COMPARISON OF ADVANCED LEARNING ALGORITHMS FOR SHORT-TERM LOAD FORECASTING

V.S. KODOGIANNIS

12-99

Preprint no. 12-99/1999

Department of Computer Science
University of Ioannina
451 10 Ioannina, Greece

Comparison of advanced learning algorithms for short-term load forecasting.

V.S. Kodogiannis

Department of Computer Science, University of Ioannina, Ioannina, GR-45110, Greece,
Tel: +30-651-97308, Fax: +30-651-48131, Email: vassilis@cs.uoi.gr

ABSTRACT: Neural networks are currently finding practical applications, ranging from 'soft' regulatory control in consumer products, to the accurate modelling of nonlinear systems. This paper presents the development of improved neural-network-based short-term electric load forecasting models for the power system of the Greek Island of Crete. Several approaches, including radial basis function networks and dynamic neural networks, have been proposed, and are discussed in this paper. Their performances are evaluated through a simulation study, using metered data provided by the Greek Public Power Corporation. The results indicate that the load-forecasting models developed in this way provide more accurate forecasts, compared to conventional backpropagation network forecasting models. Finally, the embedding of the new model capability in a modular forecasting system is presented.

Key words: Short-term load forecasting, neural networks, radial basis functions, dynamic neural networks.

1. INTRODUCTION

Load forecasting has been a central and integral process in the planning and operation of electric utilities. Load forecasting, with lead-times from a few minutes to several days, helps the system operator to schedule spinning reserve allocation efficiently. In addition, load forecasting can provide information that can be used for possible energy interchange with other utilities. On the other hand, load forecasting is also useful for power-system security. If applied to the system security assessment problem, valuable information can be obtained, in order to detect vulnerable situations in advance.

The load-prediction period may be a month or a year for the long- and medium-term forecasts, and a day or an hour for short-term forecasts. The long- and medium-term forecasts are used to determine the capacity of generation, transmission, or distribution system additions, and the type of facilities required in expansions in transmission planning, annual hydro and thermical maintenance scheduling, etc. The short-term forecasts are needed for the control and scheduling of the power system, and also as inputs to load flow studies or contingency analyses.

The short-term load forecast (one to twenty-four hours) is of importance in the daily operations of a power utility. It is required for unit commitment, energy transfer scheduling and load dispatch. With the emergence of load-management strategies, the short-term

forecast is playing a broader role in utility operations. The development of an accurate, fast and robust short-term load-forecasting methodology is of importance to both the electric utility and its customers, thus introducing higher accuracy requirements.

Many techniques have been proposed in the last few decades for short-term load forecasting (STLF) (IEEE Comm. Rep., 1980). The traditional techniques that have been used for STLF include Kalman filtering, the Box and Jenkins method, regression models, the autoregressive model and the spectral expansion techniques (Moghram *et al.*, 1989).
The Kalman filter approach requires the estimation of a covariance matrix. The possible high non-stationarity of the load pattern, however, may typically preclude the making of accurate estimates (Mehra, 1969). The Box-Jenkins method requires an autocorrelation function for the identification of proper ARMA models. This can be accomplished by using pattern-recognition techniques. A major obstacle here is its slow performance (Vemuri *et al.*, 1981). An interesting application to STLF using regression procedures has been proposed by Irisarri *et al.* (1982). However, their Generalised Linear Square Algorithm was faced with numerical instabilities when applied to a large database. Additionally, enormous computational efforts are required to produce reasonable results.

The AR model is used to describe the stochastic behaviour of the hourly load pattern on a power system. This model assumes that the load at a specific hour can be estimated by a linear combination of the previous few hours. Generally, the larger the data set the better the result in terms of accuracy. A longer computational time, however, is required for the parameter identification.

Recently, with the developments in artificial intelligence, alternative solutions to the STLF problem have been proposed. Expert systems have been successfully applied to STLF (Rahman and Bhatnager, 1988). This approach, however, presumes the existence of an expert who is capable of making accurate forecasts, and who will train the system. Neural networks, on the other hand, are a more promising area of artificial intelligence since they do not rely on human experience but attempt to learn by themselves, the functional relationship between system inputs and outputs.

Interest in applying artificial neural networks (ANNs) to STLF began recently. Park *et al.* (1991a) proposed the use of a multi-layer network with three layers, *i.e* one input, one hidden, one output. The training of the network is performed through a simple back-propagation algorithm. Using load and weather information the system produces three different forecast variables, *i.e.* peak load, total daily load and hourly load. Peng *et al.* (1992) presented a search procedure for selecting the training cases for ANNs to recognise the relationship between the weather changes and the load shape. It utilises a neural-network algorithm that includes a combination of linear and nonlinear terms, which map past load and temperature inputs to the load forecast output. The single forecast output is the total daily load. Lu *et al.* (1993) used a similar feedforward neural network incorporating the previous load demands, the day of the week, the hour of the day and temperature information for load forecasting. Papalexopoulos *et al.* (1994) proposed the inclusion of additional input variables, such as a seasonal factor and a cooling/heating degree, into a single neural network. Rather questionable results are reported by Bakirtzis *et al.* (1996), where a 24-hour prediction can be

obtained by an enormous three-layer neural network. Although the number of hidden nodes is much smaller than the input dimension, the authors claim that this number does not significantly affect the forecasting accuracy. Lee *et al.* (1991) treat electric load demands as a non-stationary time-series, and they modelled the load profile by a recurrent neural network. The forecast was made separately for weekends and weekdays, using load data only. Of the above neural based forecasting techniques, most of them can generally be classified into two categories in accordance with techniques they employ. One approach treats the load pattern as a time-series signal, and predicts the future load by using the techniques already mentioned. In the second approach, the load pattern is considered to be heavily dependent on both weather variables (temperature, humidity, etc.) and previous load patterns. Such models, that include weather variables, are limited in their use by problems such as the inaccuracy of weather forecasts and difficulties in modelling the weather-load relationship (Park *et al.*, 1991b).

This paper presents algorithms that follow the time-series approach. In this case, ANNs trace previous load patterns, and predict a load pattern using recent load data without using weather information for modelling. The systems developed here identify the load model that reflects the stochastic behaviour of the hourly load demand for the island of Crete in Greece. The remainder of this paper contains a comparative study of various prediction techniques used to develop the STLF for the power system of Crete.

2. THE SHORT-TERM LOAD-FORECASTING PROBLEM

This section deals with the application of the STLF models to the power system of the island of Crete. The objective of STLF is to predict the 24-hourly loads for the next day. Here, a modular forecasting system is proposed, where 24 neural blocks with a single output have to be developed and trained separately, to represent the 24 hourly loads respectively. The outline of the proposed architecture is illustrated in Fig. 1.

Each neural block is fed by the previous one. Hence, step by step, 24-hour load prediction can be obtained. In this work, the training data set was created from the whole year of 1994, and the testing one for the first four months of 1995. Both training and testing sets were classified into 24 time series, each one corresponding to a different hour of the day. Therefore, for each neural block, the following nonlinear load model is proposed for one-hour-ahead forecasting:

$$x(t) = F\{ x(t-24i), x(t-j), x(t-24i-j) \}$$
$$i = 1,\ldots,p \qquad j = 1,\ldots,q \tag{1}$$

where, p and q indicate the number of previous days and hours respectively. The main objective of the proposed system is the development of sufficiently accurate blocks, representing the individual hourly loads. The investigation of neural networks (NNs) for system modelling has been carried out for the prediction of all 24 hours, each one trained separately, one for each hour. An assumption has been made in the case of blackouts, which occurred during the whole year. All the zero load values have been removed from both

training and testing sets, and were replaced by the mean value of the preceding and subsequent load value.

Eq. (1) is used to provide the input variables to the individual blocks. According to Eq. (1), the requested load is forecast using not only the load data of previous days, but also the forecast load data for the same day at previous time steps. In the following sections, only results that correspond to hours with the maximum (14:00h) and minimum (02:00h) load consumption are illustrated.

3. AUTOREGRESSIVE LINEAR MODELLING

Although this work is primarily oriented to neural-network approaches to STLF, it was assumed that it was necessary to test the well-known stochastic autoregressive model for use with this type of data. An autoregressive (AR) model of a time series assumes that the current value of the time series is equal to a weighted average of a finite number of previous values, with the addition of a constant offset and a random noise term.

The original AR model is expressed as

$$x_t = \phi_0 + \phi_1 x_{t-1} + \phi_2 x_{t-2} + \ldots + \phi_p x_{t-p} + \alpha_t \qquad (2)$$

where $x_t, x_{t-1}, \ldots, x_{t-p}$ are the terms of time series, $\phi_1, \phi_2, \ldots, \phi_p$ are the unknown coefficients of the model , ϕ_o is a constant term, and α_t is the random noise term.

The main objective here, is the computation of the unknown parameters of the AR model $\phi_1, \phi_2, \ldots, \phi_p$. The recursive least-squares method was used, in conjunction with the well-known U-D algorithm (Press and Teukolsky, 1992) for optimisation of the numerical calculations. The order p of the AR model was approximated using the well-known Akaike's information criterion. Hence, for the majority of the test cases, it has been found that p=4. Attempts to increase the model's order for the sake of improved accuracy resulted a severe deterioration of its performance.

4. BACKPROPAGATION ALGORITHMS FOR STLF PREDICTION.

Some artificial neural-network architectures exhibit the capability to form complex mappings between an input and an output, which enable the network to approximate general non-linear mathematical functions. The multi-layer perceptron (MLP) neural network, trained by the standard back-propagation (BP) algorithm, is probably the most widely used network, and its mathematical properties for nonlinear function approximation are well documented (Rumelhart and McClelland, 1986). The generalised delta rule is applied for the adjustment of the weights of the feedforward networks in order to minimise a predetermined cost error function. The rule for adjusting the weights is given by the following equation:

$$w_{ij}^{p}(t+1) = w_{ij}^{p}(t) + \eta \delta_j^{p} y_i^{p} + \alpha \Delta w_{ij}^{p}(t). \qquad (3)$$

where η is the learning rate parameter, α the momentum term, and δ is the negative derivative of the total square error with respect to the neuron's output.

It is well known that the training of MLPs using the standard backpropagation algorithm is plagued by slow convergence. In this work, a simple heuristic strategy, the Adaptive learning rate Backpropagation Neural Network (ABP), which relates the learning rate to the total error function E have been adopted, in order to accelerate the convergence speed (Vogl *et al.*, 1988). The algorithm uses the batch update mode, and the update rule for the weights is

$$w_{ij}^{p}(t+1) = w_{ij}^{p}(t) - \frac{\rho(E)\dfrac{\vartheta E}{\vartheta w_{ij}^{p}}}{\left\| \dfrac{\vartheta E}{\vartheta w_{ij}^{p}} \right\|^{2}} \tag{4}$$

with $\rho(E)$ is some function of the error E, which is given by

$$\rho(E) = \begin{cases} \eta \\ \eta E \\ \eta \tanh\left(\dfrac{E}{E_o}\right) \end{cases} \tag{5}$$

where η and E_o are constant, non-negative numbers, representing the learning rate and the error normalisation factor, respectively. The main advantage of using these formulas is the dependence of the learning rate on the instantaneous value of the total squared error E. Therefore, faster convergence of the algorithm is achieved.

In order to provide sufficient information to model each neural block using an MLP, a structure with two hidden layers and eight inputs was used. The inputs were selected from the model of Eq. (1) by setting the parameters p and q equal to 2.

A common approach, taken to enable a neural network to capture the dynamics of a non-linear system, as the load forecasting problem, is to configure and train a network to represent a non-linear, autoregressive (NAR) model structure. It is natural to reflect the dynamic nature of the problem by sequential information processing. There are two approaches to processing inputs in the time domain: one is to window the inputs and then treat the time domain like another spatial domain; alternatively, it is possible to use some internal storage to maintain a current state.

In the first approach, the network input consists of a moving window of time-delayed system outputs, and the network is trained to predict the system output at the next time step. It has been found that a major factor affecting the neural model prediction accuracy is the method by which data are coded in the network. This is especially true in the case when the network is acting as a recurrent model. The reason for this is that any errors in the predicted output will tend to accumulate. The conventional method of conditioning the data is to re-scale and represent then using a single node at the input or output layers of the network. An alternative representation, called spread encoding (SE), has been shown to enable a network to maintain a high degree of accuracy (Kodogiannis *et al.*, 1993).

In the SE technique, each data value is represented as the mean value of a sliding Gaussian pattern of excitation over several nodes at the network input and output. A similar and reverse procedure is applied at the network output to decode the output back into the

original variable range. This approach has similarities with data-fuzzification techniques, where the scalar dimensional space of each variable is fuzzified to a space of higher dimensions. Also, decoding of the network output using the SE method consists of computing a weighted summation of the node excitations, which is analogous to the conventional centre-of-gravity defuzzification technique. Thus, a network utilising spread encoding can be considered as a fuzzy-neural-type network. The SE method is also in keeping with the heritage of neural networks from biological systems where information is often represented by the combined activity of a population of receptors, as in the retina of the eye (Marr and Hildreth, 1980). Fig. 2 illustrates the internal architecture of this technique.

Analytically, this data-conditioning method of SE consists of mapping each network variable, $x \in [x_{min}, x_{max}]$, onto a sliding Gaussian activation pattern of N network nodes, which includes additional nodes on either side of the variable range to contain the overspill resulting from the use of a mapping function with wide support. The level of activation of each node is confined to the range [0.1, 0.9], similar to the conventional normalisation technique. Each node is assigned a value, α_i, linearly spaced by a distance, δ, to span the range of x, and the centre of the Gaussian excitation pattern corresponds to the coded value (Kodogiannis, 1994).

The SE algorithm is derived by creating a discrete map that represents the mean value of a continuous probability distribution, $\varphi(\alpha)$, within each class interval. This then provides a simple mechanism for retrieving the original coded value as a sum of the activity of the node excitations, each weighted by the value at the centre of the relevant class interval α_i. For a particular value of x, the excitation of each node is defined by

$$\psi_i(x) = \frac{\int_{\alpha_i - \delta/2}^{\alpha_i + \delta/2} \alpha \, \varphi(\alpha - x) d\alpha}{\alpha_i} \tag{6}$$

which satisfies the requirement that

$$\sum_{i=1}^{N} \alpha_i \psi_i(x) = \int \alpha \, \varphi(\alpha - x) d\alpha = \bar{\alpha} = x \tag{7}$$

It is assumed that the distribution $\varphi(\alpha)$ has a unit area.

The activation of a particular node can be evaluated from Eq. (6) by integration by parts:

$$\alpha_i \psi_i(x) = [\alpha \, \Phi(\alpha - x)]_{\alpha_i - \delta/2}^{\alpha_i + \delta/2} - \int_{\alpha_i - \delta/2}^{\alpha_i + \delta/2} \Phi(\alpha - x) d\alpha \tag{8}$$

where $\Phi(\alpha)$ is a parent cumulative distribution with $\varphi(\alpha) = \Phi'(\alpha)$. In the investigations reported in this paper, the integral term in Eq. (8) was approximated using the first two terms in the trapezium rule, resulting in

$$\psi_i(x) \approx \Phi(\alpha_i + \delta/2 - x) - \Phi(\alpha_i - \delta/2 - x) \tag{9}$$

which was found to provide sufficient accuracy in these studies. The relationship between this coding technique and conventional fuzzification techniques is illustrated by considering a first approximation of the integral term in Eq. (8), resulting from a Taylor series expansion of the cumulative function about the interval centre, α_i, and keeping the linear term in the expansion. This leads to

$$\psi_i(x) \approx \varphi(\alpha_i - x) \tag{10}$$

which is analogous to the use of membership functions in fuzzy logic (Zadeh, 1965).

The practical advantage of spread encoding is that it leads to more-accurate models using static feed-forward neural networks, than representing normalised physical variables using single nodes. The main reason for this is that signal noise is reduced in the spread encoding representations by suitable matching of the coding function with the interval width spanned by each node, hence exploiting the network's fault tolerance.

The spread encoding algorithm was implemented by initially scaling the data to a normalised range, where the original data range $r \in [r_{min}, r_{max}]$ was represented by $x \in [0, N-2N_0]$ with N the total number of nodes, N_0 the number of nodes on either side of the variable range, and δ=1. The data coding isperformed using the following relation:

$$\psi_i(x) = \Phi(\alpha_i + 1/2 - x) - \Phi(\alpha_i - 1/2 - x), \quad i = 1,..:,N \tag{11}$$

where

$$\alpha_i = i - N_0 - c \tag{12}$$

and the cumulative Gaussian distribution function was approximated by the sigmoidal function centred at x:

$$\Phi(\alpha - x) = \frac{1}{1+e^{-\beta(\alpha - x)}} \tag{13}$$

In Eq. (12), c is an offset term that shifts the position of the range limits on the nodes. The width of the node excitations is inversely controlled by the parameter β in Eq. (13). Errors arise in decoding using straight-forward applications of Eq. (7) because the node excitations, $\psi_i(x)$, are calculated by an approximation, Eq. (9). The accuracy of decoding is improved by dividing the weighted sum by the sum of the node excitations. Thus, the network output is decoded back to the normalised range using

$$x = \frac{\sum_{i=1}^{N} \alpha_i \psi_i(x)}{\sum_{i=1}^{N} \psi_i(x)} \tag{14}$$

which is analogous to the conventional centre-of-gravity defuzzification technique. In this work, the parameters used in the spread encoding algorithm were N=6, N_0=2, c=0.5 and β=2.3 which were found to provide sufficiently accurate coding and decoding in the application reported in this paper.

Formal techniques for determining an optimum number of nodes in the hidden layers are still an area of current research; presently, this task is often achieved by experimentation. The resulting MLP network topology with the spread encoding applied to the network data, was 48 input nodes, 34 and 16 nodes for the two hidden layers, and 6 output nodes.

5. RADIAL BASIS FUNCTIONS NETWORKS

An alternative model to the multilayer networks for the time-series identification is the neural network employing radial basis functions (RBFs). An RBF is a function that has an in-built distance criterion with respect to a centre. A typical RBF neural network consists of three layers (input, hidden, output). The activation of a hidden neuron is determined in two steps:

the first involves computing the distance (usually by using the Euclidean norm) between the input vector and a center c_i, which represents the i^{th} hidden neuron. Second, a function h (which is usually bell-shaped) is applied, using the previously distance to get the final activation of the hidden neuron. In this case, the well-known Gaussian function

$$G(x)= \exp(-\frac{x^2}{\sigma^2}) \tag{15}$$

was used. The parameter σ is called the «unit width», and is determined using the «*global first nearest-neighbor*» heuristic rule (Moody and Darken, 1989). It uses a uniform average width for all units, using the Euclidean distance in the input space between each unit m and its nearest neighbor n. All the widths in the network are fixed to the same value σ, and this result in a simpler training strategy.

The activation of a neuron in the output layer is determined by a linear combination of the fixed nonlinear basis functions, *i.e.*

$$F^*(x)= \sum_{i=1}^{M} w_i \phi_i(x) \tag{16}$$

where $\phi_i(x)= G(\|x - c_i\|)$ and w_i are the adjustable weights that link the output nodes with the appropriate hidden neurons. These weights in the output layer can then be learnt using the least-squares method.

The present study adopts a systematic approach to the problem of centre selection. Because a fixed centre corresponds to a given regressor in a linear regression model, the selection of RBF centres can be regarded as a problem of subset selection. The orthogonal least-squares (OLS) method can be employed as a forward selection procedure that constructs RBF networks in a rational way. The algorithm chooses appropriate RBF centres, one by one, from training data points until a satisfactory network is obtained. Each selected centre minimises the increment to the explained variance of the desired output, and the ill-conditioning problems that frequently occur in a random selection of centres can thereby automatically be avoided. In contrast to most learning algorithms, which can only work if a fixed network structure has first been specified, the OLS algorithm is a structural identification technique where the centres and estimates of the corresponding weights can be simultaneously determined in a very efficient manner during learning. The OLS learning procedure generally produces an RBF network that is smaller than that produced by a randomly selected RBF network (Chen *et al.*, 1991). Due to its linear computational procedure at the output layer, the RBF has a faster training time, compared to its backpropagation counterpart.

A major drawback of this method is associated with dimensionality of the input space. For large numbers of inputs units, the number of radial basis functions required, can become excessive. If too many centres are used, the large number of parameters available in the regression procedure will cause the network to be over-sensitive to the details of the particular training set, and will result in poor generalisation performance (overfit). To avoid this problem, a regularisation method has been proposed by Orr (Orr, 1993). Both the forward selection and zero-order regularisation techniques were proposed to construct parsimonious RBF networks with improved generalisation properties. Using the regularized

forward selection algorithm (RFS), the error criterion for minimisation is given by the equation

$$J = e^T e + \lambda w^T w \tag{17}$$

instead of the standard $J = e^T e$, where λ is the regularisation parameter. However, the proposed algorithm does not utilise an orthogonalisation scheme; it is therefore computationally more expensive than the standard OLS algorithm.

An efficient combination of the zero-order regularisation and the OLS algorithm has been proposed by Chen and Chng (1996). Similarly, the new error criterion for minimisation in the ROLS algorithm is

$$J = e^T e + \lambda\, g^T g \tag{18}$$

where g is the orthogonal weight vector which satisfies the triangular system

$$g = AW \tag{19}$$

and A is an upper triangular matrix which is computed directly from the OLS regression procedure (Chen *et al.*, 1991).

In the case described in this paper, the ROLS algorithm was employed to model the hourly demand load. The best results were obtained using 10 inputs, selected from the general Eq. (1), by setting the parameters p, q equal to 3. The proposed «*global first nearest-neighbor*» method for width definition was found in practice to be inadequate for the problem. A rather heuristic method, taking half the maximum Euclidean distance, was finally adopted for the simulations. Although an elegant approach to the selection of the regularisation parameter λ is to adopt Bayesian techniques, in this work this parameter was set by trial and error to small positive values, which satisfy the optimal problem solution.
The λ parameter was set equal to 0.0002 and 0.0008 respectively for the two cases. As a result, the corresponding centres were found by the OLS procedure to be equal to 27 and 13 respectively. Just for comparison purposes, it should be noted that for the first case (14h), the original OLS algorithm, without the use of λ parameter, gave a network with a similar accuracy but at the computational cost of 69 centres.

6. INTERNAL STATE RECURRENT NEURAL NETWORK

In the previous sections, the so called *windowed input network* has been applied for modelling the electric load. Another approach has been that of explicitly including the time dependency into the network structure. The commonly used backpropagation algorithm contains no *memory*, hindering the learning of temporal patterns. Here, the alternative is to use a dynamic network that is given some kind of memory to encode past history. In this section, two novel network architectures for the load-forecasting problem, called the modified Elman recurrent network and the autoregressive recurrent network (ARNN) are proposed.

6.1. Modified Elman recurrent network

Recurrent networks with feedback exhibit dynamic behaviour, incorporating a temporal aspect that is not well conveyed by feedforward networks. They also create their own state variables and delays, thus requiring less information about the system being modelled. The

recurrent network developed by Elman has a simple architecture, and it can be trained using the standard backpropagation learning algorithm. The context units of the Elman network memorise some past states of the hidden units, so the output of the network depends on an aggregate of the previous states and the current input. This is the reason that the Elman network possesses the characteristics of a dynamic memory.

In this architecture, in addition to the input, hidden and output units, there are also context units. The input and output units interact with the outside environment, while the hidden and context units do not. The input units are only buffer units, which pass the signals without changing them. The output units are linear units that sum the signals fed to them. The hidden units have nonlinear sigmoidal functions. The context units are used only to memorise the previous activations of the hidden units, and can therefore be considered to function as one-step time delays. The feedforward connections are modifiable, but the recurrent ones are fixed.

At a specific time *t*, the previous activations of the hidden units, at time *(t-1)* and the current inputs, at time *t*, are used as inputs to the network. These inputs are propagated forward to produce the output. The standard BP learning rule is then employed to train the network. After this training step, the activations, at time *t*, of the hidden units are sent back through the recurrent links to the context units, and are saved there for the next training step, *i.e:* time *(t+1)*.

This network has been proved to be effective for modelling linear systems, not higher than the first order (Pham and Liu, 1993). For this reason, an idea based on the work of Hertz (Hertz *et al.*, 1991) was employed to configure a modified Elman network. Here, self-connections are introduced in the context units of the network in order to give these units a certain amount of inertia. The internal structure of this type of network is shown in Fig. 3.

The introduction of self-feedback in the context units increases the capability of the Elman network to model high-order systems. Thus the output of the j^{th} context unit in the modified Elman network (M.ELMAN) is given by

$$x_{cj}(t+1) = \alpha\, x_{cj}(t) + x_j(t) \tag{20}$$

It can be shown that:

$$x_{cj}(t+1) = x_j(t) + \alpha\, x_j(t-1) + \alpha^2 x_j(t-2) + \ldots \tag{21}$$

Usually, α is between 0 and 1. A value of α nearer to 1 enables the network to trace further back into the past.

In this section, this network architecture is used for the load-forecasting problem. For this purpose, the output of the jth context unit in the modified Elman network structure is given by

$$x_{cj}(t+1) = x_j(t) + \alpha\, x_j(t-1) + \alpha^2 x_j(t-2) + \alpha^3 x_j(t-3) + \alpha^4 x_j(t-4) + \alpha^5 x_j(t-5) \tag{22}$$

The five 'memories' in each node at the context layer allow the network to encode state information. Thus, the network can be considered as a model that is capable of carrying out five-step-ahead accurate predictions [Kodogiannis *et. al.*, 1994].

In order to enhance the network's performance, an extra hidden layer has been added, and the linear output function was replaced with a standard sigmoidal one. Therefore a 4/12/14/1 Elman network was applied with self-feedback in the 12 context units, with α equal

to 0.3. The inputs were selected from the model of Eq. (1) by setting the parameters p and q equal to 2. Figs 4-5 illustrate the load prediction results for 14.00h and 2.00h, which correspond to the hours with the maximum and minimum load consumptions respectively.

6.2. Autoregressive recurrent neural network

In the standard multilayer perceptron, the input to a neuron are multiplied by feedforward weights and summed, along with a node bias term. The sum is then passed through a smooth sigmoidal transfer function, producing the neuron's output value. This neural model has no memory, because the output value is not expilicitly dependent upon previous outputs.

In this section, a novel network architecture for the load forecasting problem, called the autoregressive recurrent network (ARNN) is proposed (Kodogiannis, 1994). It was designed to contain internal memory of the previous states, while training rapidly using a generalised BP algorithm. The idea is that we still use a recurrent neural network model but the recurrent neurons are decoupled so that each neuron only feeds back to itself. With this modification the ARNN model is considered to converge easier and to need less training cycles than a fully recurrent network.

The ARNN is a hybrid feedforward / feedback neural network, with the feedback represented by recurrent connections appropriate for approximating the dynamic system. The structure of the ARNN is shown in Fig. 6. There are two hidden layers, with sigmoidal transfer functions, and a single linear output node. The ARNN topology allows recurrence only in the first hidden layer. For this layer, the memoryless backpropagation model has been extended to include an autoregressive memory, a form of self-feedback where the output depends also on the weighted sum of previous outputs.

A modified backpropagation algorithm was developed to train the ARNN that includes dynamic recursive equations in time. The mathematical definition of the ARNN is shown below:

$$y(t) = O(t) = \sum_{l} W_{l}^{O} Q_{l}(t), \quad Q_{l} = f(S_{l}), \quad S_{l} = \sum_{j} W_{jl}^{H} Z_{j}(t) \tag{23}$$

and

$$Z_{j}(t) = f(H_{j}(t)), \quad H_{j}(t) = \sum_{k=1}^{k=n} W_{jk}^{D} Z_{j}(t-k) + \sum_{i} W_{ij}^{I} I_{i} \tag{24}$$

where $I_i(t)$ is the i^{th} input to ARNN, $H_j(t)$ is the sum of inputs to the j^{th} recurrent neuron in the first hidden layer, $Z_j(t)$ is the output of the j^{th} recurrent neuron, $S_l(t)$ is the sum of inputs to the l^{th} neuron in the second hidden layer, $Q_l(t)$ is the output of the l^{th} neuron in the second hidden layer and $O(t)$ is the output of the ARNN. Here, $f(\bullet)$ is the sigmoid function and W^I, W^D, W^H and W^O are input, recurrent, hidden and output weights, respectively. The index n indicates the number of internal memories in the hidden nodes at reccurent layer and for this application was set to five. Thus the network can be considered as a model capable to carry out five-steps-ahead accurate predictions. The five memories in each node at the first hidden layer allow the network to encode state information.

Eight inputs were selected form the model of Eq.(1) by setting the parameters p and q equal to 2. Hence a structure of 8/20/14/1 nodes has been used for the simulation of the

STLF problem. The results for the 14.00h and 2.00h that correspond to hours with maximum and minimum load consumption respectively, are illustrated in Tables 1 and 2.

From the results, that are shown in a next section, it was proved that the approach of producing a dynamic memory is clearly simpler than the other proposed techniques, with the result that the computational burden to be substantially reduced (Kodogiannis *et al.*, 1996). This fact, in conjunction with the decrease in network complexity, proves that the current approach is better suited to this kind of sequence processing.

7. DISCUSSION OF RESULTS

This section presents the results and the statistics of the forecasts obtained from the application of the developed STLF models to the power system of the island of Crete. Case studies for the proposed methods were carried out for 24-hour load forecasting.

Several structures of neural networks, with algorithms ranging from backpropagation learning to OLS methods, were tested. Note that the neural models differ from the usual time-series models. Since they include loads for both previous times and previous days, they represent the hourly variations better than other existing methods do.

An obvious advantage of the proposed modular architecture is that, since the complete system consists of 24 neural blocks, each one with a single output, training is easier and faster compared to traditional neural approaches, which treat the output as a 24x1 vector. The results were analysed, on the basis of the following indices:

Standard error deviation

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}[y_i - \hat{y}_i]^2}$$

Percent relative error

$$\varepsilon = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| * 100 / y_i$$

Root mean square error

$$\varepsilon_1 = \sqrt{\frac{1}{N}\sum_{i=1}^{N}[\{y_i - \hat{y}_i\}/y_i]^2} * 100$$

The complete results for the STLF problem, for the hours with minimum and maximum load consumptions, are illustrated in Tables 1 and 2 respectively.

The standard MLP with ABP learning algorithm, which is widely used for such application, was considered in this work as a testbed case. In an alternative representation, the SE structure has been shown to enable a network to maintain a higher degree of accuracy, compared with the classic MLP structure. Although this considerable improvement in performance is generally obtained at the expense of a larger network, the use of the proposed structure has significant advantages in applications requiring long-range predictions. The performance of a classical MLP will severely deteriorate when it is acting as a recurrent model, because any errors of the predicted output will tend to accumulate. This problem is

avoided in this specific structure, by splitting the error across several nodes, thus exploiting the network's fault tolerance.

An alternative method to MLPs could be to consider a neural model employing radial basis functions. Both aspects, training time and improved accuracy, were satisfied with the use of ROLS. An additional advantage of the specific algorithm was the avoidance of the overfitting problem by using the regularised parameter λ. Due to this factor, the RBF network enjoys a very compact structure, compared to the other proposed neural architectures.

The use of dynamic neural networks presents an innovation to the specific problem. Here, the objective was to use a dynamic network (M.ELMAN, ARNN) which was given some kind of memory to encode past history, with the additional requirements of short training time.

The improved results, compared to the standard MLP structures, reveal the advantages of using memory neuron structures. The inclusion of five memories and the related recurrence in the context layer, enable the network to carry out five-steps-ahead accurate predictions. Although this method is dependent on the number of 'memories' and therefore it can be considered as a partially recurrent network, its use in the proposed modular architecture will allow the expansion of the prediction horizon beyond the limit of 24 hours.

8. CONCLUSIONS

This paper is based on a comparative analysis of neural-network-based STLF techniques. These methods were developed for a one-day-ahead prediction of hourly electric load, using a modular architecture. Several neural architectures were tested, including multilayer perceptrons, fuzzy-neural-type networks, radial basis functions and memory neuron networks. The building block of the existing forecasting systems is an MLP trained with the BP algorithm. However, in such dynamic applications, the real computing power of connectionist models can be exploited only if the networks themselves are dynamic in nature. Two approaches to processing inputs in the time domain have been applied: one is to window the inputs and then treat the time domain like another spatial domain, with the development of the SE network as a result. The results obtained from the SE model reveal the advantage of the proposed neural approach, compared to the widely used multilayer perceptron employed with the standard BP algorithm.

Alternatively, it is possible to introduce dynamics into the network by transferring the regular network neuron state to another set of '*duplication*' neurons called memory neurons. The modified Elman network and the ARNN are examples of such architecture. Their performance is characterised by a high degree of accuracy, as well as fast training times, similar to the performance of the RBF network based on the ROLS algorithm.

It should be noted that the proposed structures differ from the usual time-series prediction models, since they include only loads for previous times and previous days for hourly load forecasting. In future work, the present approach will be enhanced by using advanced neuro-fuzzy models and additional load and weather information, such as illumination levels, temperature, humidity, wind direction and industrial load.

REFERENCES

1. Bakirtzis, A., Petridis, V., and Klartzis, S., A neural network short term load forecasting model for the greek power system. *IEEE Trans. on Power Systems*, 1996, **11**(2), 858-863.
2. Chen, S., Cowan, C.F., and Grant, P., Orthogonal least-squares learning algorithm for radial basis function networks. *IEEE Trans. on Neural Networks*, 1991, **2**, 302-309.
3. Chen, S. and Chng, E., Regularised orthogonal least squares algorithm for constructing radial basis function networks. *Int. J. Control*, 1996, **64**(5), 829-837.
4. Cybenko, G., Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 1989, **2**, 303-314.
5. Gross, G and Galiana, F., Short term load forecasting. *Proc. IEEE*, 1987, **75**(12), 1558-1573.
6. Hertz, J., Krogh, A. and Palmer, R.G., *Introduction to Theory of Neural Computation*, Addison-Wesley, 1991.
7. IEEE Committee Report, Load forecasting bibliography Phase I. *IEEE Trans. on Power App. and Sys.*, 1980, **99**, 53-58.
8. Irisarri, G and Widergren, S., On-line load forecasting for energy control center applications. *IEEE Trans. on Power App. and Sys.*, 1982, **101**(1), 71-78.
9. Kodogiannis, V.S., Lisboa, P.J.G. and Lucas, J., Neural network modelling and control for underwater vehicles. *Artificial Intelligence in Engineering*, 1996, **1**, 203-212.
10. Kodogiannis, V.S., Lisboa, P.J.G. and Lucas, J., Long Range Predictive Controller for Underwater Robotic Vehicles using Recurrent Neural Networks. *2nd IEEE Mediterranean Symposium on New Directions in Control & Automation*, Chania, Crete, Greece, June 1994, pp. 217-224.
11. Kodogiannis, V.S., Neural network techniques for modelling and learning control of an underwater robotic vehicle. PhD thesis, Liverpool University, May 1994.
12. Kodogiannis, V.S., Lisboa, P.J.G. and Lucas, J., A Neural Predictive Controller for Underwater Robotic Applications. *Proc. 2nd IEE/BCS Workshop on Neural Network Applications and Tools*, Liverpool, September 1993, IEEE Computer Society Press, pp. 126-133.
13. Lee, K., Cha, Y. and Ku, C., A study on neural networks for short-term load forecasting. *Proc. of ANNPS '91*, Seattle, July 1991, 26-30.
14. Lu, C.N., Wu, H.T. and Vemuri, S., Neural network based short term load forecasting. *IEEETrans. on Power Systems*, 1993, **8**(1), 337-342.
15. Marr, D. and Hildreth, E., Theory of edge detection. *Proc. Roy. S ,moc. B*, 1980, 207.
16. Mehra, R., On the Identification of variance and Adaptive Kalman filtering. *Proc. of JACC*, 1969, 494-505.
17. Moghram, I. and Rahman, S., Analysis and evaluation of five short-term load forecasting techniques. *IEEE Trans. on Power Systems*, 1989, **4**(4), 1484-1491.

18. Moody, J. and Darken, C., Fast learning in networks of locally tuned processing units. *Neural Computation*, 1989, **1**, 281-294.
19. Orr, M., Regularised centre recruitment in radial basis function networks. Research report No 59, Centre for Cognitive Science, University of Edinburgh, U.K, 1993.
20. Papalexopoulos, A., How, S. and Peng, T.M., An implementation of a neural network based load forecasting model for the EMS. Paper 94 WM PWRS presented at the IEEE/PES 1994 Winter Meeting, 207-209.
21. Park, D.C, El-Sharkawi, M.A., Marks II, R.J., Atlas, L. and Damborg, M., Electric load forecasting using an artificial neural network. *IEEE Trans. on Power Systems*, 1991a, **6**(2), 442-449.
22. Park, J.H., Park, Y.M. and Lee, K., Composite modeling for adaptive short term load forecasting. *IEEE Trans. on Power Systems*, 1991b, **6**(2), 450-456.
23. Peng, T.M., Hubele, N.F. and Karady, G.G., Advanement in the application of neural networks for short term load forecasting. *IEEE Trans. on Power Systems*, 1992, **7**(1), 250-258.
24. Pham, D.T. and Liu, X, Identification of linear and nonlinear dynamics systems using recurrent neural networks. *Artificial Intelligence in Engineering* 1993, **8**, 67-75.
25. Press, W. and Teukolsky, S., *Numerical Recipes in C: The art of scientific computing*. Cambridge University Press, 1992.
26. Rahman, S. and Bhatnager, R., An expert system based algorithm for short-term load forecast. *IEEE Trans. PWRS*, 1988, **3**, 392-399.
27. Ross, T., *Fuzzy Logic with Engineering Applications*, McGraw Hill, 1995.
28. Rumelhart, D. and McClelland, T. L., (eds), *Parallel Distributed processing: Explorations in the Microstructure of Cognition. Vol. 1: Foundations*, MIT Press, 1986.
29. Vemuri, S., Huang, W., and Nelson, D., On-line Algorithm for forecasting hourly loads of an electric utility. *IEEE Trans. on Power App. and Sys.*, 1981, **100**, 3775-3784.
30. Vogl, T.P., Mangis, J.K. and Rigler, A.K., Accelerating the convergence of the backpropagation method. *Biological Cybernetics*, 1988, 257-263.
31. Zadeh, L.A., Fuzzy sets. *Information and Control*, 1965, **8**, 338-353.

	Models	AR	ABP	SE	RBF	M.ELMAN	ARNN
Statistical Properties							
Relative Error %		3.86297	2.7346	1.5174	1.35119	1.21426	1.2233
Standard Deviation Error		5.90894	3.91741	2.1345	1.96127	1.95812	1.83768
RMSE %		5.80199	3.80634	2.0314	1.86722	1.80738	1.71771

Table 1: *Results for the 2:00h load*

	Models	AR	ABP	SE	RBF	M.ELMAN	ARNN
Statistical Properties							
Relative Error %		11.1837	11.9674	3.4001	2.79244	2.56073	2.73292
Standard Deviation Error		25.6068	24.3568	7.716	5.77855	5.8887	5.92847
RMSE %		17.2049	17.6778	4.6194	3.60609	3.48519	3.68308

Table 2: *Results for the 14:00h load*

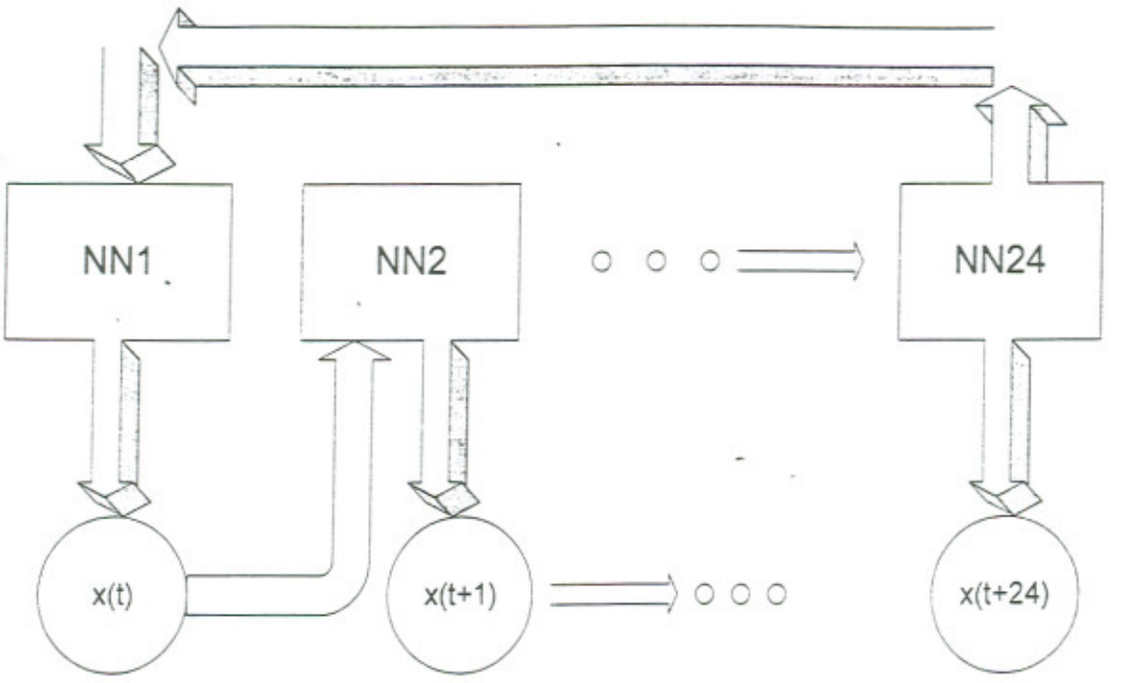

Figure 1: *Proposed modular architecture for the STLF problem*

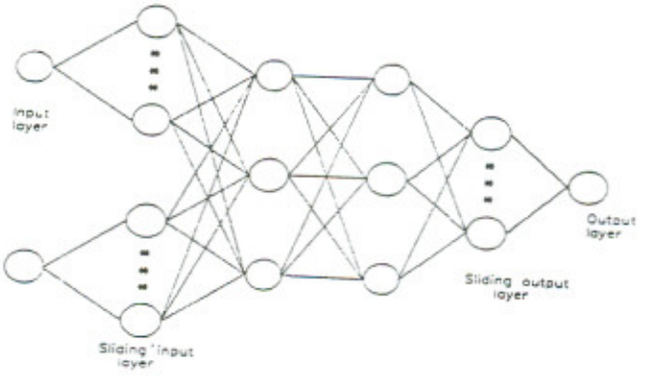

Figure 2: *Spread encoding neural architecture*

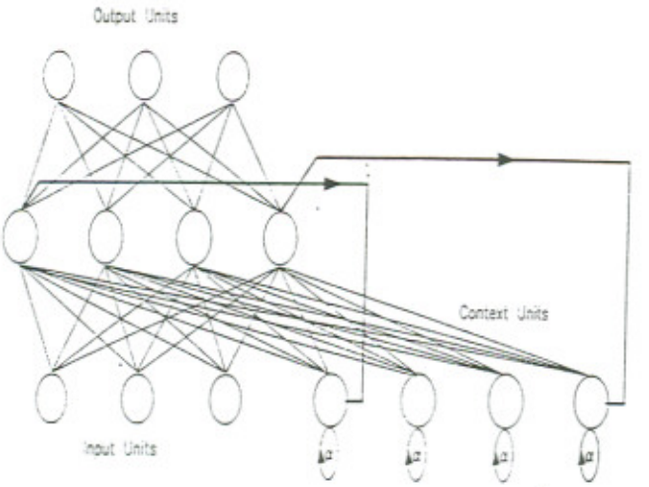

Figure 3: *Modified Elman architecture*

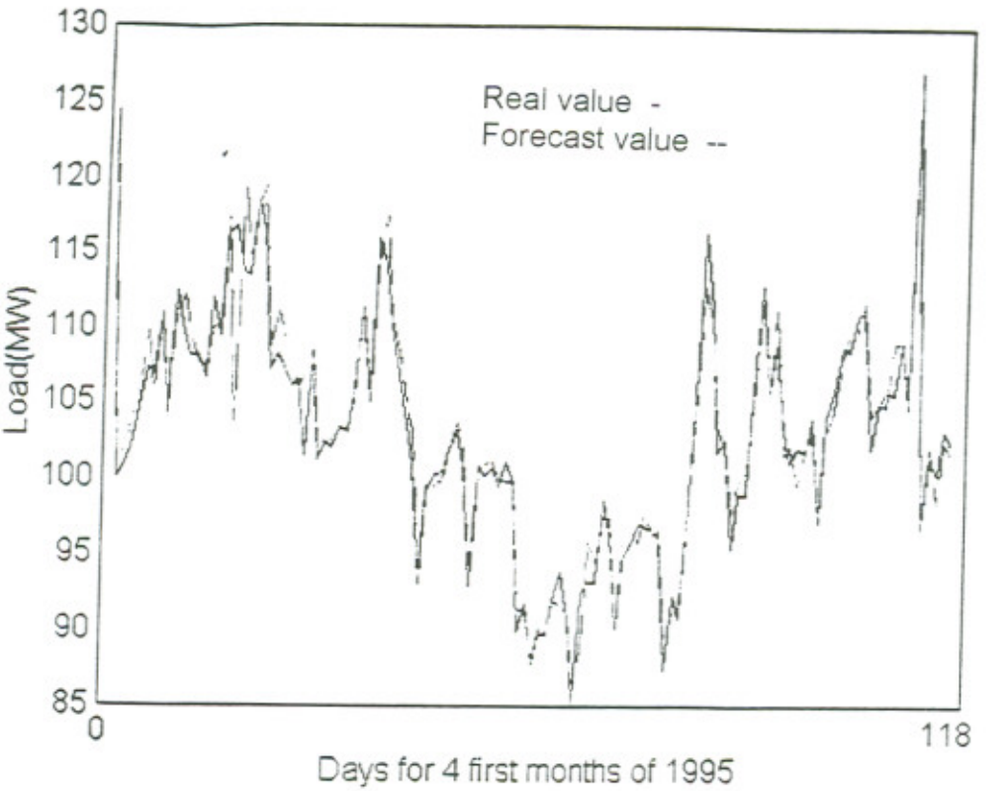

Figure 4: *Load forecasting of 2:00h using the M.Elman model*

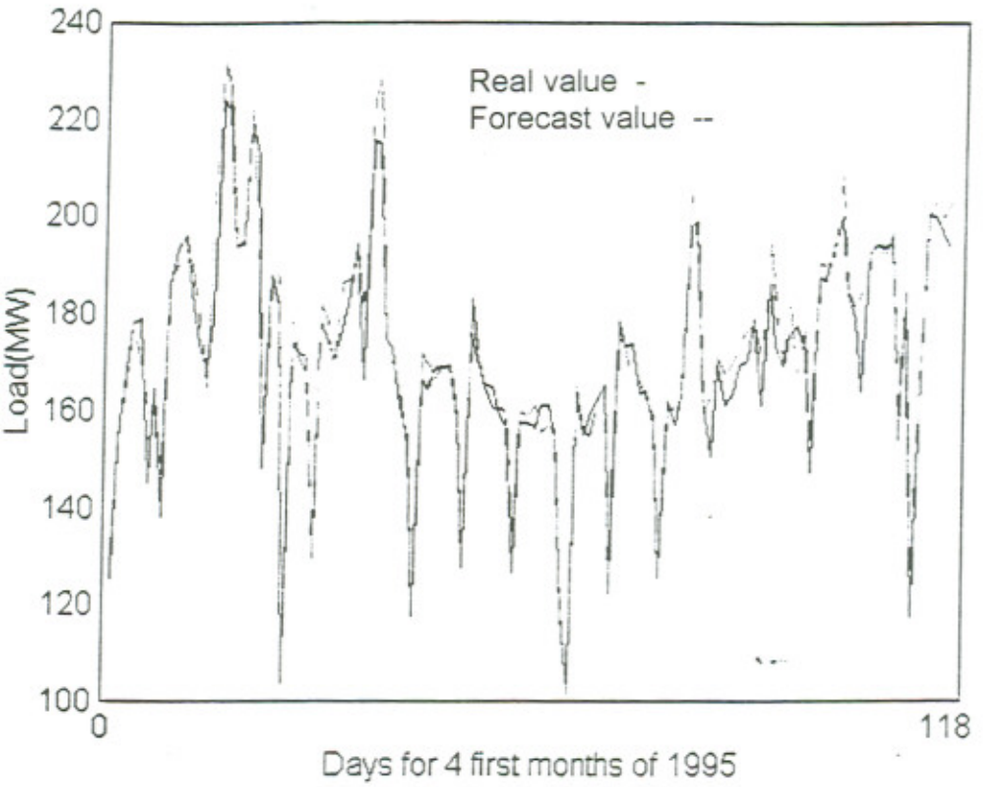

***Figure 5**: Load forecasting of 14:00h using the M.Elman model*

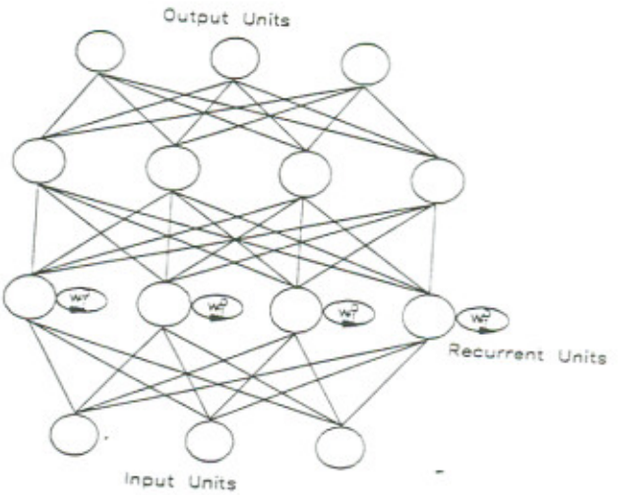

***Figure 6**: ARNN architecture*