# Building a Fuzzy Expert System for Electric Load Forecasting Using a Hybrid Neural Network

P. K. Dash and A. C. Liew

National University of Singapore, 10, Kent Ridge Crescent, Singapore

S. Rahman

Department of Electrical Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA

G. Ramakrishna

Regional Engineering College, Rourkela-769 008, India

**Abstract**—*This paper presents the development of a hybrid neural network to model a fuzzy expert system for time series forecasting of electricc load. The hybrid neural network is trained to develop fuzzy logic rules and find optimal input/output membership values of load and weather parameters. A hybrid learning algorithm consisting of unsupervised and supervised learning phases is used for training the fuzzified neural network. In the supervised learning phase, both back-propagation and linear Kalman filter algorithms are used for the adjustment of weights and membership functions. Extensive tests have been performed on a 2-year utility data for the generation of peak and average load profiles in 24 h, 48 h, and 168 h ahead time frame during summer and winter seasons. From the simulation results, it is observed that the fuzzy expert system using the Kalman filter-based algorithm gives faster convergence and more accurate prediction of a load time series.*

## 1. INTRODUCTION

It is well known that fuzzy logic provides an inference strategy that enables approximate human reasoning capabilities to knowledge-based systems. Also, it provides a mathematical morphology to emulate certain perceptual and linguistic attributes associated with human cognition. Although fuzzy theory provides an inference mechanism under cognitive uncertainty, computational neural networks offer exciting advantages such as learning, adaptation, fault tolerance, parallelism, and generalization. The computational neural networks, comprising processing elements called neurons, are capable of coping with computational complexity, nonlinearity, and uncertainty.

To enable a system to deal with cognitive uncertainties in a manner more like humans, one may incorporate the concepts of fuzzy logic into neural network. Although fuzzy logic is a natural mechanism for modelling cognitive uncertainty, it may involve an increase in the amount of computation required. This can be readily offset by using fuzzy neural network approaches having the potential for parallel computations with high flexibility.

The application of an artificial neural network (ANN) and fuzzy logic-based decision support system to time-series forecasting has gained much attention recently. ANN-based load forecasts give large errors when the weather profile changes very fast. Also extremely slow training or even training failure occurs in many cases due to difficulties in selecting proper structures of the neural network paradigm being used, and due to the errors in associated parameters such as learning rates, activation functions, etc., which are fundamental to any back-propagation neural network. On the other hand, the development of a fuzzy decision system (fuzzy expert system) for load forecasting requires detail analysis of data, and the fuzzy rule base has to be developed

Requests for reprints should be sent to P. K. Dash, Centre for Intelligent Systems, Electrical Engineering Department of Regional Engineering College, Rourkela 769008, India.

heuristically for each season. The rules fixed in this way may not always yield the best forecast. Thus, a hybrid neural network model is used that combines the idea of fuzzy logic-based decision system and neural network structure and learning abilities into an integrated framework. Such a hybrid model provides human understandable meaning to the normal feedforward neural network in which the internal units are always opaque to the users. This structure also avoids the rule matching time of the inference engine in the traditional fuzzy logic system and results in enhanced learning speed and prediction accuracy.

The present work is aimed at achieving the said objective of a robust load forecast with much improved accuracy using a fuzzy expert system modelled by the hybrid ANN architecture. The two types of fuzzy expert system models, abbreviated $FES_1$ and $FES_2$ are based on the argument that any fuzzy expert system employing one block of rules may be approximated by a neural network (feedforward, multilayered). The input vector to $FES_1$ and $FES_2$ consists of differences in weather parameters between the reference and the forecasted day or hour. The output of the $FES_1$ and $FES_2$ gives the load correction that, when added to the past load, gives the forecasted load. The learning algorithm for $FES_1$ and $FES_2$ combines unsupervised and supervised learning procedures to build the rule nodes and train the membership functions. The supervised learning procedure for $FES_1$ uses a gradient back-propagation algorithm for finding the optimum weights and membership functions. Although for $FES_2$, the supervised learning procedure uses a linear Kalman filter-based learning algorithm suggested by Scalero and Tepedelenlioglu (1992), which is similar to the least square adaptive learning techniques. The least square adaptive filtering techniques are known to have rapid convergence properties over the back-propagation algorithm.

A few examples of peak load forecasting and daily average load forecasting of a typical utility with 24 h and 168 h lead time are shown in this paper. The approach presented in this paper is also applicable to load forecasts with longer lead times.

## 2. FUZZY EXPERT SYSTEM

### 2.1. Fuzzy Statements

This section portrays inexact information presented by fuzzy statements, and explains both fuzzy conditional statements and inference mechanisms. Typically, an engineering model requires the use of exact or mathematical statements. These statements correspond to precise information such as $x=3.0$, $2<\theta<0$, or $y=3t+24$. The value of $x=3.0$ has a grade membership comparable to 100% ( = 1), for all other values (2.8, 2.9, 3.1, 3.2), the grades of membership in the solution is zero. In the case of real world values, however, this grade of membership is not true due to the imprecision of tools, the influence of the observer, etc. Additionally, human reasoning is often imprecise, that is, inexact statements such as, "the value of $x$ is not big" or "the temperature is about 4°." Therefore, a theory to correctly express the grade of membership is desirable.

The Fuzzy Set Theory allows manipulation of both exact and inexact (fuzzy) statements. This is very important for load forecasting because there are so many fuzzy factors that are difficult to characterize by a number. An instance of this could be weather conditions such as temperature, humidity, cloud cover, etc. For example, "the temperature this morning will be about 11°." Temperature is an important factor in load forecasting, but is not easy to characterise by an exact numerical quantity. In addition, the linguistic hedges (such as small, medium, large) can be modified by other linguistic hedges (such as not, very, very very) (Zadeh, 1965) For example, "value of $x$ is not big," where big is a linguistic hedge and not a modifier.

### 2.2. Fuzzy Rule Base

A fuzzy expert system consists of a collection of fuzzy IF-THEN rules. For example, the rule base $R$ is written as

$$R=[R^1,R^2,\ldots,R^M] \tag{1}$$

with

$$R^l{:}IF\ (x_1 \text{ is } F_1^l \text{ and} \ldots \text{and } x_p \text{ is } F_p^l)$$
$$\text{THEN } (y_1 \text{ is } G_1^l \text{ and} \ldots y_q \text{ is } G_q^l) \tag{2}$$

where $\underline{x}=(x_1,\ldots,x_n)^T$ and $\underline{y}=(y_1,\ldots,y_m)^T$ are the input and output vectors to the fuzzy system, respectively. $F_i^l$ and $G_i^l$ are labels of fuzzy sets $U_i$ and $V_j$, respectively, and $1 \leqslant p \leqslant n$; $1 \leqslant q \leqslant m$, and $l=1,2,\ldots,M$.

### 2.3. Fuzzy Inference Engine

A fuzzy inference engine uses the rules in the fuzzy rule base to determine a mapping from fuzzy sets in $U$ to fuzzy sets in $V$ based on fuzzy logic operations. In the fuzzy inference engine, the IF part of $R_j^l$ defines a cartesian product of $F_1^l,\ldots,F_p^l$ and the $R_j^l$ itself is viewed as a fuzzy implication $F_1^l\ x \ldots x\ F_p^l \Rightarrow G_j^l$. If $A$ is an arbitrary fuzzy set in $U$, then each $R^l$ of eqn. (2) determines a fuzzy set $AR^l$ in $V$ based on the sup-star composition

$$\mu A R_1(y_{A\circ R^l}) = \sup_{\underline{x}\in U}[\mu_A(\underline{x}\cdot y_j)$$
$$F_1{*}\ldots{*}F_p^l \Rightarrow G_j^l$$
$$= \sup_{\underline{x}\in U}[\mu_A(\underline{x}){*}\mu_{F_1^l}(x_1){*}\ldots{*}\mu_{F_p^l}(x_p){*}\mu_{G_j^l}(y_j)] \tag{3}$$

where $y_j \in V_j$.

The final fuzzy set $A \circ R_j$ ($R_j = [R_j^1, \ldots, R_j^M]$) in $V_j$ determined by the fuzzy inference engine is obtained by combining eqn. (2) for $l = 1, 2, \ldots, M$ using the triangular co-norm

$$\mu_{A \circ R_j}(y_j) = \mu_{A \circ R_j^1}(y_j) + \ldots + \mu_{A \circ R_j^M}(y_j). \tag{4}$$

*2.3.1. Fuzzification and Defuzzification.* In many cases it is convenient to express the membership function of a fuzzy subset in terms of a standard nonlinear function. The following Gaussian membership function suggested by Lin and Lee (1991) is used for the fuzzification of input and output linguistic parameters of the fuzzy expert system:

$$\mu_{A_i}(x) = \exp\left\{\frac{-(x-a)^2}{b}\right\} \tag{5}$$

where $a$ and $b$ are the center and width of the Gaussian membership function, respectively.

The defuzzifier is needed to give a crisp output for any practical application like forecasting. The output of the fuzzy inference engine is a fuzzy set $A \circ R$ in $V$, therefore, the defuzzifier maps $A \circ R$ into a crisp point $\underline{y} \in V$. The most commonly used centroid defuzzifier is defined as

$$y = \frac{\sum_{l=1}^{M} \bar{y}_j^l \left( \mu_{A \circ R_j^l}(\bar{y}_j^l) \right)}{\sum_{l=1}^{M} \mu_{A \circ R_j^l}(\bar{y}_j^l)} \tag{6}$$

## 3. FUZZY EXPERT SYSTEM MODEL USING NEURAL NETWORK APPROACH

An alternative to the neural network-based load forecast is the expert system approach. A fuzzy expert system approach for load forecast consists of rules similar to the rules $R^1, \ldots, R^M$ given in Section 2.2. One of the difficulties with the fuzzy expert system is the rule matching and composition time, apart from the time-consuming process of adapting the rules. The neural network eliminates the rule matching process and stores the knowledge in the link weights. The decision signals can be pumped out immediately after the input data are fed in. Figure 3 shows the proposed hybrid neural network to model the fuzzy expert system using the ANN architecture. The fuzzy expert system clusters the differential temperatures and humidities of the $i$th and $(i+n)$th days into fuzzy term sets. Here $n$ is the lead time for the forecast (i.e., $n = 24$ for 24 h ahead forecast, $n = 48$ for 48 h ahead forecast, and so on). The output of the expert system is the final load correction ($\hat{e}_{LC}$). Hence, the forecasted load on $(i+n)$th day ($P_f(i+n)$) is given by:

$$P_f(i+n) = P(i) + \hat{e}_{LC}(i). \tag{7}$$

The fuzzy expert system modelled as a hybrid neural network has a total of five layers. Nodes at layer 1 are the input linguistic nodes. Layer 5 is the output layer and consists of two nodes [one is for the actual load correction ($\hat{e}_{LC}$) and the other is the desired load correction ($e_{LC}$)]. Nodes at layer 2 and 4 are term nodes that act as membership functions to represent the term sets of the respective linguistic variable. Each node at layer 3 represents the preconditions of the rule nodes, and layer 4 links define the consequence of the rules. The functions of each layer is described as follows:

a) Layer 1: The nodes in this layer just transmit the input feature $x_i$, $i = 1, 2$ to the next layer.
b) Layer 2: Each input feature $x_i$, $i = 1, 2$ is expressed in terms of membership values $\mu_x^j(a_{ij}, b_{ij})$, where $i$ corresponds to the input feature and $j$ corresponds to the number of term sets for the linguistic variable $x_i$.

   The membership function $\mu_{x_i}^j$ uses the Gaussian membership function (5).
c) Layer 3: The links in this layer are used to perform precondition matching of fuzzy logic rules. Hence, the rule nodes perform the product operation (or AND operation).

$$\mu_{R_P} = \Pi\, \mu_{x_i}^j \tag{8}$$

   where $R_P = 1, 2, \ldots, n$. $R_P$ corresponds to the rule node and $n$ is the maximum number of rule nodes. However, if the fuzzy AND operation is used

$$\mu_{R_p} = \min\{\mu_{x_i}^j\} \tag{9}$$

d) Layer 4: The nodes in this layer have two operations (i.e., forward and backward transmission). In forward transmission mode, the nodes perform the fuzzy OR operation to integrate the fired rules, which have the same consequence:

$$\mu^4 = \sum_{i=1} u_i^4 \tag{10}$$

   where $p$ corresponds to the links terminating at the node. In the backward transmission mode, the links function exactly the same as the layer 2 nodes.
e) Layer 5: There are two nodes in this layer (i.e., for obtaining the actual and desired output load correction, respectively). The desired output load correction ($e_{LC}$) is fed into the hybrid neural network during learning whereas the actual load correction ($\hat{e}_{LC}$) is obtained by using the centroid defuzzification method (6).

## 4. HYBRID LEARNING ALGORITHM FOR THE FUZZY EXPERT SYSTEM (FES)

The hybrid learning scheme consists of two phases. In phase I, the initial membership functions of the input and output linguistic variables are fixed and an initial form of the network is constructed. Then, during the learning process, some nodes and links of this initial network are deleted or combined to form the final structure of the network. In phase II learning, the input and output membership functions are optimally adjusted to obtain the desired outputs. Phase I represents the unsupervised learning phase. Phase II represents the supervised learning phase.

### 4.1. Phase-I: Unsupervised Learning Phase

Given the training input data $x_i(t)$, $i = 1, 2$, the desired output load correction ($e_{\mathrm{LC}}(t)$) and the fuzzy partitions $|\mu^j_{x_i}|$. We want to locate the membership functions (i.e., $a_{ij}$ and $b_{ij}$) and find the fuzzy logic rules.

The Kohonen's feature maps algorithm (Lin & Lee, 1991) is used to find the values for $a_{ij}$ and $b_{ij}$:

$$\|x(t)-a_{i,\,\mathrm{closest}}(t)\| = \min_{1\le j\le t}\left\{\|x_i(t)-a_{ij}(t)\|\right\} \tag{11}$$

$$a_{i,\,\mathrm{closest}}(t+1) = a_{i,\,\mathrm{closest}}(t) + \eta(t)\left[x_i(t)-a_{i,\,\mathrm{closest}}(t)\right] \tag{12}$$

$$a_{ij}(t+1) = a_{ij}(t) \quad \text{for} \quad a_{ij} \ne a_{i,\,\mathrm{closest}} \tag{13}$$

where $\eta(t)$ is the monotonically decreasing learning rate and $t = |T(x_i)|$ (i.e., the number of term sets for the linguistic variable $x_i$).

This adaptive formulation runs independently for each input linguistic variable $x_i$:

The width, $b_{ij}$ is determined heuristically at this stage (Lin & Lee, 1991) as follows:

$$b_{ij} = \frac{|a_{i,j}-a_{i,\,\mathrm{closest}}|}{r} \tag{14}$$

where $r$ is an overlap parameter. After the parameters of the membership functions have been found, the weights in layer 4 are obtained by using the competitive learning algorithm (Rumelhart & Zipses, 1985) as follows:

$$W_{ij} = LI^4_j(LI^3_j - W_{ij}) \tag{15}$$

where $LI^3_j$ serves as the win–loss index of the rule node at layer 3 and $LI^4_j$ serves as the win–loss index of the $j$th term node at layer 4, respectively.

After the competitive learning through the whole training data set, the link weights at layer 4 represent the strength of the existence of the corresponding rule consequence. If a link weight between rule node and the term node of the output linguistic node is very small, then all the corresponding links are deleted, meaning that this rule node has little or no relation to the output.

After the consequences of rule nodes are determined, the rule combination is performed to reduce the number of rules in the following manner. The criteria for the choice of rule nodes are:

1. they have the same consequences,
2. some preconditions are common to all the rule nodes in this set,
3. the union of other preconditions of these rule nodes composes the whole term set of some input linguistic variables.

The rule nodes that satisfy these criteria are replaced by a new rule node with common preconditions.

### 4.2. Phase II: Supervised Learning Phase

Once the fuzzy logic rules have been found, the phase II learning is used to find the optimum weights and the input and output membership functions by using the gradient descent backpropagation algorithm (Scalero & Tepedelenlioglu, 1992].

a) The weight update equation for layer 4 is:

$$W_{ij}(t) = W_{ij}(t) + \eta\left[\frac{-\partial E}{\partial W_{ij}}\right]. \tag{16}$$

The error function $E$ is given by

$$E = \frac{1}{2}\left[e_{\mathrm{LC}}(t) - \hat{e}_{\mathrm{LC}}(t)\right]^2. \tag{17}$$

Because

$$\hat{e}_{\mathrm{LC}} = \frac{\sum (a_{ij}b_{ij})\mu^5_i}{\sum b_{ij}\mu^5_i} \tag{18}$$

using centroid defuzzification method (6)

$$\mu^5_i = \mu^4 = \sum_{i=1}^{p} \mu^4_i W_{ij} \tag{19}$$

where $W_{ij} = 1$ for $t = 1$.
Now,

$$\frac{\partial E}{\partial W_{ij}} = \frac{\partial E}{\partial \hat{e}_{\mathrm{LC}}}\frac{\partial \hat{e}_{\mathrm{LC}}}{\partial \mu^5_i}\frac{\partial \mu^5_i}{\partial W_{ij}}. \tag{20}$$

Therefore,

$$\frac{\partial E}{\partial W_{ij}} = [e_{LC}(t) - \hat{e}_{LC}(t)]$$

$$\times \left\{ \frac{a_{ij} b_{ij} (\Sigma b_{ij} \mu_i^5) - (\Sigma a_{ij} b_{ij} \mu_i^5) b_{ij}}{(\Sigma b_{ij} \mu_i^5)^2} \right\} \mu_i^4. \quad (21)$$

Training the output membership functions at layer 5:

$$a_{ij}(t+1) = a_{ij}(t) + \eta_1 \left[ \frac{-\partial E}{\partial a_{ij}} \right] \quad (22)$$

where $\eta_1$ is the learning rate.
Now,

$$\frac{\partial E}{\partial a_{ij}} = \frac{\partial E}{\partial \hat{e}_{LC}} \frac{\partial \hat{e}_{LC}}{\partial a_{ij}}. \quad (23)$$

Therefore, using eqns. (17) and (18) we get

$$a_{ij}(t+1) = a_{ij}(t) + \eta_1 (e_{LC}(t) - \hat{e}_{LC}(t)) \left\{ \frac{b_{ij} \mu_i^5}{(\Sigma b_{ij} \mu_i^5)} \right\}. \quad (24)$$

Similarly,

$$b_{ij}(t+1) = b_{ij}(t) = \eta_2 \left[ \frac{-\partial E}{\partial b_{ij}} \right] \quad (25)$$

where $\eta_2$ is the learning rate.
Now,

$$\frac{\partial E}{\partial b_{ij}} = \frac{\partial E}{\partial \hat{e}_{LC}} \frac{\partial \hat{e}_{LC}}{\partial b_{ij}}. \quad (26)$$

Therefore, using eqns. (17) and (18) we get

$$b_{ij}(t+1) = b_{ij}(t) + \eta_2 (e_{LC}(t) - \hat{e}_{LC}(t))$$

$$\times \left\{ \frac{a_{ij} \mu_i^5 (\Sigma b_{ij} \mu_i^5) - (\Sigma a_{ij} b_{ij} \mu_i^5) \mu_i^5}{(\Sigma b_{ij} \mu_i^5)^2} \right\} \quad (27)$$

c) Tuning the input membership functions at layer 2: The error signal $\delta_i^4$ at layer 4 is given by

$$\delta_i^4 = [e_{LC}(t) - \hat{e}_{LC}(t)]$$

$$\times \left\{ \frac{a_{ij} b_{ij} (\Sigma b_{ij} \mu_i^5) - (\Sigma a_{ij} b_{ij} \mu_i^5) b_{ij}}{(\Sigma b_{ij} \mu_i^5)^2} \right\} \quad (28)$$

where $(a_{ij}, b_{ij})$ correspond to the output term set.

The error signal at layer 3 is found by performing the summation over the consequences of a rule node (i.e., layer 4). Therefore,

$$\delta_i^3 = \Sigma \delta_i^4. \quad (29)$$

The adaptive rule for $a_{ij}$ (layer 2) is derived as:

$$a_{ij}(t+1) = a_{ij}(t) + \eta_3 \left[ \frac{-\partial E}{\partial a_{ij}} \right] \quad (30)$$

where $\eta_3$ is the learning rate.
Now,

$$\frac{\partial E}{\partial a_{ij}} = \delta_i^2 e \left( \frac{-(x_i - a_{ij})^2}{b_{ij}} \right) \left\{ \frac{2(x_i - a_{ij})}{b_{ij}} \right\}. \quad (31)$$

Also,

$$\delta^2 = \sum \delta^3, \quad (32)$$

in a similar way as eqn. (29).
Therefore,

$$a_{ij}(t+1) = a_{ij}(t) - \eta_3 \delta_i^2 e \left( \frac{-(x_i - a_{ij})^2}{b_{ij}} \right) \left\{ \frac{2(x_i - a_{ij})}{b_{ij}} \right\}. \quad (33)$$

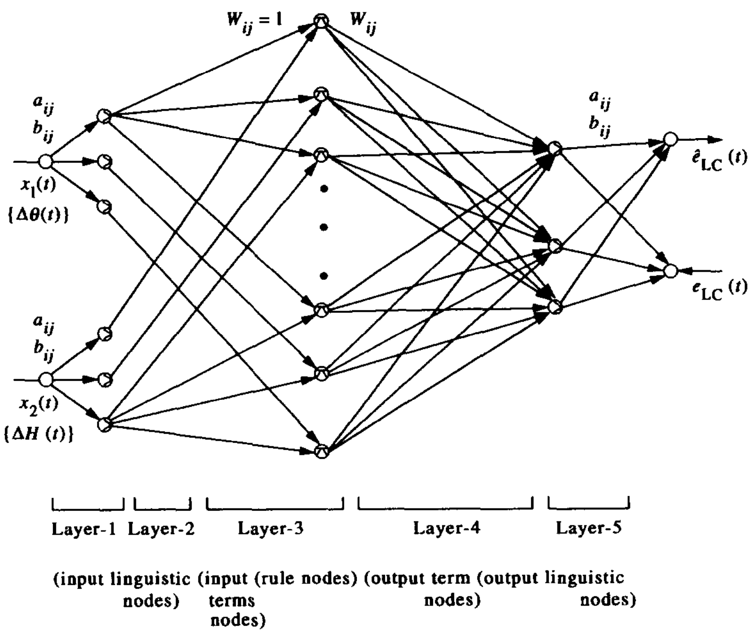
Similarly, the adaptive rule for $b_{ij}$ (layer 2) is derived as

$$b_{ij}(t+1) = b_{ij}(t) - \eta_4 \delta_i^2 e \left( \frac{-(x_i - aij)^2}{b_{ij}} \right) \left\{ \frac{(x_i - a_{ij})^2}{b_{ij}^2} \right\}. \quad (34)$$

The convergence speed of the phase II supervisory learning scheme for fuzzy expert systems is found to be superior to the supervisory learning scheme for ordinary fuzzy neural network approach (Dash et al., 1993) because the phase I unsupervised learning process had done much of the learning in advance. The convergence speed of the supervised learning process can be improved further by solving the weight update equations at layer 3 and the input and output membership function at layer 1 and layer 2 by linear Kalman filter equations (Singhal & Wu, 1989).

## 5. FUZZY EXPERT SYSTEM USING LINEAR KALMAN FILTER (FES$_2$)

Referring to Figure 1, the tuning of Gaussian membership function at layers 2 and 4 $(a_{ij}, b_{ij})$ is similar to the weight update equations at layer 3. In (Scalero & Tepedelenlioglu, 1992) it has been shown that the



$\Delta\theta$ - Differential temperature　　$t$ - Iteration No.

$\Delta H$- Differential humidity　　$W_{ij}$ - weight

$\hat{e}_{LC}$ - Actual load correction　　$e_{LC}$ - Desired load correction

**FIGURE 1. Hybrid neural network modelled as a fuzzy expert system.**

presence of nonlinear function (in this study the Gaussian membership function) between the weights and the error makes the back-propagation algorithm different from standard least square adaptive filtering techniques, which are known to have rapid convergence (Haykin, 1986).

The FES$_2$ model uses the linear Kalman filter equations for updating the weights and the membership function. Unlike the back-propagation technique, this algorithm assumes that the estimated weight matrix is nonstationary and hence will allow the tracking of a time varying data like that of load forecasting.

This algorithm defines locally at each node a gradient based on present and past data, and updates the weights of each node using the linear Kalman filter equations to bring this gradient identically to zero whenever an update is made. Performing the update and defining the gradient in this manner ensures that maximum use is made of available information.

The gradient for the linear combiner in each is defined as

$$G=R\,W-C. \qquad (35)$$

The weight vector that makes $G=R\,W-C$ zero is the solution to the normal equations.

Here $R$ is the auto correlation matrix for each layer and is calculated as

$$R=\sum_{np=1}^{NP} f^{NP-np} x_{np}\, x_{np}^T \qquad (36)$$

and $C$ is the cross-correlation matrix and is given by

$$C=\sum_{np=1}^{NP} f^{NP-np} d_{np}\, x_{np}^T \qquad (37)$$

where $NP$ denotes the total number of patterns. $d_{np}$ and $x_{np}$ are the summation output and the output of the nonlinearity (Gaussian membership function), respectively, for the layer 2 and layer 5 nodes, respectively. As the layer 4 nodes contain no nonlinearity term, $d_{np}=x_{np}$. The weights given by eqn. (35) are updated by using a Kalman filter at each layer with a variable forgetting factor ($f$) as shown in eqns. (36) and (37). A variable forgetting factor is used to take care of the new estimates giving less weightage to old estimates.

The Kalman gain vector $K_j\,(t)$ at each layer $j$ is given by

$$K_j(t)=\left\{\frac{R_j^{-1}(t)x_i(t)}{f_j+x_i^T(t)\,R_j^{-1}(t)x_i(t)}\right\} -\mu_j\left\{\frac{E(t-1)-E(t-2)}{K_j(t-1)-K_j(t-2)}\right\} \qquad (38)$$

where $x_i(t)$ corresponds to the previous layer, $\mu_j$ is the learning rate, and $t$ denotes the iteration number. The error, $E$, at each iteration is obtained from eqn. (17). The first term in eqn. (38) expresses the Kalman gain, $K_j$, in

**TABLE 1**
**The Learned Fuzzy Logic Rules for 24 Hour Ahead Peak Load Forecasting Using FES$_1$ in Winter**

| | Term Sets | | |
|---|---|---|---|
| | Preconditions | | Consequence |
| Rule | $\Delta \Theta_{max}(i, i+1)$ | $\Delta H_{max}(i, i+1)$ | $\hat{e}_{LC}(i)$ |
| 0 | 0 | 3 | 7 |
| 1 | 0 | 4 | 7 |
| 2 | 1 | 0 | 8 |
| 3 | 1 | 1 | 7 |
| 4 | 1 | 2 | 7 |
| 5 | 1 | 3 | 6 |
| 6 | 1 | 4 | 6 |
| 7 | 2 | 0 | 8 |
| 8 | 2 | 1 | 7 |
| 9 | 2 | 2 | 7 |
| 10 | 2 | 3 | 6 |
| 11 | 2 | 4 | 7 |
| 12 | 3 | 1 | 2 |
| 13 | 3 | 2 | 4 |
| 14 | 3 | 3 | 6 |
| 15 | 4 | 2 | 5 |
| 16 | 4 | 3 | 6 |
| 17 | 4 | 4 | 1 |
| 18 | 5 | 0 | 3 |
| 19 | 5 | 1 | 2 |
| 20 | 5 | 2 | 1 |
| 21 | 5 | 3 | 1 |
| 22 | 5 | 4 | 0 |
| 23 | 6 | 0 | 1 |
| 24 | 6 | 1 | 1 |
| 25 | 6 | 2 | 0 |

terms of the inverse autocorrelation matrix $R_j$ and the input to the previous layer $x_i$. The second term is used to track the error along a gradient descent surface to attain the final convergence (similar to tuning of membership functions). This helps in a faster rate of convergence.

During training the network is randomly initialized. The original training pairs become obsolete as the adjacent layers adapt their weights according to their perceived observations. Hence, the $R_j(t)$ matrix for a particular layer, containing the sum total of all prior observations, becomes outdated. Because the Kalman filter minimizes the prediction effort of all prior observations, parameter estimates become biased towards initial conditions and the solution to this problem is the modification of the covariance matrix and the forgetting factor in the following manner:

$$f_j(t+1) = f_0 - \gamma_1 \bar{e}(t) \tag{39}$$

$$R_j^{-1}(t+1) = [R_j^{-1}(t) - K_j(t) x_i^T(t) R_j^{-1}(t)]) / f_j \tag{40}$$

where

$$\bar{e}(t) = \frac{r(t)}{1 + x_i^T(t) R_j^{-1}(t) x_i(t)} \tag{41}$$

and $\gamma_1$ is a constant that is inversely proportional to the prediction error, $e(t)$, at that layer. As a result, $f_j$ remains close to 1 when $R_j$ is already large and the weight estimates are sensitive to parameter variations. A lower bound $f_0$ and $f_j$ is used to prevent the forgetting factor from becoming excessively small, resulting in large estimate fluctuations in spite of small prediction errors. The strategy for employing this forgetting factor is to initially allow it to assume small values during learning such that the initial conditions are quickly forgotten. However, as the average prediction error decreases, it is important to cause $f_j$ to remain near one to avoid the winding up of the weight estimates. This can be simply implemented by setting $\gamma_1$ smaller as the mean squared error decreases.

The unsupervised learning phase of FES$_2$ is same as for FES$_1$. The supervised learning phase of FES$_2$ is modified using the linear Kalman filter equations instead of the back-propagation method used for FES$_1$. Therefore, the update equations are modified as follows:

a) The weight update equation for layer 4 is:

$$W_{ij}(t) = W_{ij}(t) + \eta K_j(t) \left[ \frac{-\partial E}{\partial W_{ij}} \right] \tag{42}$$

where $\partial E / \partial W_{ij}$ is given by eqn. (21) and $K_j(t)$ is given by eqn. (38).

b) The update equations for $a_{ij}$ and $b_{ij}$ at layer 5 are:

$$a_{ij}(t+1) = a_{ij}(t) + \eta_1 K_j(t) \left[ \frac{-\partial E}{\partial a_{ij}} \right] \tag{43}$$

where $\partial E / \partial a_{ij}$ is given by eqn. (23) and $K_j(t)$ is given by eqn. (38).

$$b_{ij}(t+1) = b_{ij}(t) + \eta_2 K_j(t) \left[ \frac{-\partial E}{\partial b_{ij}} \right] \tag{44}$$

where $\partial E / \partial d_{ij}$ is given by eqn. (26) and $K_j(t)$ is given by eqn. (38).

c) The update equations for $a_{ij}$ and $b_{ij}$ at layer 2 are:

$$a_{ij}(t+1) = a_{ij}(t) + \eta_3 K_j(t) \left[ \frac{-\partial E}{\partial a_{ij}} \right] \tag{45}$$

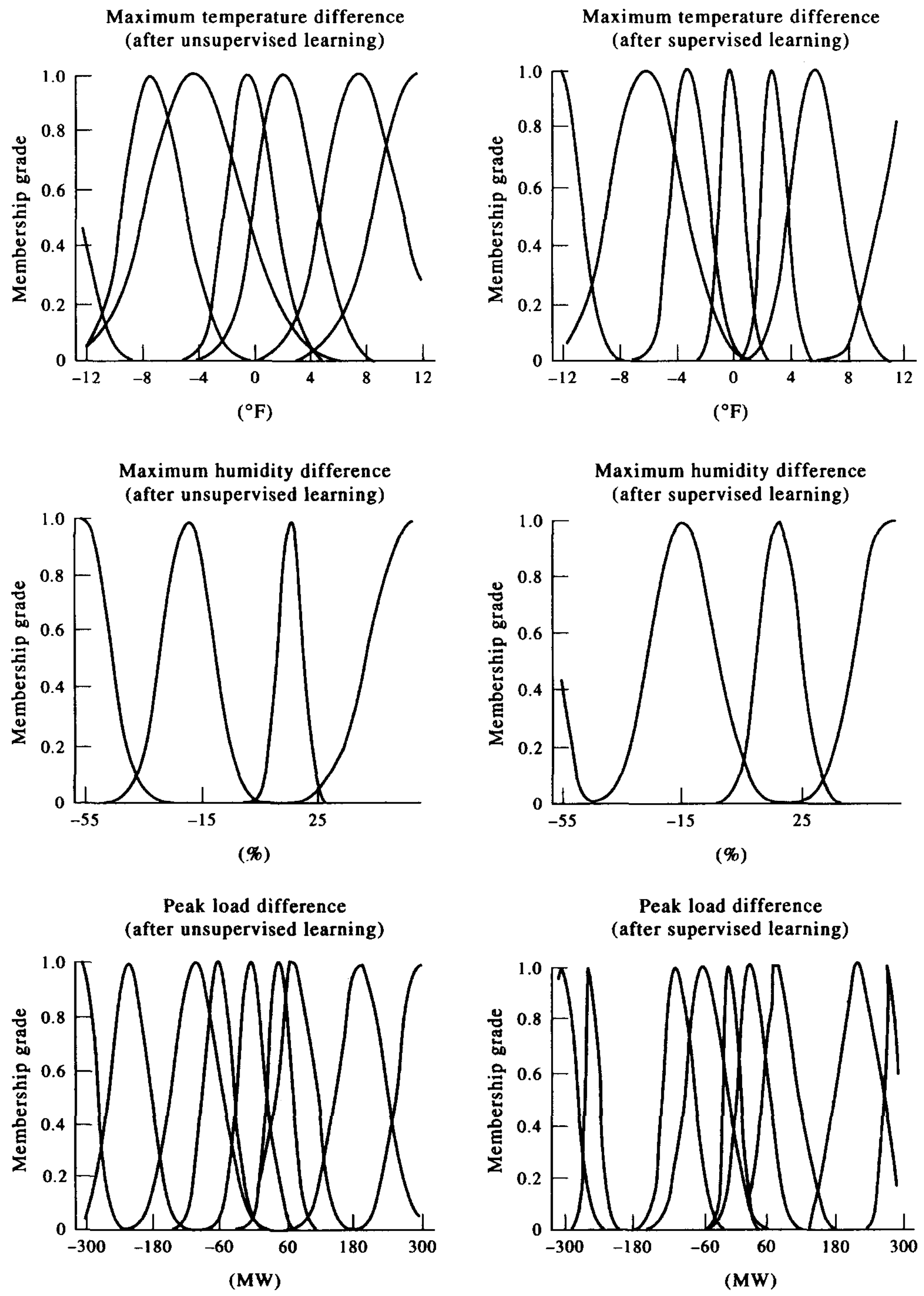


**FIGURE 2. Learned membership functions for 24 h ahead peak load forecasting, in January (winter) using FES$_1$.**

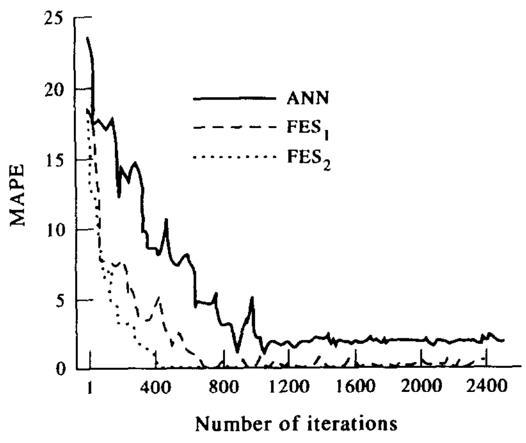


**FIGURE 3. Comparison of the mean absolute percentage errors versus the iteration number for 24 h ahead forecast in January (winter).**

where, $\delta E/\partial a_{ij}$ is given by eqn. (31) and $K_j(t)$ is given by eqn. (38).

Similarly,

$$b_{ij}(t+1) = b_{ij}(t) - \eta_4 K_j(t)\delta_i^2 e\left(\frac{-(x_i-a_{ij})^2}{b_{ij}}\right) \times \left\{\frac{(x_i-a_{ij})^2}{b_{ij}^2}\right\} \quad (46)$$

where $d_i^2$ is given by eqn (32) and $K_j(t)$ is given by eqn. (38).

The convergence properties of the supervisory learning scheme for FES$_2$ are found to be superior to FES$_1$ using the Kalman filter equation in the weight and membership update equations, as shown in Sections 6.3 and 6.4. Also, the accuracy of the forecast increases for a similar reason.

$$R_j^{-1}(t+1) = f_j(t)R_j^{-1}(t) + x_i^T(t)x_i(t) \quad (46)$$

and the forgetting factor $f_j$ is modified as

$$f_j(t+1) = 1 - \gamma\bar{e}(t+1) \quad (47)$$

**TABLE 2**
**Peak Load Forecasting in June (Summer) Using 24 Hour Ahead Forecast**

| Day | Actual Load (MW) | Peak Forecast (NN) (MW) | P E (NN) | Peak Forecast (FES$_1$) (MW) | P E (FES$_1$) | Peak Forecast (FES$_2$) (MW) | P E (FES$_2$) |
|---|---|---|---|---|---|---|---|
| 1 | 1848 | 1872.8 | −1.34 | 1823.3 | 1.34 | 1825.7 | 1.21 |
| 2 | 1850 | 1883.7 | −1.82 | 1844.7 | 0.29 | 1831.8 | 0.99 |
| 3 | 1852 | 1827.8 | 1.31 | 1833.6 | 0.99 | 1868.0 | −0.86 |
| 4 | 1687 | 1627.6 | 3.52 | 1669.7 | 1.03 | 1688.9 | −0.11 |
| 5 | 1573 | 1499.6 | 4.67 | 1539.4 | 2.14 | 1544.1 | 1.84 |
| 6 | 1752 | 1752.4 | −0.02 | 1747.6 | 0.25 | 1749.1 | 0.17 |
| 7 | 1679 | 1661.4 | 1.05 | 1671.1 | 0.47 | 1668.5 | 0.62 |
| 8 | 1674 | 1726.6 | −3.14 | 1652.6 | 1.28 | 1698.5 | −1.46 |
| 9 | 1712 | 1750.3 | −2.24 | 1686.0 | 1.52 | 1715.3 | −0.19 |
| 10 | 1778 | 1858.3 | −4.52 | 1752.7 | 1.42 | 1761.8 | 0.91 |
| 11 | 1614 | 1660.4 | −2.87 | 1620.6 | −0.41 | 1602.7 | 0.70 |
| 12 | 1520 | 1456.7 | 4.16 | 1474.5 | 3.00 | 1559.6 | −2.60 |
| 13 | 1730 | 1705.0 | 1.45 | 1747.4 | −1.01 | 1719.5 | 0.61 |
| 14 | 1663 | 1709.5 | −2.80 | 1674.5 | −0.69 | 1657.8 | 0.31 |
| 15 | 1722 | 1755.8 | −1.96 | 1755.7 | −1.96 | 1741.7 | −1.14 |
| 16 | 1710 | 1745.5 | −2.08 | 1681.4 | 1.67 | 1693.5 | 0.97 |
| 17 | 1806 | 1838.3 | −1.79 | 1805.1 | 0.05 | 1803.3 | 0.15 |
| 18 | 1730 | 1788.2 | −3.37 | 1757.7 | −1.60 | 1718.3 | 0.68 |
| 19 | 1622 | 1584.9 | 2.29 | 1649.1 | −1.67 | 1644.7 | −1.40 |
| 20 | 1752 | 1775.7 | −1.35 | 1772.9 | −1.20 | 1738.9 | 0.75 |
| 21 | 1658 | 1648.6 | 0.57 | 1671.4 | −0.81 | 1662.6 | −0.28 |
| 22 | 1672 | 1684.1 | −0.72 | 1685.3 | −0.80 | 1680.3 | −0.50 |
| 23 | 1701 | 1643.7 | 3.37 | 1718.8 | −1.05 | 1727.7 | −1.57 |
| 24 | 1736 | 1669.5 | 3.83 | 1702.9 | 1.91 | 1714.4 | 1.24 |
| 25 | 1588 | 1556.1 | 2.01 | 1621.0 | −2.08 | 1619.2 | −1.96 |
| 26 | 1482 | 1413.4 | 4.63 | 1463.9 | 1.22 | 1462.9 | 1.29 |
| 27 | 1720 | 1672.0 | 2.79 | 1695.7 | 1.41 | 1705.4 | 0.85 |
| 28 | 1699 | 1750.4 | −3.02 | 1708.2 | −0.53 | 1665.9 | 1.95 |
| 29 | 1684 | 1666.6 | 1.03 | 1705.2 | −1.26 | 1667.5 | 0.98 |
| 30 | 1693 | 1757.5 | −3.81 | 1676.9 | 0.95 | 1721.2 | −1.66 |
| MAPE | | | 2.45 | | 1.20 | | 1.00 |
| Iterations required | | | 970 | | 520 | | 440 |

**TABLE 3**
**Peak Load Forecasting in January (Winter) Using 48 Hour Ahead Forecast**

| Day | Actual Load (MW) | Peak Forecast (NN) (MW) | P E (NN) | Peak Forecast ($FES_1$) (MW) | P E ($FES_1$) | Peak Forecast ($FES_2$) (MW) | P E ($FES_2$) |
|---|---|---|---|---|---|---|---|
| 1 | 2690 | 2553.1 | 5.09 | 2663.6 | 0.98 | 2673.3 | 0.62 |
| 2 | 2628 | 2554.4 | 2.80 | 2577.5 | 1.92 | 2663.2 | −1.34 |
| 3 | 2703 | 2657.0 | 1.70 | 2672.7 | 1.12 | 2728.7 | −0.95 |
| 4 | 2592 | 2631.9 | −1.54 | 2560.6 | 1.21 | 2561.4 | 1.18 |
| 5 | 2530 | 2583.9 | −2.13 | 2509.5 | 0.81 | 2544.9 | −0.59 |
| 6 | 2574 | 2590.2 | −0.63 | 2591.2 | −0.67 | 2555.5 | 0.72 |
| 7 | 2389 | 2383.7 | 0.22 | 2400.0 | −0.46 | 2367.0 | 0.92 |
| 8 | 2513 | 2482.6 | 1.21 | 2493.4 | 0.78 | 2497.7 | 0.61 |
| 9 | 2500 | 2372.0 | 5.12 | 2595.8 | −3.83 | 2568.0 | −2.72 |
| 10 | 2450 | 2366.7 | 3.40 | 2490.9 | −1.67 | 2478.9 | −1.18 |
| 11 | 2551 | 2504.3 | 1.83 | 2569.9 | −0.74 | 2557.4 | −0.25 |
| 12 | 2763 | 2889.0 | −4.56 | 2849.2 | −3.12 | 2811.6 | −1.76 |
| 13 | 2603 | 2563.4 | 1.52 | 2623.3 | −0.78 | 2620.2 | −0.66 |
| 14 | 2914 | 2781.7 | 4.54 | 2875.8 | 1.31 | 2895.9 | 0.62 |
| 15 | 2761 | 2865.1 | −3.77 | 2780.9 | −0.72 | 2783.4 | −0.81 |
| 16 | 2514 | 2425.0 | 3.54 | 2567.3 | −2.12 | 2552.0 | −1.51 |
| 17 | 2543 | 2647.8 | −4.12 | 2511.5 | 1.24 | 2516.3 | 1.05 |
| 18 | 2435 | 2500.0 | −2.67 | 2486.6 | −2.12 | 2414.8 | 0.83 |
| 19 | 2496 | 2468.8 | 1.09 | 2487.0 | 0.36 | 2488.3 | 0.31 |
| 20 | 2551 | 2484.2 | 2.62 | 2505.6 | 1.78 | 2522.9 | 1.10 |
| 21 | 2813 | 2882.8 | −2.48 | 2773.3 | 1.41 | 2794.2 | 0.67 |
| 22 | 2537 | 2427.4 | 4.32 | 2591.3 | −2.14 | 2574.8 | −1.49 |
| 23 | 2381 | 2282.2 | 4.15 | 2323.1 | 2.43 | 2333.1 | 2.01 |
| 24 | 2459 | 2378.3 | 3.28 | 2432.7 | 1.07 | 2433.2 | 1.05 |
| 25 | 2505 | 2487.0 | 0.72 | 2521.3 | −0.65 | 2517.3 | −0.49 |
| 26 | 2429 | 2500.2 | −2.93 | 2451.3 | −0.92 | 2450.6 | −0.89 |
| 27 | 2438 | 2320.7 | 4.81 | 2379.0 | 2.42 | 2403.1 | 1.43 |
| 28 | 2748 | 2771.4 | −0.85 | 2711.7 | 1.32 | 2702.9 | 1.64 |
| 29 | 2388 | 2503.3 | −4.83 | 2347.2 | 1.71 | 2350.5 | 1.57 |
| 30 | 2175 | 2132.4 | 1.96 | 2208.7 | −1.55 | 2142.2 | 1.51 |
| 31 | 2539 | 2466.1 | 2.87 | 2520.0 | 0.75 | 2521.0 | 0.71 |
| MAPE | | | 2.82 | | 1.42 | | 1.07 |
| Iterations required | | | 1560 | | 880 | | 700 |

$$\bar{e}(t+1) = \frac{[e(t)]^2}{1 + x^T(t)R_j(t-1)x(t)} \tag{48}$$

where $\gamma$ is a constant inversely proportional to the prediction error $e(t)$.

## 6. IMPLEMENTATION RESULTS

In order to evaluate the performance of the fuzzy expert system, load forecasting is performed on a typical utility data. The models ANN, $FES_1$, and $FES_2$ are tested on a 2-year utility data for generating peak and average load profiles and some of the results are given in the subsequent subsections. In (Bunn and Farmer (1989) and Brace (1991)) it has been shown that ANN gives the best prediction and accuracy compared to conventional approaches. So in this case the results of $FES_1$ and $FES_2$ are compared to that of the ANN approach.

The training sets are formed separately for each of the seven day types (i.e., Tuesdays through Thursdays, Mondays, Fridays, Saturdays, Sundays, Holidays). The selection of training patterns and the selection of variable ranges are given in Sections 6.1 and 6.2.

### 6.1. Optimum Selection of Training Patterns

The utility data studied here are susceptible to large and sudden changes in weather and load, so selection of appropriate training cases plays a vital role in training the network. Several techniques for the selection of training patterns have been suggested in Peng, Hubele, and Karady (1992).

The following load model is used for peak load forecasting:

$$P_{\max}(i) = f(P_{\max}(i-n), P_{\max}(i-n-1), \ldots, P_{\max}(i-n-n_1),$$
$$\Theta_{\max}(i-n), \Theta_{\max}(i), \Theta_{\max}(i-1), \ldots,$$
$$\Theta_{\max}(i-n_2), H_{\max}(i-n), H_{\max}(i), H_{\max}(i-1),$$

TABLE 4
**Peak Load Forecasting in January (Winter) Using 168 Hour Ahead Forecast**

| Day | Actual Load (MW) | Peak Forecast (NN) (MW) | P E (NN) | Peak Forecast ($FES_1$) (MW) | P E ($FES_1$) | Peak Forecast ($FES_2$) (MW) | P E ($FES_2$) |
|---|---|---|---|---|---|---|---|
| 1 | 2690 | 2534.5 | 5.78 | 2630.0 | 2.23 | 2742.2 | −1.94 |
| 2 | 2628 | 2556.0 | 2.74 | 2564.7 | 2.41 | 2571.0 | 2.17 |
| 3 | 2703 | 2610.8 | 3.41 | 2649.5 | 1.98 | 2666.5 | 1.35 |
| 4 | 2592 | 2651.4 | −2.29 | 2636.6 | −1.72 | 2551.0 | 1.58 |
| 5 | 2530 | 2616.5 | −3.42 | 2580.1 | −1.98 | 2480.7 | 1.95 |
| 6 | 2574 | 2503.5 | 2.74 | 2519.9 | 2.10 | 2628.8 | −2.13 |
| 7 | 2389 | 2433.2 | −1.85 | 2351.5 | 1.57 | 2354.8 | 1.43 |
| 8 | 2513 | 2433.1 | 3.18 | 2444.9 | 2.71 | 2449.7 | 2.52 |
| 9 | 2500 | 2353.8 | 5.85 | 2370.8 | 5.17 | 2378.5 | 4.86 |
| 10 | 2450 | 2324.3 | 5.13 | 2340.5 | 4.47 | 2360.1 | 3.67 |
| 11 | 2551 | 2620.1 | −2.71 | 2510.7 | 1.58 | 2519.6 | 1.23 |
| 12 | 2763 | 2953.9 | −6.91 | 2886.0 | −4.45 | 2874.1 | −4.02 |
| 13 | 2603 | 2531.9 | 2.73 | 2555.9 | 1.81 | 2649.6 | −1.79 |
| 14 | 2914 | 2761.6 | 5.23 | 2853.1 | 2.09 | 2972.3 | −2.00 |
| 15 | 2761 | 2894.6 | −4.84 | 2658.3 | 3.72 | 2660.2 | 3.65 |
| 16 | 2514 | 2389.6 | 4.95 | 2408.4 | 4.20 | 2415.2 | 3.93 |
| 17 | 2543 | 2697.4 | −6.07 | 2653.9 | −4.36 | 2651.8 | −4.28 |
| 18 | 2435 | 2494.7 | −2.45 | 2454.7 | −0.81 | 2424.8 | 0.83 |
| 19 | 2496 | 2463.8 | 1.29 | 2509.2 | −0.53 | 2481.5 | 0.58 |
| 20 | 2551 | 2456.6 | 3.70 | 2493.9 | 2.24 | 2514.5 | 1.43 |
| 21 | 2813 | 2886.4 | −2.61 | 2864.5 | −1.83 | 2762.9 | 1.78 |
| 22 | 2537 | 2382.0 | 6.11 | 2441.9 | 3.75 | 2462.2 | 2.95 |
| 23 | 2381 | 2279.8 | 4.25 | 2285.0 | 4.03 | 2285.0 | 4.03 |
| 24 | 2459 | 2540.6 | −3.32 | 2518.5 | −2.42 | 2507.4 | −1.97 |
| 25 | 2505 | 2575.4 | −2.81 | 2538.8 | −1.35 | 2537.8 | −1.31 |
| 26 | 2429 | 2500.7 | −2.95 | 2480.5 | −2.12 | 2462.3 | −1.37 |
| 27 | 2438 | 2281.2 | 6.43 | 2351.0 | 3.57 | 2370.2 | 2.78 |
| 28 | 2748 | 2839.0 | −3.31 | 2808.2 | −2.19 | 2784.3 | −1.32 |
| 29 | 2388 | 2513.8 | −5.27 | 2481.8 | −3.93 | 2474.2 | −3.61 |
| 30 | 2175 | 2125.4 | 2.28 | 2198.9 | −1.10 | 2159.3 | 0.72 |
| 31 | 2539 | 2456.2 | 3.26 | 2476.3 | 2.47 | 2489.2 | 1.96 |
| MAPE | | | 3.87 | | 2.61 | | 2.29 |
| Iterations required | | | 2050 | | 1170 | | 890 |

$$\ldots, H_{max}(i-n_3)). \qquad (49)$$

For average load forecast a similar model as eqn. (49) is chosen, given by:

$$P_{av}(i) = f(P_{av}(i-n), P_{av}(i-1), \ldots, P_{av}(i-n_1), \Theta_{max}(i-n), \Theta_{max}(i), \Theta_{max}(i-1), \ldots, \Theta_{max}(i-n_2), \Theta_{min}(i-n), \Theta_{min}(i), \Theta_{min}(i-1), \ldots, \Theta_{min}(i-n_2), H_{max}(i-n), H_{max}(i), H_{max}(i-1), \ldots, H_{max}(i-n_3)). H_{min}(i-n), H_{min}(i), H_{min}(i-1), \ldots, H_{min}(i-n_3)). \qquad (50)$$

Here, $n$ indicates the lead time of the forecast ($n \neq n_1, n_2, n_3$). $P$, $\Theta$, and $H$ stand for load, temperature, and humidity, respectively. Also different values of $n_1$, $n_2$, and $n_3$ were tested to find the effect of the past data on the load forecast. With $n_1$, $n_2$, $n_3 > 0$ there was no marked improvement in the results. Therefore, we chose, $n_1 = n_2 = n_3 = 0$ in this study. However, the choice of $n_1$, $n_2$, $n_3$ depends entirely on the utility data concerned.

## 6.2. Scaling of the Variable Range

Because the input variables and the estimated ones from the hybrid neural network have wide variations in magnitudes, they will cause convergence problem and the system will behave completely erratic. To circumvent this problem, the variables are scaled between 0.1 and 0.9 (Rahman, Drezga, & Rajagopalan, 1993). This is performed so as to maximise accuracy and minimise training time. The following scheme is adopted to scale the variables between 0.1 and 0.9. Let $X_{i,max}$ and $X_{i,min}$ denote the upper and lower bounds of the observed range of feature $X_i$ in all the patterns in the historical data base considering numerical values only. Similarly, $O_{i,max}$ and $O_{i,min}$ denote the upper and lower bounds of the observed range of outputs. Then $X_i$ is normalised as

$$X_i^n = 0.1 + [0.8(X_i - X_{i,min})/(X_{i,max} - X_{i,min})] \qquad (51)$$

where ($i$, $j$) corresponds to the $j$th pattern of the $i$th training set.

**TABLE 5**
**Average Load Forecasting in January (Winter) Using 24 Hour Ahead Forecast**

| Day | Actual Load (MW) | Peak Forecast (NN) (MW) | P E (NN) | Peak Forecast (FES$_1$) (MW) | P E (FES$_1$) | Peak Forecast (FES$_2$) (MW) | P E (FES$_2$) |
|---|---|---|---|---|---|---|---|
| 1 | 2166 | 2223.4 | −2.65 | 2161.2 | 0.22 | 2163.5 | 0.12 |
| 2 | 2153 | 2209.8 | −2.64 | 2192.4 | −1.83 | 2120.8 | 1.49 |
| 3 | 2233 | 2218.1 | 0.67 | 2242.3 | −0.42 | 2216.6 | 0.73 |
| 4 | 2093 | 2092.1 | 0.04 | 2092.6 | 0.02 | 2089.0 | 0.19 |
| 5 | 2050 | 2041.0 | 0.44 | 2050.1 | −0.00 | 2050.0 | −0.00 |
| 6 | 2109 | 2109.3 | −0.01 | 2107.9 | 0.05 | 2108.2 | 0.04 |
| 7 | 1937 | 1933.4 | 0.18 | 1934.2 | 0.14 | 1936.9 | 0.00 |
| 8 | 1983 | 1974.9 | 0.41 | 1969.4 | 0.68 | 1976.2 | 0.34 |
| 9 | 2045 | 1991.6 | 2.61 | 2008.7 | 1.78 | 2080.1 | −1.72 |
| 10 | 2003 | 2036.6 | −1.68 | 2007.1 | −0.21 | 1998.2 | 0.24 |
| 11 | 1982 | 1954.9 | 1.37 | 1980.2 | 0.09 | 1980.5 | 0.07 |
| 12 | 2117 | 2116.2 | 0.04 | 2116.8 | 0.01 | 2116.6 | 0.02 |
| 13 | 2064 | 2035.6 | 1.37 | 2042.8 | 1.03 | 2049.9 | 0.70 |
| 14 | 2237 | 2262.3 | −1.13 | 2241.5 | −0.20 | 2240.6 | −0.17 |
| 15 | 2198 | 2239.1 | −1.87 | 2171.0 | 1.23 | 2221.3 | −1.06 |
| 16 | 2068 | 2065.3 | 0.13 | 2065.2 | 0.14 | 2067.1 | 0.04 |
| 17 | 2021 | 2057.0 | −1.78 | 2047.9 | −1.33 | 1994.9 | 1.29 |
| 18 | 1976 | 1953.7 | 1.13 | 1970.8 | 0.26 | 1982.1 | −0.31 |
| 19 | 2019 | 2019.9 | −0.04 | 2030.1 | −0.55 | 2007.3 | 0.58 |
| 20 | 2073 | 2090.0 | −0.82 | 2053.3 | 0.95 | 2087.2 | −0.68 |
| 21 | 2195 | 2224.6 | −1.35 | 2175.8 | 0.87 | 2203.5 | −0.39 |
| 22 | 2015 | 1982.8 | 1.60 | 2041.4 | −1.31 | 2003.3 | 0.58 |
| 23 | 1917 | 1866.8 | 2.62 | 1935.1 | −0.94 | 1936.8 | −1.03 |
| 24 | 1977 | 1943.5 | 1.70 | 1961.3 | 0.80 | 1961.0 | 0.81 |
| 25 | 2021 | 2017.5 | 0.17 | 2017.1 | 0.20 | 2024.1 | −0.15 |
| 26 | 1977 | 1961.6 | 0.78 | 1965.6 | 0.58 | 1970.9 | 0.31 |
| 27 | 1964 | 2010.8 | −2.38 | 1921.8 | 2.15 | 1939.0 | 1.27 |
| 28 | 2086 | 2124.2 | −1.83 | 2078.7 | 0.35 | 2077.9 | 0.39 |
| 29 | 1876 | 1889.6 | −0.72 | 1889.4 | −0.72 | 1883.0 | −0.37 |
| 30 | 1772 | 1751.3 | 1.17 | 1757.2 | 0.83 | 1765.5 | 0.37 |
| 31 | 1960 | 200.1 | −2.05 | 1928.1 | 1.63 | 1924.6 | 1.81 |
| MAPE | | | 1.21 | | 0.69 | | 0.56 |
| Iterations required | | | 1700 | | 1020 | | 680 |

Similarly, the output $O_i$ can be expressed

$$O_i^n = 0.1 + [0.8(O_i - O_{i,\min})/(O_{i,\max} - O_{i,\min})] \quad (52)$$

where $X_i^n$ and $O_i^n$ denote the normalised input and output vectors, respectively.

The normalised predicted values can be converted back to the actual values using the above expressions.

The ideas discussed in sections 6.1. and 6.2 are used for obtaining the peak and average load forecasts.

### 6.3. Peak Load Forecasting

For $n$ h ahead peak load forecasting, the following training data are used for the ANN:

Input pattern: $P_{\max}(i)$, $\Theta_{\max}(i)$, $H_{\max}(i)$, $\Theta^f_{\max}(i+n)$, $H^f_{\max}(i+n)$

Output pattern: $P_{\max}(i+n)$ for the ANN.

Superscript $f$ denotes the forecasted values for $\Theta$ and $H$. The forecasted values for $(i+n)$th day are used to get a more realistic forecast.

For FES$_1$ and FES$_2$ the training patterns used are:

Input pattern: $\Delta\Theta_{\max}(i, i+n)$ and $\Delta H_{\max}(i, i+n)$

Output pattern: $e_{\mathrm{LC}}(i)$, the desired load correction.

The $P_{\max}(i+n)$ for FES$_1$ and FES$_2$ is obtained using eqn. (7).

Table 1 gives the learned membership functions for 24 h ahed peak load forecasting in winter using FES$_1$, for example, rule 0 is interpreted as:

RO: IF $\Delta\Theta_{\max}$ is term 0 and $\Delta H_{\max}$ is term 3
THEN $\hat{e}_{\mathrm{LC}}$ is term 7.

Figure 2 gives the learned membership functions after the first phase (unsupervised learning phase) and the

**TABLE 6**
**Average Load Forecast in June (Summer) Using 48 Hour Ahead Forecast**

| Day | Actual Load (MW) | Peak Forecast (NN) (MW) | P E (NN) | Peak Forecast ($FES_1$) (MW) | P E ($FES_1$) | Peak Forecast ($FES_2$) (MW) | P E ($FES_2$) |
|---|---|---|---|---|---|---|---|
| 1 | 1521 | 1541.1 | −1.32 | 1509.7 | 0.74 | 1516.7 | 0.28 |
| 2 | 1531 | 1562.5 | −2.06 | 1512.3 | 1.22 | 1549.5 | −1.21 |
| 3 | 1473 | 1455.2 | 1.21 | 1453.1 | 1.35 | 1456.5 | 1.12 |
| 4 | 1336 | 1380.9 | −3.36 | 1370.3 | −2.57 | 1305.9 | 2.25 |
| 5 | 1285 | 1254.0 | 2.41 | 1260.8 | 1.88 | 1272.9 | 0.94 |
| 6 | 1433 | 1446.2 | −0.92 | 1443.5 | −0.73 | 1423.5 | 0.66 |
| 7 | 1406 | 1375.5 | 2.17 | 1386.6 | 1.38 | 1418.5 | −0.89 |
| 8 | 1403 | 1442.3 | −2.80 | 1437.0 | −2.42 | 1418.0 | −1.07 |
| 9 | 1428 | 1461.0 | −2.31 | 1450.7 | −1.59 | 1448.8 | −1.46 |
| 10 | 1431 | 1467.3 | −2.54 | 1459.8 | −2.01 | 1411.3 | 1.38 |
| 11 | 1301 | 1272.8 | 2.17 | 1286.0 | 1.15 | 1322.7 | −1.67 |
| 12 | 1235 | 1206.7 | 2.29 | 1217.8 | 1.39 | 1219.9 | 1.22 |
| 13 | 1418 | 1394.7 | 1.64 | 1419.3 | −0.09 | 1419.8 | −0.13 |
| 14 | 1408 | 1395.5 | 0.89 | 1401.9 | 0.43 | 1399.8 | 0.58 |
| 15 | 1421 | 1451.3 | −2.13 | 1443.0 | −1.55 | 1439.6 | −1.31 |
| 16 | 1447 | 1480.1 | −2.29 | 1468.4 | −1.48 | 1429.9 | 1.18 |
| 17 | 1440 | 1420.7 | 1.34 | 1425.3 | 1.02 | 1428.3 | 0.81 |
| 18 | 1352 | 1313.9 | 2.82 | 1324.8 | 2.01 | 1330.5 | 1.59 |
| 19 | 1305 | 1279.8 | 1.93 | 1282.6 | 1.72 | 1290.3 | 1.13 |
| 20 | 1430 | 1407.5 | 1.57 | 1416.6 | 0.94 | 1440.3 | −0.72 |
| 21 | 1383 | 1387.8 | −0.35 | 1395.3 | −0.89 | 1389.2 | −0.45 |
| 22 | 1414 | 1434.1 | −1.42 | 1425.0 | −0.78 | 1401.1 | 0.91 |
| 23 | 1423 | 1386.9 | 2.54 | 1404.1 | 1.33 | 1410.9 | 0.85 |
| 24 | 1406 | 1361.6 | 3.16 | 1370.4 | 2.53 | 1373.1 | 2.34 |
| 25 | 1267 | 1230.6 | 2.87 | 1250.3 | 1.32 | 1248.6 | 1.45 |
| 26 | 1221 | 1254.3 | −2.73 | 1207.9 | 1.07 | 1234.4 | −1.10 |
| 27 | 1397 | 1371.7 | 1.81 | 1401.7 | −0.34 | 1395.9 | 0.08 |
| 28 | 1401 | 1374.1 | 1.92 | 1384.3 | 1.19 | 1384.9 | 1.15 |
| 29 | 1403 | 1415.2 | −0.87 | 1416.5 | −0.96 | 1389.3 | 0.98 |
| 30 | 1411 | 1376.1 | 2.47 | 1393.6 | 1.23 | 1390.1 | 1.48 |
| MAPE | | | 2.01 | | 1.31 | | 1.08 |
| Iterations required | | | 1630 | | 1520 | | 1070 |

second phase (supervised learning phase). Figure 3 gives the mean absolute percentage errors (MAPE's) versus the number of iterations. The results in Figures 2 and 3 are obtained for 24 h ahead load forecasting in January (winter). From Figure 3 we find that the $FES_2$ gives fastest convergence followed by $FES_1$ and ANN. The convergence speed of the $FES_2$ was found to be superior because of the linear Kalman filter equations used for weight update and the error-dependent forgetting factor was responsible for driving the MAPE low during the first few hundred iterations until the bias introduced by the initial conditions was eliminated.

Table 2 gives the peak load forecasting results in terms of mean absolute percentage errors (MAPEs) for ANN, $FES_1$ and $FES_2$ in the month of June (winter) using 24 h ahead forecast. Tables 3 and 4 give the results for the month of January (winter) using 48 h and 168 h ahead forecast, respectively.

From Tables 2, 3 and 4 we see that the $FES_1$ and $FES_2$ give better prediction accuracy compared to ANN. Also we find that the results for 48 h and 168 h predictions are comparable with that of the 24 h ahead predictions. This is because the load forecasting was performed as a one-step process (i.e. looking 24 h ahead, 48 h ahead, and so on). However, as the lead time increased to 168 h the PEs were found to be greater than 4% even with $FES_2$. As the primary aim of this paper was to make a comparative assessment between ANN, $FES_1$ and $FES_2$, no attempt was made to improve the forecast errors further.

## 6.4. Daily Average Load Forecast

For $n$ h ahead average load forecasting, the following training data are used for ANN.

Input pattern: $P_{av}(i)$, $\Theta_{max}(i)$, $\Theta_{min}(i)$, $H_{max}(i)$, $H_{min}(i)$, $\Theta^f_{max}(i+n)$, $\Theta^f_{min}(i+n)$, $H^f_{max}(i+n)$, $H^f_{min}(i+n)$

Output pattern: $P_{av}(i+n)$ for ANN.

For $FES_1$ and $FES_2$, the training patterns used are:

**TABLE 7**
**Average Load Forecasting in June (Summer) Using 168 Hour Ahead Forecast**

| Day | Actual Load (MW) | Peak Forecast (NN) (MW) | P E (NN) | Peak Forecast ($FES_1$) (MW) | P E ($FES_1$) | Peak Forecast ($FES_2$) (MW) | P E ($FES_2$) |
|---|---|---|---|---|---|---|---|
| 1 | 1521 | 1558.6 | −2.47 | 1542.9 | −1.44 | 1538.2 | −1.13 |
| 2 | 1531 | 1558.9 | −1.82 | 1551.8 | −1.36 | 1550.6 | −1.28 |
| 3 | 1473 | 1436.0 | 2.51 | 1452.7 | 1.38 | 1456.5 | 1.12 |
| 4 | 1336 | 1409.3 | −5.49 | 1388.2 | −3.91 | 1364.9 | −2.16 |
| 5 | 1285 | 1244.1 | 3.18 | 1250.8 | 2.66 | 1261.6 | 1.82 |
| 6 | 1433 | 1409.4 | 1.65 | 1428.4 | 0.32 | 1434.6 | −0.11 |
| 7 | 1406 | 1377.5 | 2.03 | 1385.3 | 1.47 | 1385.9 | 1.43 |
| 8 | 1403 | 1441.2 | −2.72 | 1435.3 | −2.30 | 1431.1 | −2.00 |
| 9 | 1428 | 1484.1 | −3.93 | 1471.8 | −3.07 | 1448.7 | −1.45 |
| 10 | 1431 | 1469.6 | −2.70 | 1447.0 | −1.12 | 1416.5 | 1.01 |
| 11 | 1301 | 1358.0 | −4.38 | 1345.8 | −3.44 | 1330.0 | −2.23 |
| 12 | 1235 | 1202.5 | 2.63 | 1212.5 | 1.82 | 1221.7 | 1.08 |
| 13 | 1418 | 1390.3 | 1.95 | 1403.0 | 1.06 | 1432.0 | −0.99 |
| 14 | 1408 | 1394.9 | 0.93 | 1399.7 | 0.59 | 1411.0 | −0.21 |
| 15 | 1421 | 1484.4 | −4.46 | 1453.7 | −2.30 | 1451.8 | −2.17 |
| 16 | 1447 | 1482.3 | −2.44 | 1473.2 | −1.81 | 1463.8 | −1.16 |
| 17 | 1440 | 1402.7 | 2.59 | 1412.1 | 1.94 | 1415.5 | 1.70 |
| 18 | 1352 | 1308.5 | 3.22 | 1357.7 | −0.42 | 1358.1 | −0.45 |
| 19 | 1305 | 1269.2 | 2.74 | 1290.0 | 1.15 | 1295.0 | 0.77 |
| 20 | 1430 | 1404.0 | 1.82 | 1406.3 | 1.66 | 1441.9 | −0.83 |
| 21 | 1383 | 1350.4 | 2.36 | 1363.2 | 1.43 | 1399.7 | −1.21 |
| 22 | 1414 | 1463.1 | −3.47 | 1448.6 | −2.45 | 1396.6 | 1.23 |
| 23 | 1423 | 1371.5 | 3.62 | 1393.0 | 2.11 | 1404.8 | 1.28 |
| 24 | 1406 | 1359.5 | 3.31 | 1372.1 | 2.41 | 1389.7 | 1.16 |
| 25 | 1267 | 1219.4 | 3.76 | 1237.6 | 2.32 | 1254.2 | 1.01 |
| 26 | 1221 | 1256.7 | −2.92 | 1241.6 | −1.69 | 1236.1 | −1.24 |
| 27 | 1397 | 1372.7 | 1.74 | 1384.6 | 0.89 | 1387.9 | 0.65 |
| 28 | 1401 | 1373.1 | 1.99 | 1379.7 | 1.52 | 1380.5 | 1.46 |
| 29 | 1403 | 1414.4 | −0.81 | 1405.5 | −0.18 | 1399.5 | 0.25 |
| 30 | 1411 | 1359.6 | 3.64 | 1374.9 | 2.56 | 1378.1 | 2.33 |
| MAPE | | | 2.78 | | 1.76 | | 1.23 |
| Iterations required | | | 2650 | | 2480 | | 1310 |

Input pattern: $\Delta\Theta_{max}(i, i+n)$, $\Delta\Theta_{min}(i, i+n)$, $\Delta H_{max}(i, i+n)$, $\Delta H_{min}(i, i+n)$
Output pattern: $e_{LC}(i)$, the desired load correction.

Again forecasted temperature and humidity values are used for the day of the forecast.

The $P_{av}(i+n)$ for $FES_1$ and $FES_2$ is obtained using eqn. (7).

Table 5 gives the average load forecasting results, number of iterations for convergence, PEs and the MAPEs for the ANN, $FES_1$, and $FES_2$ models in the month of January (winter) using 24 h ahead forecast. Tables 6 and 7 give the results for 48 h and 168 h ahead predictions in the month of June (summer).

Again, from Tables 5, 6, and 7 we find the improved performance of $FES_2$ in terms of faster convergence and improved overall accuracy over the ANN and $FES_2$.

## 7. DISCUSSION

The fuzzy expert system presented in this paper is constructed from training examples by machine learning techniques, and the neural network model is trained to develop fuzzy logic rules and find input/output membership functions. By combining both unsupervised and supervised learning schemes, the learning speed for time series forecasting problems converges much faster than original back-propagation learning algorithm. Further, by using a linear Kalman filter for the supervised learning phase, the learning time is shortened with rapid convergence due to the adaptive nature of the learning algorithm. The fuzzy expert system using a linear Kalman filter produces a more accurate load forecast for lead times varying from 24 h to 168 h in comparison to the one using gradient descent back-propagation algorithm. The forecasting results presented in Section 6 are based upon only true temperature information because all the data for this utility are historical. However, in reality the temperature information will be the forecasted value and this will add to the forecast error. The developed expert systems are used to forecast the load during weekends but no attempt is made to forecast for days with unusual events. Such events will require extensive data analysis to track the events in that day and to select training cases accordingly from the previous days with similar events. Such activities are continuing, and will be reported in a future paper.

As a final note, a shorter period is used for training as load patterns change very fast. But training and prediction can be done over longer periods with less chaotic data having strong correlation to any particular weather and environmental variable. This flexibility permits the adaptation of the proposed fuzzy expert systems for producing accurate load forecasts for different geographic areas.

## 8. CONCLUSIONS

This paper presents the development of a fuzzy expert system using a hybrid neural network approach to predict peak and daily average load profiles in an energy management system. The fuzzy expert system is modelled as a hybrid neural network and uses fuzzy membership values of load and weather parameters. A hybrid learning scheme consisting of both self-organized and supervised learning phases is used for training the hybrid neural network. Further, the paper presents simulation results using both back-propagation and Kalman filter-based algorithms, the latter producing faster convergence during training and more accurate predictions.

**Acknowledgement**—The authors gratefully acknowledge the funds from the National Science Foundation (NSF Grant No. INT-9209103 and INT-9117624), USA for undertaking this research.

## REFERENCES

Box, G. E. P., & Jenkins, G. M. (1976). *Time series analysis forecasting and control*, Oakland, CA: Holden-Day.

Brace, M. C. (1991). A comparison of the forecasting accuracy of neural networks with other established techniques, *First International Forum on Applications on Neural Networks to Power Systems*, Seattle, WA, July, 1991, pp. 23–26.

Buckley, J. J., Hayashi, Y., & Czogala, E. (1993) On the equivalence of neural nets and fuzzy expert systems, *Fuzzy Sets Systems*, **53** 129–134.

Bunn, D. W., & Farmer, E. D. (1989). *Comparative models for electric load forecasting*, New York: John Wiley and Sons.

Dash, P. K., Dash, S., Rama Krishna, G., & Rahman, S. (1993). Forecasting of a load time series using a fuzzy expert system and fuzzy neural networks, *Engineering International Systems*, **1**(2), 103–117.

Dash, P. K., Satpathy, J. K., Rama Krishna, G., & Rahman, S. (1993). An improved kalman filter based neural network approach for short-term load forecasting, *Proceedings of the Third International Symp. on Electricity Distribution and Energy Management*, Vol. 1, pp. 3–8.

El-Sharkawi, M. A., Oh, S., Marks, R. J., Dambourg, M. J., & Brace, C. M. (1991). Short term electric load forecasting using an adaptively trained layered perceptron, *First International Forum on Applications of Neural Networks to Power Systems*, Seattle WA., July 23–26, 1991, pp. 3–6.

Hayashi, Y., Buckley, J. J., & Czogala, E. (1992). Fuzzy expert systems versus neural networks, *Proceedings of International Joint Conference on Neural Networks*, Baltimore, Vol. II, June 7–12 pp. 720–726.

Haykin, S. S. (1986). *Adaptive filter theory* (pp. 312–314). Englewood Cliffs, NJ: Prentice-Hall.

Ho, K. L., Hsu, Y. Y., & Yang, C. C. (1992). Short term load forecasting using a multilayer neural network with an adaptive learning algorithm, *IEEE Transactions on Power Systems*, **7**(1), 141–150.

Lee, C. C. (1990). Fuzzy logic in control systems: Fuzzy logic controller, part 2, *IEEE Transactions Systems, Man & Cybernetics*, **20**(3), 266–275.

Lee, K. Y., Cha, Y. T., & Park, J. H. (1992). Short term load forecasting using an artificial neural network, *IEEE Transactions on Power Systems*, **7**(1), 124–133.

Lin, C. T., & Lee, G. C. S. (1991). Neural network-based fuzzy logic control and decision system, *IEEE Transactions on Computers*, **40**(12), 1320–1336.

Moghram, I., & Rahman, S, (1989). Analysis and evaluation of five short-term load forecasting techniques, *IEEE Transactions on Power Systems*, **4**(4), 1484–1490.

Pal, S. K., & Mitra, S. (1992). Multilayer perceptrons, fuzzy sets and classifications, *IEEE Transactions on Neural Networks*, **3**(5), 683–698.

Park, D. C., El-Sharkawi, M. A., & Marks, R. J., (1991). Adaptively trained neural networks, *IEEE Transactions on Neural Networks*, 224–245.

Park, J. H., Park, Y. M., & Lee, K. Y. (1991). Composite models for adaptive short term load forecasting, *IEEE Transactions on Power Systems*, **1**(2), 450–457.

Peng, T. M., Hubele, N. F., & Karady, G. G. (1992). Advancement in the application of neural networks for short term load forecasting, *IEEE Transactions on Power Systems*, **7**(1), 250–258.

Rahman, S., Drezga, I., & Rajagopalan, J. (1993). Knowledge enhanced connectionist models for short-term electric load forecasting, *International Conference on ANN applications to Power Systems*, Yokohama, Japan, April.

Rumelhart, D. E., & Zipses, D. (1985). Feature discovery by competitive learning. *Cognitive Science*, **9**, 75–112.

Scalero, R. S., & Tepedelenlioglu, N. (1992). A fast algorithm for training feedforward neural networks, *IEEE Transactions on Signal Processing*, **40**(1), 202–210.

Singhal, S., & Wu, L. (1989). Training feed-forward network with the extended Kalman algorithm, *Proceedings of ICASSP*, pp. 1187–1190.

Singhal, S., & Wu, L. (1989). Training feed-forward network with the extended Kalman algorithm. *Proceedings of ICASSP*, pp. 1187–1190.

Wang, L.-X. & Mendel, J. M. (1992). Generating fuzzy rules by learning from examples, *IEEE Transactions on Systems Man & Cybernetics*, **22**(6), 1414–1427.

Zadeh, L. A. (1965). Fuzzy sets, *Information and Control*, **8**, 338–358.