The Development of a Weighted Evolving Fuzzy Neural Network

Pei-Chann Chang[1], Chen-Hao Liu[1], Chia-Hsuan Yeh[2], and Shih-Hsin Chen[1]

[1] Department of Industrial Engineering and Management, Yuan-Ze University, 135 Yuan-Dong Rd., Taoyuan 32026, Taiwan, R.O.C.
iepchang@saturn.yzu.edu.tw,
{s929510, s939506}@mail.yzu.edu.tw
[2] Department of Information Management, Yuan-Ze University, 135 Yuan-Dong Rd., Taoyuan 32026, Taiwan, R.O.C.
imcyeh@saturn.yzu.edu.tw

Abstract. This study modifies the Evolving Fuzzy Neural Network Framework (EFuNN framework) proposed by Kasabov (1998) and adopts a weighted factor to calculate the importance of each factor among these different rules. In addition, an exponential transfer function (exp (-D)) is employed to transfer the distance of any two factors into the value of similarity among different rules, thus a different rule clustering method is developed accordingly. The intensive experimental results show that the WEFuNN performs very well when applied in the PCB sales forecasting.

1 Introduction

The field of Artificial Intelligence (AI), such as Neural Network (NN), Fuzzy Theory, Expert System (ES), Genetic Algorithms (GA), and Rule Induction has been rapidly developed in recent years. Each algorithm has their own strengths but there are still some limitations of them. In order to reduce these limitations, the hybrid algorithm was developed from combining two or three different A.I. approaches and then draws on the strength of each to offset the weakness of the others. Thus, the main ideal of this research is to develop a better weighted evolving fuzzy neural network for various applications.

This study modifies the Evolving Fuzzy Neural Network Framework (EFuNN framework) proposed by Kasabov [14] and adopts a weighted factor to calculate the value of similarity of different rules.

2 Literature Review

Soft computing algorithm which combined fuzzy theory with neural network has found a variety of applications in various fields ranging from industrial environment control system, process parameters, semi-conductor machine capacity forecasting, business environment forecasting, financial analysis, stock index fluctuation forecasting, consumer loan, medical diagnosis and electricity demand forecasting.

The study by Lin and Lee [19] is the most earliest study to combines the fuzzy theory with neural network. They proposed a hybrid model which combines the idea

D.-S. Huang, K. Li, and G.W. Irwin (Eds.): ICIC 2006, LNAI 4114, pp. 212 – 221, 2006.

of fuzzy logic controller, neural network structure and learning abilities into an integrated neural-network-based fuzzy logic control and decision system. Subsequently, several researchers also investigate some related studies with regard to the application of this combined approach and then continually developed several kinds of approaches [4, 5, 6, 7, 8, 12, 16, 17, 18, 20, 21, 22, and 23]. The commonly used methods are as follows:

1. Fuzzy Adaptive Learning Control Systems (FALCON) [Zhong, 1999]
2. Fuzzy Back-Propagation Network (FBPN)
3. Adaptive Neuro-Fuzzy Inference Systems (ANFIS)
4. Fuzzy Hyper Rectangular Composite Neural Networks (FHRCNNs)
5. Fuzzy Neural Network (FuNN)

Some of the previous studies concerning the application of fuzzy neural network in the forecast aspect will be presented as follows:

Kasabov [15] modified fuzzy neural network and then proposed the method of FuNN/2. Under the framework of FuNN, the author employed the function of Genetic Algorithms which has the ability to search quickly in large spaces and then offset the insufficiency of neural network's parameters setting. The EFuNN combined unsupervised learning and supervised learning was proposed by Kasabov [14]. The first stage: in order to achieve the objective of connecting fuzzy rules, Kasabov employed unsupervised learning to adjust the connection weights of fuzzy input and fuzzy rule. The second stage: the supervised learning of BPN was used to adjust the connection weights of fuzzy rule and fuzzy output so as to achieve the goal of inner rule of incoming vectors and outgoing vectors.

Abraham and Baikunth [1] proposed a method of Fuzzy Neural Network that employed hybrid supervised and unsupervised learning to forecast short-term electricity demand in the State of Victoria. The authors regarded "the maximum and minimum temperatures of the day", "previous day's electricity demand", "season" and "the day of week" as input factors and used EFuNN to develop a forecasting model of electricity demand. The EFuNN forecasting technique is also compared with Conjugate Gradient Algorithm (CGA), Back-propagation Network (BPN) and Box-Jenkins. Abraham et al. [2] utilized Fuzzy Neural to forecast the long-term rainfall in Kerala state. The rainfall was affected by global warming, season, storms and butterfly effect and so on. Under the BPN framework, the authors made explicit values to be fuzzy and then proceeded with fuzzy rule aggregation, clustering and extraction. Moreover, employed a process of training via BPN algorithms and adjusted the weights of concealment to output through each time of training. Abraham et al. [3] applied Fuzzy Neural Network to analyze the strategic decision-making of stock market. Firstly, the researchers choose six stock indices from the listing companies in the Nasdaq Stock Market and used conjugate gradient algorithm to forecast these stocks. Kasabov [13] investigated the short-term forecasting of the Gross Domestic Product (GDP) for totally 15 countries which are EU countries, USA and so on. The related factors considered in this study are Consumer Product Index (CPI), Interest Rate (IR), Unemployment Rate and GDP per capita. The results of this study revealed that the MSE of the 15 countries performed well after fuzzy rule re-aggregation, rule adaptation and rule extraction. It also reveled that the results of EFuNN forecasting are highly accurate in various kinds of environments.

3 Development of WEFuNN

The main ideal of this research is to modified the EFuNN [13], generate better weight combination so as to improve the forecasting accuracy. The WEFuNN will be used to forecast the demand of PCB product and it is a five-layer network (Figure. 1) where nodes and connections are created/connected as data examples are presented.

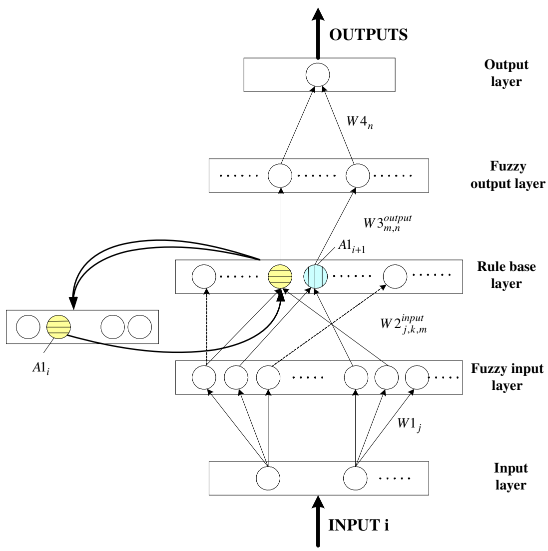

Fig. 1. Architecture of WEFuNN

3.1 Feed-Forward Learning Phase

For each training case, there are four characteristics to be processed by using the following steps:

3.1.1 Data Fuzzification Step

Triangular membership function is used to transfer the input characteristics to the fuzzy membership function $\tilde{\mu}_x$ (Figure. 2 & Eq. 1).

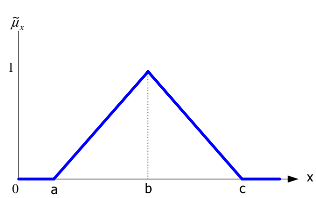

Fig. 2. The triangular membership function

$$\tilde{\mu}_x = \begin{cases} 0, x < a \\ \dfrac{x-a}{b-a}, a \le x < b \\ \dfrac{c-x}{c-b}, b \le x < c \\ 0, x \ge c \end{cases} \tag{1}$$

In this research, we added the dynamic weight ($W1_j$) for character j between the 1st layer and the 2nd layer. This process can derive better forecasting result because the triangular membership function will become a non-isosceles one.

For the first data, $MF_{j,k}^{input}(1)$ is its input fuzzy membership function for the j-th characters and the k-th region. $MF_n^{output}(1)$ is its output fuzzy membership function for the n-th fuzzy region.

3.1.2 Initial Network Setting Step

In the beginning, there is no fuzzy rule in the network. Thus, the first rule should be built by the first data input. The connection weights between the 2nd layer and the 3rd layer are $W2_{j,k,m}^{input}$; between the 3rd layer and the 4th layer are $W3_{m,n}^{output}$; between the 4th layer and the output layer are $W4_n$. m is the rule number.

3.1.3 Network Constructing Step

In order to construct the fuzzy rule and weights of the network, all of the training data must process the following sub-steps:

Step 1. Similarity Computing

Instead of Kasabov's fuzzy distance function, weighted Euclidean distance $D_{i,m}$ is used to generate the distance between the i-th case and the m-th rule, which have been determined.

$$D_{i,m} = \sqrt{\sum_j \sum_k W2_{j,k,m}^{input} \times [MF_{j,k}^{input}(i) - R_{j,k}(m)]^2} \tag{2}$$

In this weighted distance function, the direct distance between case i and rule m is computed to represent the difference of them. This study further takes the weights into account in an effort to express the degree of importance between each factor. Moreover, to present the relation between distance and similarity, an exponential transfer function is used to transfer the distance to the similarity. Which,

$$A_{i,m} = \exp(-D_{i,m}) \tag{3}$$

Comparing with the linear transfer function, we will get much larger similarity in the close distance and much smaller similarity in the large distance from this function.

Step 2. Rule Determining

Find the most similar rule of case i, where $A1_i = \max(A_{i,m})$. If $A1_i > S$, here S is the similarity threshold, it presents that the case has to merge in this rule. Thus, go to *Step 3*. Otherwise, create a new fuzzy rule and compute its connection weights, $W2_{j,k,m}^{input}$ and $W3_{m,n}^{output}$, then go to *Network Generating Step*.

Step 3. Output Computing

In this sub-step, saturating linear transfer function will be used to transfer the fuzzy membership function of case i to the fuzzy forecast output,

$$A2_{i,m} = Satlin(W3_{m,n}^{output} \times A1_i) \tag{4}$$

The figure of Saturating linear transfer function will be shown as follows:

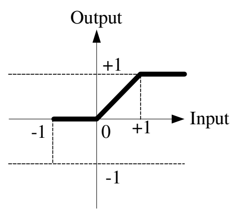

Fig. 3. Saturating linear transfer function

Step 4. Error Computing

Compute the error between fuzzy forecast of the case i and its actual fuzzy demand A_i,

$$Err_i = | A2_{i,m} - A_i | \tag{5}$$

If $Err_i < E_{thr}$ retain this forecasting result, where E_{thr} is the error threshold. Otherwise, create a new fuzzy rule and compute its connection weights, $W2_{j,k,m}^{input}$ and $W3_{m,n}^{output}$, then go to *Network Generating Step*.

Step 5. Defuzzification

Each fuzzy forecast output has been defuzzified to the real forecast output. where,

$$O_i = W4_n \times A2_{i,m} \tag{6}$$

Step 6. Weights Updating

For the occurrence of each merging process, the connection weights $W2_{j,k,m}^{input}$ and $W3_{m,n}^{output}$ should be updated as follows,

$$dist = [MF_{j,k}^{input}(i) - R_{j,k}(m)_{old}] \tag{7}$$

$$R_{j,k}(m)_{new} = R_{j,k}(m)_{old} + \alpha_1 \times dist \tag{8}$$

$$MF_n^{output}(i)_{new} = MF_n^{output}(i)_{old} + \alpha_2 \times (Err) \times (A1_i) \tag{9}$$

where, α_1 is the learning rate of $R_{j,k}(m)$ and α_2 is the learning rate of $MF_n^{output}(i)$. The diagram of weights updating process will be presented below.

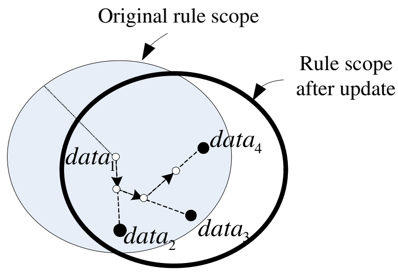

Fig. 4. The diagram of rule scope before/after update

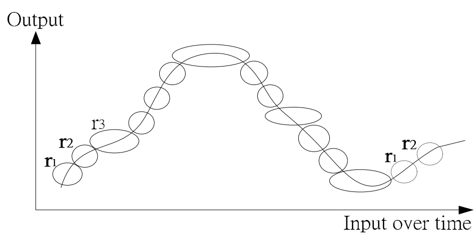

Fig. 5. The Diagram of fuzzy rule generating

3.1.4 Network Generating Step

For each training data, go to *Network Constructing Step* to generate our network which includes the connection weights and fuzzy rules.

After network feed-forward training phase, we can get the fuzzy rules of these training data. And these fuzzy rules will be used to forecast in recall forecast phase. The original concept of fuzzy rule after generating will be similar as Figure 5.

3.2 Recall Forecast Phase

In general, the most similar rule can be retrieved from the rule base for each testing data according to the similarity $D_{i,m}$. In addition, the fuzzy output from this similar rule will be treated as the fuzzy forecast of this testing data. After defuzzification, we can get the forecast demand. However, the concept of K-NN (k-nearest neighbor method, Gordon, 1997) is utilized to find the forecast in this research.

The most similar k rules of each testing data will be considered, the distances between the k-nearest rules and the testing data present the ratios in summarizing the forecasts of k rules. Thus, we can get the combined result from many different rules. To sum up, we expect that we can get more accurate forecasting results.

3.3 Evaluating Indices

The forecasting methods of evaluating indices are presented as follows:

3.3.1 Mean Absolute Percentage Error (MAPE)

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\frac{|F_t - A_t|}{A_t} \tag{10}$$

where F_t is the expected value for period t, A_t is the actual value for period t, n is the total number of periods.

3.3.2 Mean Absolute Deviation (MAD)

$$MAD = \frac{1}{n}\sum_{t=1}^{n}|F_t - A_t| \tag{11}$$

3.3.3 Root of Mean Squared Error (RMSE)

$$RMSE = \sqrt{\sum_{t=1}^{n}(F_t - A_t)^2 \Big/ n} \tag{12}$$

4 Integrated Comparisons

The best parameter combination of WEFuNN was collected from the Taguchi experimental design, and the result is shown as Table 1:

Table 1. Detail parameter setting of WEFuNN

Factors	The best parameters
The no. of fuzzy region in X_1	9
The no. of fuzzy region in X_4	9
Weight of X_1	0.26
Weight of X_2	0.2
Weight of X_4	0.35

In this section, it will focus on the comparisons between WEFuNN and other forecasting models. As shown in Table 2, this study employed MAPE and MAD to evaluate the degree of accuracy. Among these five forecasting models, WEFuNN is the most accurate forecasting model since the value of MAPE and MAD is 2.11% and

16445 square foot respectively. In addition, RMSE is used as an evaluating index for the degree of precise. The value of RMSE is 24909 square foot. The results show that WEFuNN has the best degree of precise among these five forecasting models. Accordingly, it could be concluded that the WEFuNN forecasting model proposed in this study has considerably accurate forecasting ability.

Table 2. The comparison of each forecasting methods

	WEFuNN	EFuNN	BPN	MRA	GANN
MAPE	2.11%	6.44%	8.76%	9.88%	3.06%
MAD	16445	47072	72494	82673	23991
RMSE	24909	59081	114785	131115	33497

According to the experimental results above, it could draw some conclusions:

1. The model taken other domains into consideration rather than just focus on the trend factors and seasonal factors will get a better result no matter in the average or the standard deviation of error. Therefore, to take other domains into account can reduce the error and standard deviation effectively and then increase the accuracy and stability of the model.
2. The forecasting model taken all input factors into account and offered these factors different weight value will derive more accurate results than the forecasting model with the same weight value between each factor.
3. Considering the accuracy and the precise of the forecasting model, the WEFuNN is superior to other forecasting models since it has the minimum value of MAPE and RMSE.
4. As for the cost variance caused by the forecasting errors, WEFuNN has the minimum error cost when compared with other forecasting models.
5. The WEFuNN usually takes one epoch to get the results. It is faster than other forecasting models such as BPN.
6. The forecasting ability of Multiple Regression Analysis is the worst one among the five forecasting models as it is only suitable for the linear question and its forecasting ability for the non-linear question is comparatively poor and weak.

5 Discussions

According to the experimental results shown in the previous section, it could be concluded that the forecasting model proposed in this study is superior to the others. The possible reasons might be as follows:

1. WEFuNN utilized weight which is allocated according to the degree of importance in each factor to calculate the similarity. This approach is much better than EFuNN with the same weight in each factor.
2. The original WEFuNN used Fuzzy Distance Function to calculate the value of similarity. Thus, the value of similarity is 1-D when the distance between the two factors is D. However, in this study, exponential transfer function (exp(-D)) is

employed to transfer the distance to the value of similarity, and the result shows that the outcome is better after using the exponential transfer function.
3. At the recall stage, the method of combining WEFuNN with K-Nearest-Neighbor to get the fuzzy rule is much accurate than the method of WEFuNN which only find out the most similar fuzzy rule. It is because that K-Nearest-Neighbor can modify the accuracy of the forecasting results and relatively adapt to the complicated environmental change.
4. As can been seen from the experimental results, the forecasting results of WEFuNN, EFuNN and GANN are better than that of BPN which employed the Gradient Steepest Descent Method. According to Chang (2005), this result is in line with the previous finding that the performance of the Gradient Steepest Descent Method is not good.

References

1. Abraham, A., Baikunth, N.: A Neuro-Fuzzy Approach for Modeling Electricity Demand in Victoria. Applied Soft Computing. (2001) 127–138
2. Abraham, A., Steinberg, D., Philip, N.S.: Rainfall Forecasting Using Soft Computing Models and Multivariate Adaptive Regression Splines. IEEE SMC Transactions, Special Issue on Fusion of Soft Computing and Hard Computing in Industrial Applications (2001)
3. Abraham, A., Baikunth, N., Mahanti, P.K.: Hybrid Intelligent Systems for Stock Market Analysis. Lecture Notes in Computer Science, Vol. 2074. Springer-Verlag, Berlin Heidelberg New York (2001) 337–345
4. Arifovic, J., Gencay, R.: Using Genetic Algorithms to Select Architecture of a Feedback Artificial Neural Network. Physica A. (2001) 574–594
5. Chang, P.C., Hsieh, J.C.: A Neural Network Approach for Due-Date Assignment in a Wafer Fabrication Factory. Int. J. Industrial Engineering. 10 (2003) 55–61
6. Chang, P.C., Wang, Y.W., Tsai, C.Y.: Evolving Neural Network for Printed Circuit Board Sales. Expert Systems with Applications. 29 (2005) 83–92
7. Chang, P.C., Lai, C.Y.: A Hybrid System Combining Self-Organizing Maps with Case-Based Reasoning in Wholesaler's New-release Book Forecasting. Expert Systems with Applications. 29 (2005) 183–192.
8. Chang, P.C., Wang, Y.W., Liu C.H.: Fuzzy Back-Propagation Network for PCB sales forecasting. Lecture Notes in Computer Science, Vol. 3610. Springer-Verlag, Berlin Heidelberg New York (2005) 364–373
9. Chang, T.C., Lin, S.J.: Grey Relation Analysis of Carbon Dioxide Emissions from Industrial Production and Energy Uses in Taiwan. Journal of Environmental Management. 56 (1999) 247–257
10. Chung, I.F.: Reinforcement Neural Fuzzy Inference Networks and its Applications. PHD thesis, Department of electronic and control engineering, National Chaio-Tung University, Taiwan (1999)
11. Elalfi, A.E., Haque, R., Elalami, M.E.: Extracting Rules from Trained Neural Network Using GA for Managing E-businesss. Applied Soft Computing. 4 (2004) 65–77
12. Kasabov, N.: Evolving Fuzzy Neural Networks for On-line Knowledge Discovery. The Information Science Discussion Paper Series (2001)
13. Kasabov, N.: Evolving Fuzzy Neural Networks-Algorithms, Applications and Biological Motivation. In Proc. of Iizuka'98, Iizuka, Japan (1998)

14. Kasabov, N., Kim, J., Watts, M.J.: Funn/2-A Fuzzy Neural Network Architecture for Adaptive Learning and Knowledge Acquistion. Information Sciences. 101 (1997) 155–175
15. Kim, K.J., Han, I.: Genetic Algorithms Approach to Feature Discretization in Artificial Neural Networks for the Prediction of Stock Price Index. Expert Systems with Application. 19 (2000) 125–132
16. Kuo, R.J., Chen, J.A.: A Decision Support System for Order Selection in Electronic Commerce Based on Fuzzy Neural Network Supported by Real-coded Genetic Algorithm. Expert Systems with Application. 26 (2004) 141–154
17. Liang, R.H.: Application of Grey Relation Analysis to Hydroelectric Generation Scheduling. Int. J. Electrical Power & Energy Systems. 21 (1999) 357–364
18. Lin, C.T., Lee, C.G.: Neural-Network-Based Fuzzy Inference System. IEEE Transactions on Computer. 40 (1991) 1320–1336
19. Liu, Z., Liu, A., Wang, C., Niu, Z.: Evolving Neural Network Using Real Coded Genetic Algorithm for Multispectral Image Classification. Future Generation Computer System. 20 (2004) 1119–1129
20. Montana, D., Davis, L.: Training Feed Forward Neural Networks Using Genetic Algorithms. Proceedings of 11th International Joint Conference on Artificial Intelligence, San Mateo, CA, Morgan Kaufmanns. (1989) 762–767
21. Sexton, R.S., Gupta, J.N.D.: Comparative Evaluation of Genetic Algorithm and Backpropagation for Training Neural Networks. Information Sciences. 129 (2000) 45–59
22. Srinivasan, D.: Evolving Artificial Neural Networks for Short Term Load Forecasting. Neural computing. 23 (1998) 265–276
23. Yao, X.: Evolving Artificial Neural Networks. Proceedings of the IEEE. 87 (1999) 1423–1447