A Genetic Algorithm Based Fuzzy-Tuned Neural Network

S.H. Ling, H.K. Lam, F.H.F. Leung and Y.S. Lee
Centre for Multimedia Signal Processing,
Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

***Abstract*--This paper presents a fuzzy-tuned neural network, which is trained by the genetic algorithm (GA). The fuzzy-tuned neural network consists of a neural-fuzzy network and a modified neural network. In the modified neural network, a novel neuron model with two activation functions is employed. The parameters of the proposed network are tuned by GA with arithmetic crossover and non-uniform mutation. Some application examples are given to illustrate the merits of the proposed network.**

***Index Terms*—Neural Network, Neural Fuzzy Network, Genetic Algorithm.**

I. Introduction

It is well known that a 3-layer feed-forward neural network (NN) can approximate any nonlinear continuous function to an arbitrary accuracy [1]. NNs are widely applied in areas such as prediction [3,8], system modeling and control [1]. Owing to its particular structure, NNs can be used to realize a learning process [9] using some algorithms such as the genetic algorithm (GA) [5, 10] and back propagation [9]. Traditionally, a feed-forward NN [2] has its nodes connected in a layer-to-layer manner.

Neural fuzzy networks (NFNs) [16] have been used to deal with variable linguistic information. By processing fuzzy information, reasoning with respect to a linguistic knowledge base can be realized.

GA is a directed random search technique [10] that is widely applied in optimization problems [9-11]. It can help find the globally optimal solution over a domain [9-11]. GA has been applied in different areas such as fuzzy control [7, 12-13], path planning [14], greenhouse climate control [15], modeling and classification [6] etc.

In this paper, a fuzzy-tuned NN, which consists of a traditional NFN [16] and a modified NN [4], is proposed. For the modified NN, two different activation functions are used in the neurons of the hidden layer. By introducing the proposed neurons, the freedom of the searched domain can be enhanced. Some parameters of the activation functions in the proposed neurons are tuned by the NFN. The values of these parameters are therefore governed by some fuzzy rules. Comparing with the traditional feed-forward NN [2], the proposed fuzzy-tuned NN can give a better performance with a similar number of parameters. In the proposed network, all parameters are trained by the GA with arithmetic crossover and non-uniform mutation [10]. Two application examples will be used to show the merits of the proposed network.

This paper is organized as follows. In section II, the proposed fuzzy-tuned NN is presented. In section III, training of the parameters of the proposed fuzzy-tuned NN using the GA will be discussed. In section IV, some application examples will be given. A conclusion will be drawn in session V.

II. Fuzzy-Tuned Neural Network

The block diagram of the proposed fuzzy-tuned NN is shown in Fig. 1. The proposed fuzzy-tuned NN consists of two parts, namely the modified NN [4] and the NFN [16]. In the modified NN, each neuron in the hidden layer has two activation functions inside: static activation function (SAF) and dynamic activation function (DAF). The parameters of the SAF are fixed. The parameters of the DAF are provided by the NFN.

A. Modified Neural Network

Fig. 2 shows the proposed neuron with two activation functions. The parameters of the SAF are fixed, and its output depends on the inputs of the neuron. For the DAF, the parameters depend on the outputs of the NFN. With this proposed neuron, the connection of the modified NN is shown in Fig. 3, which is a 3-layer NN. $\mathbf{p}^U = \left[p_1^U,\ p_2^U,\ \ldots\ p_{n_h}^U\right]$ and $\mathbf{p}^L = \left[p_1^L,\ p_2^L,\ \ldots\ p_{n_h}^L\right]$ are obtained by the NFN, where n_h denotes the number of hidden nodes.

Proposed neuron model

For the SAF, let v_{ik} be the synaptic connection weight from the i-th input node z_i to the k-th neuron. The output κ_k of the k-th neuron's SAF is defined as,

$$\kappa_k = net_s^k\left(\sum_{i=1}^{n_{in}} z_i v_{ik}\right),\ i = 1, 2, \ldots, n_{in};\ k = 1, 2, \ldots, n_h \qquad (1)$$

where n_{in} denotes the number of inputs, and $net_s^k(\cdot)$ is a static activation function defined as:

$$net_s^k\left(\sum_{i=1}^{n_{in}} z_i v_{ik}\right) = \begin{cases} e^{\frac{-\left(\sum_{i=1}^{n_{in}} z_i v_{ik} - m_s^k\right)^2}{2\sigma_s^{k\,2}}} - 1 & \text{if } \sum_{i=1}^{n_{in}} z_i v_{ik} \le m_s^k \\ 1 - e^{\frac{-\left(\sum_{i=1}^{n_{in}} z_i v_{ik} - m_s^k\right)^2}{2\sigma_s^{k\,2}}} & \text{otherwise} \end{cases} \qquad (2)$$

where m_s^k and σ_s^k are the static mean and static standard deviation for the k-th SAF respectively. They are fixed after

the training process. By using the activation function of (2), the output value is ranged from −1 to 1. The shapes of the SAFs are shown in Fig. 4. The static mean is used to control the bias as shown in Fig. 4a and the static standard deviation influences the sharpness as shown in Fig. 4b.

For the DAF, the neuron output ζ_k of the k-th neuron is defined as,

$$\zeta_k = net_d^k(\kappa_k, p_k^U, p_k^L),\ k = 1, 2, \ldots, n_h \tag{3}$$

and,

$$net_d^k(\kappa_k, p_k^U, p_k^L) = \begin{cases} e^{\frac{-(\kappa_k - p_k^U)^2}{2p_k^{L2}}} - 1 & \text{if } \kappa_k \le p_k^U \\ 1 - e^{\frac{-(\kappa_k - p_k^U)^2}{2p_k^{L2}}} & \text{otherwise} \end{cases} \tag{4}$$

p_k^U and p_k^L are the parameters of the DAF, which are effectively the dynamic mean and the dynamic standard deviation respectively of the k-th DAF. From (1) to (4), the input-output relationship of the proposed neuron is given by,

$$\zeta_k = net_d^k(net_s^k(\sum_{i=1}^{n_{in}} z_i v_{ik})) \tag{5}$$

Connection of the modified neural network

The modified NN has n_{in} nodes in the input layer, n_h nodes in the hidden layer, and n_{out} nodes in the output layer. In the hidden layer, the proposed neuron model is employed. In the output layer, a static activation function is used. Considering an input-output pair $(\mathbf{z}, \mathbf{y})$ of the NN, the output of the k-th node of the hidden layer is given by (5). The output of the modified NN (Fig. 5) is given by,

$$y_l = net_o^l(\sum_{k=1}^{n_h} \zeta_k w_{kl}) \tag{6}$$

$$= net_o^l(\sum_{k=1}^{n_h} net_d^k(net_s^k(\sum_{i=1}^{n_{in}} z_i v_{ik}))w_{kl}) \tag{7}$$

where w_{kl}, $k = 1, 2, \ldots, n_h$; $l = 1, 2, \ldots n_{out}$ is the weight of the link between the k-th hidden and the l-th output nodes; $net_o^l(\cdot)$ denotes the activation function of the output neuron:

$$net_o^l(\sum_{k=1}^{n_h} \zeta_k w_{kl}) = \begin{cases} e^{\frac{-\left(\sum_{k=1}^{n_h} \zeta_k w_{kl} - m_o^l\right)^2}{2\sigma_o^{l2}}} - 1 & \text{if } \sum_{k=1}^{n_h} \zeta_k w_{kl} \le m_o^l \\ 1 - e^{\frac{-\left(\sum_{k=1}^{n_h} \zeta_k w_{kl} - m_o^l\right)^2}{2\sigma_o^{l2}}} & \text{otherwise} \end{cases} \tag{8}$$

where m_o^l and σ_o^l are the mean and the standard deviation of the output node activation function respectively.

From Fig. 3, the first layer simply distributes the input variables. In the hidden layer, the SAF determines hyper planes as switching surfaces. Owing to the DAF, the input $\mathbf{p}^U$ concerns the bias term while the input $\mathbf{p}^L$ influences the sharpness of the edges of these hyper-planes. They eventually combine into convex regions by the output layer. Some parameters of the modified NN can be trained by the GA and some other parameters are provided by the NFN.

B. Neural-Fuzzy Network

In the modified NN, the parameters of $\mathbf{p}^U$ and $\mathbf{p}^L$ of the DAFs are determined by the NFN as shown in Fig. 6. The input and output variables of the NFN are z_i and p_j respectively; where $i = 1, 2, \ldots, n_{in}$, $j = 1, 2, \ldots, q$, and $q=2n_h$ is the number of output variables. The behavior of the NFN is governed by m fuzzy rules of the following format:

R_h: IF $z_1(t)$ is A_1^h AND $z_2(t)$ is A_2^h AND ... AND $z_{n_{in}}(t)$ is $A_{n_{in}}^h$ THEN $p_j(t)$ is g_{hj}, $t = 1, 2, \ldots, u$ (9)

where u is the number of input-output data pairs; $h = 1, 2, \ldots, m$, is the rule number; g_{hj} is an output singleton in rule h. In this network, the membership function is a bell-shaped function given by,

$$\mu_{A_i^h}(z_i(t)) = e^{\frac{-(z_i(t) - \bar{z}_i^h)^2}{2\sigma_i^{h2}}} \tag{10}$$

where the parameter $\bar{z}_i^h$ and σ_i^h are the mean value and the standard deviation respectively of the membership function $\mu_{A_i^h}$. The grade of membership of each rule is defined as,

$$\mu_h(t) = \prod_{i=1}^{n_{in}} \mu_{A_i^h}(z_i(t)) \tag{11}$$

The output of the NFN, $p_j(t)$, is defined as,

$$p_j(t) = \frac{\sum_{h=1}^{m} \mu_h(t) g_{hj}}{\sum_{h=1}^{m} \mu_h(t)} \tag{12}$$

which is a parameters of the DAF. The number of outputs of the NFN doubles the number of hidden nodes of the modified NN. Referring to Fig. 3 and Fig. 6, $p_1 = p_1^U$, $p_2 = p_1^L$, ..., $p_{q-1} = p_{n_h}^U$, $p_q = p_{n_h}^L$. The weights of the NFN are tuned by the GA.

III. Training of Parameters

In this section, the proposed fuzzy-tuned NN is trained to learn the input-output relationship of an application. GA with arithmetic crossover and non-uniform mutation [10] is used to realize the training. A population of chromosomes P is initialized and then evolves. First, two parents are selected from P by the method of spinning the roulette wheel [10]. Then a new offspring is generated from these parents using the crossover and mutation operations, which are governed by the probabilities of crossover and mutation respectively. These probabilities are chosen by trial and error through experiments for good performance. The new population thus generated replaces the current population. These procedures are repeated until a certain termination condition is satisfied, e.g. a predefined number of generations has been reached.

Based on Fig. 1, let the input-output relationship of an application be given by,

$$\mathbf{y}^d(t) = \mathbf{g}\left(\mathbf{z}^d(t)\right),\ t = 1, 2, \ldots, n_d \tag{13}$$

where $\mathbf{z}^d(t) = \left[z_1^d(t) \quad z_2^d(t) \quad \cdots \quad z_{n_{in}}^d(t)\right]$ and $\mathbf{y}^d(t) = \left[y_1^d(t) \quad y_2^d(t) \quad \cdots \quad y_{n_{out}}^d(t)\right]$ are the given inputs and the desired outputs of an unknown nonlinear function $\mathbf{g}(\cdot)$ respectively. n_d denotes the number of input-output data pairs. The fitness function is defined as,

$$fitness = \frac{1}{1+err} \tag{14}$$

$$err = \frac{\sum_{t=1}^{n_d}\sum_{k=1}^{n_{out}}\left|y_k^d(t) - y_k(t)\right|}{n_d n_{out}} \tag{15}$$

The objective is to maximize the fitness value of (14) (minimize *err* of (15)) using GA by setting the chromosome to be $\left[v_{ik} \quad m_s^k \quad \sigma_s^k \quad w_{kl} \quad m_o^l \quad \sigma_o^l \quad \bar{z}_i \quad \sigma_i \quad g_{hj}\right]$ for all h, i, j, k, l. In this paper, $v_{ik}, w_{kl}, g_{hj} \in [-1 \quad 1]$, $m_s^k, m_o^l, \bar{z}_i \in [-0.5 \quad 0.5]$ and $\sigma_s^k, \sigma_o^l, \sigma_i \in [0.01 \quad 0.5]$. The range of the fitness value of (14) is [0, 1].

VI. Application Examples

Two application examples will be given in this section to illustrate the merits of the proposed fuzzy-tuned NNs.

A. Forecasting the Sunspot Number

The sunspot numbers [3, 8] from 1700 to 1980 are shown in Fig. 7. The cycles generated are non-linear, non-stationary, and non-Gaussian which are difficult to model and predict. We use the proposed fuzzy-tuned NN for the sunspot number forecasting. The inputs, z_i, of the proposed network are $z_1(t) = y^d(t-1)$, $z_2(t) = y^d(t-2)$ and $z_3(t) = y^d(t-3)$, where t denotes the year and $y^d(t)$ is the sunspot number at the year t. The sunspot numbers of the first 180 years (i.e. $1705 \le t \le 1884$) are used to train the proposed NN. The number of hidden nodes (n_h) is set at 3 and the number of membership functions (h) in the NFN is set at 2 (the total number of parameter is 44). Referring to (7), the proposed network used for the sunspot forecasting is governed by,

$$y(t) = net_o\left(\sum_{k=1}^{3} net_d^k\left(net_s^k\left(\sum_{i=1}^{3} z_i v_{ik}\right)\right) w_k\right) \tag{16}$$

GA is used to tune the parameters of the proposed fuzzy-tuned NN of (16). The fitness function is defined as follows,

$$fitness = \frac{1}{1+err} \tag{17}$$

$$err = \sum_{t=1705}^{1884} \frac{\left|y^d(t) - y(t)\right|}{180} \tag{18}$$

The objective is to maximize the fitness function of (17). The population size used for the GA is 10. The chromosomes are $\left[v_{ik} \quad m_s^k \quad \sigma_s^k \quad w_k \quad m_o \quad \sigma_o \quad \bar{z}_i \quad \sigma_i \quad g_{hj}\right]$ for all h, i, j, k. The initial values of the parameters are randomly generated. For comparison purpose, a traditional feed-forward NN trained by the GA is also applied in forecasting the sunspot number. The number of hidden node of the traditional NN is 9 (total number of parameter is 46). For both approaches, the number of iteration to train the NN is 1000. For the GA, the probability of crossover is set at 0.8. The probability of mutation is set at 0.2 and 0.25 for the proposed and the traditional approaches respectively. The networks are used to forecast the sunspot number during the years 1885-1979. Fig. 8 shows the simulation results of the forecasting using the proposed network (dashed lines) and the traditional network (dotted lines), as compared with the actual sunspot numbers (solid lines). The fitness value, the training error (governed by (18)) and the forecasting error (governed by $\sum_{t=1885}^{1980} \frac{\left|y_1^d(t) - y_1(t)\right|}{96}$) are tabulated in Table I. It can be observed from Table I that our approach performs better than the traditional approach. The training error and the forecasting error of the proposed approach in terms of mean absolute error (MAE) are 10.15 and 13.03 respectively.

B. Pattern Recognition

In this section, an application on hand-written graffiti pattern recognition will be presented. Numbers are assigned to pixels on a two-dimensional plane, and 10 numbers are used to characterize the 10 uniformly sampled pixels of a given graffiti. A 10-input-3-output network is used. The ten inputs nodes, z_i, i = 1, 2, ..., 10, are the numbers representing the graffiti pattern. Three standard patterns are to be recognized: rectangle, triangle and straight line (Fig. 9). We use 300 sets of 10 normalized samples points for each pattern to train the NN. Hence, we have 900 sets of data for training. The three outputs, $y_l(t)$, l = 1, 2, 3, indicates the similarity between the input pattern and the three standard patterns respectively. The desired outputs of the pattern recognition system are $\mathbf{y}(t) = [1 \quad 0 \quad 0]$, $\mathbf{y}(t) = [0 \quad 1 \quad 0]$ and $\mathbf{y}(t) = [0 \quad 0 \quad 1]$ for rectangles, triangles and straight lines respectively. After training, a larger value of $y_l(t)$ implies that the input pattern matches more closely to the corresponding graffiti pattern. For instance, a large value of $y_1(t)$ implies that the input pattern is near to a rectangle. In the proposed network, the number of hidden nodes (n_h) is set at 6 and the number of membership functions (h) is set at 4 in the NFN (the total number of parameter is 224), which are chosen by trial and error through experiments for good performance. Referring to (7), the proposed network used for the pattern recognition is governed by,

$$y_l(t) = net_o^l\left(\sum_{k=1}^{3} net_d^k\left(net_s^k\left(\sum_{i=1}^{10} z_i v_{ik}\right)\right) w_{kl}\right), \quad l = 1, 2, 3. \tag{19}$$

GA is employed to tune the parameters of the proposed network of (19). The fitness function is defined as follows,

$$fitness = \frac{1}{1+err} \tag{20}$$

$$err = \frac{\sum_{t=1}^{300}\sum_{k=1}^{3}\left(\left(\frac{y_k(t)}{\|\mathbf{y}(t)\|}\right)^2 - \left(\frac{y_k^d(t)}{\|\mathbf{y}^d(t)\|}\right)^2\right)}{300\times 3} \tag{21}$$

(21) indicates the mean square error (MSE) of the recognition. The population size used for the GA is 10. The chromosomes are $\left[v_{ik} \quad m_s^k \quad \sigma_s^k \quad w_{kl} \quad m_o^l \quad \sigma_o^l \quad \bar{z}_i \quad \sigma_i \quad g_{hj}\right]$ for all h, i, j, k, l. The initial values of the parameters of the NN are randomly generated. For comparison, a traditional NN trained by the GA is used to recognize the patterns. The number of hidden node of the traditional NN is 16 (the total number of parameter is 227). For both approaches, the number of iteration to train the NN is 2000. For the GA, the probability of crossover is set at 0.8 for both approaches. The probability of mutation is set at 0.05 and 0.06 for the proposed and traditional approaches respectively.

After training, we use 600 (200×3) sets of data for testing. The results are tabulate in Table II. From this Table, it can be seen that the training error (governed by (21)) and forecasting error (governed by $err = \frac{\sum_{t=1}^{200}\sum_{k=1}^{3}\left(\left(\frac{y_k(t)}{\|\mathbf{y}(t)\|}\right)^2 - \left(\frac{y_k^d(t)}{\|\mathbf{y}^d(t)\|}\right)^2\right)}{200\times 3}$) of the proposed network are smaller. The recognition accuracy is governed by

$$\text{Accuracy} = \left(\frac{n_{recognized}}{n_{testing}}\right)\times 100\%. \tag{22}$$

where $n_{testing}$ and $n_{recognized}$ are the total number of testing patterns and the number of successfully recognized patterns respectively.

V. Conclusion

A fuzzy-tuned NN has been proposed. The fuzzy-tuned NN consists of a modified NN and a neural-fuzzy network. A modified neuron model with two activation functions has been introduced in the hidden layer neurons of the modified NN. Some parameters of the proposed neuron are tuned by the NFN. By employing this neuron model and the network structure, the proposed network is found to perform better than the traditional NN. The parameters of the proposed network can be tuned by GA. Examples of sunspot number forecasting and pattern recognition have been given.

Acknowledgement

The work described in this paper was substantially supported by a grant from the Research Grants Council of the Hong Kong Special Administration Region, China (Project Nos. PolyU 5127/00E).

References

[1] M. Brown and C. Harris, *Neuralfuzzy adaptive modeling and control.* Prentice Hall, 1994.

[2] A.E. Bryon and Y.C. Ho, *Applied optimal control.* Blaisdell, 1969.

[3] F.H.F. Leung, H.K. Lam, S.H. Ling, and P.K.S. Tam, "Tuning of the structure and parameters of neural network using an improved genetic algorithm," *IEEE Trans. Neural Networks*, vol. 14, no. 1, 00. 79-88, Jan. 2003.

[4] S.H. Ling, F.H.F. Leung, H.K. Lam, Y.S. Lee and P.K.S. Tam, "A novel GA-based neural network for short-term load forecasting," *IEEE Trans. Industrial Electronics*, (to be published)

[5] S.H. Ling, H.K. Lam, F.H.F. Leung and P.K.S. Tam, "Parameter learning of neural network using fuzzy genetic algorithm," in *Proc. CEC2002, WCCI*, Honolulu, Hawaii, 12-17, May 2002, pp. 1928-1933.

[6] H.K. Lam, S.H. Ling, K.F. Leung, F.H.F. Leung and P.K.S. Tam, "On interpreting graffiti commands for eBooks using a neural network and an improved genetic algorithm," in *Proc. 10th IEEE Int. Conf. Fuzzy Systems*, Australia, Dec. 2001, pp. 1464-1467.

[7] H.K. Lam, S.H. Ling, F.H.F. Leung, and P.K.S. Tam, "Optimal and stable fuzzy controllers for nonlinear systems subject to parameter uncertainties using genetic algorithm," in *Proc. 10th IEEE Int. Conf. Fuzzy Systems*, Australia, Dec. 2001, pp. 908-911.

[8] M. Li, K. Mechrotra, C. Mohan, and S. Ranka, "Sunspot numbers forecasting using neural network," in *Proc. 5th IEEE Int. Symp. Intelligent Control, 1990*, 1990, pp.524-528.

[9] D.T. Pham and D. Karaboga, *Intelligent optimization techniques, genetic algorithms, tabu search, simulated annealing and neural networks*. Springer, 2000.

[10] J.H. Holland, *Adaptation in natural and artificial systems*. University of Michigan Press, Ann Arbor, MI, 1975.

[11] Z. Michalewicz, *Genetic Algorithm + Data Structures = Evolution Programs*, 2nd Ed. Springer-Verlag, 1994.

[12] B.D. Liu, C.Y. Chen, and J.Y. Tsao, "Design of adaptive fuzzy logic controller based on linguistic-hedge concepts and genetic algorithms," *IEEE Trans. Systems, Man and Cybernetics, Part B*, vol. 31 no. 1, pp. 32-53, Feb. 2001.

[13] K. Belarbi and F. Titel, "Genetic algorithm for the design of a class of fuzzy controllers: an alternative approach," *IEEE Trans. Fuzzy Systems*, vol. 8, no. 4, pp. 398-405, Aug. 2000.

[14] H. Juidette and H. Youlal, "Fuzzy dynamic path planning using genetic algorithms," *Electronics Letters*, vol. 36, no. 4, pp. 374-376, Feb. 2000.

[15] R. Caponetto, L. Fortuna, G. Nunnari, L. Occhipinti, and M. G. Xibilia, "Soft computing for greenhouse climate control," *IEEE Trans. Fuzzy Systems*, vol. 8, no. 6, pp. 753-760, Dec. 2000.

[16] M. Brown and C. Harris, *Neuralfuzzy Adaptive Modeling and Control.* Prentice Hall, 1994.

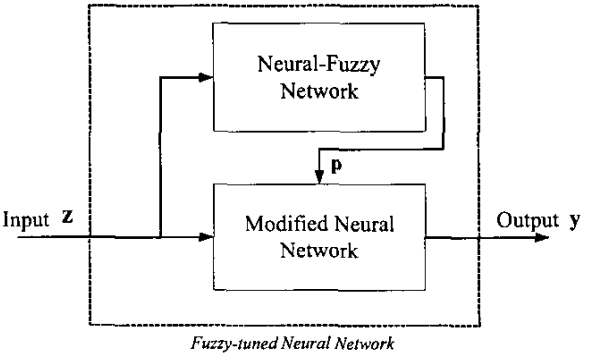


Fig. 1. Block diagram of the proposed fuzzy-tuned neural network.

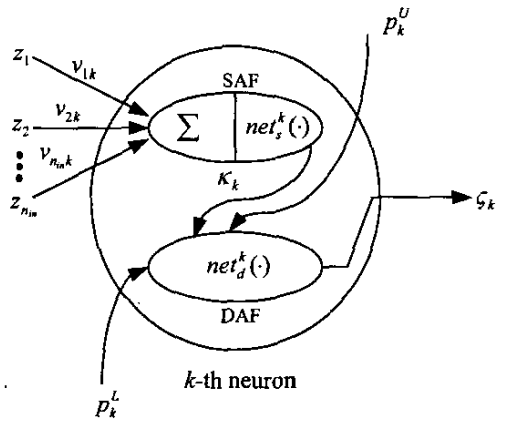


Fig. 2. Proposed neuron.

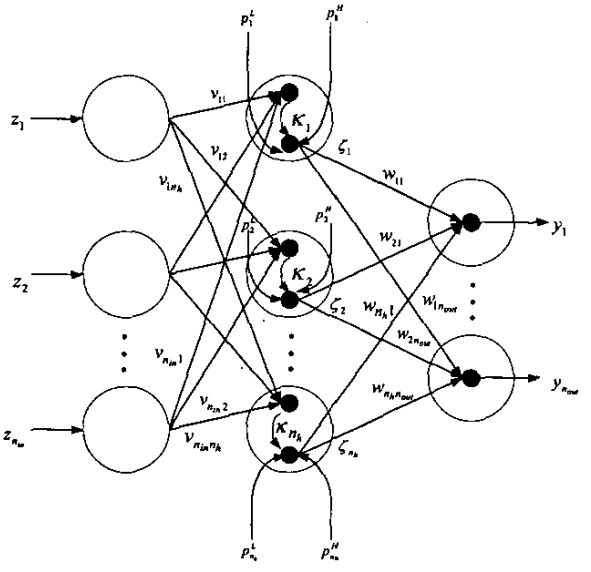
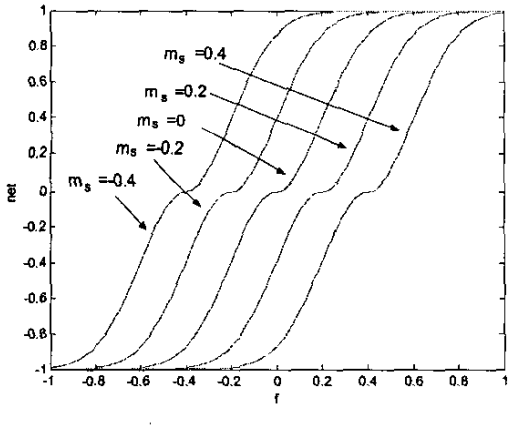
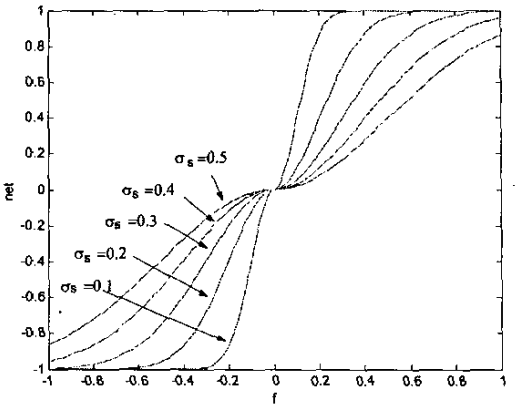


Fig. 3. Connection of the modified neural network.



(a)



(b)

Fig. 4. Sample activation functions of the proposed neuron: (a) $\sigma_s = 0.2$, (b) $m_s = 0$.

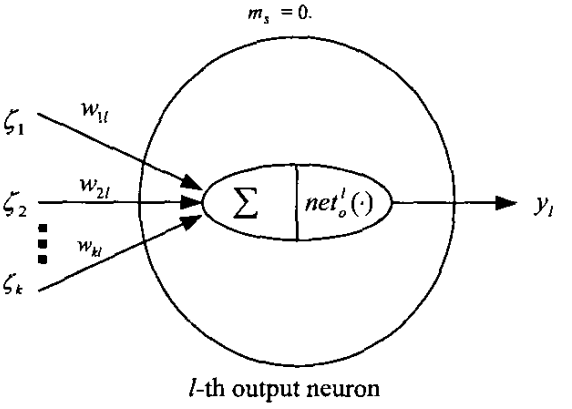


Fig. 5. Model of the proposed output neuron.

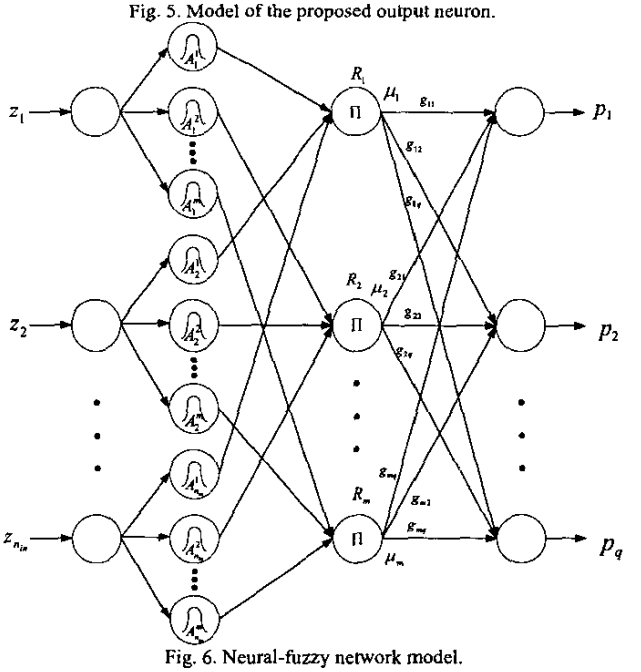


Fig. 6. Neural-fuzzy network model.

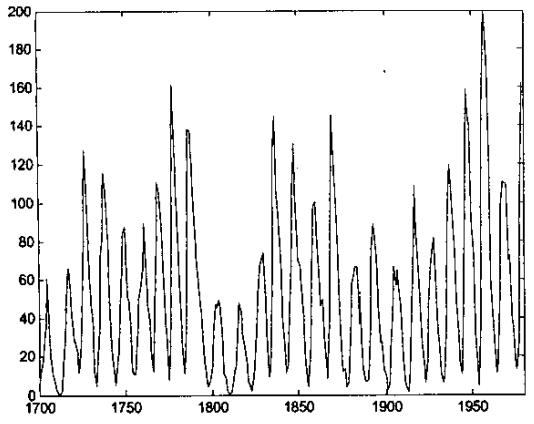


Fig. 7. Sunspot numbers from year 1700 to 1980.

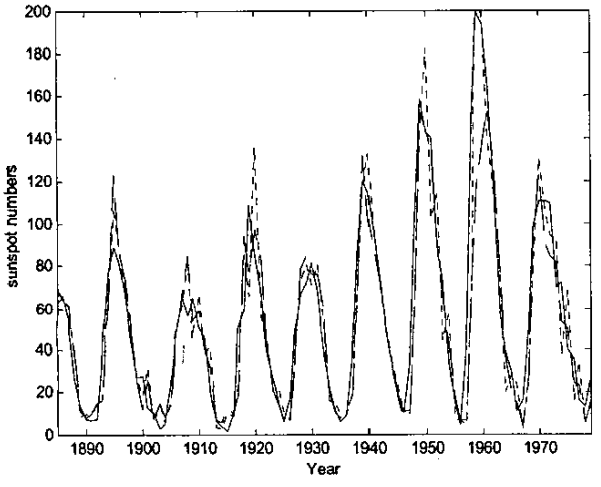
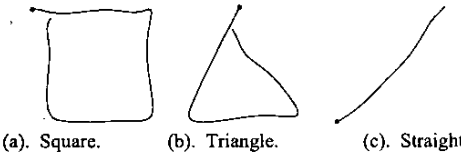


Fig. 8. Simulation results of 95-year prediction using the proposed network (dashed line) and the traditional network (dotted line), as compared with actual sunspots numbers (solid line) for the year 1885-1979.



(a). Square. (b). Triangle. (c). Straight line.

Fig. 9. Samples of the graffiti patterns.

	Proposed Network	Traditional Network
Fitness value	0.9517	0.9432
No. of parameters	44	46
Training error (MAE)	10.15	12.04
Forecasting error (MAE)	13.03	13.93

Table. I. Simulation results of the proposed and traditional neural networks trained by GA for the forecasting of sunspot number.

	Proposed Network	Traditional Network
Fitness value	0.9837	0.9717
No. of parameters	224	227
Training error (MSE)	0.0166	0.0291
Forecasting error (MSE)	0.0211	0.0411
Recognition accuracy	96.17%	93.33%

Table II. Results of the proposed neural network and the traditional neural network for hand-written graffiti pattern recognition.