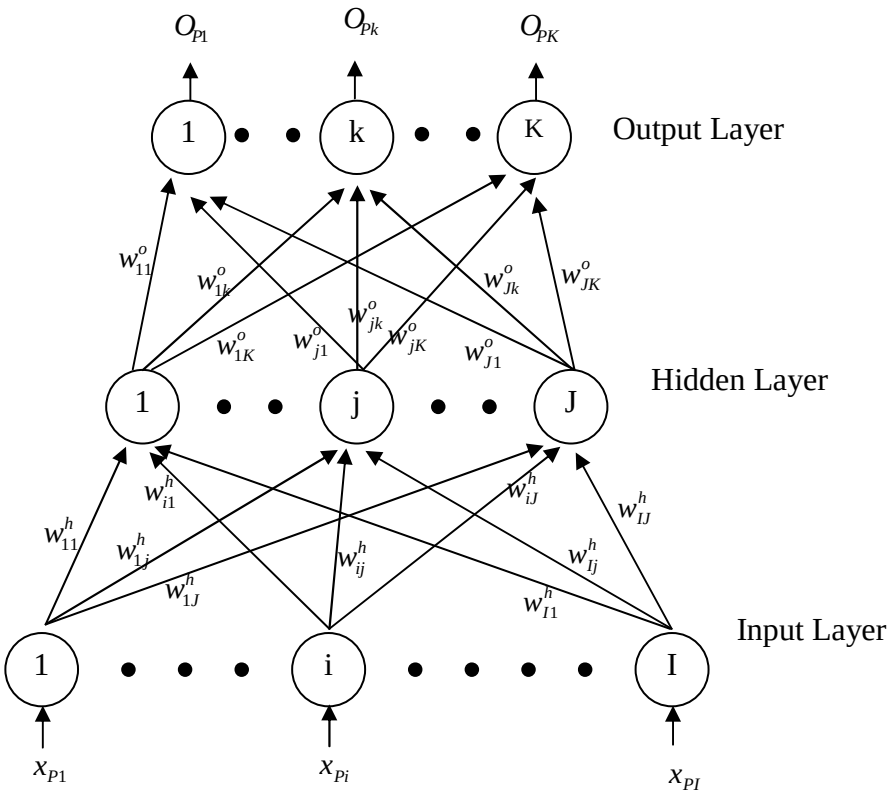

Figure 2: Three-Layer Feed-Forward Neural Network Architecture.

Selection of Activation Function

Suitable activation function for the Hidden Units , is needed to introduce non-linearity into the network, which gives the power to capture nonlinear relationship between input and output. Three commonly used activated functions are logistic, linear, tanh. Since the expected output is a continuous variable (not a classification problem with unbounded function), linear activation function (g(x)) was used (in stead of logistic sigmoidal or tanh functions mostly used for classification problems and for bipolar output ranges, i.e., between -1 and +1). The form of the linear activation function is as below:

g(x) = x

g'(x) = 1

Determination of Optimum Number of Iteration

Error back propagation with momentum can be viewed as gradient descent with smoothing. To stabilize the weight trajectory by making the weight change a combination of the gradient-decreasing term plus a fraction of the previous weight change, a specific momentum parameter is selected in any ANN architecture. The rate of learning by the network is computed by a factor called learning rate. In standard back-propagation, too low a learning rate makes the network learn very slowly and too high a learning rate makes the weights and objective function diverge, leading to

no learning at all. In the present case, Since the training is continues (non-batch type), the training rate can be maintained constant throughout the training.

The effect of number of iteration on errors (at different learning rates and momentum parameters) for varying number of hidden nodes was studied. When learning rate is 0.5 then the error term is common to 0.0554 for all momentum parameter. Based on the analysis, it found that the minimum number of iteration for different number of nodes was 6000 for minimum errors. In this section, we proved the minimum number of iterations required to get the expected minimum errors and this work will be used in section 3.

Design of Adaptive Neuro-Fuzzy Inference System

Applying a neuro-fuzzy inference system to model this real life problem is quite natural, because of the similarity to real life human decision-making. Some design issues were set up based on information from share market experts, while others were learned or experimented. The initial investment and portion of investment on various sub-sectors of share market were fuzzified for initial training. Later on we may use linguistic descriptions (low, medium, and high) from investing details for various sub-sectors of the share market. Some of the design issues were the following:

(1) Number of input membership functions: the fuzzy membership functions were set up based on knowledge of investment of sub-sectors decisions. We determined that three membership functions (low, medium, high) for each of the six soft constraints model BSE index. Due to increased computational time, we take the average data for recent 42 months BSE index data collection.

(2) Type of input membership functions: based on the properties of the soft constraints we primarily consider triangular membership functions, yet trapezoid, Gaussian curve and generalized bell-shaped membership functions were tested as well. The membership function

of this type is given by $\mu_A(x) = \begin{cases} 0 & \text{for } x < a \\ \dfrac{x - a}{b - a} & \text{for } a \le x \le b \\ \dfrac{c - x}{c - b} & \text{for } b < x \le c \\ 0 & \text{for } x > c. \end{cases}$ where a, b and c are the extreme points of the triangular model.

(3) Type of output membership functions: we used a single output, obtained using weighted average defuzzification. All output membership functions had the same type and were either constant or linear (zeroth and first order Sugeno type system).

(4) The number of output membership functions: it ranged from 2 to 243.

(5) The number of rules: for a well defined fuzzy system we need to define control actions (fuzzy output) for every possible combination of input membership

function values. If the training set doesn't include examples for given combination of values, but the testing set does then in the testing phase we may still "guess" the output value using the aggregate gain(return on investment) function from all the learned fuzzy rules. This can be done before the actual testing phase begins resulting in a fully defined fuzzy system. In our case six constraints, each with three membership functions result in 729 fuzzy rules, where the linguistic values were not negated, and they were connected with "and" relation. However, through rule extraction and combination of rules we drastically decrease the number of rules later.

(6) Performance function: some of the widely used where SSE is Sum of Squared Error, l is a confidence factor and w is weight. To verify training performance we can also verify the correct classification rate (R.Kozma et al., (1996)).

(7) Optimization methods: back-propagation and hybrid (mixed least squares and back propagation) methods had been used as optimization methods.

(8) Partitioning the data into training, cross-validation and testing sets: the range of the training data set size was 50% to 90%. The cross-validation and testing data sets each took half of the rest of the data (5%-25%). The use of cross validation is optional but in our implementation is important, to avoid overinvestment.

(9) Number of epochs: in most runs it was set up to 6000. Through an adaptive neuro-fuzzy inference system the range of the membership functions are learned, fuzzy rules are created and their weights are adjusted in order to better model the training data. The performance function values are calculated, and classification is provided.

Results and Rule Extraction

For implementation, we used Matlab 7.0 environment with fuzzy logic toolbox. Various types of input membership functions have been tried, and as we expected the triangular membership functions performed best closely followed by trapezoid, Gaussian curve and generalized bell-shaped membership functions. Linear output membership functions performed better than constant ones.

For optimization method the back-propagation and hybrid optimization method performed similarly regarding the performance function and classification, but the hybrid method running time was about 5 times as long as that of the back-propagation. To avoid over-investment we used cross-validation. In some cases the minimum cross-validation error occurred within the first epoch. This meant that the given set of membership functions were not a good choice for modeling the training data. This also indicated that either more data need to be selected for training, or we need to modify the membership functions (both the number of membership functions and their types). After training, the FIS parameters were set to be associated with the minimum cross-validation error.

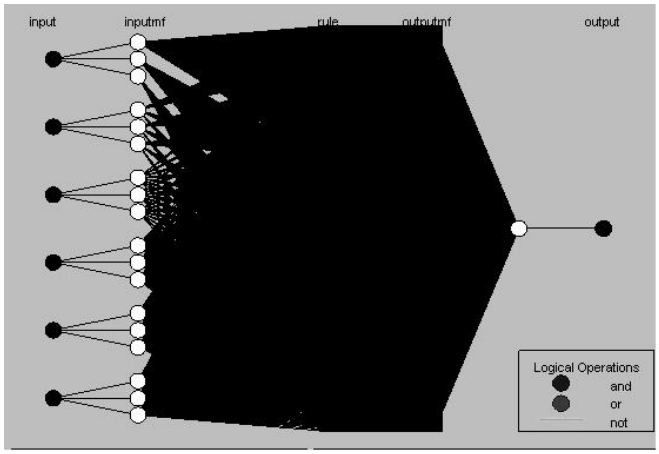

Figure 3: ANFIS model structure.

Figure 3 depicts the 6- layered architecture of single output ANFIS and the functionality of each layer is as follows:

Layer 1. Input neurons: 11 input neurons, namely BSEIT, BSEED, BSEFMCG, BSEHC, BSECG, TECH, BSEPSU, BANKEX, AUTO, METAL and OILGAS and each of them have three soft constraints (low, medium and high). The fuzzified BSE index value will be the input and output of this layer 1. The output of this layer 1 will be the input of layer 2. For reducing the time complexity, we select six sub-indexes with high return on investment with low risk by using K.S.Ravichandran and et al., (2006).

Layer 2. Input membership functions: three triangular membership functions for each input neuron may be established and each node of layer 2 is connected with all inner hidden nodes. The number of hidden layers may be generated dynamically while using Matlab 7.0. The output of layer 2 is getting by applying fuzzy BSE index value to the triangular membership function and this membership function will give the output of layer 2.

Layer 3. Fuzzy rule left hand sides: each connected to 4 input membership functions.

Layer 4. Output membership functions (right hand sides): the right hand side rules are in one to one relation with the left hand side rules.

Layer 5 Aggregated output: each output membership function gets aggregated along with the maximum weight they carry.

Layer 6. Output (decision): The input and output membership functions and its relations are giving below:

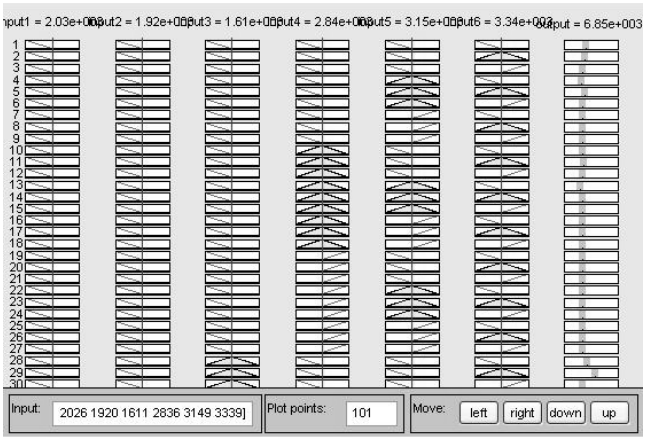

Figure 4: FIS rule.

The surface diagram for the above is given below.

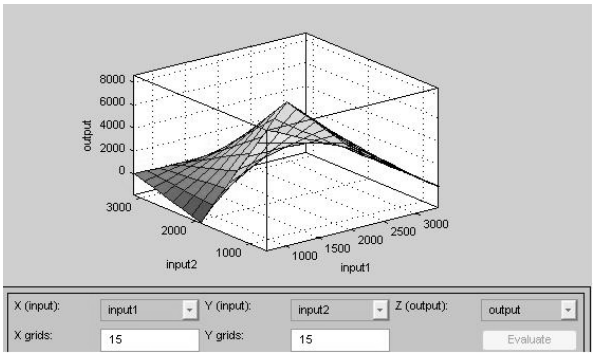

Figure 4: Surface diagram.

The following results have obtained by the training implemented through hybrid method and during its training period the following nodes and parameters have automatically created except the input and output nodes and its initial BSE index values. The number of nodes are 1503, number of linear parameters are 729, number of non-linear parameters are 54, number of linear parameters are 783, number of training data pairs 12 and the number of fuzzy design rules generated are 729. For three triangular input membership functions, 500 epochs, linear output membership functions, the Sum Squared Error between the actual and desired outputs was 0.177.

Based on 729 fuzzy design rules and its aggregated function value, the optimized design rule has been established (the percentage of amount per 100 rupees and its scripts given below). While making a fuzzy membership function for the various fuzzy sub-indexes the risk factors are also considered. Ranking of the various sub-indexes are arranged for taking both the return on investment and risk factor using K.S.Ravichandran and et al., (2006).

A typical running time on a 3.3 GHz Pentium IV processor was about 96 minutes for 6000 epochs when back-propagation optimization method was used with three

triangular membership functions for each input. Other methods took considerably longer time, particularly the hybrid optimization method.

Therefore, the optimum priority rule out of 729, if (METAL is medium) and (BSEPSU is medium) and (BANKEX is high) and (BSECG is high) and (OILGAS is low) and (BSEIT is high) then the output is maximum according to Sugeno aggregation function. The above result is obtained by taking care of high return on investment and less risk. For every 100 rupees investment, 58 rupees should be evenly invested in high profile shares, 33 rupees should be evenly invested in medium profile shares and 9 rupees should be evenly invested in low profile shares. Further enhancement to this work, even among the high profile shares, we can able to investigate the percentage of amount invested.

From 11 sub-sectors of indexes, if we use ranking of the above 11 sub-sectors of indexes with minimum risk (K.S.Ravichandran and et al., (2006)) then we select 6 among them. If we use ANFIS to the above six indexes then we get 729 rules. If we use all the 11 indexes then we get 3^{11} (= 177147) rules and the optimum rule will be selected among 3^{11} rules is very difficult. Therefore, we need a help of non-traditional techniques called genetic algorithms or the association rules of data mining (R.Agrawal et al., (1996)) and then rules extraction can be obtained by any one of the above said methods. This may be the further work of this paper.

Conclusion

The study reveals that a high potential of adaptive neuro-fuzzy inference system predicting the Indian share market investment BSE. Results have been shown that the human-kind decisions can be achieved with the model. The Sugeno FIS rules are extracted from direct verification method and the maximum output was tested. In this work, one can easily select an optimum rule that will give the maximum return on investment with low risk among the selected 729 rules. For further work, if we use either genetic algorithm or association rules in data mining that will help a lot to identify the script selection among the high, medium and low investment group but it will take a lot of complexity including a very large computational time.

References

[1] Maier, H.R. and Dandy, G.C (1996), “The use of artificial neural networks for the prediction of water quality parameters”, Water Resources Research, 32(4), pp.1013-1022.

[2] Stokelj, T, Paravan, D and R. Golob (2002), “Enhanced artificial neural network inflow forecasting algorithm for run-of-river hydropower plants, Journal of Water Resources Planning and Management”, ASCE, 128(6), pp.415-423.

[3] Garson, G.D.,(1991), "Interpreting neural network connection weights," AI Expert, April 1991, pp.47-51.

[4] Milne, L.K.,(1995), "Feature selection with neural networks with contribution measures," Proceedings of the Australian Conference on Artificial Intelligence AI'95, Canberra.

[5] Gedeon, T.D,(1997), "Data mining of inputs: Analysing magnitude and functional measures," International Journal of Neural Systems, 8, pp.209-218.

[6] Kim, C.Y., Bae, G.J., Hong, S.W., Park, C.H., Moon, H.K. and Shin, H.S.(2001), "Neural network based prediction of ground surface settlements due to tunneling", Computers and Geotechnics, 28, 517-547.

[7] D.E. Rumelhan, J.I.. McClelland, el. al.,(1986), “Parallel Distributed Processing: Explorations in the Microstructure or Cognition”; Volume I: Foundations, MIT Press, Cambridge, Mass.

[8] D.E. Rumelhart. B. Widrow and M.A. Lehr,(1994), “The basic ideas in neural networks”, Communications of the ACM, 37, pp. 87-92.

[9] Masters, T.(1993), **“**Practical Neural Network Recipes in C++**”,** San Diego: Academic Press.

[10] JJ. Vidal, (1991), “Exploring Artificial Neural Networks with Mathematics”. Mathcmatica notebook.

[11] P.D. Wassennan,(1989), ‘Neural Computing: Theory and Practice”, Van Nostrand Reinhold, New York.

[12] K.G. Schwcller and A.L. Plagman,(1989), “Neural nets and alphabets: introducing students to neural networks”, SIGCSE Bulletin, 21., pp. 2-10.

[13] B. Widrow, D.E. RumelhaJrl and M.A. Lchr,(1994), “Neural networks: applications in industry, business and science”, Communications of the ACM, 37, pp. 93-105.

[14] K.S.Ravichandran et al.(2006), “A Study on Market Sentiments (Investments): Fuzzy Approach” , China Journal of Finance, China, to appear. (No.3, Vol.3, 2006).

[15] R. Kozma, M. Sakuma, Y. Yokoyama and M. Kitamura,(1996), “On the Accuracy of Mapping Backpropagation with Forgetting,” Neurocomputing, Vol. 13, No. 2-4, pp. 295-311.

[16] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen and A. Verkamo,(1996), “Fast Discovery of Association Rules,” in Advances in Knowledge Discovery and Data Mining, MIT Press, pp. 307-328.