SHORT TERM LOAD FORECASTING USING A MULTILAYER NEURAL NETWORK WITH AN ADAPTIVE LEARNING ALGORITHM

Kun-Long Ho, Student Member IEEE Yuan-Yih Hsu, Senior Member IEEE, Chien-Chuen Yang

Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan, ROC

Abstract – A multilayer neural network with an adaptive learning algorithm is designed for short term load forecasting. Extensive studies have been performed on the effect of various factors such as learning rate, momentum, the number of presentations in an iteration, etc. on the efficiency and accuracy of the backpropagation-momentum learning method which is employed in the training of artificial neural networks. To speed up the training process, a new learning algorithm for the adaptive training of neural networks is presented. The effectiveness of the neural network with the proposed adaptive learning algorithm is demonstrated by short term load forecasting of Taiwan power system. It is found that, once trained by the proposed learning algorithm, the neural network can yield the desired hourly load forecast very efficiently and accurately. Moreover, the proposed adaptive learning algorithm converges much faster than the conventional backpropagation-momentum learning method.

Keywords: load forecasting, artificial neural networks, artificial intelligence, machine learning systems.

1. INTRODUCTION

The main objective of short term load forecasting is to predict the hourly loads, one day or even one week beforehand, which are necessary for the operational planning of a power system [1]. Since the electric load demand is a function of weather variables and human social activities, both statistical techniques [2–7] and expert system approaches [8–10] have been proposed. Usually, the statistical approaches are effective for the forecasting of normal days but fail to yield good results for those days with special events, e.g., the superbowl day. To take these special events into account, operators' experience and heuristic rules must be employed. This motivated the development of rule-based expert systems for load forecasting [8–10].

In this paper, a new approach using artificial neural networks (ANN) is proposed for short-term load forecasting. Among the various artificial neural networks presented so far, the multilayer feedforward neural network [11–17] is employed. To determine the connection weights between neurons, the conventional backpropagation-momentum learning technique with constant learning rate η and momentum α [11] is first employed in the training process. The effect of different learning rates and momenta on the convergence property of the learning process is extensively studied. In addition, the number of presentations in an iteration and the number of input/output patterns in a presentation are also varied to see their effect on the convergence rate. To speed up the convergence rate of the learning process, an adaptive learning algorithm in which the momentum is automatically adapted in the training process is presented. Results from the present study indicate that the proposed adaptive learning algorithm converges much faster than the conventional backpropagation-momentum technique.

To demonstrate the effectiveness of the artificial neural network, short-term load forecasting is performed on Taiwan power system. The neural network has 46 input nodes, 60 hidden nodes, and one output node. In the training process, the weather variables and load data in the past few days are used. Once trained, the neural network is capable of forecasting the hourly loads with a mean absolute error (MAE) of 0.7%.

In the next section, the problem of load forecasting is briefly described. This is followed by an introduction of the artificial neural network and the backpropagation-momentum learning algorithm. Then, we present a new adaptive learning algorithm for the neural network. An example is then given to demonstrate the effectiveness of the proposed neural network.

2. PROBLEM FORMULATION

The objective of short-term load forecasting is to predict the 24 hourly loads L(i) , i = 1, 2, ... , 24, of the next day. There are four main steps involved in the forecasting process [10].

Step 1. Read in the day to be forecasted and the forecasted weather variables.

Step 2. Identify day type and get the 24 normalized hourly loads Ln (i), i = 1, 2, ... , 24, for that particular day type. These normalized hourly loads have been stored in the data base to represent the load pattern of each day type.

Step 3. Compute the forecasted daily peak load and valley load using the following equations.

$$L_p = g_p * T_p + h_p \quad (1)$$

$$L_t = g_t * T_t + h_t \quad (2)$$

where

L_p = peak load of the day
L_t = valley load of the day
T_p = equivalent forecasted high temperature of the system on the day
T_t = equivalent forecasted low temperature of the system on the day

91 SM 450-7 PWRS A paper recommended and approved by the IEEE Power System Engineering Committee of the IEEE Power Engineering Society for presentation at the IEEE/PES 1991 Summer Meeting, San Diego, California, July 28 - August 1, 1991. Manuscript submitted July 9, 1990; made available for printing June 19, 1991.

0885-8950/92$03.00©1992 IEEE

g_p, h_p, g_t, and h_t = coefficients determined by least-square-error technique using the load and weather data in the data base.

Step 4. Compute the forecasted hourly loads L(i) as follows.

$$L(i) = L_n(i)\,(L_p - L_t) + L_t \qquad i = 1, 2, \ldots, 24 \tag{3}$$

Details of the aforementioned procedures are given in [10].

It has been noted during the course of our previous work [10] that the forecasted hourly loads depend heavily on the forecasted peak and valley loads. In fact, the mean absolute error (MAE) can be reduced to a great extent if the actual peak and valley loads are employed to replace the forecasted peak and valley loads. This implies that the rule-based expert system developed in our previous work [10] is good at identifying day type and providing the 24 normalized hourly loads $L_n(i)$, i = 1, 2, ... , 24, for each day type. However, to have even better forecast results, the approach for peak and valley load forecasting employed in that work needs further refinement. Thus, the present study will be focused on the application of neural networks to forecast the peak load L_p and valley load L_t of Taiwan power system. These peak load and valley load values, together with the normalized hourly loads $L_n(i)$ obtained in our previous work [10], can be used to forecast the hourly loads L(i). Of course, the proposed neural network can also be employed to achieve the normalized hourly loads L(i). But our main effort in this study will be placed on peak and valley load forecasting.

3. MULTILAYER NEURAL NETWORKS

Consider the multilayer feedforward neural network as shown in Fig. 1 [11].

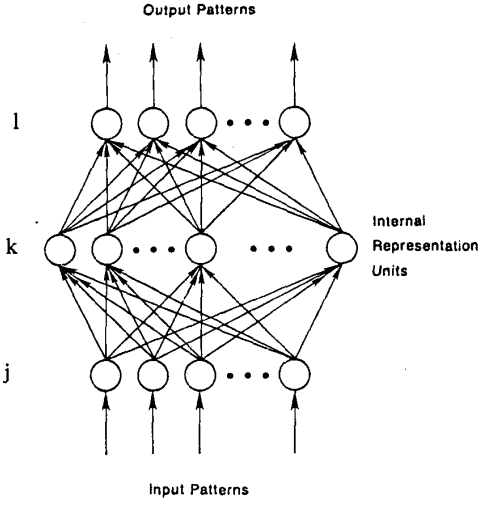

Fig. 1 A multilayer neural network

The nodes in the neural network can be divided into three layers: the input layer, the output layer and one or more hidden layers. The nodes in the input layer receive input signals form external sources and the nodes in the output layer provide the desired output signals. Since only one hidden layer is employed in the present study, it is called a three-layer neural network according to Rumelhart's definition [11]. However, since the input nodes simply pass the inputs to the nodes in the hidden layer and do not perform the activation function and output function that normal neurons are supposed to carry out, some people regard this kind of network as a two-layer neural network [18]. The important thing is to note that only the nodes in the hidden layer and output layer are regarded as ordinary neurons which perform activation function and output function. It should also be noted that, in the employed feedforward network structure, signal propagation is allowed only from the input layer to the hidden layer and from the hidden layer to the output layer. Connections between nodes within the same layer or from the input layer directly to the output layer are not permitted.

For each neuron k in the hidden layer and neuron ℓ in the output layer, the net inputs are given by

$$net_k = \sum_{j=1}^{NJ} w_{kj}\, O_j \qquad k = 1, \ldots, NK \tag{4}$$

and

$$net_\ell = \sum_{k=1}^{NK} w_{\ell k}\, O_k \qquad \ell = 1, \ldots, NL \tag{5}$$

respectively. The neuron outputs are given by

$$O_j = net_j \tag{6}$$

$$O_k = \frac{1}{1 + e^{-(net_k + \theta_k)}} = f_k\,(net_k, \theta_k) \tag{7}$$

$$O_\ell = \frac{1}{1 + e^{-(net_\ell + \theta_\ell)}} = f_\ell\,(net_\ell, \theta_\ell) \tag{8}$$

where net_j is the input signal from the external sources to node j in the input layer.

Let the connection weights w_{kj} and $w_{\ell k}$ be updated after each presentation. In other words, they are changed after NQ input/output patterns (pairs) have been presented. Then for a presentation p, the sum of squared errors to be minimized is given by

$$E_p = \sum_{q=1}^{NQ} \sum_{\ell=1}^{NL} (t_{pq\ell} - O_{pq\ell})^2 \tag{9}$$

where $t_{pq\ell}$ and $O_{pq\ell}$ are the target output value and computed output value at output node ℓ for the qth input/output pair in presentation p, respectively.

By minimizing the error E_p using the technique of gradient descent, the connection weights between hidden unit k and output unit ℓ can be updated by using the following equations [11]

$$\Delta w_{\ell k}(p) = \eta \sum_{q=1}^{NQ} \frac{\delta_{pq\ell}}{NQ} O_{pqk} + \alpha \Delta w_{\ell k}(p-1) \quad (10)$$

where

$$\delta_{pq\ell} = (t_{pq\ell} - O_{pq\ell}) f'_{\ell}(net_{pq\ell}) \quad (11)$$

and O_{pqk} is the output of hidden unit k for the qth input/output pair in presentation p. Note that the learning rate η and the momentum α give the relative weights for the present error and the error in the previous presentation. These factors affect the convergence rate of the learning process to a great extent. Details on the selection of η and α will be addressed later. The connection weights between input unit j and hidden unit k can be updated using similar equations

$$\Delta w_{kj}(p) = \eta \sum_{q=1}^{NQ} \frac{\delta_{pqk}}{NQ} O_{pqj} + \alpha \Delta w_{kj}(p-1) \quad (12)$$

where

$$\delta_{pqk} = f'_k(net_{pqk}) \sum_{\ell=1}^{NL} \delta_{pq\ell} w_{\ell k} \quad (13)$$

Note that the threshold θ is also updated in the learning process using an equation similar to eq. (12).

The overall training (learning) process is summarized by the flow chart in Fig. 2.

In applying the above learning rule, there are several issues which should be addressed.

(1) What are the optimal values of learning rate (η) and momentum (α)? Most works on feedforward neural nets use constant values of η and α. Rumelhart [11] recommended that a combination of $\eta = 0.25$ and $\alpha = 0.9$ can yield good results for most problems. But there is still no consensus as to what values of η and α should be used in the learning process. In fact, the optimal values of η and α may be problem-dependent. In the present work, extensive studies have been carried out on the effect of different values of η and α on the convergence rate of the learning method. It is found that the combination of $\eta = 2.5$ and $\alpha = 0.9$ is a good choice for the load forecasting problem. To further speed up the convergence rate, an adaptive algorithm as described in next section is developed to adapt the momentum during the learning process.

(2) What are the optimum values for NP (number of presentations in an iteration) and NQ (number of input/output patterns in a presentation) given a fixed value of NR (number of input/output patterns in an iteration)? If any input/output pattern is allowed to appear in only one presentation, then

$$NR = NP \cdot NQ \quad (14)$$

It is obvious that we will have less presentations in an iteration if we use more patterns per presentation. In the present study, the convergence properties for various combinations of NP and NQ are investigated. We also examine the case in which an input/output pattern is allowed to appear in two presentation. In this case, we define

NS = the number of input/output patterns in the current presentation that will appear in next presentation

Then, we have

$$NR = NP \cdot NQ - (NP - 1) \cdot NS \quad (15)$$

The effect of different values of NS on convergence rate is also examined.

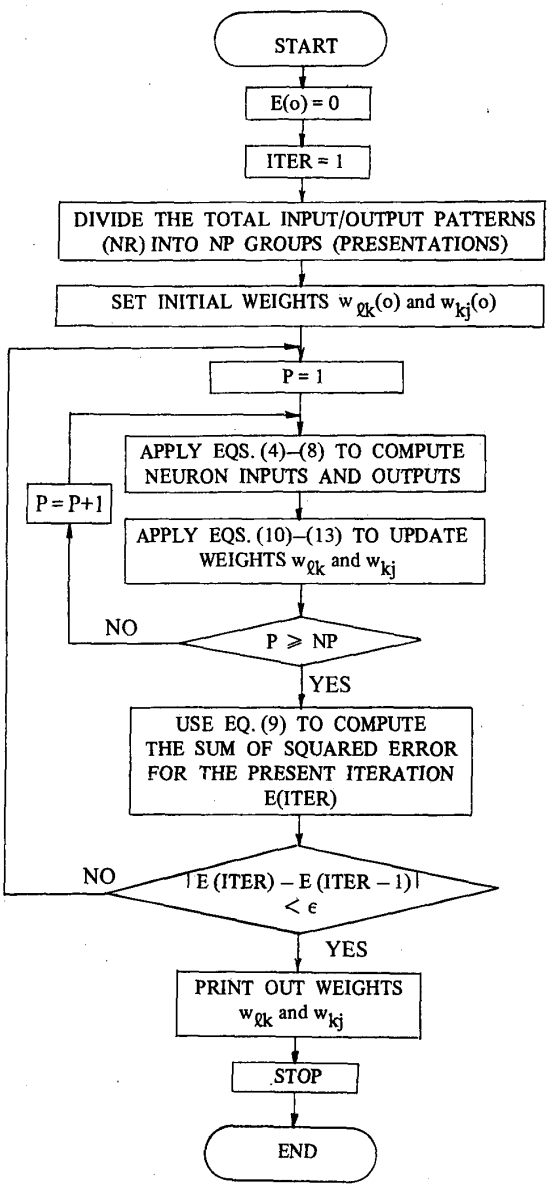

Fig. 2 The learning procedure

(3) In the learning procedure described in Fig. 2, convergence is checked at the end of each iteration when the connect weights have been updated NP times. It may be argued that convergence can also be checked after each presentation. Indeed, there is no reason why we can not check convergence after each presentation in order to save some presentations (or even iterations). However, we made the decision to check convergence only at the time when an iteration is completed because we found that the computation of the sum of squared error, E, is very time consuming and we do not want to spend a lot of CPU time in computing E. In other words, we believe that the required CPU time per iteration is equally important as the required number of iterations since the total required CPU time in the learning process is the product of the two figures. Thus, in presenting different learning algorithms in our load-forecast example, comparisons will be made based on the required CPU times as well as the number of iterations.

4. AN ADAPTIVE LEARNING ALGORITHM

To make the learning process converge more rapidly than the conventional method in which both the learning rate and momentum are kept constant during the learning process, a new adaptive training algorithm is developed to adapt momentum in the training process. The proposed adaptation rule is as follows.

$$\alpha(n+1) = \begin{cases} 1.01\ \alpha(n) & , \ \Delta E_n > 0 \\ 0.99\ \alpha(n) & , \ \text{otherwise} \end{cases} \tag{16}$$

where $\alpha(n)$ is momentum at iteration n, and $\Delta E_n = E(n-1) - E(n)$ with E(n) being the sum of squared errors at the end of nth iteration. Here, the basic idea is to increase α when ΔE_n is positive and decrease α when ΔE_n is negative. Note that ΔE_n is positive when the error is decreasing ($E(n) < E(n-1)$), which implies that the connection weights are updated to the correct direction. In this case, it is reasonable to maintain this update direction in next iteration. The way we achieve this is to increase the momentum in next iteration. On the other hand, if the connection weights are moved to the wrong direction, causing the error to increase, we should try to ignore this direction in next iteration by reducing the momentum term.

5. EXAMPLE

To demonstrate the effectiveness of the proposed adaptive learning algorithm, load forecasting is performed on Taiwan power system using the neural network with the conventional algorithm and the adaptive algorithm. Table 1 gives the forecast results for a typical day (August 19, 1987) using the rule-based expert system in [10]. It is observed that the MAE is 1.79% and the errors in peak load and valley load are 19.42 MW (1.78%) and 171.63 MW (1.57%), respectively.

To further improve the accuracy in the forecasted hourly loads, it is desirable to reduce the errors in peak load and valley load forecasting. In the present work, we propose to use neural networks to forecast peak and valley load.

The inputs to the neural net contain the forecasted high (low) temperatures in the three areas of Taiwan for the day load forecast is being conducted, the recorded area high (low) temperatures in the previous day, and the recorded area high (low) temperatures and peak (valley) load in the past ten days with the same load pattern as the forecast day. Thus, we have 46 input nodes in all. The neural network has 60 hidden units and one output nodes which gives the peak (valley) load.

Table 1. Load forecast results using the rule-based expert system (August 19, 1987)

HOUR	ACTUAL LOADS(MW)	FORECASTED LOADS(MW)	ERROR(MW)	% ERROR
1	7619.00	7524.28	94.72	8.66E-03
2	7326.00	7218.27	107.73	9.84E-03
3	7074.00	6988.41	85.59	7.82E-03
4	6895.00	6862.72	32.28	2.95E-03
5	6791.00	6683.49	107.51	9.82E-03
6	6735.00	6593.15	141.85	1.30E-02
7	6747.00	6563.37	183.63	1.68E-02
8	7395.00	7142.06	252.94	2.31E-02
9	9538.00	9235.29	302.71	2.77E-02
10	10201.00	9937.47	263.53	2.41E-02
11	10485.00	10267.33	217.67	1.99E-02
12	10616.00	10372.71	243.29	2.22E-02
13	9539.00	9279.03	259.97	2.38E-02
14	10823.00	10588.48	234.52	2.14E-02
15	10944.00	10749.54	194.42	1.78E-02
16	10785.00	10575.12	209.88	1.92E-02
17	10558.00	10373.14	184.86	1.69E-02
18	9715.00	9531.42	183.58	1.68E-02
19	9732.00	9479.65	252.35	2.31E-02
20	9977.00	9797.35	179.65	1.64E-02
21	9721.00	9560.17	160.83	1.47E-02
22	9352.00	9080.00	272.00	2.49E-02
23	8994.00	8752.46	241.54	2.21E-02
24	8522.00	8234.76	287.24	2.62E-02

** MAE = 1.79 % ** RMS = 1.90 %
** FORECASTED PEAK = 10749.54 MW
** ACTUAL PEAK = 10944 MW ERROR = 194.42 MW(1.78%)
** FORECASTED VALLEY = 6563.37 MW
** ACTUAL VALLEY = 6735 MW ERROR= 171.63 MW(1.57%)

In the training process, 30 input/output patterns selected from the data base of Taiwan Power Company are employed to determine the weights for the neural network. The convergence criterion used for training is to have a root mean square error (RMSE) of 0.01 or less ($\epsilon = 0.01$ in Fig. 2).

The efficiency and accuracy of the neural network using the conventional learning algorithm with fixed learning rate and momentum are compared in Tables 2 and 3 for different combinations of η, α, NS, NP, and NQ. Note that the learning efficiency is evaluated by the number of iterations and the required CPU time in the learning process. Once the network has been trained, the accuracy of the neural net can be evaluated by testing the neural net with 7 load forecasts and record the maximum error and root-mean-square error of the forecasts. Table 2 gives the results for the case in which each presentation uses different input/output patterns (NS = 0) while Table 3 gives the results for the case in which a certain number of input/output patterns in the current presentation will appear in next presentation (NS $\neq$ 0). We use a set of 30 input/output training patterns in the achieving the results in Table 2 and Table 3 (case 1). To see how the input/output patterns affect the final results, training is also performed using another set of 30 input/output patterns. The results for this case (case 2) are summarized in Tables 4 and 5. The results obtained by using the adaptive learning algorithm are given in Table 6.

Table 2. The results for case 1 using conventional learning algorithm (NS=0)

NQ	NP	$\eta = 0.25$ $\alpha = 0.9$				$\eta = 2.5$ $\alpha = 0.9$			
		learning efficiency		test results		learning efficiency		test results	
		number of iterations	CPU time (sec)	maximum error	root-mean-square error	number of iterations	CPU time (sec)	maximum error	root-mean-square error
1	30	566	9154	1.947%	1.405%	diverge			
5	6	429	3920	1.419%	0.835%	153	1420	1.197%	0.841%
10	3	888	7272	1.403%	0.827%	119	977	1.044%	0.533%
15	2	1376	10870	1.299%	0.745%	141	1116	1.063%	0.585%
30	1	2794	21303	1.307%	0.751%	292	2286	1.058%	0.564%

Table 3. The results for case 1 using conventional learning algorithm (NQ=10)

NS	NP	$\eta = 0.25$ $\alpha = 0.9$				$\eta = 2.5$ $\alpha = 0.9$			
		learning efficiency		test results		learning efficiency		test results	
		number of iterations	CPU time (sec)	maximum error	root-mean-square error	number of iterations	CPU time (sec)	maximum error	root-mean-square error
8	11	674	15906	0.586%	0.362%	33	784	0.810%	0.429%
6	6	477	6695	1.347%	0.751%	82	1151	0.377%	0.266%
5	5	572	6928	1.332%	0.760%	78	950	1.125%	0.675%
0	3	888	7272	1.403%	0.827%	119	977	1.044%	0.533%

Table 4. The results for case 2 using conventional learning algorithm (NS=0)

NQ	NP	$\eta = 0.25$ $\alpha = 0.9$				$\eta = 2.5$ $\alpha = 0.9$			
		learning efficiency		test results		learning efficiency		test results	
		number of iterations	CPU time (sec)	maximum error	root-mean-square error	number of iterations	CPU time (sec)	maximum error	root-mean-square error
1	30	126	2017	2.741%	1.214%	diverge			
5	6	493	4532	2.166%	1.004%	346	3154	1.931%	0.870%
10	3	997	8180	2.218%	1.008%	158	1329	1.779%	0.871%
15	2	1493	11790	2.196%	1.002%	363	2992	1.687%	0.818%
30	1	3007	23170	2.225%	1.008%	436	3415	1.958%	0.879%

Table 5. The results for case 2 using conventional learning algorithm (NQ=10)

NS	NP	$\eta = 0.25$ $\alpha = 0.9$				$\eta = 2.5$ $\alpha = 0.9$			
		learning efficiency		test results		learning efficiency		test results	
		number of iterations	CPU time (sec)	maximum error	root-mean-square error	number of iterations	CPU time (sec)	maximum error	root-mean-square error
8	11	292	6942	1.812%	1.126%	35	808	2.196%	0.926%
6	6	468	6570	2.068%	0.977%	362	5081	1.812%	0.885%
5	5	543	6575	2.198%	1.003%	165	1998	1.652%	0.809%
0	3	997	8180	2.218%	1.008%	158	1329	1.779%	0.871%

Table 6. Comparison of the conventional algorithm and the adaptive algorithm (NS=0, NQ=10, initial value for $\alpha = 0.9$)

algorithm	$\eta = 0.25$				$\eta = 2.5$			
	learning efficiency		test results		learning efficiency		test results	
	number of iterations	CPU time (sec)	maximum error	root-mean-square error	number of iterations	CPU time (sec)	maximum error	root-mean-square error
conventional	888	7272	1.403%	0.827%	119	977	1.044%	0.533%
adaptive	84	688	2.176%	1.519%	81	665	1.229%	0.608%

From the results in Tables 2-6, the following observations can be made:

(1) For the conventional learning algorithm with NS = 0 (Table 2) the choice of $\eta = 2.5$ and $\alpha = 0.9$ can yield good results in terms of training efficiency and accuracy. In the training process, the number of iterations and the required CPU time on a VAX 11/780 computer are in the order of 100 and 1000 seconds, respectively. However, both the number of iterations and CPU time will increase drastically (eight times as great) if the learning rate is reduced to 0.25.

(2) From the results in Table 2, it seems that, in our study, the combination of NP = 3 and NQ = 10 ($\eta = 2.5$) can achieve most efficient learning if each input/output pair is used in only one presentation.

(3) By comparing the results in Table 2 and Table 3, it can be concluded that more efficient learning can be achieved by allowing some input/output patterns to appear in more than one presentation. To be specific, the required CPU time can be reduced from 977 seconds in Table 2 to 784 seconds in Table 3. It is also noted the convergence rate will be affected by the number of input/output patterns in the current presentation that will appear in next presentation (NS).

(4) As evidenced by the results in Tables 2-3 and Tables 4-5, the previous observations also hold for case 2. Thus, it is concluded that the above observations remain valid regardless of the set of training patterns used. In fact, many other sets of input/output patterns and different values of η and α have been tested. The results are not presented due to limited space.

(5) Table 6 compares the convergence rates of the conventional learning algorithm and the adaptive learning algorithm. It is observed that both the required CPU time and the number of iterations can be significantly reduced by the proposed adaptive learning algorithm. Thus, the adaptive algorithm converges much faster than the conventional algorithm. One characteristic feature of the adaptive learning algorithm is that, no matter what value of η is used, the adaptive learning algorithm can make the learning process very efficient. For example, even a value of 0.25 is picked, the required CPU time can be reduced from 7272 seconds to 688 seconds by the adaptive algorithm. Thus, it is concluded that, with the conventional learning algorithm, the values of learning rate and momentum must be carefully selected in order to achieve efficient learning. However, if the adaptive learning algorithm is employed, the selection of learning rate and initial value for momentum will have only a minor effect on the convergence rate. Thus, we do not have to worry about the selection of η and α, if we use the adaptive algorithm.

(6) To see how the momentum is changed by the adaptive algorithm throughout the learning process, Fig. 3 depicts the value of α as a function of iteration number for the adaptive algorithm with $\eta = 2.5$.

(7) To examine the convergence characteristics of the conventional algorithm and the adaptive algorithm, the root mean square (RMS) errors in the learning process are depicted in Figs. 4-6. Fig. 4 shows the RMS errors for the conventional algorithm with NS $\neq$ 0 and $\eta = 2.5$. Figs. 5 and 6 compares the RMS errors obtained by using the conventional algorithm and the adaptive algorithm for $\eta = 0.25$ and $\eta = 2.5$, respectively. The excellent convergent characteristic of the adaptive algorithm can be easily observed.

(8) It was mentioned earlier that the error is computed only at the end of one iteration to save the computational time. Since there would be more than one presentation in an iteration and the connection weights are updated after each presentation, it is possible that the RMS error at one presentation within an iteration is smaller than that at the end of the iteration. Fig. 7 depicts the situation for the conventional algorithm

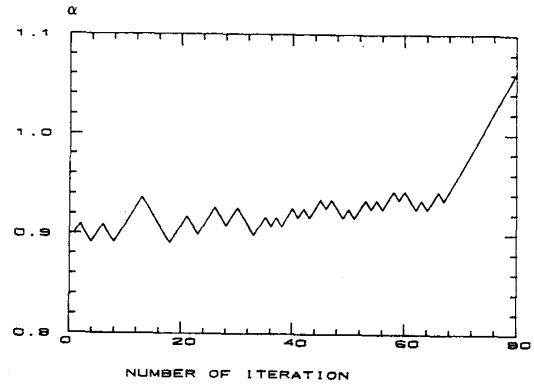

Fig. 3 The momentum as a function of iteration number for the adaptive algorithm ($\eta = 2.5$)

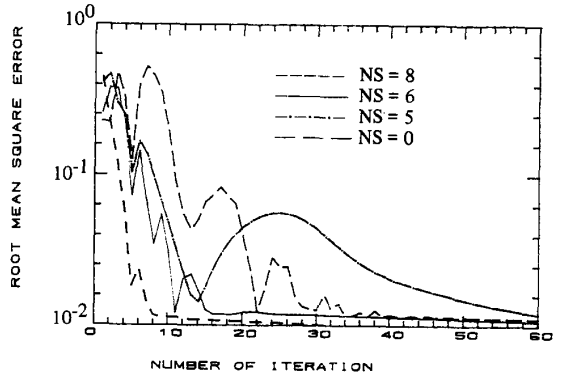

Fig. 4 The RMS error as a function of iteration number for the conventional algorithm (Table 3, NQ = 10, η = 2.5)

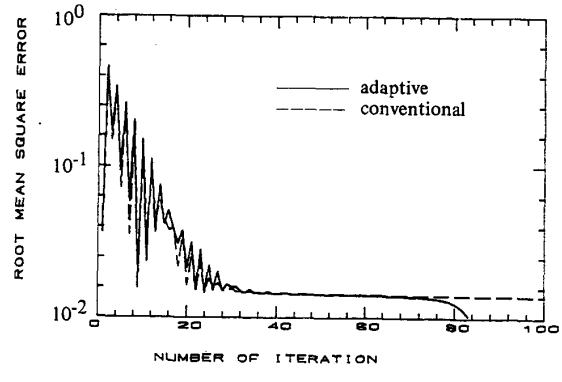

Fig. 5 Comparison of the RMS errors as a function of iteration number (η = 0.25)

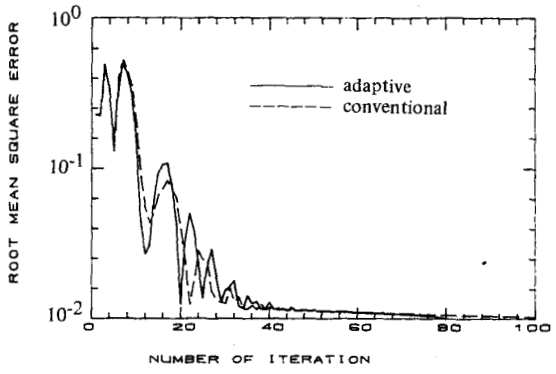

Fig. 6 Comparison of the RMS error as a function of iteration number ($\eta = 2.5$)

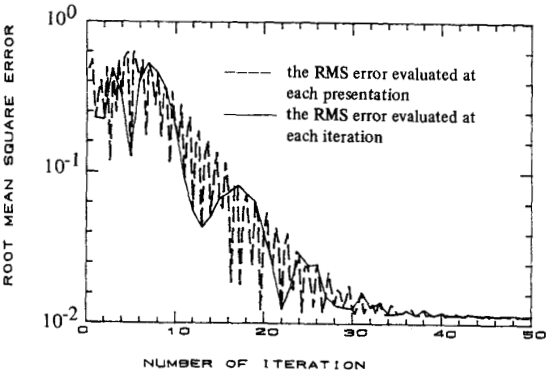

Fig. 7 The RMS errors at each presentation and each iteration

with NS = 0, NQ = 10, and $\eta = 2.5$ (Table 2). It is obvious that the required number of iterations can be reduced if we evaluate the error after each presentation. However, we do not employ this policy for the reason mentioned above.

(9) The connection weights are updated after each presentation. Fig. 8 shows how one of the connection weights is varied at each presentation for the conventional algorithm with NS = 0, NP = 3 and $\eta = 2.5$.

(10) Test results in Table 6 reveal that the neural network is capable of yielding very accurate results (with an RMS error of 0.608%) once it is trained by the proposed adaptive algorithm. The neural network trained by the conventional algorithm also gives very good accuracy (with an RMS error of 0.533%). It is not surprising that the two algorithms give different RMS errors since they will converge to different points in the weight space. However, the difference in the two RMS errors is insignificant.

(11) The initial values for the connection weights have been generated by the random number generator.

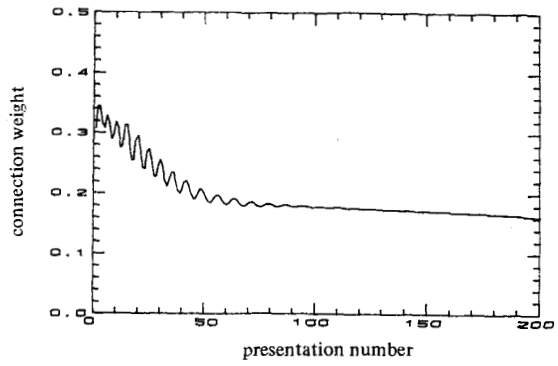

Fig. 8 The connection weight as a function of the presentation number

Now, the neural network has been trained and tested, it can be applied to forecast the required peak load. A separate neural network is also trained for valley load forecasting. Table 7 lists the forecasted peak load, valley load, and hourly loads. By comparing the results in Table 1 and Table 7, it is observed that the error in peak load can be reduced from 1.78% to 0.41% while the error in valley load is reduced from 1.57% to 0.56%. The resultant MAE for hourly load forecasting is reduced from 1.79% to 0.7%.

To see how the neural network performs in the other seasons of the year, short term load forecasting is also performed on other days of the year. Due to limited space, only the results for one day in spring (May 7, 1987) are presented in Table 8.

Table 7. Load forecast results using the neural network (August 19, 1987)

HOUR	ACTUAL LOADS(MW)	FORECASTED LOADS(MW)	ERROR(MW)	% ERROR
1	7619.00	7643.48	-24.48	-2.24E-03
2	7326.00	7334.56	-8.56	-7.82E-04
3	7074.00	7102.50	-28.50	-2.60E-03
4	6895.00	6975.60	-80.61	-7.37E-03
5	6791.00	6794.67	-3.67	-3.36E-04
6	6735.00	6703.47	31.53	2.88E-03
7	6747.00	6673.41	73.59	6.72E-03
8	7395.00	7257.61	137.39	1.26E-02
9	9538.00	9370.83	167.17	1.53E-02
10	10201.00	10079.71	121.29	1.11E-02
11	10485.00	10412.72	72.28	6.60E-03
12	10616.00	10519.12	96.88	8.85E-03
13	9539.00	9414.99	124.01	1.13E-02
14	10823.00	10736.94	86.06	7.86E-03
15	10944.00	10899.58	44.42	4.06E-03
16	10785.00	10723.46	61.54	5.62E-03
17	10558.00	10519.55	38.45	3.51E-03
18	9715.00	9669.79	45.21	4.13E-03
19	9732.00	9617.53	114.47	1.05E-02
20	9977.00	9938.25	38.75	3.54E-03
21	9721.00	9698.82	22.18	2.03E-03
22	9352.00	9214.06	137.94	1.26E-02
23	8994.00	8883.39	110.61	1.01E-02
24	8522.00	8360.75	161.25	1.47E-02

** MAE = 0.70 % ** RMS = 0.82 %
** FORECASTED PEAK = 10899.58 MW
** ACTUAL PEAK = 10944 MW ERROR = 44.42 MW (0.41%)
** FORECASTED VALLEY = 6673.41 MW
** ACTUAL VALLEY = 6735 MW ERROR = 61.59 MW(0.56%)

Table 8. Load forecast results for May 7, 1987

HOUR	ACTUAL	OPERATOR		RULE-BASED EXPERT SYSTEM		NEURAL NETWORK	
		FORECASTED LOAD(MW)	%ERROR	FORECASTED LOAD(MW)	%ERROR	FORECASTED LOAD(MW)	%ERROR
1	5916	5840	8.71E-03	5873.31	4.89E-03	5973.07	-6.54E-03
2	5758	5750	9.16E-04	5681.81	8.73E-03	5771.71	-1.57E-03
3	5634	5670	-4.12E-03	5570.97	7.22E-03	5655.15	-2.42E-03
4	5576	5650	-8.48E-03	5500.02	8.70E-03	5580.55	-5.21E-04
5	5555	5630	-8.59E-03	5496.06	6.75E-03	5576.38	-2.45E-03
6	5623	5760	-1.57E-02	5621.52	1.69E-04	5708.31	-9.77E-03
7	5780	5980	-2.29E-02	5773.06	7.95E-04	5867.66	-1.00E-02
8	6262	6420	-1.81E-02	6208.80	6.09E-03	6325.86	-7.32E-03
9	7884	8040	-1.79E-02	7752.51	1.51E-02	7949.14	-7.46E-03
10	8371	8420	-5.61E-03	8309.37	7.06E-03	8534.70	-1.98E-02
11	8730	8880	-1.72E-02	8593.86	1.56E-02	8833.86	-1.19E-02
12	8691	8500	2.19E-02	8549.35	1.62E-02	8787.05	-1.10E-02
13	7377	7600	-2.55E-02	7250.40	1.45E-02	7421.15	-5.06E-03
14	8479	8590	-1.27E-02	8330.74	1.70E-02	8557.18	-8.95E-03
15	8636	8570	7.56E-03	8496.61	1.60E-02	8731.60	-1.10E-02
16	8631	8610	2.41E-03	8486.61	1.65E-02	8721.08	-1.03E-02
17	8648	8500	1.70E-02	8448.90	2.28E-02	8681.43	-3.83E-03
18	8218	8130	1.01E-02	8021.18	2.25E-02	8231.66	-1.56E-03
19	8267	8400	-1.52E-02	8173.01	1.08E-02	8391.31	-1.42E-02
20	8249	8060	2.16E-02	8191.99	6.53E-03	8411.27	-1.86E-02
21	7915	7840	8.59E-03	7850.66	7.37E-03	8052.35	-1.57E-02
22	7364	7280	9.62E-03	7319.20	5.13E-03	7493.50	-1.48E-02
23	6917	7000	-9.51E-03	6889.32	3.17E-03	7041.46	-1.43E-02
24	6428	6400	3.21E-03	6411.34	1.91E-03	6538.84	-1.27E-02
PEAK	8730	8880	1.72%	8593.86	1.56%	8833.86	1.19%
VALLEY	5555	5630	0.86%	5496.06	0.68%	5576.38	0.25%
MAE	--	1.22%		1.01%		0.92%	
RMS	--	1.40%		1.19%		1.06%	

Table 9. Load forecast results for Feb. 25, 1987

OUR	ACTUAL LOADS(MW)	FORECASTED LOADS(MW)	ERROR(MW)	% ERROR
1	5507.00	5538.70	-31.70	-5.76E-03
2	5366.00	5365.80	0.20	3.73E-05
3	5255.00	5262.10	-7.10	-1.35E-03
4	5196.00	5197.20	-1.20	-2.31E-04
5	5198.00	5207.80	-9.80	-1.89E-03
6	5339.00	5330.10	8.90	1.67E-03
7	5615.00	5543.20	71.80	1.28E-02
8	6028.00	5940.10	87.90	1.46E-02
9	7515.00	7446.40	68.60	9.13E-03
10	7924.00	7849.90	74.10	9.35E-03
11	8148.00	8090.40	57.60	7.07E-03
12	8065.00	8092.40	-27.40	-3.40E-03
13	6712.00	6782.70	-70.70	-1.05E-02
14	7794.00	7761.60	32.40	4.16E-03
15	7891.00	7910.70	-19.70	-2.50E-03
16	7864.00	7902.70	-38.70	-4.92E-03
17	7927.00	7986.70	-59.70	-7.53E-03
18	7613.00	7655.00	-42.00	-5.52E-03
19	7697.00	7865.80	-168.80	-2.19E-02
20	7506.00	7645.70	-139.70	-1.86E-02
21	7218.00	7326.60	-108.60	-1.50E-02
22	6760.00	6842.90	-82.90	-1.23E-02
23	6329.00	6460.70	-131.70	-2.08E-02
24	5838.00	6003.00	-165.00	-2.83E-02

** MAE= 0.91 % ** RMS error 1.18 %
** FORECASTED PEAK = 8092.4
** ACTUAL PEAK = 8065 MW ERROR = 27.4MW (0.34%)
** FORECASTED VALLEY = 5197.2 MW
** ACTUAL VALLEY = 5196.0 MW ERROR =1.2MW(0.02%)

Table 10. Load forecast results for Nov. 27, 1987

OUR	ACTUAL LOADS(MW)	FORECASTED LOADS(MW)	ERROR(MW)	% ERROR
1	6094.00	6152.90	-58.90	-9.67E-03
2	5934.00	5926.50	7.50	1.26E-03
3	5794.00	5778.30	15.70	2.71E-03
4	5667.00	5681.60	-14.60	-2.58E-03
5	5690.00	5681.00	9.00	1.58E-03
6	5817.00	5808.50	8.50	1.46E-03
7	6021.00	5957.10	63.90	1.06E-02
8	6469.00	6417.80	51.20	7.91E-03
9	8174.00	8221.10	-47.10	-5.76E-03
10	8773.00	8792.80	-19.80	-2.26E-03
11	9067.00	9168.50	-101.50	-1.12E-02
12	9038.00	9172.30	-134.30	-1.49E-02
13	7724.00	7833.60	-109.60	-1.42E-02
14	8876.00	8961.30	-85.30	-9.61E-03
15	9037.00	9120.00	-83.00	-9.18E-03
16	9065.00	9096.90	-31.90	-3.52E-03
17	9167.00	9120.00	47.00	5.13E-03
18	9057.00	8859.10	197.90	2.19E-02
19	8860.00	8884.20	-24.20	-2.73E-03
20	8597.00	8573.30	23.70	2.76E-03
21	8189.00	8196.40	-7.40	-9.04E-04
22	7616.00	7629.30	-3.30	-1.75E-03
23	7126.00	7168.00	-42.00	-5.89E-03
24	6542.00	6641.90	-99.90	-1.53E-02

** MAE= 0.69 % ** RMS error 0.88 %
** FORECASTED PEAK = 9172.3
** ACTUAL PEAK = 9167 MW ERROR = 5.3 MW (0.05%)
** FORECASTED VALLEY = 5681.0 MW
** ACTUAL VALLEY = 5667.0 MW ERROR = 14MW(0.2%)

From the results in Table 8, it is observed that, by using the artificial neural network, the error in peak load is reduced from 1.56% to 1.19% while the error in valley load is reduced from 0.68% to 0.25%. The resultant MAE for hourly load forecasting is reduced from 1.01% to 0.92%.

By comparing the effectiveness of the neural network in different seasons (see Tables 7–10), it is found that the improvement in peak load and valley load forecast is more pronounced in summer than in spring. This is due to the fact that, in a summer peak system such as Taiwan power system, peak load is more dependent on temperature in summer than in other seasons. The neural network improves the forecasted peak load by simulating the complicated relationship between temperature and peak load using the neurons and their associated weights.

It is also observed from Table 8 that the neural network yields slightly better forecast results than the system operator. Moreover, the neural network approach gives smaller forecast errors than the rule-based expert system by using a more sophisticated algorithm to reduce the error in both the forecasted peak load and the forecasted valley load. Note that the two approaches use the same hourly load pattern, $L_n(i)$, and it is the peak load, L_p, and the valley load, L_t, that causes the difference in forecast error.

6. NOMENCLATURE

subscripts j, k, ℓ = any node in the input layer, hidden layer, and output layer, respectively.
NJ, NK, NL = number of nodes in the input layer, hidden layer, and output layer, respectively.
NP = number of presentations in an iteration
NQ = number of input/output patterns in a presentation
NR = number of input/output patterns in an iteration
NS = number of input/output patterns in the current presentation that will appear in next presentation
θ = threshold
w = connection weight between neurons
t = target (desired) output value
o = computed output value using the neural net
E = sum of squared error
η = learning rate
α = momentum

7. CONCLUSIONS

A multilayer feedforward neural network is proposed for short-term load forecasting. The effect of learning rate and momentum on the efficiency of conventional learning algorithm, in which the learning rate and momentum are both fixed, is extensively studied. It is found that the convergence rate of the conventional algorithm is significantly affected by the learning rate and momentum. To speed up the learning process, an adaptive learning algorithm, in wich the momentum is adapted in the learning process, is developed. It is found that the adaptive algorithm converges much faster than the conventional algorithm. Moreover, the convergence property of the adaptive learning algorithm will not be affected by the learning rate and initial value of momentum. This makes the adaptive algorithm more convenient than the conventional algorithm. The proposed neural network with the adaptive algorithm has been applied to load forecasting of Taiwan power system. It is found that accurate load forecasting results can be achieved by the neural network in a very efficient way.

8. ACKNOWLEDGEMENTS

The authors would like to thank the reviewers for their valuable comments. Sincere gratitude is also given to Messrs. C.C. Liang, K.K. Chen, T.S. Lai, and B.S. Chang of the System Operation Department of Taiwan Power Company for providing the valuable system data. Financial support given to this work by the National Science Council of ROC under contract number NSC–78–0404–E002–30 is appreciated.

9. REFERENCES

[1] G. Gross and F.D. Galiana, "Short term load forecasting," Proc. IEEE, Vol. 75, No. 12, pp. 1558-1573, 1987.

[2] W.R. Christianse, "Short-term load forecasting using general exponential smoothing," IEEE Trans. PAS, Vol. 90, pp. 900-911, 1971.

[3] J. Davey, J.J. Soachs, G.W. Cunningham and K.W. Priest, "Practical application of weather sensitive load forecasting to system planning," IEEE Trans. PAS, Vol. 91, pp. 917-977, 1972.

[4] R.P. Thompson, "Weather sensitive demand and energy analysis on a large geographically diverse power system-application to short-term hourly electric demand forecasting," IEEE Trans. PAS, Vol. 95, pp. 384-393, 1976.

[5] A.D. Papalexopoulos and T.C. Hesterberg, "A regression-based approach to short-term system load forecasting," 1989 PICA Conference, pp. 414-423.

[6] F. Meslier, "New advances in short term load forecasting using Box and Jenkins Approaches, "Paper A78 051-5, presented at the IEEE/PES 1978 Winter Meeting.

[7] G.D. Irisarri, S.E. Widergren, and P.D. Yehsakul, "On-line load forecasting for energy control center application," IEEE Trans. PAS, Vol. 101, pp. 71-78, 1982.

[8] S. Ranman and R. Bhatnagar, "An expert system based algorithm for short term load forecast," IEEE Trans. PWRS, Vol. 3, pp. 392-399, 1988.

[9] K. Jabbour, J.F.V. Riveros, D. Landsbergen, and W. Meyer, "ALFA: automated load forecasting assistant," IEEE Trans. PWRS, Vol. 3, pp. 908-914, 1988.

[10] K.L. Ho, Y.Y. Hsu, C.F. Chen, T.E. Lee, C.C. Liang, T.S. Lai, and K.K. Chen, "Short term load forecasting of Taiwan power system using a knowledge-based expert system," Paper 90 WM 259-2 PWRS, presented at the IEEE/PES 1990 Winter Meeting.

[11] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning internal representations by error propagation," in D.E. Rumelhart and J.L. McClelland, Parallel Distributed Processing, Vol. 1, Ch. 8.

[12] D.J. Sobajic and Y.H. Pao, "Artificial neural-net based dynamic security assessment for electric power systems," IEEE Trans. PWRS, Vol. 4, pp. 220-228, 1989.

[13] M. Aggoune, M.A. El-Sharkawi, D.C. Park, M.J. Damborg, and R.J. Marks II, "Preliminary results on using artificial neural networks for security assessment," 1989 PICA Conference, pp. 252-258.

[14] M.I. Santoso and O.T. Tan, "Neural-net based real-time control of capacitors installed on distribution systems," Paper 89 SM 768-3 PWRD, presented at the IEEE/PES 1989 Summer Meeting.

[15] E.H.P. Chan, "Application of neural-network computing in intelligent alarm processing," 1989 PICA Conference, pp. 246-251.

[16] R. Fischl, M. Kam, J.C. Chow and S. Ricciardi, "Screening power system contingencies using a back-propagation trained multiperception," Proc. 1989 International Symposium on Circuits and Systems, pp. 486-489.

[17] M.E. Aggoune, L.E. Atlas, D.A. Cohn, M.J. Damborg, M.A. El-Sharkawi, and R.J. Marks II, "Artificial neural networks for power system static security assessment," Proc. 1989 International Symposium on circuits and Systems, pp. 490-494.

[18] R.P. Lippmann, "An introduction to computing with neural nets," IEEE ASSP Magazine, pp. 4-22, 1987.

Kun-Long Ho was born in 1964. He received his B.S. degree in electrical engineering from Chung-Yuan University, Chungli, Taiwan. He has been working toward his Ph.D. degree in the Electrical Engineering Department of National Taiwan University, Taipei, Taiwan.

At present, his research interests include security control, load forecasting, and the application of expert systems to power system problems.

Yuan-Yih Hsu was born in Taiwan on June 19, 1955. He received his B. Sc., M. Sc., and Ph.D. degrees, all in electrical engineering from National Taiwan University, Taipei, Taiwan.

Since 1977, he has been with National Taiwan University, where he is now a professor. He worked at the University of Calgary, Alberta, Canada, as a postodctoral research fellow and instructor from 1982 to 1983. From 1988 to 1989, he was a visiting scholar at the University of California, Berkeley.

At present, his research interests include power system dynamics, stability analysis, reliability analysis and the application of expert systems to power system problems.

He is a senior member of IEEE.

Chien-Chuen Yang was born in 1962. He received his B.Sc. degree in electrical engineering from Chung-Yuan University, Chungli, Taiwan. His M. Sc. degree in electrical engineering was received from National Taiwan University, Taipei, Taiwan in 1990. He has been working toward his Ph.D. degree in the Electrical Engineering Department of National Taiwan University.

At present, his research interest include security control, load forecasting, and the application of expert systems to power system problems.