The 30th Annual Conference of the IEEE Industrial Electronics Society, November 2 - 6, 2004, Busan, Korea

Genetic Algorithm Based Variable-Structure Neural Network and its Industrial Application[1]

S.H. Ling[2], *Student Member, IEEE*, F.H.F. Leung, Senior *Member, IEEE*, and H.K. Lam, *Member, IEEE*

Abstract: **This paper presents a neural network model with a variable structure, which is trained by an improved genetic algorithm (GA). The proposed variable-structure neural network (VSNN) consists of a Neural Network with Link Switches (NNLS) and a Network Switch Controller (NSC). In the NNLS, switches in its links between the hidden and output layers are introduced. By introducing the NSC to control the switches in the NNLS, the proposed neural network can model different input patterns with variable network structures. The proposed network gives better results and increased learning ability than conventional feed-forward neural networks. An industrial application on short-term load forecasting in Hong Kong is given to illustrate the merits of the proposed network.**

I. INTRODUCTION

Neural networks [1] can approximate any nonlinear continuous function to an arbitrary accuracy. They are widely applied in areas such as prediction [3], load forecasting [4-5, 13-14], classification [8], system modeling and control [1]. Owing to its particular structure, a neural network is good at learning [9] using some algorithms such as Genetic Algorithm (GA) [3, 7] and back propagation [1]. A fixed network structure is usually used in applications. However, an oversized network may introduce excessive computations for both training and applying. In [3], it was shown that a partly connected neural network can perform better than a fully connected neural network because some links in the fully connected network could be redundant. It would be good if different network structures are used for different applications.

In general, there are two classes of tuning algorithms: local and global search algorithms. Gradient descent techniques, e.g. the back-propagation algorithm [1], are kinds of search algorithms for the local optima. They are good for optimizing some simple functions, e.g. convex or concave functions, in a fast convergence rate. However, they only converge to local minima and are sensitive to initial values. Genetic algorithm [9], simulated annealing [9] and tabu search [9] are kinds of search algorithms for the global optima over a defined domain. They are good for optimizing some complex functions. In [3], it was shown that neural networks trained by GA could perform better than neural networks trained with gradient descent techniques. In this paper, an improved GA [6] with modified genetic operations of crossover and mutation will be employed for tuning the neural networks.

A GA based variable-structure neural network (VSNN) is proposed. It consists of a Neural Network with Link Switch (NNLS) and a Network Switch Controller (NSC). In the NNLS, switches introduced in some links between the hidden and output layers are proposed to facilitate the tuning of the network structure. The on/off states of the link switches are controlled by the NSC. The NSC is a 3-layer feed-forward neural network. By considering an application with a large domain of input-output mappings, a VSNN should be able to perform the mapping task in a more efficient way in terms of accuracy and network complexity (number of parameters). In practice, a VSNN divides the input-output domain into several sub-domains, and we employ different network structures to perform the mapping task for different input-output sub-domains. Effectively, it is an adaptive network capable of handling different input patterns, and exhibits a better performance.

An accurate load forecasting in power systems is important. Correct forecasts have significant influences to system operations such as unit commitment, load dispatch and maintenance scheduling. The forecasting information can be used to aid optimal energy interchange between utilities, thereby saving valuable fuel costs. The short-term load forecasting problems are non-linear, random, and time-varying. In particular, artificial neural network offers a promising and effective platform for tackling the short-term electric load forecasting problems [13-14]. In this paper, an industrial application on short-term load forecasting in Hong Kong is presented. The actual load data are obtained from the local utility company, the CLP Power Hong Kong Limited. By employing the proposed VSNN on short-term load forecasting, the proposed network can model different input load patterns with variable structures and give better results and increased learning ability than conventional neural networks.

This paper is organized as follows. In Section II, the proposed VSNN is presented. In Section III, training of the parameters of the proposed network using the improved GA will be presented. In Section IV, the

[1] The work described in this paper was substantially supported by a grant from the Hong Kong Polytechnic University (Project No. G-YX31). The data in the application example are offered by CLP Power Hong Kong Limited. The authors are with the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

[2] E-mail: ensteve@eie.polyu.edu.hk

0-7803-8730-9/04/$20.00 ©2004 IEEE

industrial application on short-term load forecasting in Hong Kong will be given to show the merits of the proposed network. A conclusion will be drawn in Section V.

II. Variable-Structure Neural Network Model

The block diagram of the proposed VSNN is shown in Fig. 1. It consists of the Neural Network with Link Switch (NNLS) and the Network Switch Controller (NSC). In the NNLS, link switches are introduced between the hidden layer and the output layer. The states of the switches are controlled by the NSC, which depend on the network inputs.

A. Neural Network with Link Switch (NNLS)

NNLS is the core of the VSNN. By employing switches in the links, the structure of the network becomes variable. By using the NSC to control the state of switches, it is equivalent to divide the input-output domain into several sub-domains and employ networks of different structures to perform the mapping task for different sub-domains. The proposed structure of the VSNN is shown in Fig. 3, which 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. The output of the hidden layer of the NNLS is given by,

$$\zeta_j = \text{logsig}\left(\sum_{i=1}^{n_{in}} z_i v_{ij} - c_j^1\right), j = 1, 2, \ldots, n_h \quad (1)$$

where z_i, i = 1, 2, ..., n_{in} are the input variables; n_{in} denotes the number of inputs; n_h denotes the number of hidden nodes; v_{ij} denotes the weight of the link between the i-th input and the j-th hidden node; c_j^1 denotes the bias for the hidden nodes; logsig(·) denotes the logarithmic sigmoid function:

$$\text{logsig}(\eta) = \frac{1}{1+e^{-\eta}}, \ \eta \in \Re, \quad (2)$$

Link switches between the hidden and output layers are present. A unit step function is used to realize the link switches,

$$\delta(\alpha) = \begin{cases} 0 \text{ if } \alpha < 0 \\ 1 \text{ if } \alpha \geq 0 \end{cases}, \alpha \in \Re \quad (3)$$

In this proposed VSNN, the number of switches (n_{sw}) is variable and is given by,

$$n_{sw} = k_f \times n_h \times n_{out} \quad (4)$$

where $k_f \in (0 \ \ 1]$, is a constant factor governing the percentage of links with switches in the network, e.g. $k_f = 0.1$ means 10% of links have switches; $k_f = 1$ means all links have switches. The value of the constant k_f depends on the application and the network size. As the size of network is larger, the value of k_f is smaller. With the link switches introduced, the output of the proposed neural network (Fig. 3) is given by,

$$y_l = \text{logsig}\left(\sum_{j=1}^{n_h} (\beta_{jl} w_{jl}) \zeta_j - c_l^2\right), \ l = 1, 2, \ldots, n_{out} \quad (5)$$

$$= \text{logsig}\left(\sum_{j=1}^{n_h} (\beta_{jl} w_{jl}) \text{logsig}\left(\sum_{i=1}^{n_{in}} z_i v_{ij} - c_j^1\right) - c_l^2\right) \quad (6)$$

where w_{jl}, j = 1, 2, ..., n_h; l = 1, 2, ... n_{out} denotes the weight of the link between the j-th hidden and the l-th output nodes; β_{jl} denotes the on/off states of the switches; c_l^2 denotes the bias for the output nodes. Referring to Fig. 3, let

$$\begin{bmatrix} s_{11} & s_{12} & \cdots & s_{n_h 1} \\ s_{12} & s_{22} & \cdots & s_{n_h 2} \\ \vdots & \vdots & \ddots & \vdots \\ s_{1n_{out}} & s_{2n_{out}} & \cdots & s_{n_h n_{out}} \end{bmatrix} = \begin{bmatrix} \varphi_1 & \varphi_2 & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & \varphi_{n_{sout}-2} & \varphi_{n_{sout}-1} & \varphi_{n_{sout}} \end{bmatrix},$$

$$\left[\beta_{jl}\right] = \begin{bmatrix} \delta(s_{11}) & \delta(s_{21}) & \cdots & \delta(s_{n_h 1}) \\ \delta(s_{12}) & \delta(s_{22}) & \cdots & \delta(s_{n_h 2}) \\ \vdots & \vdots & \ddots & \vdots \\ \delta(s_{1n_{out}}) & \delta(s_{2n_{out}}) & \cdots & \delta(s_{n_h n_{out}}) \end{bmatrix} \quad (7)$$

where s_{jl} denotes the parameter of the link switch and φ_h, h = 1, 2, ..., n_{sout} denotes the outputs of the NSC; n_{sout} denotes the number of output of the NSC. If the link has no switch, $\delta(\cdot)$ = 1. For instance, when $k_f = 0.5$ (half of the links have switches), n_h =4 and n_{out} =3, n_{sw} will be equal to $0.5 \times 4 \times 3 = 6$ and $\left[\beta_{jl}\right]$ can be equal to:

$$\left[\beta_{jl}\right] = \begin{bmatrix} \delta(s_{11}) & \delta(s_{21}) & 1 & 1 \\ \delta(s_{12}) & \delta(s_{22}) & 1 & 1 \\ \delta(s_{13}) & \delta(s_{23}) & 1 & 1 \end{bmatrix} \quad (8)$$

The parameters of this NNLS can be trained by an improved GA [6] and the parameters of the link switches are offered by the NSC.

B. Network Switch Controller (NSC)

In the NNLS, the switch parameters s_{jl} are provided by a Network Switch Controller (NSC), which is a 3-layer feed-forward neural network. As shown in Fig. 2, the input-output relationship of the NSC is given by,

$$\varphi_h = \text{tansig}\left(\sum_{g=1}^{n_{sh}} u_{gh} \text{logsig}\left(\sum_{i=1}^{n_{in}} t_{ig} z_i - b_g^1\right) - b_h^2\right), \ h = 1, 2, \ldots, n_{sout} \quad (9)$$

where z_i, i = 1, 2, ..., n_{in} are the input variables; n_{in} denotes the number of inputs; n_{sh} denotes the number of hidden nodes; t_{ig}, g = 1, 2, ..., n_{sh}, denotes the weight of the link between the i-th input and the g-th hidden node; u_{gh}, denotes the weight of the link between the g-th hidden and the h-th output nodes; b_g^1 and b_h^2 denote the biases for the hidden nodes and output nodes

respectively; tansig(·) denotes the hyperbolic tangent sigmoid function:

$$\text{tansig}(\eta) = \frac{2}{1+e^{-2\eta}} - 1\ ,\ \eta \in \Re, \tag{10}$$

φ_h is the h-th output of the NSC. All the parameters of the NSC are tuned by the improved GA [6].

III. Training of Parameters

The proposed VSNN can be used to learn the input-output relationship of an application using the improved GA [6]. The input-output relationship is described by,

$$\mathbf{y}^d(t) = \mathbf{g}\left(\mathbf{z}^d(t)\right),\ t = 1, 2, \ldots, n_d \tag{11}$$

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 for the GA depends on the application. The most common fitness function is given by,

$$fitness = \frac{1}{1+err} \tag{12}$$

In (12), *err* could be the mean-square-error (MSE), mean-absolute-percentage-error (MAPE), etc. The MSE is defined as:

$$err = \frac{\sum_{t=1}^{n_d}\sum_{k=1}^{n_{out}}\left(y_k^d(t) - y_k(t)\right)^2}{n_d n_{out}} \tag{13}$$

The MAPE is defined as:

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

The objective is to maximize the fitness value of (12) using GA by setting the chromosome to be $\left[v_{ij} \quad w_{jl} \quad c_j^1 \quad c_l^2 \quad t_{ig} \quad u_{gh} \quad b_g^1 \quad b_h^2\right]$ for all g, h, i, j, l. In this paper, all elements in the chromosome are inside [3,–3]. The range of the fitness value of (12) is [0, 1].

IV. Short-Term Electric Load Forecasting In Hong Kong

We consider the short-term load forecasting (STLF) for the power supply system in Hong Kong. STLF is important to power system because it plays a role in the formulation of economic, reliability, and secure operating strategies for the power system. The objectives of STLF is i) to derive the scheduling functions that determine the most economic load dispatch with operational constraints and policies, environmental and equipment limitations; ii) to assess the security of the power system at any time point; iii) to provide system dispatchers with timely information. The dispatchers need the forecasted load information to operate the system economically and reliably.

The proposed VSNN is applied to do STLF. The application of neural network to STLF has been explored extensively in the literature [13-14]. The idea is to construct seven multi-input multi-output neural networks, one for each day of a week. Each neural network has 24 outputs representing the expected hourly load for a day. A diagram of one of the seven NNs for the load forecasting is shown in Fig. 2. The network has 28 inputs and 24 outputs. Among the 28 input nodes, the first 24 nodes (z_1, ..., z_{24}) represent the previous 24 hourly loads [15] and are denoted by $z_i = L_i^d(t-1)$, i = 1, 2, ..., 24. Node 25 (z_{25}) and node 26 (z_{26}) represent the average temperatures of the previous day ($T(t-1)$)and the forecasted averaged temperatures of the present day ($T(t)$) respectively. Node 27 (z_{27}) and node 28 (z_{28}) represent the average relative humidity at the previous day ($RH(t-1)$) and the forecasted average relative humidity at the present day ($RH(t)$) respectively. The output layer consists of 24 output nodes that represent the forecasted 24 hourly loads of a day, and are denoted by $y_k(t) = L_i(t)$, i = 1, 2, ..., 24. Such a network structure is chosen based on the assumption that the consumption patterns of the seven days within a week would differ significantly among each other, while the patterns among the same day of weeks are similar. By using the past 24 hourly loads as the inputs, the relationship between a given hour's load and the 24 hourly loads of the pervious day can be considered. Temperature information (Node 25 to Node 28) is also important inputs to the STLF. For any given day, the deviation of the temperature variable from a normal value may cause such significant load changes as to require major modifications in the unit commitment pattern. Humidity is similar to temperature that affects the system load, particularly in hot and humid areas.

In this paper, we use a data set in year 2000 provided by CLP Power Hong Kong Ltd to illustrate the proposed VSNN on doing STLF. The proposed VSNN was trained using 5 months (20 weeks) load data from April 15 to August 16, 2000. Referring to (6), the proposed VSNN used for doing STLF is governed by,

$$y_l = \text{logsig}\left(\sum_{j=1}^{n_h}\left(\beta_{jl} w_{jl}\right)\text{logsig}\left(\sum_{i=1}^{28} z_i v_{ij} - c_j^1\right) - c_l^2\right),\ l = 1, 2, \ldots 23, 24. \tag{15}$$

Improved GA [6] is employed to tune the parameters of the proposed VSNN of (15). The fitness functions is defined as follows,

$$fitness = \frac{1}{1+err} \tag{16}$$

$$err = \frac{\sum_{t=1}^{20}\sum_{k=1}^{24}\dfrac{\left|y_k^d(t) - y_k(t)\right|}{y_k^d(t)}}{20\times 24} \tag{17}$$

The value of *err* indicates the mean absolute percentage error (MAPE) of the load forecasting system. For comparison, a conventional 3 layers feed-forward neural

network (fixed network) is also applied to do the same job. All parameters of both networks are trained by the improved GA [6]. For all cases, the initial values of the parameters of the neural network are randomly generated. The number of iteration to train the neural networks is 20000. For the GA [6], the probability of mutation (p_m), the weight of crossover (w), the parameters of w_t and w_f are 0.01, 1, 0.5, 0.5 respectively for both the proposed and conventional neural networks. The population size is 100. All the results are averaged ones out of 10 runs. In the proposed VSNN, the number of hidden nodes (n_h) of the NNLS is 55. It is chosen by trial and error through experiments for good performance. The training and the forecasting results for August 17 (Thursday) and August 20 (Sunday) with different values of k_f and n_{th} are tabulated in Table I and Table II. In these tables, the proposed VSNN (with a variable structure) is compared with the conventional neural network (with a fixed structure) under 2 conditions: similar number of parameters (n_{para}) and same number of hidden nodes (n_h). From these two tables, it can be shown that the proposed VSNN performs better than the conventional neural network in terms of the best average training and forecasting errors. With the proposed VSNN, the best average training errors are 2.1013% (k_f = 0.03) for Thursday and 1.8742% (k_f = 0.025) for Sunday respectively. These imply 19.6% and 19.3% improvements over the conventional NN with a similar number of parameters. The best average forecasting errors for August 17 (Thursday) and August 20 (Sunday) are 1.7574% (k_f = 0.03) and 2.2639% (k_f = 0.025) respectively. These imply 19.7% and 23.8% improvements. On the other hand, with the same number of hidden nodes, the proposed VSNN gives 23% and 26.5% improvements for Thursday and Sunday. Fig. 4 and Fig. 5 show the forecasting results of the load forecasting on August 17 (Thursday) and August 20 (Sunday). In these figures, the dashed line represents the best forecasted result using the proposed network, and the dotted line is the best forecasted result using the conventional network. The actual load is represented by a solid line. We can see that the forecasted result using the proposed neural network is better. Fig. 6 and Fig. 7 show the comparison of the average forecasted load errors (MAPE) for August 17 (Thursday) and August 20 (Sunday) respectively. In these 2 figures, most of the forecasted load errors from the proposed VSNN are smaller than those from the conventional neural network.

V. CONCLUSION

An improved GA based variable-structure neural network has been proposed. It consists of a Neural Network with Link Switches (NNLS) and a Network Switch Controller (NSC). In the NNLS, a feed-forward neural network with link switches between the hidden and output layers have been introduced. With the NSC controlling the switches in the NNLS, the proposed neural network can model different input patterns with different network structures. By using a variable structure, the performance of the proposed network is found to be better than that of the conventional neural network. All parameters of the proposed neural network are tuned by an improved GA. An industrial application on short-term load forecasting in Hong Kong has been presented. The performance of the proposed network is better than that of a conventional neural network.

REFERENCES

[1] M. Brown and C. Harris, *Neural Fuzzy Adaptive Modeling and Control*. Prentice Hall, 1994.

[2] D.E. Rumelhrt and J.L. McClelland, *Parallel Distributed Processing: Explorations in Microstructure of Cognition*, vol. 1. Cambridge, MA:MIT Press, 1986.

[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, pp.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*, vol. 50, no. 4, pp.793-799, Aug. 2003.

[5] S.H. Ling, F.H.F Leung, H.K. Lam, and P.K.S. Tam, "Short-term electric load forecasting based on a neural fuzzy network," *IEEE Trans. Industrial Electronics*, vol. 50, no. 6, pp.1305-1316, Dec. 2003.

[6] S.H. Ling, H.K. Lam, F.H.F Leung, and Y.S. Lee, "Improved genetic algorithm for economic load dispatch with valve-point loadings," in *Proc. 29th Annual Conference of the IEEE Industrial Electronics Society, IECON'2003, Virginia, USA*, Nov 2003, pp. 442-447.

[7] S.H. Ling, H.K. Lam, F.H.F. Leung and P.K.S. Tam, "Learning of neural network parameters using fuzzy genetic algorithm," *Congress on Evolutionary Computation (CEC2002), Hawaii*, May 2002, pp. 1928-1933.

[8] 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," *Proceedings of the 10th IEEE International Conference on Fuzzy Systems, FUZZ-IEEE'2001, Australia*, Dec 2001, pp. 1464-1467.

[9] D.T. Pham and D. Karaboga, *Intelligent Optimization Techniques, Genetic Algorithms, Tabu Search, Simulated Annealing and Neural Networks*. Springer, 2000.

[10] 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.

[11] M. Srinivas, L.M. Patnaik "Genetic algorithms: a survey," *IEEE Computer*, vol. 27, issue 6, pp. 17-26, June 1994.

[12] L. Davis, *Handbook of genetic algorithms*. NY: Van Nostrand Reinhold, 1991.

[13] C.C. Hsu and C.Y. Chen, "Regional load forecasting in Taiwan–applications of artificial neural networks," *Energy Conversion and Management*, vol. 44, pp. 1941-1949, Jul. 2003.

[14] T. Senjyu, H. Takara, and T. Funabashi, "One-hour-ahead load forecasting using neural network," *IEEE Trans. Power Syst.*, vol. 17, no.1, pp. 113-118, Feb. 2002.

[15] J.A. Momoh, Y. Wang, and M. Elfayoumy, "Artifical neural network based load forecasting," *IEEE International Conference on System, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation*, vol. 4, 1997, pp. 3443-3451.

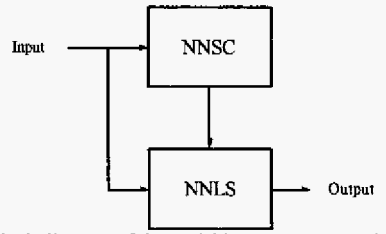

Fig. 1. Block diagram of the variable-structure neural network.

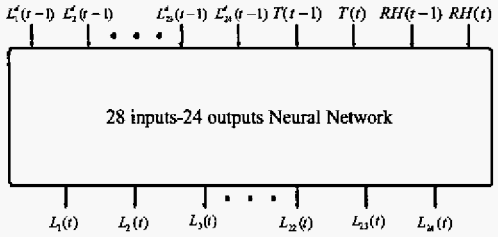

Fig. 2. Proposed neural network based short-term load forecaster.

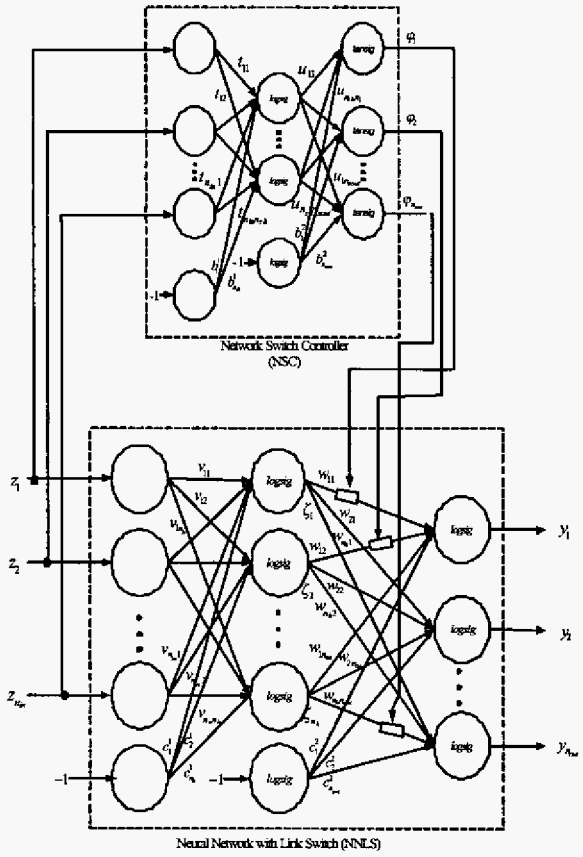

Fig. 3. Proposed structure of the VSNN.

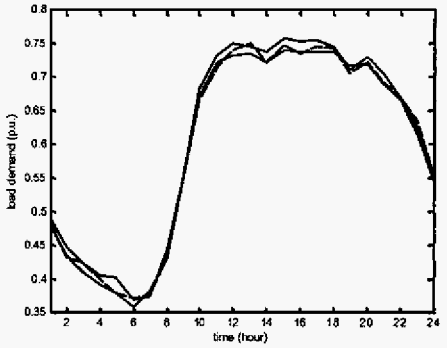

Fig. 4. Load forecasting result for August 17, 2000 (Thursday) using the proposed VSNN (dashed line) and the conventional NN (dotted line), as compared with the actual load (solid line).

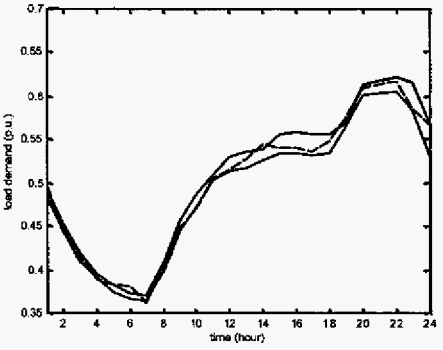

Fig. 5. Load forecasting result for August 20, 2000 (Sunday) using the proposed VSNN (dashed line) and the conventional NN (dotted line), as compared with the actual load (solid line).

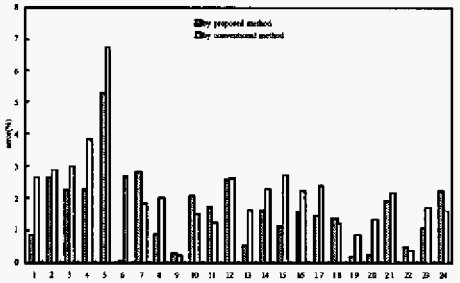

Fig. 6. Comparison of average forecasted load errors (MAPE) for August 17, 2000 (Thursday).

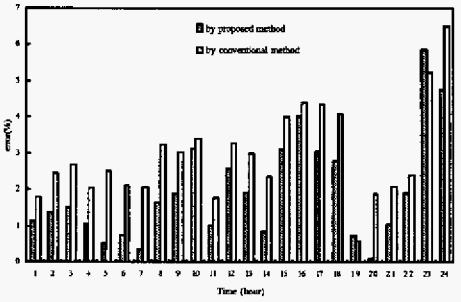

Fig. 7. Comparison of average forecasted load errors (MAPE) for August 20, 2000 (Sunday).

Proposed VSNN

n_h =55	Error (MAPE)	n_{th} = 12 k_f = 0.020 n_{para} = 3625	n_{th} = 15 k_f = 0.025 n_{para} = 3902	n_{th} = 17 k_f = 0.030 n_{para} = 4152
Average	Training	2.1533%	2.1065%	2.1013%
	Forecasting	1.9218%	1.8855%	1.7574%
Best	Training	1.9672%	1.9368%	1.9452%
	Forecasting	1.7904%	1.6144%	1.5242%
Worst	Training	2.3283%	2.3177%	2.1596%
	Forecasting	2.1306%	2.1251%	1.9457%

Conventional NN: with a similar number of parameters

	Error (MAPE)	n_h = 68 n_{para} = 3628	n_h = 74 n_{para} = 3946	n_h = 78 n_{para} = 4158
Average	Training	2.6708%	2.6064%	2.6128%
	Forecasting	2.2089%	2.2271%	2.1885%
Best	Training	2.5801%	2.5831%	2.5671%
	Forecasting	2.1736%	2.2093%	2.1417%
Worst.	Training	2.7250%	2.6383%	2.6718%
	Forecasting	2.2451%	2.2616%	2.2352%

Conventional NN: with the same number of hidden nodes

	Error (MAPE)	n_h = 55 n_{para} =2939
Average	Training	2.6791%
	Forecasting	2.2835%
Best	Training	2.6571%
	Forecasting	2.1882%
Worst	Training	2.6930%
	Forecasting	2.3505%

Table I. Results of the proposed VSNN and the conventional NN on Thursday.

Proposed VSNN

n_h =55	Error (MAPE)	n_{th} =12 k_f = 0.020 n_{para} =3625	n_{th} =15 k_f = 0.025 n_{para} =3902	n_{th} =17 k_f = 0.030 n_{para} =4152
Average	Training	1.9133%	1.8742%	1.9126%
	Forecasting	2.2755%	2.2639%	2.2814%
Best	Training	1.8517%	1.84609%	1.8632%
	Forecasting	1.9723%	1.8460%	1.9521%
Worst	Training	2.0828%	1.9826%	2.0651%
	Forecasting	2.5161%	2.4461%	2.5802%

Conventional NN: with a similar number of parameters

	Error (MAPE)	n_h = 68 n_{para} = 3628	n_h = 74 n_{para} = 3946	n_h = 78 n_{para} = 4158
Average	Training	2.3361%	2.3236%	2.3311%
	Forecasting	2.9651%	2.9721%	2.9801%
Best	Training	2.3064%	2.2851%	2.2813%
	Forecasting	2.8310%	2.8279%	2.8255%
Worst.	Training	2.3951%	2.3966%	2.4411%
	Forecasting	3.1598%	3.1489%	3.1602%

Conventional NN: with the same number of hidden nodes

	Error (MAPE)	n_h = 55 n_{para} =2939
Average	Training	2.7059%
	Forecasting	3.0804%
Best	Training	2.6117%
	Forecasting	2.9971%
Worst	Training	2.8523%
	Forecasting	3.2066%

Table II. Results of the proposed VSNN and conventional NN on Sunday.