Genetic Algorithm-Based Variable Translation Wavelet Neural Network and its Application

S.H. Ling  
Centre of Multimedia Signal Processing,  
Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

F.H.F. Leung  
Centre of Multimedia Signal Processing,  
Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

Abstract - A variable translation wavelet neural network (VTWNN) trained by genetic algorithm is presented in this paper. In the proposed wavelet neural network, the translation parameters are variables depending on the network inputs. Thanks to the variable translation parameter, the network becomes an adaptive one, providing better performance and increased learning ability than conventional wavelet neural networks. Genetic algorithm is applied to train the parameters of the proposed wavelet neural network. An application example on short-term daily electric load forecasting in Hong Kong is presented to show the merits of the proposed network.

I. INTRODUCTION

Recently, a new kind of neural networks known as the wavelet neural networks (WNNs) have been proposed [1-4], which combine feed-forward neural network with the wavelet theory [5-6]. It can provide better performance in function learning than conventional feed forward neural networks. Wavelets provide a multi-resolution approximation of discriminant functions. Researchers have successfully applied wavelet in function approximation [1], load forecasting [2], and pattern classification [3]. A typical wavelet neural network structure allows a fixed set of network parameters of which the values are learned. The network should be trained to model the input-output relationship of a given data set. However, the number of the fixed set of network parameters may not be enough for data sets that are distributed in a vast domain.

One of the important issues on neural networks is the training of the networks. The training process aims to find a set of optimal network parameters. One commonly used training method is the gradient method. However, it may only converge to a local minimum, and is sensitive to the values of the initial parameters. The function to be optimized needs to be differentiable, and the learning method may only be good for some specific network structure. Genetic algorithm (GA) [7,13] is a global search algorithm. The error functions are less likely to be trapped in a local optimum, and need not be differentiable or even continuous. Thus, GA is more suitable for searching in a large, complex, non-differentiable and multimodal domain.

The same GA can be used to train many different networks, regardless of whether they are of feed-forward [14], recurrent [14], wavelet [1] or any other structure type. This generally saves a lot of efforts in developing the training algorithms for different types of networks.

II. WAVELET THEORY

Certain seismic signals can be modelled by combining translations and dilations of an oscillatory function with finite duration called a “wavelet”.

A continuous function \( \psi(x) \) is a “mother wavelet” or “wavelet” if it satisfied the following properties:
Property 1:
\[ \int_{-\infty}^{\infty} \psi(x) \, dx = 0 \] (1)
In other words, the total positive energy of \( \psi(x) \) is equal to the total negative energy of \( \psi(x) \).

Property 2:
\[ \int_{-\infty}^{\infty} |\psi(x)|^2 \, dx < \infty \] (2)
where most of the energy in \( \psi(x) \) is confined to a finite domain and is bounded. In order to control the magnitude and the position of \( \psi(x) \), \( \psi_{a,b}(x) \) is defined as:
\[ \psi_{a,b}(x) = \frac{1}{\sqrt{a}} \psi \left( \frac{x-b}{a} \right) \] (3)
where \( a \) is the dilation parameter and \( b \) is the translation parameter. It should be noted that \( \psi_{a,b}(x) \) is scaled down as the dilation parameter \( a \) increases, and the location of the centre of the wavelet is controlled by the translation parameter \( b \).

III. GENETIC ALGORITHM BASED VARIABLE TRANSLATION WAVELET NEURAL NETWORK

In this section, the GA-based variable translation wavelet neural network (VTWNN) will be presented. The wavelet neural network (WNN) can be considered as a particular case of the feed-forward neural network model. The difference is that the hidden layer of a WNN contains transfer functions of multi-scaled wavelet functions \( \psi_{a,b}(x) \). In the proposed VTWNN, the translation parameter in the transfer function of the hidden node is variable and depends on the network inputs. With the variable translation parameters, the proposed VTWNN performs better and has higher learning ability than the conventional WNNs [1]. The tuning of the network parameters is done by GA.

The proposed VTWNN has a three-layer structure with \( n_{in} \) nodes in the input layer, \( n_h \) nodes in hidden layer, and \( n_{out} \) nodes in output layer as shown in Fig. 1. The input of the hidden layer, \( S_j \), is given by,
\[ S_j = \sum_{i=1}^{n_{in}} z_i v_{ji} , \quad j = 1, 2, ..., n_h \] (4)
where \( z_i , i = 1, 2, ..., n_{in} \) are the input variables; \( v_{ji} \) denotes the weight of the link between the \( i \)-th input and the \( j \)-th hidden node. In order to control the magnitude and the position of the wavelet, the multi-scaled wavelet function \( \psi_{a,b}(x) \) defined in (3) is used.

The dilation parameter \( a \) of the first hidden node (\( j=1 \)) is set at 1, i.e. \( \psi_{1,b}(x) = \psi(x-b) \). For the second hidden node (\( j=2 \)), the dilation parameter \( a \) is set at 2, i.e. \( \psi_{2,b}(x) = \frac{1}{\sqrt{2}} \psi \left( \frac{x-b}{2} \right) \), where the output of wavelet is scaled down by \( 1/\sqrt{2} \). Similarly, for the \( j \)-th hidden node, the dilation parameter \( a \) is set at \( j \). Hence, the output of the hidden layer of the proposed VTWNN is given by,
\[ y_{j,b}(S_j) = \frac{1}{\sqrt{j}} \psi \left( \frac{S_j-b}{j} \right) \] (5)

In this proposed network, the function of Maxican Hat [9] as shown in Fig. 2 is selected as the mother wavelet \( \psi(x) \), which is defined as:
\[ \psi(x) = e^{-x^2/2} (1-x^2) \] (6)
\( \psi(x) \) meets the Property 1 in (1) and Property 2 in (2) of wavelet. Referring to (5) and (6),
\[ y_{j,b_j}(S_j) = \frac{1}{\sqrt{j}} e^{-\frac{S_j-b_j}{j}} (1 - \frac{S_j-b_j}{j})^2 \] (7)

In this proposed network, the translation parameter \( b_j \) is variable depending on the inputs \( S_j \), and is governed by a nonlinear function \( f^j() \),
\[ b_j = f^j(S_j) \] (8)
\[ f^j(S_j) = 4 \left( 2 \frac{1}{1+e^{-K_j S_j}} - 1 \right) \] (9)
where \( K_j \) is a tuned parameter which is used to control the shape of the nonlinear function \( f^j() \). In Fig. 3, the effect of the tuned parameter \( K_j \) to \( b_j \) is shown. We see as \( K_j \to \infty \), the function reduces to a threshold function.

The output of the proposed VTWNN is defined as,
\[ y_i = \sum_{j=1}^{n_h} \psi_{j,b_j}(S_j) w_{ij} \] (10)
\[ = \sum_{j=1}^{n_h} \psi_{j,b_j}(\sum_{i=1}^{n_{in}} z_i v_{ji}) w_{ij} \] (11)
where \(w_{ij}, j = 1, 2, \ldots, n_h; l = 1, 2, \ldots, n_{out}\) denotes the weight of the link between \(j\)-th hidden node and \(l\)-th output node. All the parameters of the network are tuned by the GA [7].

![Maxican Hat mother wavelet](image)

**Fig. 2.** Maxican Hat mother wavelet

![Sample nonlinear functions with different values of parameter \(K_j\)](image)

**Fig. 3.** Sample nonlinear functions with different values of parameter \(K_j\)

\(K_j = 0.3, K_j = 0.5, K_j = 0.8, K_j = 1.0\) and \(K_j = 1.5\)

### IV. Training of Network Parameters

The proposed neural network is employed to learn the input-output relationship of an application using GA with unimodal normal distribution crossover (UNDX) [8] and non-uniform mutation [7]. More details about these two genetic operations are discussed in Appendix. A population of chromosomes \(P\) is initialized and then evolves. First, two parents are selected from \(P\) by the method of spinning the roulette wheel [7]. Then a new offspring is generated from these parents using the crossover and mutation operations, which are governed by the probabilities of crossover and mutation respectively. These probabilities are chosen by trial and error through experiments for good performance. The new population thus generated replaces the current population. The above procedures are repeated until a certain termination condition is satisfied, e.g. a predefined number of generations has been reached. The input-output relationship can be described by,

\[
y^d(t) = g(z^d(t)), \quad t = 1, 2, \ldots, n_d,\]

where \(z^d(t) = [z_1^d(t) \ z_2^d(t) \ \ldots \ z_m^d(t)]\) and \(y^d(t) = [y_1^d(t) \ y_2^d(t) \ \ldots \ y_n^d(t)]\) are the given inputs and the desired outputs of an unknown nonlinear function \(g()\) respectively; \(n_d\) denotes the number of input-output data pairs. The fitness function is defined as,

\[
fitness = \frac{1}{1 + err},
\]

\[
err = \frac{\sum_{t = 1}^{n_d} \sum_{i = 1}^{n_i} |y_i^d(t) - y_i^e(t)|}{n_d n_{out}}.
\]

The objective is to maximize the fitness value of (13) (minimize \(err\)) using the GA by setting the chromosome to be \([v_{ji} w_{j} \ k_j]\) for all \(i, j, l, i, v_{ji}, w_{j} \in [-1.5, 1.5], k_j \in [0.3, 2]\). The fitness value of (13) \(\in [0, 1]\).

### V. Application Example

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

The proposed VSNMN is applied to do STLF. The application of neural networks to STLF has been explored extensively in the literature [10-12]. The idea is to construct seven multi-input multi-output neural networks, one for each day of a week. Each neural network has 24 outputs representing the expected hourly load for a day. A diagram of one of the seven neural networks for the load forecasting is shown in Fig. 4. The network has 28 inputs and 24 outputs. Among the 28 inputs nodes, the first 24 nodes represent the previous 24 hourly loads [12] and are denoted by \(z_i = L_i(t-1), i = 1, 2, \ldots, 24\). Node 25 \((z_{25})\) and node 26 \((z_{26})\) represent the average temperatures of the previous day \((T(t-1))\) and the forecasted average temperatures of the present day \((T(t))\) respectively. Node 27 \((z_{27})\) and node 28 \((z_{28})\) represent the average relative humidity at the previous day \((RH(t-1))\) and the forecasted average relative humidity at the present day \((RH(t))\) respectively. The output layer consists of 24 output nodes that represent the forecasted 24 hourly loads of a day, and are denoted by \(y_i(t) = L_i(t), i = 1, 2, \ldots, 24\). Such a network structure is chosen based on the assumption that the consumption patterns of the seven days within a week...
would differ significantly among each other, while the patterns among the same day of weeks are similar. By using the past 24 hourly loads as the inputs, the relationship between a given hour’s load and the 24 hourly loads of the pervious day can be considered. Temperature information (Node 25 and Node 26) is important inputs to the STLF. For any given day, the deviation of the temperature variable from a normal value may cause such significant load changes as to require major modifications in the unit commitment pattern. Humidity is similar to temperature on affecting the system load, especially in hot and humid areas.

In this paper, we use a data set in year 2000 provided by CLP Power Hong Kong Ltd to illustrate the proposed VTWNN on doing STLF. The proposed VTWNN was trained using half year (22 weeks) load data for every Thursday or Sunday from Mar. 23 to Aug. 20, 2000. The load pattern for every Thursday and Sunday from Mar. 23 to Aug. 20, 2000 are shown in Fig. 5 and Fig. 6 respectively. In these two figures, we can see that the shape of every load pattern is similar but the power consumption is much different. Conventional feed-forward neural networks are only good at minimizing the average error of the system. It is difficult to model all the load patterns accurately.

In the VTWNN, each input set will be individually handled by its corresponding network parameter set. It should be able to handle the load forecasting problem better. Referring to (13), the proposed VTWNN used for doing STLF is governed by,

\[ y_i = \sum_{j=1}^{n} \psi_{j,h} \left( \sum_{l=1}^{28} z_{j,l} w_{j,l} \right) \quad i = 1, 2, \ldots, 24. \]  

GA is used to tune the parameters of the proposed VTWNN of (17). The fitness function is given by,

\[ \text{fitness} = \frac{1}{1 + \text{err}} \]

\[ \text{where } \text{err} = \frac{\sum_{i=1}^{22} \sum_{j=1}^{24} \left| y^d_{i,j}(t) - y_{i,j}(t) \right|}{22 \times 24} \]

The value of \( \text{err} \) indicates the mean absolute percentage error (MAPE) of the load forecasting system. For comparison, a wavelet neural network (WNN) [1] is also applied to do the same job. All parameters of the two networks are trained by real coded GA with unimodal normal distribution crossover (UNDX) [8] and non-uniform mutation [7]. For all cases, the initial values of the parameters of the neural networks are randomly generated. The number of iteration to train the neural networks is 10000. For the GA, the probability of crossover (\( p_c \)), the probability of mutation (\( p_m \)), and the shape parameter of the mutation operation are set at 0.8, 0.01, and 2 respectively for both the proposed VTWNN and WNN [1]. The population size is 100. All the results are averaged one out of 10 runs. The training and forecasting results are tabulated in Table I and Table II, which show the simulation results for different numbers of hidden node (\( n_h \)). The forecasting results for Aug. 24 (Thursday) and Aug. 27 (Sunday) are shown in these tables. From these tables, it can be seen that the proposed VTWNN performs better than the WNN [1] in terms of the average training and forecasting error. The best average training and forecasting error are achieved when the number of hidden node (\( n_h \)) is set at 20 for both approaches for Thursday and Sunday. With the proposed VTWNN, the average training error are 3.1266% for Thursday and 3.0213% for Sunday, which implies 12.1% and 16.7% improvements over the conventional WNN with the same number of hidden nodes. The best forecasting errors for Aug. 24 (Thursday) and Aug. 27 (Sunday) are 2.0985% and 1.8662%, which implies 16.2% and 16% improvements. Fig. 7 and Fig. 8 show the results of the load forecasting on Aug. 24 (Thursday) and Aug. 27 (Sunday). In this figure,
the dashed line represents the best forecasted result using the proposed network, and the dotted line is the best forecasted result using the conventional network. The actual load is represented by the solid line. We can see that the forecasting results using the proposed neural network are better.

VI. CONCLUSION

A GA based variable translation wavelet neural network has been presented in this paper. All network parameters are tuned by GA. Thanks to the variable translation parameters in the network, the proposed VTWNN can have a higher learning ability. The application on short-term load forecasting in Hong Kong using the proposed neural networks has been discussed. Experimental results have been given to show the merits of the proposed network.

ACKNOWLEDGMENT

The work described in this paper was substantially supported by a grant from the Hong Kong Polytechnic University (Project No. G-YX31). The data in the application example are offered by CLP Power Hong Kong Limited for illustration purpose only.

| TABLE I. RESULTS OF THE PROPOSED VTWNN AND THE CONVENTIONAL WNN FOR THURSDAY. |
|---------------------------------|----------------|----------------|----------------|----------------|
| $n_h$                          | VTWNN          | WNN [1]        |
| Average training error (MAPE)  | Average forecasting error (MAPE) | Average training error (MAPE) | Average forecasting error (MAPE) |
| 16                             | 3.2898%        | 2.4490%        | 3.6679%        | 2.8088%        |
| 18                             | 3.2237%        | 2.3511%        | 3.5823%        | 2.7154%        |
| 20                             | 3.1266%        | 2.2988%        | 3.5034%        | 2.6033%        |

| TABLE II. RESULTS OF THE PROPOSED VTWNN AND THE CONVENTIONAL WNN FOR SUNDAY. |
|---------------------------------|----------------|----------------|----------------|----------------|
| $n_h$                          | VTWNN          | WNN [1]        |
| Average training error (MAPE)  | Average forecasting error (MAPE) | Average training error (MAPE) | Average forecasting error (MAPE) |
| 16                             | 3.0502%        | 2.1199%        | 3.3912%        | 2.4891%        |
| 18                             | 3.0478%        | 2.1079%        | 3.3798%        | 2.4801%        |
| 20                             | 3.0211%        | 2.0985%        | 3.3238%        | 2.4395%        |

<table>
<thead>
<tr>
<th>REFERENCES</th>
</tr>
</thead>
</table>
A. Unimodal normal distribution crossover (UNDX)

Unimodal normal distribution crossover is defined as a mixture of three selected parents $p_1$, $p_2$, and $p_3$. The resulting offspring $o_s$ is defined as,

\[
o_s = \begin{bmatrix} o_{s1} & o_{s2} & \ldots & o_{s_{\text{no_vars}}} \end{bmatrix} = m + z_i e_1 + \sum_{i=2}^{\text{no_vars}} z_i e_i, \quad (A.1)
\]

\[
o_s' = \begin{bmatrix} o'_{s1} & o'_{s2} & \ldots & o'_{s_{\text{no_vars}}} \end{bmatrix} = m - z_i e_1 - \sum_{i=2}^{\text{no_vars}} z_i e_i, \quad (A.2)
\]

where
\[
m = \frac{(p_1 + p_2)}{2}, \quad (A.3)
\]

\[
z_i = N(0, \sigma_i^2), \quad \sigma_i = \frac{\mu_i^2}{\text{no_vars}}, \quad (A.4)
\]

\[
e_i = \left( \frac{p_2 - p_1}{|p_2 - p_1|} \right), \quad (A.6)
\]

\[
e_m \perp e_n \quad (m \neq n), \quad m, n = 1, \ldots, \text{no_vars}, \quad (A.7)
\]

where \(\text{no_vars}\) is the number of genes, \(N()\) is a normal distributed random number, \(d_i\) is the distance between the parents \(p_1\) and \(p_2\), \(d_2\) is the distance of \(p_3\) from the line connecting \(p_1\) and \(p_2\), and \(\beta\) and \(\mu\) are constants.

B. Non-uniform mutation (NUM)

Non-uniform mutation is an operation with a fine-tuning capability. Its action depends on the generation number of the population. The operation takes place as follows. If $o_s = \begin{bmatrix} o_{s1} & o_{s2} & \ldots & o_{s_{\text{no_vars}}} \end{bmatrix}$ is a chromosome and the element $o_{s_k}$ is randomly selected for mutation (the value of $o_{s_k}$ is inside $[\text{parak}_{\text{min}}, \text{parak}_{\text{max}}]$), the resulting chromosome is then given by

\[
\hat{o}_s = \begin{bmatrix} \hat{o}_{s1} & \ldots & \hat{o}_{s_k} & \ldots & \hat{o}_{s_{\text{no_vars}}} \end{bmatrix}, \quad k = 1, 2, \ldots, \text{no_vars},
\]

and

\[
\hat{o}_{sk} = \begin{cases} o_{sk} + \Delta(r, \text{parak}_{\text{max}} - o_{sk}) & \text{if } r_d = 0 \\ o_{sk} - \Delta(r, o_{sk} - \text{parak}_{\text{min}}) & \text{if } r_d = 1 \end{cases}, \quad (A.8)
\]

where \(r_d\) is a random number equal to 0 or 1 only. The function $\Delta(r, y)$ returns a value in the range $[0, y]$ such that $\Delta(r, y)$ approaches 0 as $r$ increases. The function is defined as follows,

\[
\Delta(r, y) = y \left( 1 - r \left( \frac{1 - r}{1} \right)^b \right), \quad (A.9)
\]

where $r$ is a random number in $[0, 1]$, $\tau$ is the present generation number of the population, $T$ is the maximum generation number of the population, and $b$ is a system parameter that determines the degree of non-uniformity.