A Novel GA-Based Neural Network for Short-Term Load Forecasting

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

Abstract – This paper presents a GA-based neural network with a novel neuron model. In this model, the neuron has two activation transfer functions and exhibits a node-by-node relationship in the hidden layer. This neural network provides a better performance than a traditional feed-forward neural network and fewer hidden nodes are needed. The parameters of the proposed neural network are tuned by GA with arithmetic crossover and non-uniform mutation. An application on short-term load forecasting is given to show the merits of the proposed neural network.

I. Introduction

Neural network was proved to be a universal approximator [18]. A 3-layer feed-forward neural network can approximate any nonlinear continuous function to an arbitrary accuracy. Neural networks are widely applied in areas such as prediction [19], system modeling and control [18]. Owing to its particular structure, a neural network is good at learning [2] using some learning algorithms such as Genetic Algorithm (GA) [1] and back propagation [2]. In general, the processing of a traditional feed-forward neural network is done in a layer-by-layer manner.

GA is a directed random search technique which is widely applied in optimization problems [1-7] where the number of parameters is large and the analytical solutions are difficult to obtain. GA can help finding the optimal solution over a domain globally [1-3]. It has been applied in different areas such as fuzzy control [20-23], path planning [24], greenhouse climate control [25], modeling and classification [26] etc..

A novel neural network model is proposed in this paper. Two different activation transfer functions are used in the neuron and a node-by-node relationship is proposed in the hidden layer. This network model is found to be able to give a better performance with a small number of hidden nodes (fewer parameters) than the traditional feed-forward neural network [1-3]. GA with arithmetic crossover and non-uniform mutation can help tuning the parameter of the proposed network. A numerical application example (3-inputs XOR problem) is used to test the proposed network and good results are obtained. The proposed network is then used to forecast the short-term daily load in an intelligent home. Simulation results show that the proposed network can forecast the short-term daily load successfully.

This paper is organized as follows. The proposed neural network is presented in section II. Training of the neural network with GA is presented in section III. A numerical example will be presented in section IV. A short-term daily load forecasting realized by the GA-based neural network will be presented in section V. A conclusion is drawn in section VI.

II. Neural Network Model

In this section, a novel neuron model and its application to a neural network are presented. Fig.1 shows the proposed neuron. It has two activation transfer functions to govern the input-output relationships of the neuron. The two activation transfer functions are called static activation transfer function (SATF) and dynamic activation transfer function (DATF). For the SATF, the parameters are fixed and its output depends on the input of the neuron. For the DATF, the parameters of the activation transfer function depend on the outputs of other neurons and its SATF. With this proposed neuron, the connection of the proposed neural network is shown in Fig. 2, which is a three-layer neural network. A node-by-node relationship is introduced in the hidden layer. It should be noted that the parameters of the SATF are constant. The parameters of the DATF are obtained from other two neuron outputs, which are variable. Comparing with the traditional feed-forward neural network [27-28], the proposed neural network can offer a better performance, and fewer hidden nodes are needed. The merits of the proposed neural network will be shown in the later section.

A. The neuron model

The proposed neuron model consists two activation transfer functions (SATF and DATF). We consider the SATF first. Let v_{ij} be the synaptic connection weight from the i-th input component x_i to the j-th neuron. The output κ_j of the j-th neuron's SATF is defined as,

$$\kappa_j = net_s^j\left(\sum_{i=1}^{n_{in}} x_i v_{ij}\right),\ i = 1, 2, \ldots, n_{in},\ j = 1, 2, \ldots, n_h \qquad (1)$$

where n_{in} denotes the number of input and $net_s^j(\cdot)$ is a static activation transfer function. The activation transfer function is defined as,

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

$$j = 1, 2, \ldots, n_h \qquad (2)$$

where m_s^j and σ_s^j are the static mean and static standard deviation for the j-th SATF respectively. The parameters m_s^j and σ_s^j are fixed after the training processing. Thus, the activation transfer function is static. The output of the SATF depends on the inputs of the neuron only. By using the proposed activation function in (2), the output value is ranged

0-7803-7278-6/02/$10.00 ©2002 IEEE

from −1 to 1. The shape of the proposed activation transfer function is shown in Fig. 3 and 4. Referring to Fig. 3, the effect of the mean value m to the activation transfer function is shown. The standard deviation σ of the function is fixed at 0.2 and the mean value m is chosen from −0.4 to 0.4. Referring to Fig. 4, the effect of the standard deviation σ to the activation transfer function is shown. The mean value m is fixed at 0. The standard deviation σ is chosen from 0.1 to 0.5. It can be observed from these 2 figures that $net(f) \to 1$ as $f \to \infty$ and $net(f) \to -1$ as $f \to -\infty$. Considering the DATF, the neuron output z_j of the j-th neuron is defined as,

$$z_j = net_d^j(\kappa_j),\ j = 1, 2, \ldots, n_h \tag{3}$$

where $net_d^j(\cdot)$ is the activation transfer function of the DATF. The activation transfer function is defined as follows,

$$net_d^j(\kappa_j) = \begin{cases} e^{\frac{-(\kappa_j - m_d^j)^2}{2\sigma_d^{j\,2}}} - 1 & \text{if } \kappa_j \le m_d^j,\ j = 1, 2, \ldots, n_h \\ 1 - e^{\frac{(\kappa_j - m_d^j)^2}{2\sigma_d^{j\,2}}} & \text{otherwise} \end{cases} \tag{4}$$

where

$$m_d^j = p_{j+1,j} \times z_{j+1} \tag{5}$$

$$\sigma_d^j = p_{j-1,j} \times z_{j-1} \tag{6}$$

m_d^j and σ_d^j are the dynamic mean and dynamic standard deviation for the j-th DATF. z_{j-1} and z_{j+1} represent the output of the $j-1$-th and j+1-th neurons respectively. $p_{j+1,j}$ denotes the weight of the link between the j+1-th node and the j-th node and $p_{j-1,j}$ denotes the weight of the link between the $j-1$-th node and the j-th node. It should be noted from Fig. 1 that if $j=1$, $p_{j-1,j}$ is equal to $p_{n_h,j}$, and if $j=n_h$, $p_{j+1,j}$ is equal to $p_{1,j}$. Unlike the SATF, the DATF is dynamic as the parameters of its activation transfer function depend on the outputs of the $j-1$-th and j+1-th neurons. Referring to (1) to (4), the input-output relationship of the proposed neuron is as follows,

$$z_j = net_d^j(net_s^j(\sum_{i=1}^{n_{in}} x_i v_{ij})),\ j = 1, 2, \ldots, n_h \tag{7}$$

B. Connection of the proposed neural network

The proposed neural network is presented in this section. We shall discuss the multi-input multi-output (MIMO) neural network as shown in Fig. 2. The proposed neural network has three layers with 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 neuron model presented in the previous section is employed. A node-by-node relationship is introduced in the hidden layer. The output value of a hidden node depends on its neighboring nodes and input nodes. In the output layer, a static activation transfer function is employed. Considering an input-output pair $(\mathbf{x},\mathbf{y})$, the output of the j-th node of the hidden layer is given by

$$z_j = net_d^j(net_s^j(\sum_{i=1}^{n_{in}} x_i v_{ij})),\ j = 1, 2, \ldots, n_h \tag{8}$$

The output of the proposed neural network is given by,

$$y_l = net_o^l(\sum_{l=1}^{n_{out}} z_j w_{jl}),\ l = 1, 2, \ldots, n_{out} \tag{9}$$

$$= net_o^l(\sum_{l=1}^{n_{out}} net_d^j(net_s^j(\sum_{i=1}^{n_{in}} x_i v_{ij}))w_{jl}) \tag{10}$$

where w_{jl}, $j = 1, 2, \ldots, n_h$; $l = 1,2, \ldots n_{out}$, denotes the weight of the link between the j-th hidden and the l-th output nodes; $net_o^l(\cdot)$ denotes the activation transfer function of the output neuron. The activation transfer function of the output node is defined as follows,

$$net_o^l(z_j) = \begin{cases} e^{\frac{-(z_k - m_o^l)^2}{2\sigma_o^{l\,2}}} - 1 & \text{if } z_k \le m_o^l \\ 1 - e^{\frac{-(z_k - m_o^l)^2}{2\sigma_o^{l\,2}}} & \text{otherwise} \end{cases} \tag{11}$$

where m_o^l and σ_o^l are the mean and the standard deviation of the output node activation transfer function respectively. The parameters of the proposed neural network can be trained by GA. The details about the training processing will be given in the next section.

III. Training With Genetic Algorithm

In this section, the proposed neural network is employed to learn the input-output relationship of an application using GA with arithmetic crossover and non-uniform mutation [3]. The input-output relationship is described by,

$$\mathbf{y}^d(t) = \mathbf{g}\left(\mathbf{x}^d(t)\right),\ t = 1, 2, \ldots, n_d \tag{12}$$

where $\mathbf{y}^d(t) = \left[y_1^d(t)\ \ y_2^d(t)\ \cdots\ y_{n_{out}}^d(t)\right]$ is the desired output corresponding to the input $\mathbf{x}^d(t) = \left[x_1^d(t)\ \ x_2^d(t)\ \cdots\ x_{n_{in}}^d(t)\right]$ of an unknown nonlinear function $\mathbf{g}(\cdot)$. n_d denotes the number of input-output data pairs. The fitness function is defined as,

$$fitness = \frac{1}{1 + err} \tag{13}$$

$$err = \sum_{k=1}^{n_{out}} \frac{\sum_{t=1}^{n_d} \left| y_k^d(t) - y_k(t) \right|}{n_d} \tag{14}$$

The objective is to maximize the fitness value of (13) using GA by setting the chromosome to be $[v_{ij}\ \ m_s^j\ \ \sigma_s^j\ \ p_{j+1,j}\ \ p_{j-1,j}\ \ w_{jl}\ \ m_o^l\ \ \sigma_o^l]$ for all i, j, l. The range of fitness function in (13) is [0, 1]. A larger value of *fitness* indicates a smaller *err*. By using the proposed GA-based neural network, an optimal neural network can be obtained.

IV. Numerical Simulation

An example of a three-input XOR function, which is not linearly separable, will be presented:

$$
\begin{aligned}
&(-1,-1,-1) \rightarrow -1 \\
&(-1,-1,+1) \rightarrow +1 \\
&(-1,+1,-1) \rightarrow +1 \\
&(-1,+1,+1) \rightarrow -1 \\
&(+1,-1,-1) \rightarrow +1 \\
&(+1,-1,+1) \rightarrow -1 \\
&(+1,+1,-1) \rightarrow -1 \\
&(+1,+1,+1) \rightarrow +1
\end{aligned} \tag{15}
$$

The three inputs of the proposed neural network are defined as $x_i(t)$, i = 1, 2, 3 and $y(t)$ is the network output. The number of hidden nodes (n_h) is set at 3. Referring to (10), the proposed neural network used for the three-inputs XOR classification problem is governed by,

$$y_1(t) = net_o^l\left(\sum_{j=1}^{3} net_d^j\left(net_s^j\left(\sum_{i=1}^{3} x_i v_{ij}\right)\right) w_{j1}\right) \tag{16}$$

The fitness function is defined as follows,

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

$$err = \frac{\sum_{t=1}^{8} \left| y_1^d(t) - y_1(t) \right|}{8} \tag{18}$$

GA is employed to tune the parameters of the proposed neural network of (16). The objective is to maximize the fitness function of (17). The best fitness value is 1 and the worst one is 0. The population size used for the GA is 20. The chromosomes used for the GA are $[v_{ij} \quad m_s^j \quad \sigma_s^j \quad p_{j+1,j} \quad p_{j-1,j} \quad w_{jl} \quad m_o^l \quad \sigma_o^l]$. The initial values of the parameters of the neural network are randomly generated. For comparison purpose, a traditional feed-forward neural network (3-input-single-output) [27-28] trained by GA is also used to solve the three-input XOR classification problem. The number of hidden node of the traditional neural network is 5. By using 3 hidden nodes for the proposed neural network and 5 hidden nodes for the traditional neural network, the number of parameters to be tuned is the same, which is 26. The number of iterations for training the neural network is 1000 for both approaches. The simulation results of the proposed and traditional neural networks are tabulated in Table II and shown in Fig. 5. It can be seen from Table I and Fig. 5 that the proposed neural network performs better than the traditional one.

V. Short-Term Daily Load Forecasting

Modern homes should have smart features to ensure a higher degree of home security and comfort. To realize these features, reliable channels for the communication among electrical appliances and users should be present. Appliances should be used in an efficient way to reduce the wastage of energy. Based on a power line data network in an intelligent home system [8], the AC power line network is used not only for supplying electrical power, but also serving as the data communication channel for electrical appliances. With this AC power line data network, a short-term load forecasting can be realized. An accurate load forecasting can bring the following benefits to the intelligent home: 1) increasing the reliability [4] of the AC power line data network, and 2) optimal load scheduling.

Artificial neural network have been considered as a very promising tool to short-term load forecasting [9-17]. However, the gradient-descent (GD) algorithm for parameters training of feed-forward neural networks suffers from the common problems of convergence to local minima and sensitivity to initial values of the parameters. Global search technique such as genetic algorithm (GA) may solve these problems.

The idea of the proposed daily load forecasting system is to construct seven multi-input multi-output neural network, one for each day in a week. Each network has 24 outputs representing the expected hourly load for a day. One important job in designing the short-term daily load forecasting system is the selection of the input variables. In this system, there are three main input variables:

1) Historical loads data: hourly loads were used as historical load inputs. The historical load data reflect the habit of the family on power consumption.
2) Temperature inputs: the average temperature at the previous day and the present day are used as two inputs in this forecasting system. The value of the average temperature of the present day is the forecasted one got from the weather observatory, assuming that they are accurate.
3) Rainfall index inputs: the average rainfall index at the previous day and the present day (again predicted by the weather observatory) are used as two inputs in this forecasting system. The range of the rainfall index is from 0 to 1. 0 represents no rain and 1 represents heavy rain.

Each of the seven networks serves one day-type form Monday to Sunday. A diagram of the daily load forecasting system is shown in Fig. 2. Each neural network has 28 inputs and 24 outputs. Among the 28 inputs nodes, the first 24 input nodes (z_1, ..., z_{24}) represent the previous 24 hourly loads [21] and are denoted by $z_i = L_i^d(t-1)$, where i = 1, 2, ..., 24. Nodes 25 (z_{25}) and nodes 26 (z_{26}) represent the average temperature of the previous day and present day respectively. Nodes 27 (z_{27}) and Nodes 28 (z_{28}) represent the average rainfall index at the previous day and present day respectively. The output layer consists of 24 output nodes that represent the forecasted 24 hourly loads of a day and is denoted by $y_k(t) = L_i(t)$, i = 1, 2, ..., 24. There are two methods of realizing the daily load forecasting system. One is by training the forecasting neural network off-line and the other is by training on-line. The offline training is a time consuming processing. However, once trained, the system can make the forecast quickly (as a lower number of iterations is needed). In this example, we use 12 sets historical data for off-line training with 2000 iterations. Once trained off-line, the forecasting system operates in an on-line mode, and the weights of neural network will be updated day by day with 500 iterations. Referring to (10), the proposed

neural network used for the daily load forecasting is governed by,

$$y_l(t) = net_o^l\left(\sum_{j=1}^{n_h} net_d^j\left(net_s^j\left(\sum_{i=1}^{24} x_i v_{ij}\right)\right) w_{jl}\right),\ l = 1, 2, \ldots, 24 \quad (19)$$

The number of hidden node (n_h) are changed from 3 to 9 in order to test the learning performance. The fitness function is defined as follow,

$$fitness = \frac{1}{1+err} \quad (20)$$

$$err = \frac{1}{12}\sum_{t=1}^{12}\frac{1}{24}\sum_{k=1}^{24}\frac{\left|y_k^d(t) - y_k(t)\right|}{y_k^d(t)} \quad (21)$$

GA is employed to tune the parameters and structure of the neural network of (19). The objective is to maximize the fitness function of (20). The value of *err* indicates the mean absolute percentage error (MAPE) of the forecasting result. The best fitness value is 1 and the worst one is 0. The population size used for the GA is 20. The chromosomes used for GA are $[v_{ij}\ \ m_s^j\ \ \sigma_s^j\ \ p_{j+1,j}\ \ p_{j-1,j}\ \ w_{jl}\ \ m_o^l\ \ \sigma_o^l]$ for all *i*, *j*, *l*. The number of the iterations to train the proposed neural network is 2000. The simulation data are tabulated in Table II and Table III. Table II shows the simulation results of daily load forecasting on Wednesday and Table III shows the daily load forecasting on Sunday. These tables show the fitness value and the number of trained parameters of the network. We can observe that the performance of the proposed neural network is better than the traditional one. From Table II, the worst result of proposed neural network is still better than the best result of the traditional neural network. The proposed neural network can have fewer hidden nodes and provide better results than the traditional neural network. Table IV and Table V show the average training error from week 1 to week 12 in term of MAPE and the average forecasting error from week 13 to week 14 in term of MAPE on Wednesday and Sunday respectively. The best training error is 1.7829 and 2.0776 using the proposed neural network for Wednesday and Sunday respectively. Comparing with the traditional neural network, it has 18.5% and 32% improvement. On the other hand, the best forecasting error is 1.9365 and 2.8316 using the proposed neural network for Wednesday and Sunday respectively. Comparing with the traditional neural network, it has 31.6% and 30.9% improvement.

Fig.7 shows the simulation results of the daily load forecasting on Sunday (week 13) using the proposed neural network. In this figure, the dotted line represents the forecasted result using the proposed neural network, and the dashed line represents the forecasted result using the traditional neural network. The actual load is shown in solid line. We can observe that the forecasting result using the proposed neural network is better.

VI. CONCLUSION

In this paper, a proposed GA-based neural network has been proposed. The proposed neural network is tuned by GA with arithmetic crossover and non-uniform mutation. A novel neuron model with two activation transfer function has been introduced. With this proposed neuron, the connection of the proposed neural network has a node-by-node relationship in the hidden node. By using this network, a better performance than that of the traditional feed forward neural network can be achieved, and fewer hidden nodes are needed. A multi-input XOR classification problem and a short-term daily load forecasting have been realized using the proposed GA-based neural network. The performance of the proposed network is satisfactory.

ACKNOWLEDGEMENTS

The work described in this paper is substantially supported by a Research Grant from The Hong Kong Polytechnic University (Project No. G-V 954 and A420).

REFERENCES

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

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

[3] Z. Michalewicz, *Genetic Algorithm + Data Structures = Evolution Programs*, (2n ed.), Springer-Verlag, 1994.

[4] Y. Hanaki, T. Hashiyama, and S. Okuma, "Accelerated evolutionary computation using fitness estimation," in *Proc. IEEE Int. Conf. Systems, Man, and Cybernetics*, vol. 1, 1999, pp. 643-648.

[5] K.A. De Jong, *Ph.D. Thesis*: *An analysis of the behavior of a class of genetic adaptive systems,* Ann Arbor, MI: University of Michigan, 1975.

[6] G.X. Yao and Y. Liu "Evolutionary Programming made Faster," *IEEE Trans. Evolutionary Computation*, vol. 3, no. 2, pp.82-102, July 1999.

[7] S. Amin and J.L. Fernandez-Villacanas, "Dynamic Local Search," Second International Conference On Genetic Algorithms in Engineering Systems: Innovations and Applications, pp129-132, 1997.

[8] L. K. Wong, S. H. Ling, F.H.F. Leung, Y.S. Lee, S.H. Wong, T.H. Lee, H.K. Lam, K.H Ho, D.P.K. Lun, and T.C. Hsung, "An intelligent home," in *Proc. Workshop on Service Automation and Robotics, Hong Kong*, June 2000, pp. 111-119.

[9] K.Y. Lee, Y.T. Cha, and J.H. Park, "Short-term load forecasting using an artificial neural network," *IEEE Trans. Power Systems*, Vol. 7, no. 1, pp.124-132, 1992.

[10] Y.Y. Hsu and C.C. Yang, "Design of artificial neural networks for short-term load forecasting. Part 1: Self-organizing feature maps for day type identification," *IEE Proceedings-C*, Vol. 138, no. 5, pp. 407-418, 1991.

[11] Drezga and D.S. Rahman, "Short-term load forecasting with local ANN predictors," *IEEE Trans. Power System*, Vol. 14, no. 3, pp. 844-850, Aug. 1999.

[12] J.A. Momoh, Y. Wang, and M. Elfayoumy, "Artifical neutal network based load forecasting," in *Proc. IEEE Int. Conf. Systems, Man, and Cybernetics 1997,* Vol. 4, 1997, pp. 3443-3451.

[13] D. Part, M. El-Sharkawi, R. Marks, L. Atlas, and M. Damborg, "Electric load forecasting using an artificial neural network," *IEEE Trans. Power System*, Vol. 6, no. 2, pp.442-449, 1991.

[14] C.N. Lu, H.T. Wu and S. Vemuri, "Neural network based short-term load forecasting," *IEEE Trans. Power Systems*, Vol. 8, no. 1, pp.336-342, Feb.1993.

[15] A.P. Rewagad and V.L. Soanawane, "Artificial neural network based short term load forecasting," in *Proc. 1998 IEEE Region 10 Int. Conf. Global Connectivity in Energy, Computer, Communication and Control,* Vol. 2, 1998, pp.588 –595.

[16] A.G. Bakirtzls, V. Petridls, S.J. Klartzis, M.C. Alexladls, and A.H. Malssls, "A neural network short term load forecasting model for Greed power system," *IEEE Trans. Power Systems*, vol.11, no.2, pp.858-863, May1996.

[17] O. Mohammed, D. Part, R. Merchant, T. Dinh, C. Tong and A.

Azeem, "Practical experience with an adaptive neural network short-term load forecasting system," *IEEE Trans. Power System*, vol. 10, no.1, pp.254-265, Feb1995.

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

[19] M. Li, K. Mechrotra, C. Mohan, S. Ranka, "Sunspot Numbers Forecasting Using Neural Network," *5th IEEE International Symposium on Intelligent Control, 1990. Proceedings*, pp.524-528, 1990.

[20] 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, Feb. 2001, pp. 32-53.

[21] Y.S. Zhou and L.Y. Lai, "Optimal design for fuzzy controllers by genetic algorithms," *IEEE Trans., Industry Applications*, vol. 36, no. 1, Jan.-Feb. 2000, pp. 93-97.

[22] C.F. Juang, J.Y. Lin and C.T. Lin, "Genetic reinforcement learning through symbiotic evolution for fuzzy controller design," *IEEE Trans., Systems, Man and Cybernetics, Part B*, vol. 30, no. 2, April, 2000, pp. 290–302.

[23] 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, Aug. 2000, pp. 398-405.

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

[25] 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, Dec. 2000, pp. 753-760.

[26] M. Setnes and H. Roubos, "GA-fuzzy modeling and classification: complexity and performance," *IEEE. Trans, Fuzzy Systems*, vol. 8, no. 5, Oct. 2000, pp. 509–522.

[27] A.E. Bryson and Y.C. Ho, *Applied Optimal Control*, Blaisdell, 1969.

[28] Y. Lecun, "A learning procedure for asymmetric network," *Cognitiva*, pp.599-604, 1985.

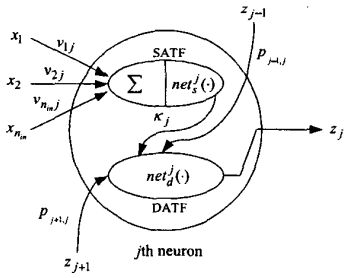


Fig. 1. Model of the proposed neuron.

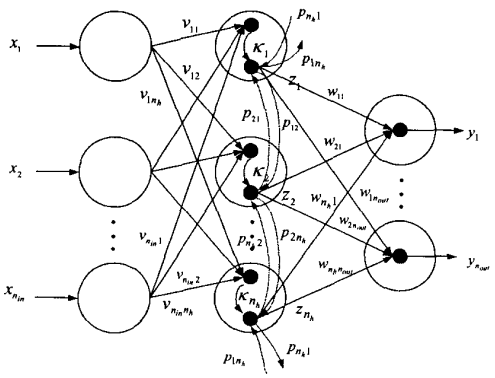


Fig. 2. Connection of the novel neural network

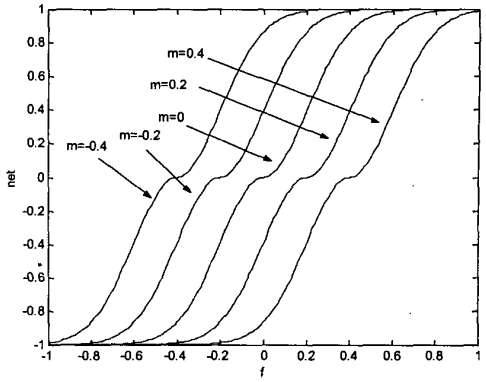


Fig. 3. Sample activation transfer function of the proposed neuron (σ =0.2)

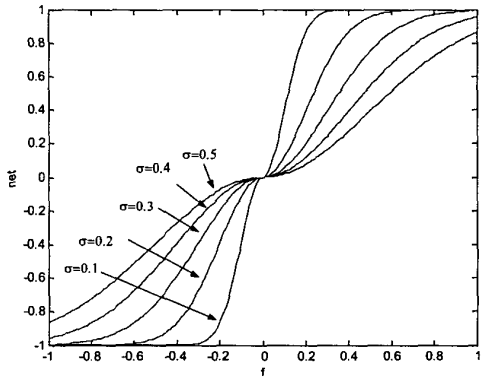
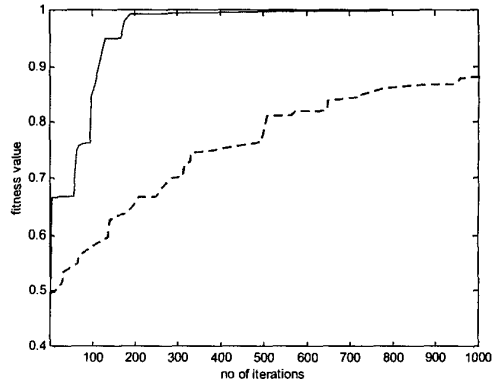
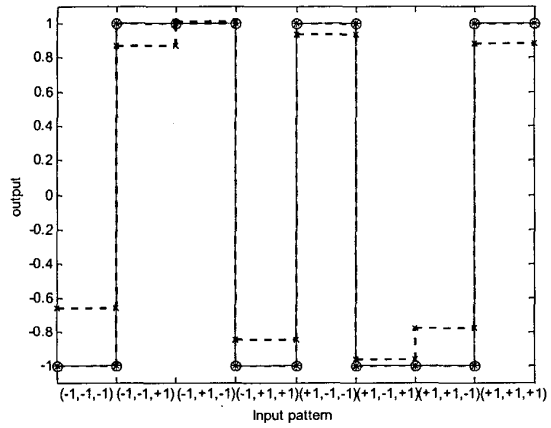


Fig. 4. Sample activation transfer function of the proposed neuron (m=0)



a) The fitness value of the 3-inputs XOR problem obtained by the proposed neural network (solid line) and the traditional neural work (dotted line) after 1000 iterations.



b) The output pattern obtained by the proposed neural network (dotted line with '+' marks) and the traditional neural network (dotted line with 'x' marks) as compared with actual output (solid line with 'o' marks).

Fig. 5. Simulation results of the XOR problem.

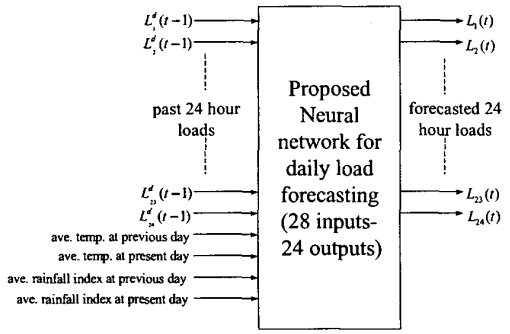


Fig. 6. Proposed neural network for daily load forecasting

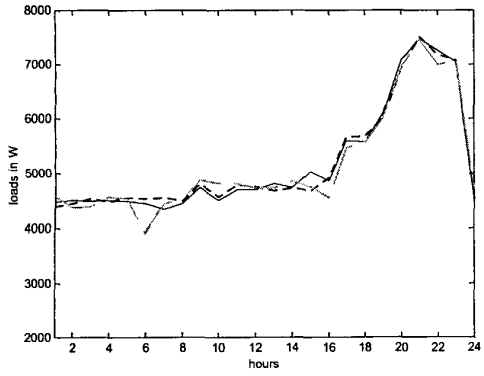


Fig. 7. Daily load forecast results on Sunday (Week13) with the proposed neural network (dotted line) and the traditional neural network (dashed line), as compared with the actual load (solid line).

	Fitness Value	MAE
Proposed Neural Network	0.999988	0.002533
Traditional Neural Network	0.880901	0.135201

Table I. Simulation results of the proposed neural network and the traditional neural network for the 3-inputs XOR classification problem after 1000 iterations.

	Proposed Neural Network		Traditional Neural Network	
n_h	Fitness Value	Number of parameters	Fitness Value	Number of parameters
3	0.977358	216	0.944286	183
4	0.978133	272	0.974208	236
5	0.977819	328	0.977089	289
6	0.981248	384	0.967182	342
7	0.977091	440	0.972413	395
8	0.982483	496	0.967235	448
9	0.981188	552	0.950723	501

Table II. Simulation results of the proposed neural network and the traditional neural network for the daily load forecasting on Wednesday.

	Proposed Neural Network		Traditional Neural Network	
n_h	Fitness Value	Number of parameters	Fitness Value	Number of parameters
3	0.979647	216	0.945921	183
4	0.979552	272	0.965002	236
5	0.977991	328	0.967021	289
6	0.977371	384	0.968254	342
7	0.975546	440	0.970349	395
8	0.976570	496	0.964480	448
9	0.977074	552	0.966030	501

Table III. Simulation results of the proposed neural network and the traditional neural network for the daily load forecasting on Sunday.

	Proposed Neural Network		Traditional Neural Network	
n_h	Training error (MAPE)	Forecasting error (MAPE)	Training error (MAPE)	Forecasting error (MAPE)
3	2.3167	2.8466	5.9002	5.4427
4	2.2356	2.4853	2.6474	3.6954
5	2.2684	2.8813	2.3448	2.8316
6	1.9110	2.0443	3.3932	3.8580
7	2.3446	2.7791	2.8370	2.8964
8	1.7829	1.9365	3.3874	3.4152
9	1.9173	1.9898	5.1831	4.3619

Table IV. Training and forecasting error in term of MAPE with the proposed neural network and the traditional neural network for the daily load forecasting on Wednesday

	Proposed Neural Network		Traditional Neural Network	
n_h	Training error (MAPE)	Forecasting error (MAPE)	Training error (MAPE)	Forecasting error (MAPE)
3	2.0776	2.3054	5.7170	5.2094
4	2.0875	1.9120	3.6268	4.0758
5	2.2504	2.5362	3.4104	3.3927
6	2.3153	2.5324	3.2786	3.2126
7	2.5067	2.6807	3.0556	2.7665
8	2.3992	2.3987	3.6828	4.1690
9	2.3464	2.2005	3.5164	4.1102

Table V. Training and forecasting error in term of MAPE with the proposed neural network and the traditional neural network for the daily load forecasting on Sunday.