# A neural network short-term load forecaster

Dipti Srinivasan, A. C. Liew and C. S. Chang
*Department of Electrical Engineering, National University of Singapore, 10 Kent Ridge Crescent, Singapore 0511 (Singapore)*

(Accepted October 5, 1993)

**Abstract**

This paper presents a neural network based approach to short-term load forecasting, which plays an important role in the day to day operation and scheduling of power systems. A four-layer feedforward neural network, trained by a back-propagation learning algorithm, has been applied for forecasting the hourly load of a power system. In this paper, the performance of the network is compared with some carefully chosen experimental methods. This new approach promises to provide results unobtainable with more traditional time series methods. It is shown that, with careful network design, the back-propagation learning procedure is an effective way of training neural networks for electrical load prediction. The choice of transfer function is an important design issue in achieving fast convergence and good generalization performance. The network is trained on real data from a power system and evaluated for short-term forecasting with hourly feedback. The network learns the training set nearly perfectly and shows accurate prediction with 1.07% error on weekdays and 1.80% error on weekends.

*Key words*: Short-term load forecasting; Neural network

## 1. Introduction

Accurate forecasting of electrical load is required in order to balance the supply and demand for electricity. Short-term forecasting (forecasts of half an hour to one day ahead) is required for unit commitment, energy transactions, system security analysis and optimum operation planning of power generation facilities; medium-term forecasting (one day to several months ahead) deals with the scheduling of fuel supplies and maintenance operations; and long-term forecasting (more than a year ahead) is useful for planning operations.

Most of the conventional load forecasting techniques [1–4] can be categorized under two approaches. Traditional prediction techniques treat the problem of prediction as a special type of parameter estimation problem where the estimated model is regarded as the predictor. This method necessitates the identification of relevant variables with strong correlation to electrical loads such as temperature, humidity, and winds, etc. The time series method of load forecasting involves the examination of historical data, extracting essential data characteristics, and effectively projecting these characteristics into the future. These statistical methods of load forecasting fail to give reasonably accurate forecasts because of their inherent limitations. The former is inefficient due to its dependence on the functional relationship between the load and the weather variables, and the latter is numerically unstable. The need to deal with increasingly complex nonlinear systems in a better way has led to a reevaluation of the conventional load forecasting methods, and investigation into new approaches [5–8]. The emerging multilayer neural networks with their massive parallelism and capability to approximate arbitrary nonlinear functions show great potential for solving nonlinear prediction problems.

This paper is inspired by previous proposals for applying a neural network approach [6, 9] to the problem of load forecasting and considers a novel architecture in which neural networks trained with the generalized delta rule [10] perform approximations of functions for predicting future load based on historical load data and weather variables. The work described in this paper is an extension of that reported by the authors in ref. 6, where a back-propagation neural network has been successfully applied for medium-term load forecasting. Parallel work is reported in ref. 9, where an attempt has been made to apply neural networks for short-term forecasting on weekdays using an adaptively trained neural network [11]. The neural network forecaster described in the present paper has been used for short-term forecasting on weekdays, as well as on weekends and public holidays. In addition, it has been shown that, with proper selection of parameters, the difficulty mentioned in ref. 9 encountered with the generalized delta rule [10] can be overcome.

## 2. Electrical load

### *2.1. Features*

Electrical load has a time varying nature and is affected by various factors, such as the kind of day, meteorological conditions and economic conditions. For short-term forecasting, the most important factors are the day of the week, temperature, humidity, seasonal effects, special days and occasions, and so on. The factors which affect the long-term forecasting include the state of the economy of the region, political aspects, industrial growth, etc. In addition to these, the electrical load contains trend components which can be weekly, monthly, or seasonal. For accurate forecasting, it is important to identify those variables with a strong correlation with the electrical load.

### *2.2. Load classification*

The shape of the load curve follows daily and weekly cycles, with small random variations due to changes in industrial activities and climatic conditions. The aim of this study was to forecast the hourly load for the following day. The load values for a day are functions of the short-term historical data and forecasted average values of weather parameters (forecasted high and low temperatures, and percentage humidity). Any abnormalities in the forecasted values can be explained by the occurrence of special events. The peak load reduces considerably before and after major public holidays. The historical data were grouped according to the type of day as follows.

(1) *Monday.* The load on Monday is different from that on other weekdays due to pick-up loads in the morning when all businesses and industries just start work.

(2) *Weekday — Tuesday, Wednesday and Thursday.* The load pattern on these days, when work is already in full swing, remains constant.

(3) *Friday.* The maximum load on Fridays is slightly lower than that of weekdays, because of its proximity to the weekend when most businesses are closed.

(4) *Saturday.* The load pattern on Saturdays is different from that on the rest of the weekdays. The peak load also takes a dip on Saturday, which is a rest day for some and a half-day for others.

(5) *Sunday/public holiday.* The shape of the load curve on Sundays is similar to that on holidays. The peak load is also much lower on these days, which are the rest days for everybody.

## 3. Multilayer neural networks

Artificial neural networks (ANNs) were first developed as structural or functional modeling systems of natural ones, featuring the ability to perform problem solving tasks. They can be thought as computing arrays consisting of series of repetitive uniform processors (neuron-like elements) placed on a grid. Learning is achieved by changing the interconnections between these processing elements.

ANNs have become the subject of widespread interest: they offer an unusual scheme-based programming standpoint and exhibit higher computing speeds than the conventional von Neumann architectures, thus easing or even enabling the handling of complex tasks, such as artificial vision, speech recognition, information recovery in noisy environments, or general pattern recognition.

In ANN systems, collective information management is achieved by means of the parallel operation of neuron-like elements, in which information processing is distributed. It is intended to exploit this highly parallel processing capability as far as possible in complex problem solving tasks.

Neural networks are characterized by their topology, that is, by the number of interconnections, the node characteristics that are classified by the type of nonlinear elements used, and the kind of learning rules implemented. A multilayer neural network is a layered network consisting of an input layer, an output layer, and at least one layer of nonlinear processing elements. The nonlinear processing elements, which sum the incoming signals and generate output signals according to some predefined functions, are called neurons or nodes.

## 3.1. *Back-propagation learning algorithm*

An error back-propagation (BP) neural network [10], which uses superpositions of sigmoidal functions to approximate a desired function [12], has been used for this load forecasting problem. In this paper, a sigmoidal transfer function is used by hidden layer neurons while the node characteristics in the output layer are governed by the linear function $s(x) = x$. The neurons in the adjacent layers are connected by terms with variable weights but no connection is made between any two neurons in the same layer. Based on solving least-square optimization problems with gradient descent, back-propagation neural network training intends to teach $N$ input–output pairs $\{(\boldsymbol{A}_p, \boldsymbol{B}_p)|p = 1, \ldots, N\}$, where $\boldsymbol{A}_p$ is a vector in the input field of the network and $\boldsymbol{B}_p = [t_{p1}, \ldots, t_{pn}]$ is a vector in the output field of the network, such that if $\boldsymbol{A}_p$ appears in the input field $\boldsymbol{B}_p$ is desired in the output field. $\boldsymbol{O}_{pf} = [o_{pf1}, \ldots, o_{pfn}]$ is defined to be the generated output which is the set of outputs of the neurons in the output (final) layer when $\boldsymbol{A}_p$ appears in the input field. The processing function $f$ of each neuron is differentiable and nondecreasing. During the forward pass, a pattern is presented to the input layer of the network, after which the network produces an output pattern. The output of the neuron $j$ (except those in the output layer) is

$$o_{pj} = f(\mathrm{net}_{pj}) \tag{1}$$

where

$$\mathrm{net}_{pj} = \sum_i w_{ji} o_{pi} \tag{2}$$

and $w_{ji}$ is the weight of the synapse between neuron $j$ and $i$, and neuron $i$ is in the layer which is one layer ahead of the layer in which neuron $j$ is situated. The processing function of the input layer is linear with unit gain. The transfer function $f$ used by the hidden neurons is given by $1/[1 + \exp(-\mathrm{net}_{pi})]$.

During training, the actual patterns of the network in the final layer are compared with their target values for the given pattern. The aim is to minimize the square of the difference between the desired output (target), $\boldsymbol{B}_p$, and the actual generalized output in the final layer, $\boldsymbol{O}_{pf}$. Let

$$E_p = \frac{1}{2} \sum_j (t_{pj} - o_{pfj})^2 \tag{3}$$

where $t_{pj}$ is the desired output of neuron $j$ for pair $p$ and $o_{pfj}$ is the generated output of neuron $j$ for pair $p$ in the output layer. Minimizing $E_p$ is the same as minimizing the absolute value of the output difference for pair $p$.

If we minimize eqn. (3), we can only bring $\boldsymbol{O}_{pf}$ closer to $\boldsymbol{B}_p$. However, changing the weights for the next pair may affect what has been achieved by the previous pairs. Since we are interested in all pairs, we want to minimize

$$E = \frac{1}{2} \sum_p \sum_j (t_{pj} - o_{pfj})^2 = \sum_p E_p \tag{4}$$

For every weight, starting at the weight put into the output units, working its way to the weight from the input nodes, a 'gradient' can be calculated, which indicates how a change in weight value affects $E$. For all these weights, the gradients are summed over all input–output patterns, and, according to the resulting gradient, the weights are modified. By using a gradient descent with respect to weight $w_{ji}$, we have

$$\dot{w}_{ji} = -\partial E / \partial w_{ji} \tag{5}$$

The gradient is related to the output by

$$\partial E / \partial w_{ji} = -\sum_p \delta_{pj} o_{pi} \tag{6}$$

Here, $\delta_{pj}$ represents the effect of a change in the net input to unit $j$ on the output of unit $i$ in pattern $p$.

The determination of $\delta$ is a recursive process that starts with the output units. If a unit is an output unit, its $\delta$ is given by

$$\delta_{pj} = (t_{pj} - o_{pfj}) f_j'(\mathrm{net}_{pfj}) \tag{7}$$

The delta term for the hidden neurons which have no specified target is determined recursively as a function of the delta terms of the neurons to which they are directly connected and their connection weights, and is given by

$$\delta_{pj} = f_j'(\mathrm{net}_{pj}) \sum_k \delta_{pk} w_{kj} \tag{8}$$

where $t_{pi}$ is the target activation, $k$ indicates the neurons in the layer next to the current layer and the derivative

$$f_j'(\mathrm{net}_{pj}) = \mathrm{d}f_j / \mathrm{d}(\mathrm{net}_{pj}) \tag{9}$$

Given $\partial E / \partial w_{ji}$ as in eqn. (6), the link weights are modified by an amount proportional to the sum of the products of the delta term (available to the neuron receiving the input) and the activation of the neuron sending the output according to the equation

$$\Delta w_{ji}(n) = \eta \sum_p \delta_{pj} o_{pi} \tag{10}$$

where index $n$ labels the iteration in the learning process and $\eta$ is the step size or learning rate.

The network is trained, that is, the interconnection weights are adjusted, until the error of the network is within some acceptable tolerance.

Rumelhart *et al.* [10] have proposed the use of a momentum term to improve the training of a BP network. It involves a slight change in the weight update rule in eqn. (10):

$$\Delta w_{ji}(n) = \eta \sum_{p} \delta_{pj} o_{pi} + \alpha \, \Delta w_{ji}(n-1) \qquad (11)$$

Here, $\alpha$ is usually taken to be $0 < \alpha < 1$, and is called the momentum term. The effect of the momentum term is to magnify the learning rate for 'flat' regions of weight space where the gradient is more or less constant. In steep regions of weight space, the momentum term focuses the movement in a downhill direction by dampening oscillations caused by altering the sign of the gradient. It provides a kind of momentum that effectively filters out the high frequency variations of the error surface in the weight space [13].

## 4. Development of the neural network based forecaster

### 4.1. Network architecture

The neural networks developed for the load forecasting were all four-layer feedforward networks. The input layer consisted of 14 neurons fed with important causal factors which have direct influence on the future load values, such as the weather conditions (the maximum and minimum values of temperature and humidity for the past 24 hours), the social condition indicator, day of the week, records of historical load data, and the hour of the forecast. The number of output neurons is determined by the number of estimated parameters, therefore only one neuron was used in the output layer to give the next hour forecast. The standard back-propagation algorithm with momentum (described earlier in this paper), as proposed by Rumelhart *et al.* [10], was used to train the networks. The gradient descent method was used to search for the optimal setting of the weights. Application of the delta learning rule eliminates the problem of the structured presentation of the training set by accumulating the weight changes over several training presentations and making the application all at once.

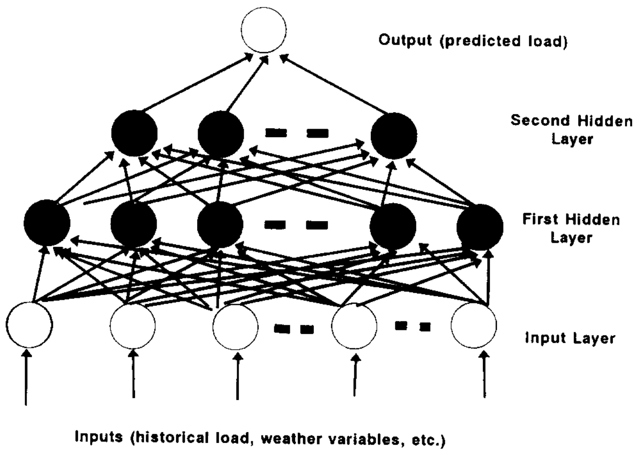

Fig. 1. Neural network structure for hourly load forecasting.

### 4.2. Number of hidden layers

Although one hidden layer is capable of modeling any function [14], two hidden layers usually result in a compact architecture and efficiency in learning with a back-propagation neural network. The network architecture used for the present load forecasting problem contains two hidden layers, as shown in Fig. 1. Complete connections are enforced between nodes in adjacent layers.

### 4.3. Number of hidden neurons

The problem of determining the optimal number of hidden neurons is a crucial one. The number of hidden neurons must be sufficiently large to realize a certain function. On the other hand, the time to train a network with an excessively large number of hidden neurons and connecting weights tends to become long and the network may become unstable and unreliable. In terms of the algebraic projection analysis [15], the solution may wander in the weight space where there are either too few hidden neurons (overdetermined case) or too many hidden neurons (undetermined case). Several structures were considered with different numbers of hidden neurons to determine the best configuration. The best results were achieved with a network with 19 neurons in the first hidden layer and 6 neurons in the second hidden layer.

### 4.4. Transfer function

Back-propagation requires that the transfer function, which determines the input–output relation of each node, should be differentiable at all points. During the learning procedure, various symmetric and asymmetric transfer functions were evaluated for use in this forecasting network. Symmetric functions yield faster convergence and better generalization performance. The best performance was achieved with the following sigmoidal transfer function for the hidden neurons:

$$y_i = 1/[1 + \exp(-\beta x_i)] \qquad (12)$$

where $x_i$ is the input and $y_i$ is the output for the $i$th neuron in the network hidden layer. Here, $\beta$ affects the steepness of the curve. High values of $\beta$ give a steplike curve and lower $\beta$ give a smoother curve.

### *4.5. Learning rate and the momentum rate*

For the load forecasting network, different learning rates, $\eta$, and momentum rates, $\alpha$, were used between consecutive layers to achieve the most rapid learning. Experiments showed that using a lower learning rate for the layers near the output node than for those near the input node gave a better learning capability. Further improvement in the learning capability was achieved by using a lower learning rate when the system reached a state where its total mean square error was decreasing at a very slow rate. At the start of training the values of the learning and momentum rates were fixed at $\eta = 0.9$ and $\alpha = 0.6$, and were gradually reduced to $\eta = 0.3$ and $\alpha = 0.1$ towards the end of the training. This gradual reduction in learning and momentum rates enabled the network to learn in small and delicate steps when the generated outputs were very close to the targets.

## 5. Implementation and results

The load data used for training and testing this neural network were obtained from the Public Utilities Board of Singapore. The network was trained on load data from year 1985 to year 1988. The main forecasting periods of this project were the first three weeks of year 1989. Five different neural networks were developed for forecasting hourly load for the following day:

(1) LFC1 for Sundays and public holidays,
(2) LFC2 for use on Mondays,
(3) LFC3 for weekdays (Tuesdays, Wednesdays and Thursdays),
(4) LFC4 for Fridays, and
(5) LFC5 for forecasting loads on Saturdays.

### *5.1. Training set*

Once the network architecture has been selected, the training set and training time determine the mapping represented by the network and its forecasting accuracy. For training this load forecasting network, the load and weather parameters were obtained from the SCADA system database. The future value of the electrical load was represented as a function of these parameters. The typical data were divided into two main types: the target field and the factor field. The target field represented the desired output the neural network was eventually required to generate, whereas the factor field was the set of factors that affect the target load value. The typical inputs for training this forecasting network consisted of the load data for the past 4 hours, the historical load data for the same day of the past three consecutive weeks, the hour of the forecast, temperature information (maximum and minimum temperature for the day, and forecasted temperature for the hour of forecast), and maximum and minimum humidity.

### *5.2. Data normalization*

In predicting the future values of the electrical load, the range of one input variable may be drastically different from that of the others. Higher valued predictor variables may tend to suppress the influence of smaller ones. To overcome this problem, this network was trained with normalized input data, leaving it to the network to learn weights associated with the connections emanating from these inputs. The raw data were scaled in the range 0.10–0.90 for use by the neural network to minimize the effects of magnitude between inputs. In this case each input or output parameter $x$ was normalized as $x_n$ before being applied to the neural network according to

$$x_n = (0.90 - 0.10)(x - x_{\min})/(x_{\max} - x_{\min}) \qquad (13)$$

where $x_{\max}$ and $x_{\min}$ are the maximum and minimum values of data parameter $x$.

### *5.3. Results*

Real data from the SCADA system installed in the Public Utilities Board of Singapore were used to test this method of short-term forecasting using an artificial neural network. The SCADA system provides the hourly weather data from the remote terminal units and the system load. The neural networks were trained on this data consisting of load and weather variables for the first three weeks of four consecutive years (i.e. 84 days). The neural network was then used to forecast the load for the first 21 days of the fifth year. A four-layer configuration with 14 input neurons, 19 neurons in the first hidden layer, 6 neurons in the second hidden layer and one output neuron gave the quickest convergence after about 150 iterations of the training samples.

The load forecasted by the neural network was then compared with the actual load, and the error was calculated. The principal statistic used to evaluate the performance of these models, mean absolute percentage error, is defined as

$$\text{Mean abs. percentage error} = \frac{1}{N}\sum_{i=1}^{N} + \frac{|\text{actual}_i - \text{forecast}_i|}{\text{actual}_i} \times 100 \qquad (14)$$

The value of $N$ was chosen as $N = 24$.

Figures 2–7 show the performance of the neural network forecaster in providing the load forecasts on a public holiday (New Year's Day), Sunday, Monday, a weekday, Friday, and Saturday, respectively.

The results of hourly load forecasting are summarized in Table 1, and compared with the results obtained from two popular methods, the multiple linear regression method and the autoregressive moving average (ARMA) model (see ref. 16 for a detailed description of these methods), for the same set of data. Both of these mathematical methods take into account the weather information, load patterns in the past, and random effects. It should be noted here that the neural network forecaster provides very accurate results, with an error of 1.07% on weekdays, 1.80% on Sundays and 2.09% on public holidays.

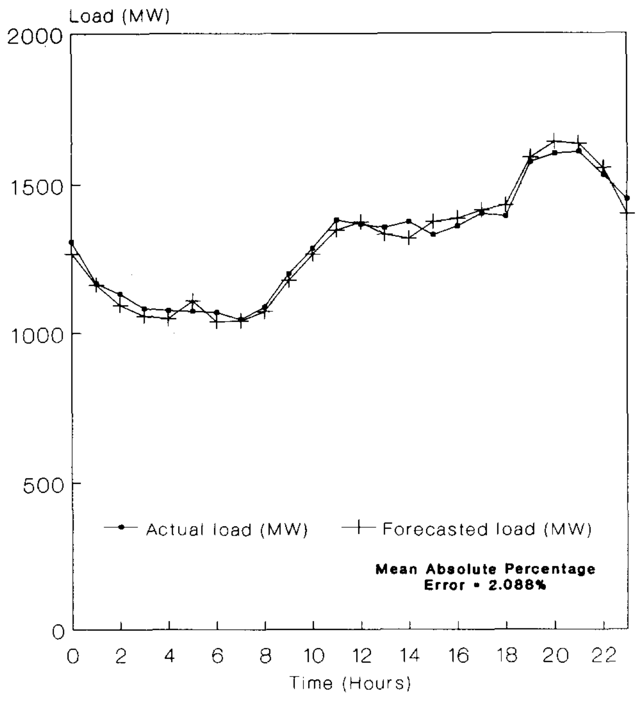

Fig. 2. Forecasting results of the model LFC1 for a public holiday (New Year's day).

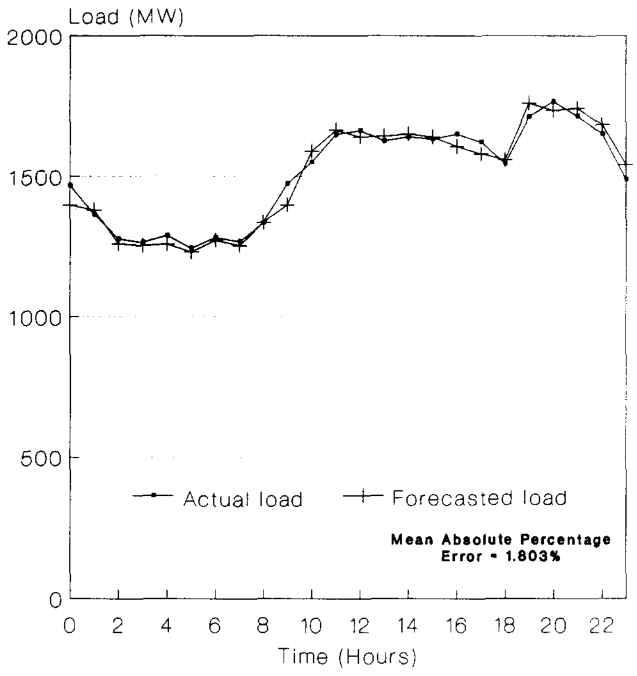

Fig. 3. Forecasting results of the model LFC1 for a Sunday.

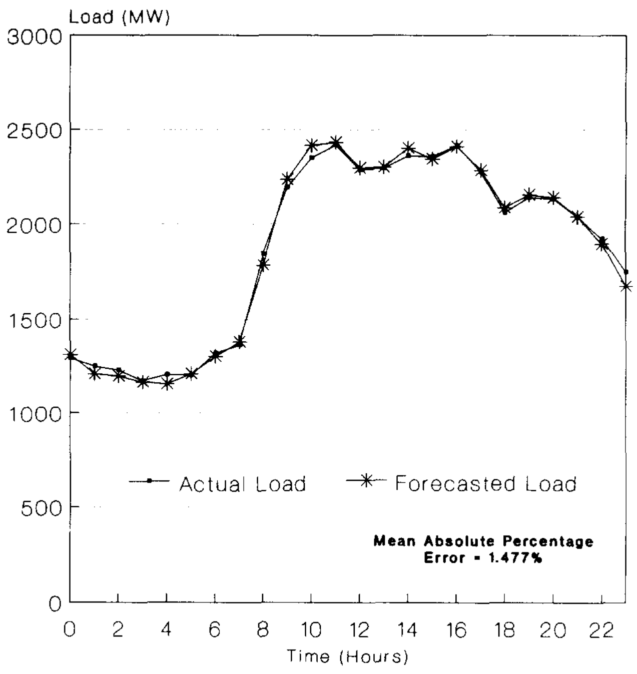

Fig. 4. Forecasting results of the model LFC2 for a Monday.

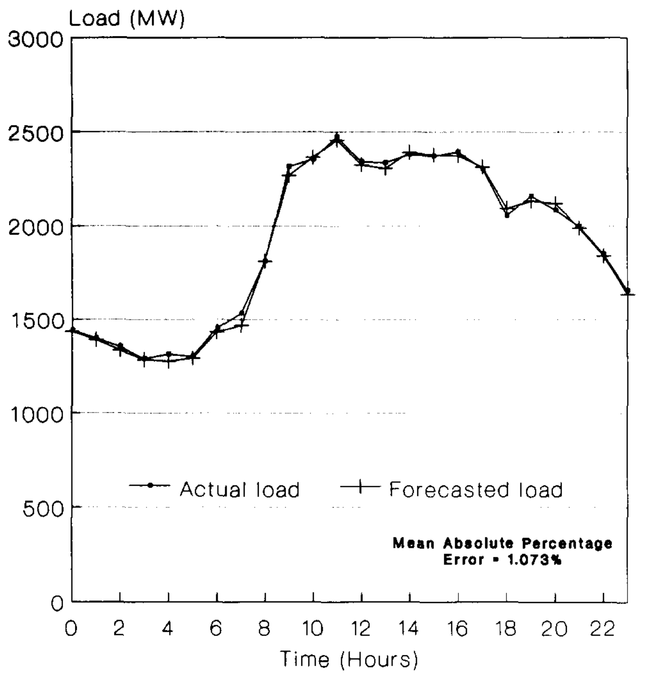

Fig. 5. Forecasting results of the model LFC3 for a weekday (Wednesday).

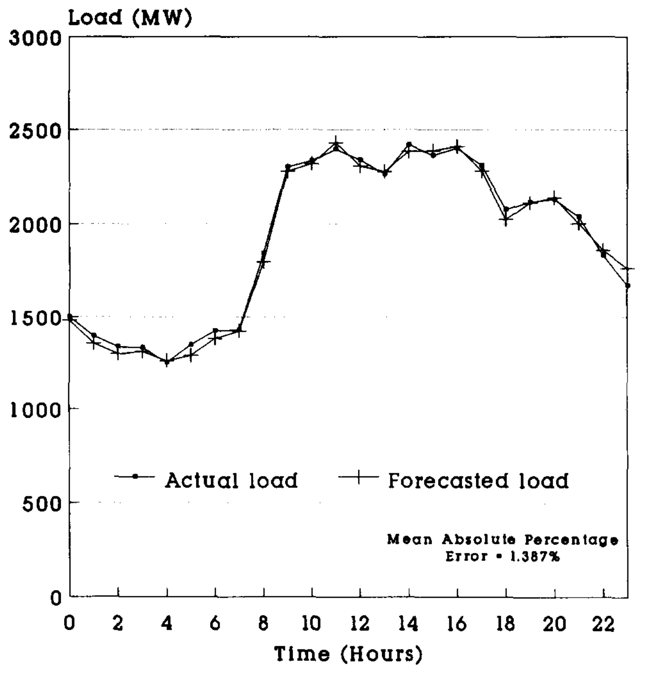

Fig. 6. Forecasting results for the model LFC4 for a Friday.

This comparison shows the supremacy of the neural network method for forecasting application. This ability arises from the implicit way in which the input data are mapped under a nonlinear transformation in the space spanned by the hidden units.

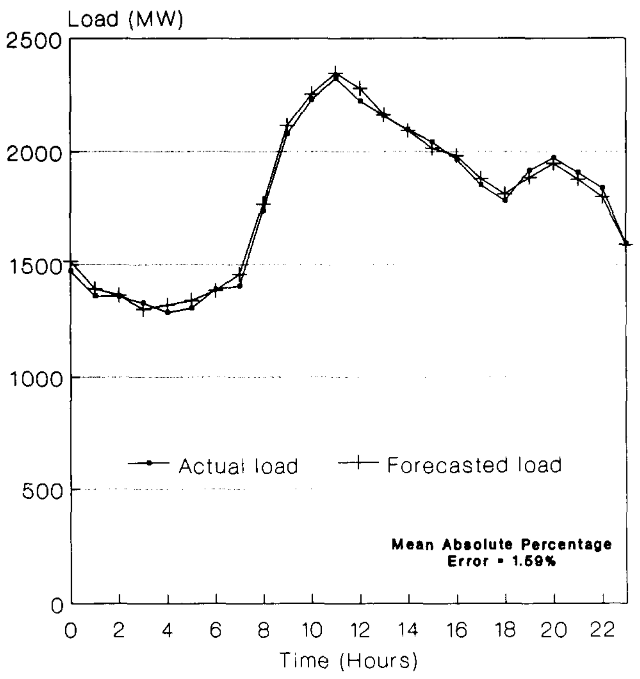

Fig. 7. Forecasting results of the model LFC5 for a Saturday.

TABLE 1. Comparison of forecasting errors from various methods

| Forecasting method | Weekday forecast | Weekend forecast | Public holiday forecast |
|---|---|---|---|
| Neural network forecaster | 1.07% | 1.80% | 2.09% |
| Multiple regression | 1.32% | 1.91% | 2.34% |
| ARMA model | 2.17% | 2.74% | 3.43% |

## 6. Conclusion

This paper describes the application of neural networks for short-term load forecasting and its implementation using multilayer perceptrons. In this paper we concentrate on the problem of short-term forecasting up to 24 hours ahead, which plays an important role in unit commitment, optimum operation planning of power generation facilities, energy transactions, and system security analysis. Actual load data for an electric utility were used to train this forecasting model. The five neural networks developed for forecasting hourly loads were extensively tested and the results were compared with those of two experimental methods. Although training the back-propagation networks was time consuming, once trained, the application of the network took only a forward pass, and was therefore very fast. With respect to the traditional approaches, which involve extensive computations, this method presents a completely new way of predicing the future load values. The simple feedforward forecasting networks described here are capable of performing complex prediction tasks better than traditional methods.

## Nomenclature

| | |
|---|---|
| $E_p$ | error (target-output) for pair $p$ |
| $f$ | transfer function |
| $o_{pj}$ | output of neuron $j$ for pair $p$ |
| $t_{pj}$ | target output of neuron $j$ for pair $p$ |
| $w_{ji}$ | weight of synapse between neuron $j$ and $i$ |
| $\alpha$ | momentum term |
| $\delta$ | gradient |
| $\eta$ | learning rate |

## References

1 W. R. Christiaanse, Short term load forecasting using general exponential smoothing, *IEEE Trans. Power Appar. Syst., PAS-90* (1971) 900–909.

2 P. C. Gupta, A stochastic approach to peak power demand forecasting in electric utility systems, *IEEE Trans. Power Appar. Syst., PAS-90* (1971) 824–832.

3 G. E. P. Box and G. M. Jenkins, *Time Series Analysis, Forecasting and Control*, Holden-Day, San Francisco, CA, 1976.

4 M. T. Hagan and S. M. Behr, The time series approach to short term load forecasting, *IEEE Trans. Power Syst., PWRS-2* (1987) 785–791.

5 T. S. Dillon, K. Morsztyn and K. Phua, Short term load forecasting using adaptive pattern recognition and self organizing techniques, *Proc. 5th Power System Computation Conf. (PSCC), Cambridge, UK, 1975.*

6 D. Srinivasan, A. C. Liew and J. S. P. Chen, A novel approach to electrical load forecasting based on a neural network, *Proc. IEEE Int. Joint Conf. Neural Networks (IJCNN), Singapore, 1991,* Vol. 2, pp. 1172–1177.

7 Y. Park and J. K. Park, An expert system for short term load forecasting by fuzzy decision, *Proc. 2nd Symp. Expert System Applications to Power Systems (ESAPS), Seattle, WA, USA, 1989,* pp. 244–250.

8 S. Rahman and R. Bhatnagar, An expert system based algorithm for short term load forecast, *IEEE Trans. Power Syst., 3* (1988) 392–399.

9 D. C. Park, M. A. El-Sharkawi, R. J. Marks, L. E. Atlas and M. J. Damborg, Electric load forecasting using an artificial neural network, *IEEE Trans. Power Syst., 6* (1991) 442–449.

10 D. E. Rumelhart, G. E. Hinton and R. J. Williams, Learning internal representations by error back propagation, in D. E. Rumelhart and J. E. McClelland (eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol. 1, MIT Press, Cambridge, MA, 1986, pp. 318–369.

11 D. C. Park, M. El-Sharkawi and R. J. Marks II, An adaptively trained neural network, *IEEE Trans. Neural Networks, 2* (1991) 334–345.

12 G. Cybenko, Approximation by superpositions of a sigmoidal function, *Mathematics of Control, Signals, and Systems*, Vol. 2, 1989, pp. 303–314.

13 J. L. McClelland and D. E. Rumelhart, *Explorations in Parallel Distributed Processing: A Handbook of Models, Programs and Exercises*, MIT Press, Cambridge, MA, 1988.

14 K. Hornich, M. Stinchcomebe and H. White, Multilayer feedforward networks are universal approximators, *Neural Networks, 2* (1989) 359–366.

15 S. Y. Kung and J. N. Hwang, An algebraic projection for optimal hidden units size and learning rates in back propagation learning, *Proc. IEEE Int. Conf. Neural Networks, San Diego, CA, USA, 1988,* pp. 363–370.

16 S. Makridakis and S. C. Wheelright, *A Handbook of Forecasting*, Wiley, Chichester, UK, 1987.