

Wavelet Neural Networks with a Hybrid Learning Approach*

CHENG-JIAN LIN

*Department of Computer Science and Information Engineering
Chaoyang University of Technology
Taichung County, 413 Taiwan
E-mail: cjlin@mail.cyut.edu.tw*

In this paper, we propose a Wavelet Neural Network with Hybrid Learning Approach (WNN-HLA). A novel hybrid learning approach, which combines the on-line partition method (OLPM) and the gradient descent method, is proposed to identify a parsimonious internal structure and adjust the parameters of WNN-HLA model. First, the proposed OLPM is an online method and is a distance-based connectionist clustering method. Unlike the traditional cluster techniques that only consider the total variation to update the one mean and deviation. Second, a back-propagation learning method is used to adjust the parameters for the desired outputs. Several simulation examples have been given to illustrate the performance and effectiveness of the proposed model. The computer simulations demonstrate that the proposed WNN-HLA model performs better than some existing models.

Keywords: wavelet neural networks, on-line partition method, identification, prediction, back-propagation learning algorithm

1. INTRODUCTION

It is well known that multilayer perceptrons (MLPs) have been a popular research subject recently. It has been successfully used for modeling complex nonlinear systems, identification and forecasting signal with relatively simple architectures. The MLP structure is less sensitive to learning gain variation and capable of converging to a lower value of mean square error. Because it uses sigmoid functions as node functions and train its weights by the back-propagation (BP) algorithm [1]. However, its multilayered structure and the greedy nature of the BP algorithm, the training processes often settle in undesirable local minima of the error surface. Thus, MLPs are still problematic in terms of their convergence performance and complex structure. A suitable approach to overcome the disadvantages of global approximation networks is the substitution of the global activation function with localized basis functions. From a function representation perspective, a radial basis function (RBF) network is a scheme that represents a function using locally supported functions [2]. One of the typical functions used by RBF is the Gaussian function. The locality of the functions makes RBF more suitable for learning functions with local variations and discontinuities. This means that RBF can represent any function that is in space spanned by the basis functions [3, 4].

Received July 19, 2004; revised January 7 & March 11, 2005; accepted March 23, 2005.

Communicated by Chin-Teng Lin.

* This research was supported in part by the National Science Council of Taiwan, R.O.C., under grant No. NSC 95-2221-E-324-028-MY2.

Following the concept of locally supported basis functions such as RBF, a class of wavelet neural network (WNN) which originates from wavelet decomposition in signal processing has become more popular lately [5, 6]. Wavelets are a class of basic elements with oscillations of effectively finite duration that makes them look like “little waves”. The self-similar, multiple resolution nature of wavelets offers a natural framework for the analysis of physical communication signals [7], prediction [8], identification [9], time series [10], and control [11]. Daubechies [7] has shown demonstrated existence of orthonormal wavelet bases, and hence follows the availability of rates of convergence for the approximation using wavelet-based networks. With wavelet’s specific features, the wavelet basis function network is considered more efficient than the conventional neural networks in approximating nonlinearities appeared in servo systems. Zhang and Benveniste [6] proposed a new notation of wavelet network as an alternative to feedforward neural networks for approximating arbitrary nonlinear functions based on wavelet transform theory. Zhang *et al.* [8] described a wavelet-based neural network for function learning and estimation, and the structure of this network is similar to a RBF network, except that the radial functions are replaced by orthonormal scaling functions. Recently, many researchers [12-15] have been done on applications of WNN, which combine the capability of MLP in learning from process and the capability of wavelet decomposition. Ikonopoulou and Endou [12] proposed the analytical ability of the discrete wavelet decomposition combined with the computational power of radial basis function networks. Members of a wavelet family were chosen through a statistical selection criterion that constructs the structure of the network. Ho *et al.* [13] used the orthogonal least squares (OLS) algorithm to purify the wavelets from their candidates, which avoided using more wavelets than required, and often resulted in an overfitting of the data and a poor situation in [6]. Lin *et al.* [14] proposed a wavelet neural network to control the moving table of a linear ultrasonic motor (LUSM) drive system. They chose an initialization for the mother wavelet based on the input domains defined by the examples of the training sequence. Huang and Huang [15] proposed an evolutionary algorithm for optimally adjusted wavelet networks. But there still has a problem with above-mentioned methods; that is, the selections of wavelet bases are based on practical experimentation or trial-and-error tests.

In this paper, a wavelet neural network with hybrid learning approach (WNN-HLA) is proposed for solving various application problems. The proposed WNN-HLA is a four-layered network structure, which is comprised of an input layer, wavelet layer, product layer, and output layer. We adopt the wavelet function as its node function in the wavelet layer. The proposed WNN-HLA is different from the RBF model [2]. The RBF model is a three-layered network structure, which is comprised of an input layer, radial function layer, and output layer. From the point of view of function representation, the RBF model can represent any function that is in the space spanned by the family of basis functions. However, the basis functions in the family are generally not orthogonal and are redundant. It means that the RBF model representation for a given function is not unique and is probably not the most efficient [8]. The on-line partition method (OLPM) is to determine the structure of WNN-HLA model. In parameter learning scheme, the back-propagation (BP) method is to adjust the shape of wavelet bases and the connection weights in WNN-HLA model. The proposed WNN-HLA model has the following advantages: 1) This study adopts the wavelet neural network to perform prediction and

approximation problems. The local properties of wavelets in the WNN-HLA model enable arbitrary functions to be approximated more effectively. 2) We use an on-line partitioning method, called OLPM, to implement a scatter partitioning of the input space for the purpose of creating wavelet functions. The proposed OLPM is different from the *partition-based* clustering techniques [24-26] to perform cluster analysis. However, such clustering techniques [24-26] require *prior* knowledge of things such as the number of clusters present in a data set. 3) As demonstrated in section 4, the WNN-HLA model is characterized by small network size and fast learning speed.

2. THE STRUCTURE OF THE WNN-HLA

The structure of the WNN-HLA model is shown in Fig. 1. It is designed as a four-layer structure, which comprised of an input layer, a wavelet layer, a product layer, and an output layer. The input data in the input layer of the network is $x = [x_1 \ x_2 \ \dots \ x_n]$, where n is the number of dimensions. The input data are directly transmitted into the wavelet nodes in the wavelet layer. For the discrete wavelet transform, the mother wavelet $\phi(x)$ describes the dilation a and the translation b as follow:

$$\phi_{a,b}(x) = |a|^{-\frac{1}{2}} \phi\left(\frac{x-b}{a}\right). \tag{1}$$

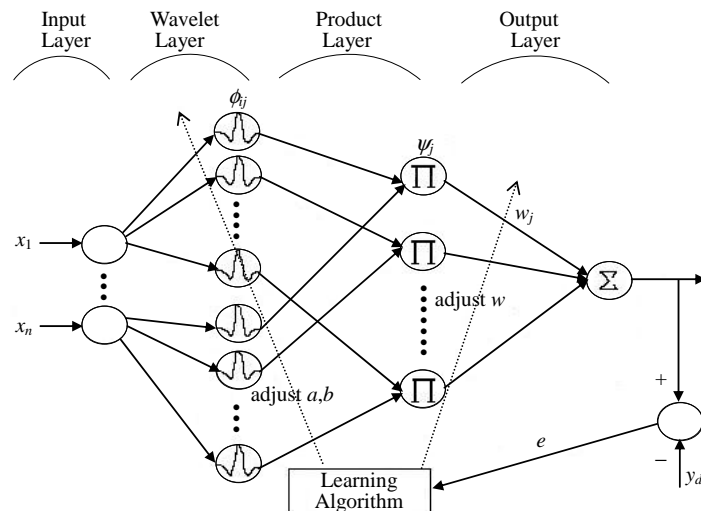


Fig. 1. The architecture of the WNN-HLA model.

In this paper, we use the Mexican-hat function (see Fig. 2) as the wavelet function. Because the wavelet functions only using one parameter, we can control it very easily. The Mexican-hat function is the second derivate of the Gaussian function and expressed as follow:

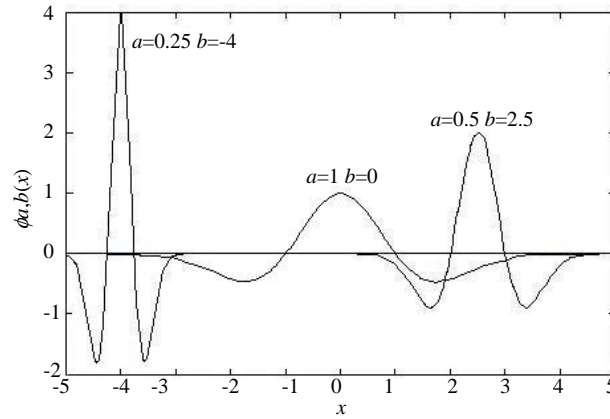


Fig. 2. The Mexican-hat function with different translation and dilation values.

$$\phi(x) = (1 - \|x\|^2) e^{-\frac{\|x\|^2}{2}}. \quad (2)$$

Therefore, the activation function of the j th wavelet node connected with the i th input data is represented as:

$$\phi_{a_j, b_j}(x_i) = |a_j|^{-\frac{1}{2}} \left(1 - \left\| \frac{x_i - b_j}{a_j} \right\|^2 \right) e^{-\frac{\left\| \frac{x_i - b_j}{a_j} \right\|^2}{2}}, \quad i = 1, \dots, n, j = 1, \dots, m. \quad (3)$$

Using this equation, we can easily adjust the parameters a and b appropriately. In Eq. (3), n is the number of input-dimensions and m is the number of the wavelets. Each wavelet in the product layer is labeled Π , i.e., the product of the j th multi-dimensional wavelet with n input dimensions of x_i . The relation is defined as

$$\psi_j(x) = \prod_{i=1}^n \phi_{a_j, b_j}(x_i). \quad (4)$$

According to the theory of multi-resolution analysis (MRA), any $f \in L^2(\mathfrak{R})$ can be regarded as a linear combination of wavelets at different resolution levels. For this reason, the function f is expressed as

$$y(x) = f(x) \approx \sum_{j=1}^m w_j \psi_j(x) \quad (5)$$

where $\psi_j, j = 1, 2, \dots, m$ is used as a nonlinear transformation function of hidden nodes, and $w_j, j = 1, 2, \dots, m$ is used as the adjustable weighting parameters to provide the function approximation. Notice that in addition to w_j , the dilation and translation factors a and b may also be used as adjustable parameters. Applying Eq. (5), we can express the WNN-HLA modeling function y .

3. THE LEARNING ALGORITHM FOR WNN-HLA

In this section, a hybrid learning approach is proposed. First, the translation and dilation parameters of wavelet function are determined by on-line partition method (OLPM). Then, the back-propagation (BP) learning algorithm is used for adjusting the translation, the dilation, and the connection weight parameters. The details of the whole learning algorithm are presented below.

3.1 The On-Line Partition Method

When a parsimonious WNN-HLA structure is being established, number of wavelet functions and approximate estimates of the corresponding parameters, such as translations and dilations, describing the wavelet functions need to be extracted from given input-output training pairs. Thus, the choice of clustering technique in networks is an important consideration. This is due to the use of *partition-based* clustering techniques, such as fuzzy *C*-means (FCM) [24], linear vector quantization (LVQ) [25], fuzzy Kohonen partitioning (FKP), and pseudo FKP [26], to perform cluster analysis. However, such clustering techniques require *prior* knowledge of things such as the number of clusters present in a data set. To solve the above problem, online-based cluster techniques were proposed [21, 27, 28]. But there still has a problem with these methods; that is, the clustering methods only consider the total variations of the mean and deviation in all dimensions per input. This is because the cluster numbers increase quickly.

In this paper we use an on-line partition method, called OLPM, to implement a scatter partitioning of the input space for the purpose of creating wavelet functions. The proposed OLPM is a distance-based connectionist clustering method and is unlike traditional cluster techniques [24-28] that only consider the total variation to update the one mean and deviation. The tendency for traditional cluster techniques to consider the total variation will cause to extend too fast. And they also cannot consider every dimensional variation in the input data. According to the above-mentioned problems, the proposed OLPM method not only considers the original mean and deviation but also considers every mean and deviation's dimensional variation in the input data.

The main advantage of the proposed method is that the OLPM is a one-pass algorithm that dynamically estimates the number of clusters in a set of data and finds their current means in the input data space. For the above reason, the OLPM can cluster the input space quickly. The details of the OLPM algorithm are described in the following steps:

Step 0: Create the first cluster C_1 by simply taking the position of the first input data as the first cluster mean $C_{C_{i-1}}$ where i means the i th input variables. And setting a dimension distance value to 0 (see Fig. 3 (a)).

Step 1: If all of the training input data have been processed, the algorithm is finished. Otherwise, the current input example, $x_i[k]$, is taken and the distances between this input example and all already created cluster mean $C_{C_{i-j}}$ are calculated:

$$D_{i,j}[k] = \|x_i[k] - C_{C_{i-j}}\| \quad (6)$$

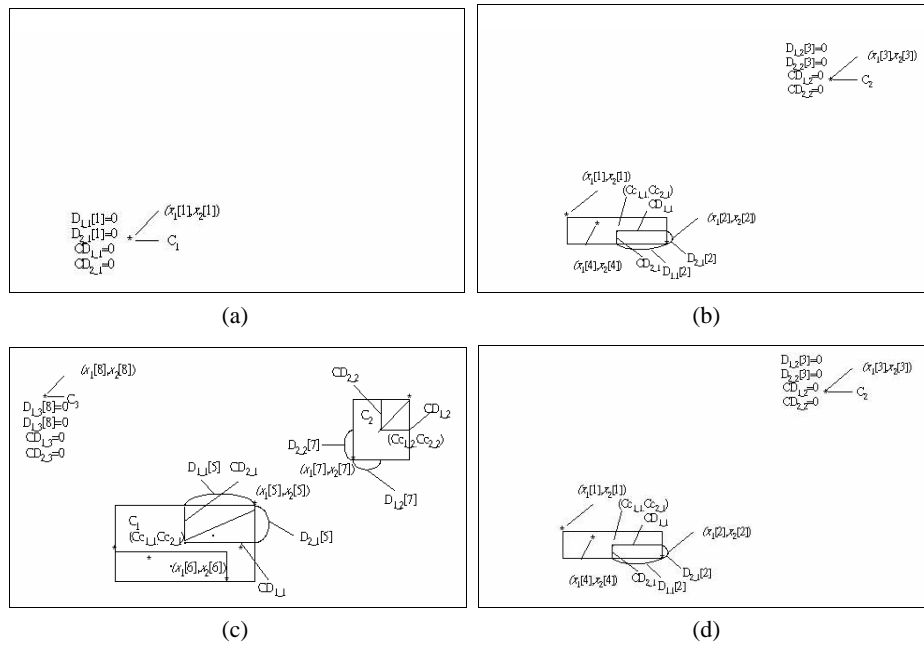


Fig. 3. A brief clustering process using OLPM with samples in 2-D space.

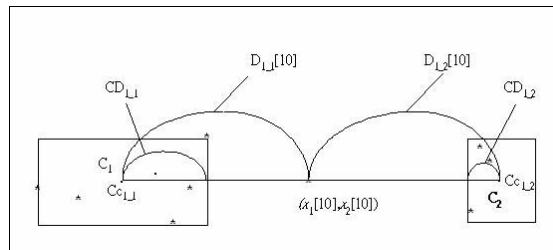


Fig. 4. The special case of OLPM.

where $j = 1, 2, \dots, R$ denotes the j th cluster, $k = 1, 2, 3, \dots, N$ represents the k th input, and $i = 1, 2, \dots, n$ represents the i th dimension. D_{i_j} represents the dimension distance in i th cluster and j th dimension.

Step 2: If the distance calculation in Eq. (6) is equal to or less than at least one of the dimension distances, CD_{i_j} , then it means that the current input example belongs to a cluster with the minimum distance:

$$D \min_j[k] = \min \left(\sum_{i=1}^n \| x_i[k] - Cc_{i_j} \| \right), \tag{7}$$

$$D \min_{i_j}[k] = \| x_i[k] - Cc_{i_j} \|. \tag{8}$$

The constraint is described as follows:

$$D \min_{i,j}[k] \leq CD_{i,j}. \tag{9}$$

If no new clusters are created or no existing clusters are updated (the cases of $(x_1[4], x_2[4])$ and $(x_1[6], x_2[6])$ in Fig. 3 (b)), the algorithm returns to step 1. Otherwise, the algorithm goes to the next step.

Step 3: Find a cluster from all existing cluster centers by calculating $S_{i,j}[k] = D_{i,j}[k] + CD_{i,j}, j = 1, 2, \dots, R$, and then choosing the cluster center with the minimum value:

$$S \min_{i,j}[k] = D \min_{i,j}[k] + CD_{i,j} = \min \left(\sum_{i=1}^n S_{i,j}[k] \right), \text{ where } j = 1, 2, \dots, R. \tag{10}$$

In Eqs. (8) and (9), the minimum distance from any cluster mean to the examples that belong to this cluster is not greater than the threshold D_{thr} , though the algorithm does not keep any information of passed examples. However, we find that the formulation only considers the distance between the input data and the cluster mean in Eq. (10). But the special situation [29] shows that the distances between the given point $x_i[10]$ and both cluster means Cc_{i-1} and Cc_{i-2} are the same as in Fig. 2. In the aforementioned technique, the cluster C_2 , which has small dimension distances CD_{i-2} , will be selected to expand according to Eq. (10). However this causes a problem in that the cluster numbers increase quickly. To avoid this problem, we state a condition, as follows:

If there are two $D \min_j[10]$ computed in Eq. (9) that $D \min_1[10] = D \min_2[10]$ and $(CD_{1-1} + CD_{2-1} > CD_{1-2} + CD_{2-2})$

$$\text{Then } D \min_{1-1}[10] = D_{1-1}[10] \tag{11}$$

$$D \min_{2-1}[10] = D_{2-1}[10] \tag{12}$$

In Eqs. (11) and (12), we find that when the distances between the input data and both clusters are the same, the formulation will choose the cluster that has the large dimension distance CD_{1-1} and CD_{2-1} .

Step 4: If $S \min_{i,j}[k]$ in Eq. (10) is greater than D_{thr} , the input example $x_i[k]$ does not belong to any existing cluster. A new cluster is created in the same way as described in step 0 (the cases of $(x_1[3], x_2[3])$ and $(x_1[8], x_2[8])$ in Fig. 3 (c)), and the algorithm returns to step 1.

Step 5: If $S \min_{i,j}[k]$ is not greater than D_{thr} , the cluster is updated by moving its mean, $Cc_{i,j}$, and increasing the value of its dimension distances. The new mean is moved to the point on the line connecting the input data, and the distance from the new mean to the point is equal to its dimension distance (the cases of $(x_1[5], x_2[5])$ and $(x_1[9], x_2[9])$ in Fig. 3 (d)). The details for updating the equations are as follows:

$$\begin{aligned} &\text{if } CD_{i,j} < (x_i[k] - Cc_{i,j} + CD_{i,j})/2 \\ &\text{then } CD_{i,j} = (x_i[k] - Cc_{i,j} + CD_{i,j})/2 \\ &\quad Cc_{i,j} = x_i[k] - CD_{i,j} \quad \text{if } Cc_{i,j} \geq x_i[k] \\ &\quad Cc_{i,j} = x_i[k] + CD_{i,j} \quad \text{if } Cc_{i,j} < x_i[k] \end{aligned} \tag{13}$$

where $k = 1, 2, 3, \dots, N$ represents the k th input, j represents the j th cluster that has a minimum distance in Eq. (13), x represents the input data, and i represents the i th dimension. After this step is performed, the algorithm returns to step 1.

The threshold parameter D_{thr} is an important parameter in the input space partition scheme. A low threshold value leads to the learning of fine clusters (such that many wavelet functions are generated), whereas a high threshold value leads to the learning of coarse clusters (such that fewer wavelet functions are generated). Therefore, the selection of the threshold value D_{thr} critically affects the simulation results, and the threshold value is determined by practical experimentation or trial-and-error tests.

3.2 The Parameter Learning Algorithm

The well-known back-propagation learning algorithm is used for the supervised learning to find the output errors of the node in each layer and analyze the error to perform parameter adjustment. The goal is to minimize the error function

$$e = y(x) - y^d(x) \quad (14)$$

and

$$E = \frac{1}{2} e^2 \quad (15)$$

where $y^d(t)$ is desired output, $y(t)$ is the model output, and E is the cost function. Then the parameter learning algorithm based on back-propagation is performed as follows:

Assuming that w is the connection weight of the output layer adjusting parameter in a node, the generally used learning rule is

$$w_j(t+1) = w_j(t) + \Delta w_j. \quad (16)$$

According to the chain rule, the Δw can be decomposed as:

$$\Delta w = -\eta \frac{\partial E}{\partial w} = -\eta \frac{\partial E}{\partial y} \frac{\partial y}{\partial w} = \eta e \psi_j \quad (17)$$

where η is the learning rate. Similarly, the updating laws of a_{ij} and b_{ij} are shown as follows:

$$a_{ij}(t+1) = a_{ij}(t) + \Delta a_{ij} \quad (18)$$

$$b_{ij}(t+1) = b_{ij}(t) + \Delta b_{ij} \quad (19)$$

where

$$\begin{aligned} \Delta a_{ij} &= -\eta \frac{\partial E}{\partial a_{ij}} = -\eta \frac{\partial E}{\partial y_j} \frac{\partial y_j}{\partial \psi_j} \frac{\partial \psi_j}{\partial \phi_{ij}} \frac{\partial \phi_{ij}}{\partial a_{ij}} \\ &= -\eta e w_j \phi_{ij} a_{ij}^{-\frac{3}{2}} \left(-\frac{1}{2} + a^{-1}(x_i - b_{ij})^2 \left(1 + \frac{5}{2} a^{-1} - a^{-2}(x_i - b_{ij})^2 \right) \right) \exp \left(\frac{(x_i - b_{ij})^2}{2} \right), \quad (20) \end{aligned}$$

$$\begin{aligned} \Delta b_{ij} &= -\eta \frac{\partial E}{\partial b_{ij}} = -\eta \frac{\partial E}{\partial y_j} \frac{\partial y_j}{\partial \psi_j} \frac{\partial \psi_j}{\partial \phi_{ij}} \frac{\partial \phi_{ij}}{\partial b_{ij}} \\ &= -\eta ew_j \phi_j a_{ij}^{-\frac{5}{2}} (x_i - b_{ij}) (3 - (x_i - b_{ij})^2 a_{ij}^{-2}) \exp\left(\frac{\left(\frac{x_i - b_{ij}}{a_{ij}}\right)^2}{2}\right). \end{aligned} \tag{21}$$

4. ILLUSTRATIVE EXAMPLES

In this section, there are four examples are given to demonstrate the validity of the presented WNN-HLA. The first and second examples are prediction of chaotic signals [17, 20]. The third and fourth examples are identification of nonlinear function [6, 20].

Example 1: Prediction of a Chaotic Signal

In this example, the proposed WNN-HLA model was used to predict a simple chaotic signal. The classical time series prediction problem is a one-step-ahead prediction that is described in [19]. The following equation describes the logistic function:

$$x(k + 1) = ax(k)(1 - x(k)). \tag{22}$$

The behavior of the time series generated by this equation depends critically upon the value of the parameter a . If $a < 1$, the system has a single fixed point at the origin, and from a random initial value between $[0, 1]$ the time series collapses to a constant value. For $a > 3$, the system generates a periodic attractor. Beyond the value $a = 3.6$, the system becomes chaotic. In this study, we set the parameter value a to 3.8. The first 60 pairs (from $x(1)$ to $x(60)$), with the initial value $x(1) = 0.001$, were the training data set, while the remaining 100 pairs (from $x(1)$ to $x(100)$), with initial value $x(1) = 0.9$, were the testing data set used for validating the proposed method.

In this example we set the threshold value in OLPM to 0.2. Table 1 shows the results after the OLPM clustering method was used, where $Cc_{i,j}$ and $D_{i,j}$ represent the translation and dimension distance values of i th input dimension and j th cluster.

Table 1. Translation and dimension distance values after OLPM.

Cluster	$Cc_{i,j}$	$D_{i,j}$
1	0.0274	0.0264
2	0.235	0.0538
3	0.5614	0.1288
4	0.90689	0.0429
5	0.77894	0.0623
6	0.32243	0.0084

After the parameter learning, the final rms error of the predicted output approximates 0.00089. Fig. 5 shows the prediction results of the desired output and the model output through the learning process of 500 training steps. The notation “*” represents the desired output of the time series, and the notation “+” represents the output of the

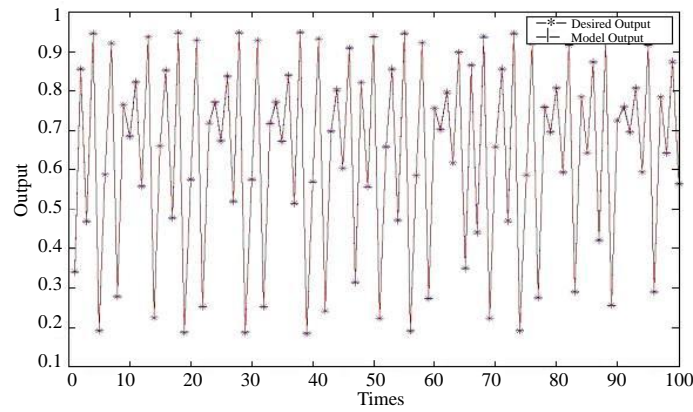


Fig. 5. The prediction results of the proposed WNN-HLA model.

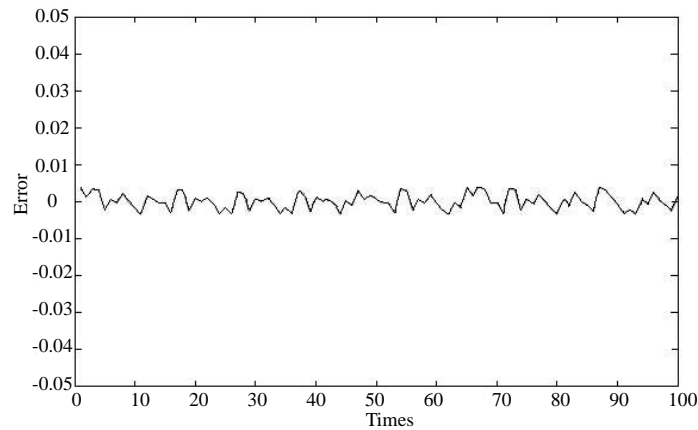


Fig. 6. The prediction errors of the proposed WNN-HLA model.

WNN-HLA model. The prediction errors of the proposed method are shown in Fig. 6. The experimental results demonstrate the perfect prediction capability of the WNN-HLA model.

We now discuss the performance of the WNN-HLA model compared with the performance of MLP [16] and RBF [2] models. Figs. 7 and 8 show the prediction results by using the MLP and RBF models. The prediction errors of the MLP and RBF models are shown in Figs. 9 and 10. Table 2 shows the performance comparison of WNN-HLA, MLP, and RBF models. Computer simulations demonstrated that the proposed method performs better performance than MLP and RBF models. Fig. 11 shows the learning curves of the WNN-HLA and MLP models. In this figure, we find that the proposed WNN-HLA model with localized basis functions converges quickly and obtains a lower rms error than MLP model.

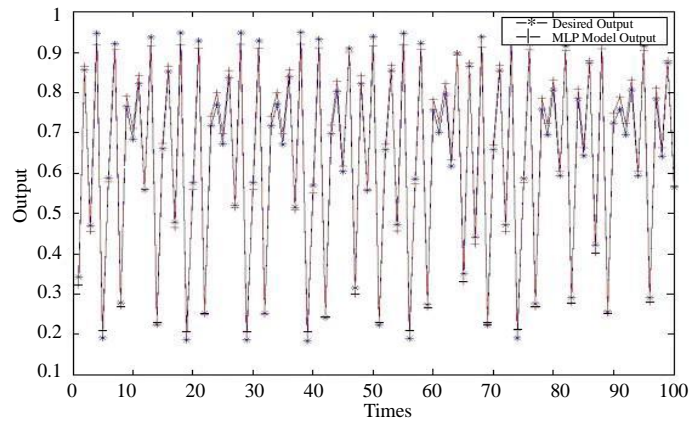


Fig. 7. The prediction results of the MLP model.

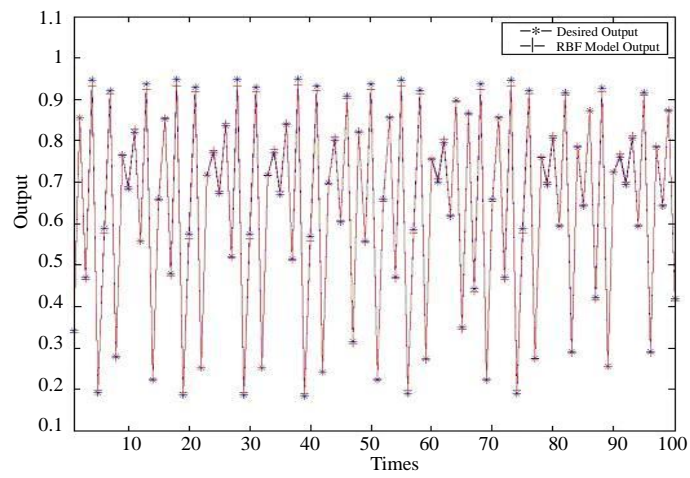


Fig. 8. The prediction results of the RBF model.

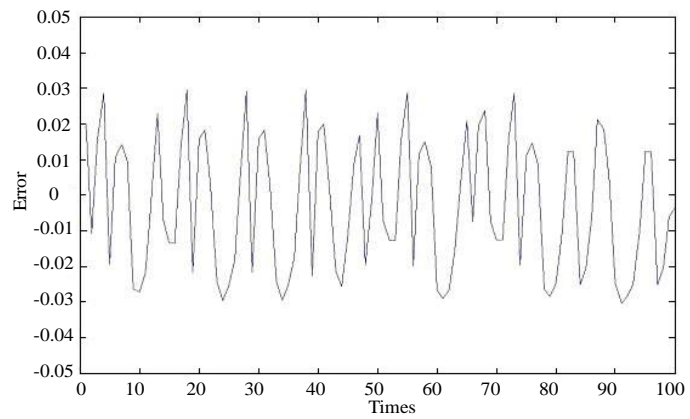


Fig. 9. The prediction errors of the MLP model.

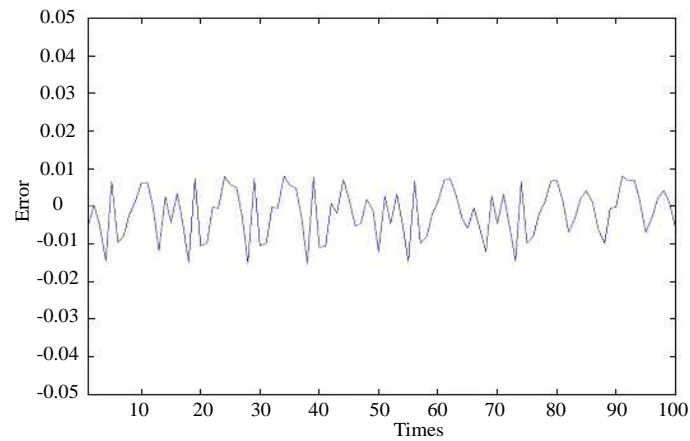


Fig. 10. The prediction errors of the RBF model.

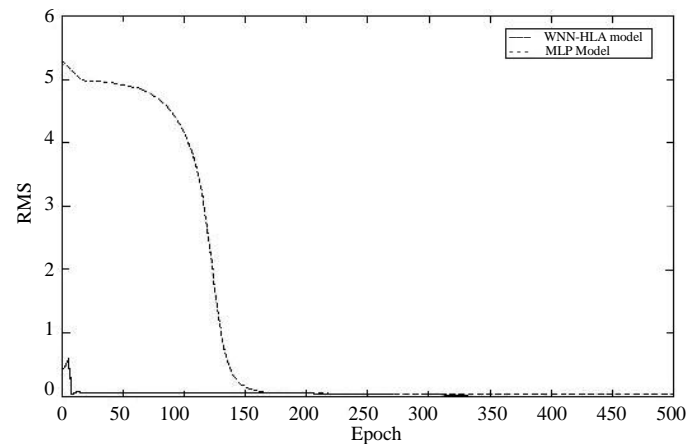


Fig. 11. The learning curves of the proposed WNN-HLA model and MLP model.

Table 2. Performance comparison of some existing models on prediction problem.

	WNN-HLA	RBF	MLP
Training Steps	500	500	500
Number of rules or wavelet bases	6	6	--
Number of Parameters	18	18	19
RMS errors	0.00089	0.0015	0.0019

Example 2: Prediction of the Chaotic Time Series

The Mackey-Glass chaotic time series $x(t)$ in consideration here was generated from the following delay differential equation:

$$\frac{dx(t)}{dt} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t). \quad (23)$$

Crowder [18] extracted 1000 input-output data pairs $\{x, y^d\}$ which consist of four past values of $x(t)$, i.e.

$$[x(t - 18), x(t - 12), x(t - 6), x(t); x(t + 6)] \tag{24}$$

where $\tau = 17$ and $x(0) = 1.2$. There were four inputs to the WNN-HLA model, corresponding to these values of $x(t)$, and one output representing the value $x(t + \Delta t)$, where Δt is a time prediction into the future. The first 500 pairs (from $x(1)$ to $x(500)$) were the training data set, while the remaining 500 pairs (from $x(501)$ to $x(1000)$) were the testing data set used for validating the proposed method. We set the threshold value in OLPM to 0.4. Table 3 shows the translation and dimension distance values after the OLPM clustering method was used.

Table 3. Translation and dimension distance values after OLPM.

Cluster	Cc_{1-j}	Cc_{2-j}	Cc_{3-j}	Cc_{4-j}	D_{1-j}	D_{2-j}	D_{3-j}	D_{4-j}
1	1.001	0.9601	0.9187	0.8793	0.0904	0.0907	0.0855	0.0770
2	0.7840	0.6323	0.7124	0.7475	0.1167	0.1045	0.0912	0.1165
3	0.8603	0.9112	0.9592	1.002	0.1103	0.1031	0.0929	0.0815
4	1.1365	1.1592	1.1679	1.1581	0.122	0.1070	0.0910	0.0778
5	1.1125	1.1034	1.1334	0.9919	0.1208	0.0762	0.0675	0.1087
6	0.9318	0.8663	0.8160	0.7832	0.0721	0.0603	0.0607	0.0759
7	0.7043	0.6654	0.6233	0.6149	0.0967	0.0994	0.0958	0.1219
8	0.6794	0.7629	0.8121	0.8234	0.0771	0.0587	0.0654	0.0995
9	1.2400	1.2619	1.2427	1.1932	0.0558	0.0428	0.0586	0.0891
10	0.5321	0.4853	0.5243	0.5582	0.0822	0.0509	0.0938	0.1163
11	0.5817	0.6185	0.6656	0.7647	0.0843	0.0696	0.0517	0.0792

After the parameter learning, the final rms error of the prediction output approximates 0.006. Fig. 12 shows the prediction outputs of the chaotic time series from $x(501)$ to $x(1000)$, when 500 training data from $x(1)$ to $x(500)$ were used. The solid line represents the desired output while the dotted line represents the model output. The prediction errors between the proposed WNN-HLA model and the desired output are shown in Fig. 13. We also perform this example by using RBF model [2]. The prediction results and the prediction errors are shown in Figs. 14 and 15. Computer simulations demonstrated that the proposed method performs better performance than RBF model.

In this example, as with example 1, we also compared the performance of the WNN- HLA model with other methods. Table 4 lists the generalization capabilities of other methods [2, 18, 19, 23]. The generalization capabilities were measured by using each model to predict 500 points immediately following the training data set. Clearly, simulation results show that the proposed model can obtain smaller RMSE than other methods.

Example 3: Identification of the Nonlinear Dynamic System

In this example, the proposed WNN-HLA model is used to identify a dynamic system. The identification model has the form

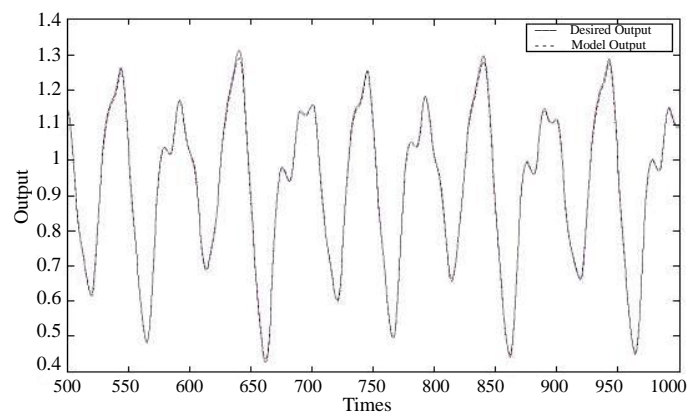


Fig. 12. The prediction results of the proposed WNN-HLA model.

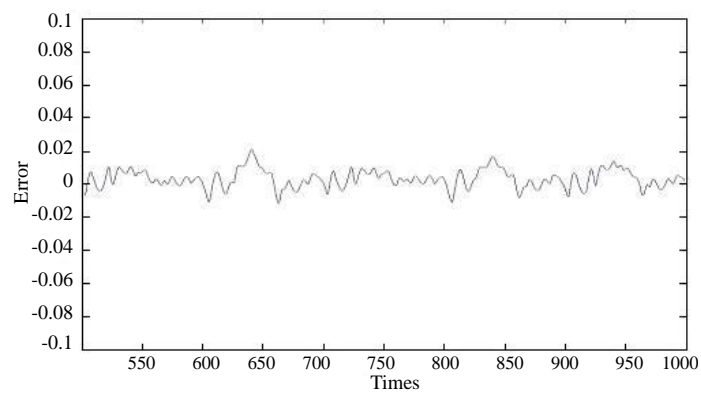


Fig. 13. The prediction errors of the proposed WNN-HLA model.

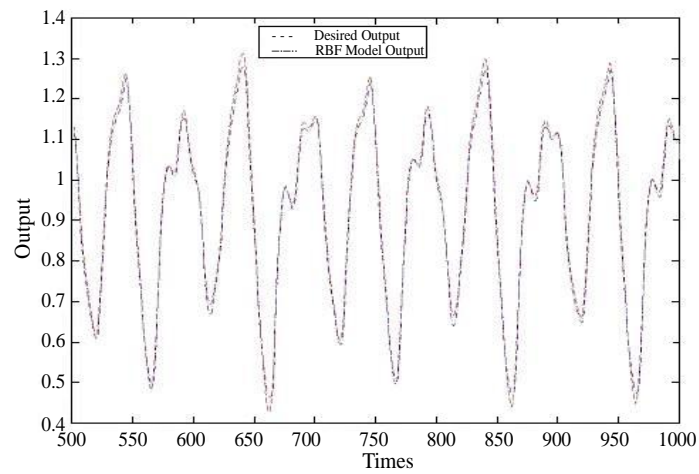


Fig. 14. The prediction results of the RBF model.

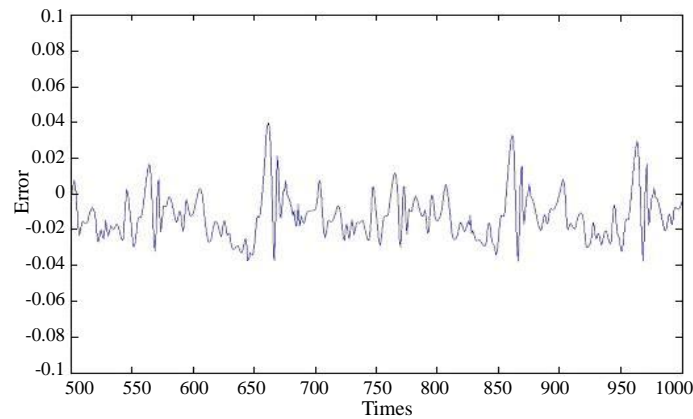


Fig. 15. The prediction errors of the RBF model.

Table 4. Performance comparison of some existing models.

LEARNING ALGORITHM	Training cases	RMSE _{train}
WNN-HLA	500	0.006
RBF [2]	500	0.0072
ANFIS [23]	500	0.007
Back-propagation NN	500	0.02
Six-order polynomial	500	0.04
Cascaded-correlation	500	0.06
Linear predictive model	2000	0.55

$$\hat{y}(k + 1) = \hat{f}[u(k), u(k + 1), \dots, u(k - p + 1), y(k), y(k + 1), \dots, y(k - q + 1)]. \quad (25)$$

Since both the unknown plant and WNN-HLA are driven by the same input, the WNN-HLA adjusts itself with the goal of causing the output of the identification model to match that of the unknown plant. Upon convergence, their input-output relationship should match. The plant to be identified is guided by the difference equation

$$y(k + 1) = \frac{y(k)}{1 + y^2(k)} + u^3(k). \quad (26)$$

The output of the plant depends nonlinearly on both its past values and inputs, but the effects of the input and output values are additive. The 200 training input patterns are generated from $u(k) = \sin(2\pi k/100)$ and training process is continued for 100 epochs. In Fig. 16, the outputs of the WNN-HLA are represented as “- -” while the plant outputs are represented as “—”. The simulation results show the perfect identification capability of the well-trained WNN-HLA. Fig. 17 shows the prediction errors of the proposed method. The final distribution of wavelet bases in $u[k]$ and $u[k - 1]$ dimensions are shown in Figs. 18 and 19.

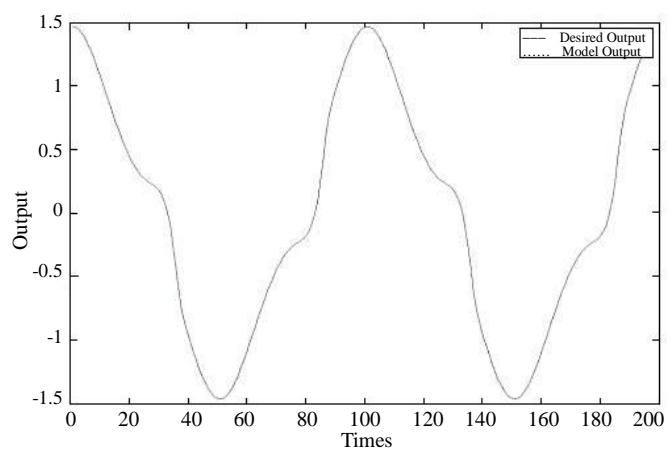


Fig. 16. The identification results of the proposed WNN-HLA model.

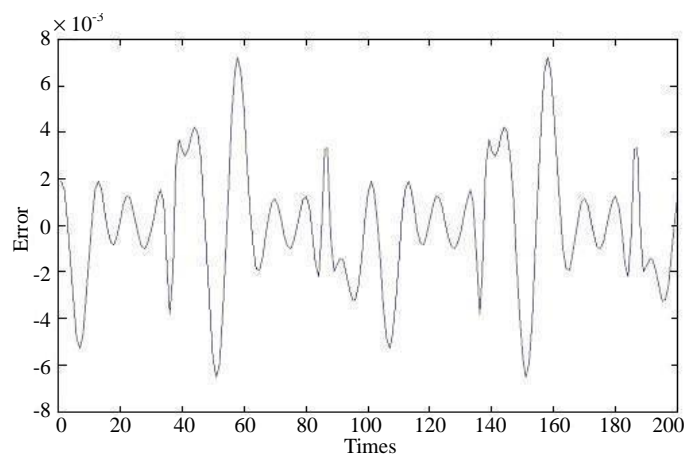


Fig. 17. The identification errors of the proposed WNN-HLA model.

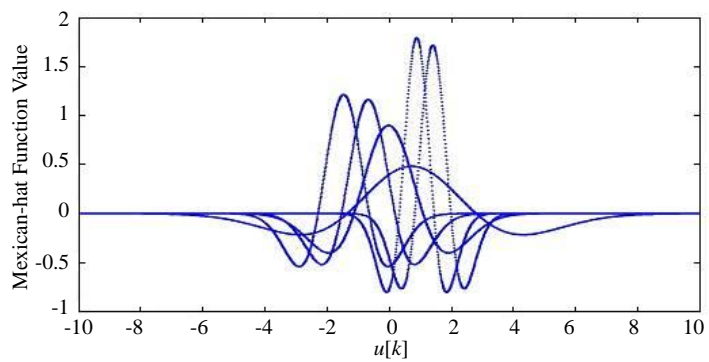


Fig. 18. The final distribution of wavelet bases in $u[k]$ dimension.

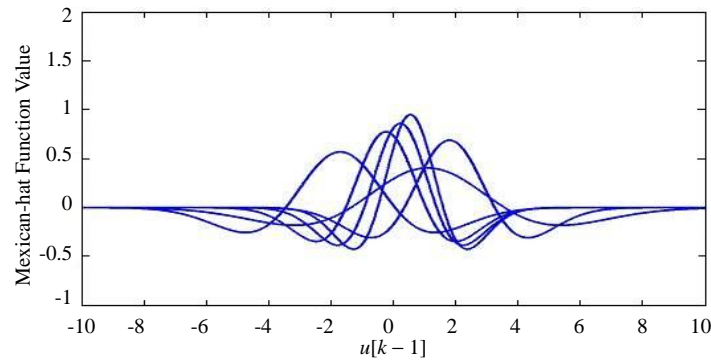


Fig. 19. The final distribution of wavelet bases in $u[k - 1]$ dimension.

Table 5. Performance comparison of some existing models on identification problem.

	WNN-HLA	FALCON [20]	SONFIN [21]	SCFNN [22]
Training Steps	30000	60000	50000	30000
Number of rules or wavelet bases	6	6	5	22
Number of parameters	30	54	29	110
RMS errors	0.012	0.02	0.013	0.017

We now compare the performance of WNN-HLA model with other models [20-22]. The performance indexes considered include training steps, wavelet or rule numbers, parameter numbers, and rms errors. The comparison results are tabulated in Table 5. The results show that the proposed WNN-HLA needs fewer adjustable parameters and rms errors than FALCON [20] and SCFNN [22] models.

Example 4: Approximation of a Nonlinear Function

The example is a simulation study to approximate the nonlinear function

$$f(x) = (x_1^2 - x_2^2) \sin(0.5x_1), \quad -10 \leq x_1, x_2 \leq 10. \tag{27}$$

A data set of 400 training pairs was collected from random inputs that were distributed uniformly over the domain $D = [-10, 10] \times [-10, 10]$. We set the threshold value in OLPM to 2.4. There are eight clusters generated after the OLPM clustering method was used. Figs. 20 and 21 illustrate the original form of $f(x)$ over D , and its approximate was realized by using the WNN-HLA model.

For the purpose of comparing our model with other wavelet networks, we postulate the measure in

$$J = \sqrt{\frac{\sum_{l=1}^L (Y_l - Y_l^d)^2}{\sum_{l=1}^L (Y_l^d - \bar{Y})^2}} \quad \text{with} \quad \bar{Y} = \frac{1}{L} \sum_{l=1}^L Y_l^d \tag{28}$$

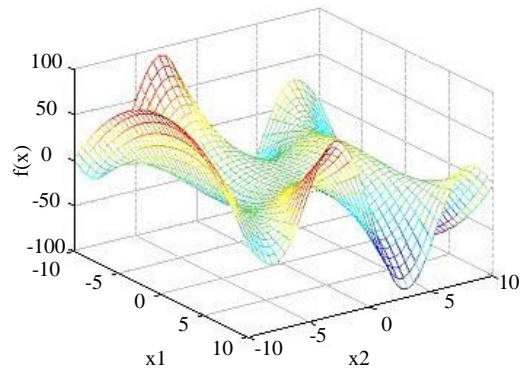


Fig. 20. The original function $f(x) = (x_1^2 - x_2^2) \sin(0.5x_1)$.

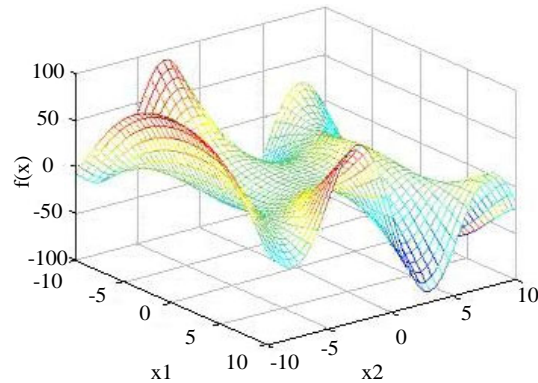


Fig. 21. Approximation of the function $f(x) = (x_1^2 - x_2^2) \sin(0.5x_1)$ using the WNN-HLA model.

Table 6. Performance comparison of some models on approximation problem.

	WNN-HLA	RBF [2]	WNN [8]	WN[6]
Training Steps	10000	10000	10000	10000
Number of rules or wavelet bases	6	6	10	22
Number of parameters	80	80	80	96
Performance index J	0.019	0.05	0.03	0.033

as the performance index for this example. In Eq. (28), Y_l^d is the l th desired output, and Y_l is the l th estimated output from the WNN-HLA model.

Table 6 shows the approximation results obtained by the WNN-HLA, RBF [2], WN [6], and WNN [8] models. Simulation results show that the proposed WNN-HLA model obtained a smaller performance index (J) than other wavelet networks [6, 8].

5. CONCLUSION

In this paper, a novel hybrid learning algorithm is proposed. First, a structure learning scheme is used to determine proper input space partitioning and to find the transla-

tion and dilation values of each cluster. With the proposed on-line partition method, a flexible partitioning of the input space is achieved. The number of nodes is relatively small compared to the grid partition [23]. Second, a back-propagation learning scheme is used to adjust the parameters for the desired outputs. Four simulation examples have been given to illustrate the performance and effectiveness of the proposed model. We also compare the performance of WNN-HLA model with other wavelet models. The computer simulations demonstrate that the proposed WNN-HLA model can obtain a better performance than other methods for prediction and identification problems.

REFERENCES

1. G. J. Gibson and C. G. N. Cowan, "On the decision regions of multilayer perceptrons," in *Proceedings of the IEEE*, Vol. 78, 1990, pp. 1590-1594.
2. J. A. Leonard and M. A. Kramer, "Radial basis function networks for classifying process faults," *IEEE Control Systems Magazine*, Vol. 11, 1991, pp. 31-38.
3. S. Chen, S. A. Billings, and P. M. Grant, "Recursive hybrid algorithm for nonlinear system identification using radial basis function networks," *International Journal of Control*, Vol. 55, 1992, pp. 1051-1070.
4. B. Mulgrew, "Applying radial basis function," *IEEE Signal Processing Magazine*, Vol. 13, 1996, pp. 53-63.
5. I. Daubechies, "The wavelet transform, time-frequency localization and signal analysis," *IEEE Transforms on Information Theory*, Vol. 36, 1990, pp. 961-1005.
6. Q. Zhang and A. Benveniste, "Wavelet networks," *IEEE Transactions on Neural Networks*, Vol. 3, 1992, pp. 889-898.
7. I. Dubechies, "Orthonormal base of compactly supported wavelets," *Communications on Pure and Applied Mathematics*, Vol. 91, 1995, pp. 909-996.
8. J. Zhang, G. G. Walter, Y. Miao, and W. N. W. Lee, "Wavelet neural networks for function learning," *IEEE Transactions on Signal Processing*, Vol. 43, 1995, pp. 1485-1497.
9. J. Chen and D. D. Bruns, "WaveARX neural network development for system identification using a systematic design synthesis," *Industrial and Engineering Chemistry Research*, Vol. 34, 1995, pp. 4420-4435.
10. P. Cristea, R. Tuduce, and A. Cristea "Time series prediction with wavelet neural networks," in *Proceedings of the IEEE Neural Network Applications in Electrical Engineering*, 2000, pp. 5-10.
11. T. Yamakawa, E. Uchino, and T. Samatsu, "Wavelet neural networks employing over-complete number of compactly supported non-orthogonal wavelets and their applications," in *Proceedings of the IEEE Conference on Neural Networks*, Vol. 3, 1994, pp. 1391-1396.
12. A. Ikonomopoulos and A. Endou, "Wavelet decomposition and radial basis function networks for system monitoring," *IEEE Transactions on Nuclear Science*, Vol. 45, 1998, pp. 2293-2301.
13. D. W. C. Ho, P. A. Zhang, and J. Xu, "Fuzzy wavelet networks for function learning," *IEEE Transactions on Fuzzy Systems*, Vol. 9, 2001, pp. 200-211.
14. F. J. Lin, R. J. Wai, and M. P. Chen, "Wavelet neural network control for linear ul-

- trasonic motor drive via adaptive sliding-mode technique," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, Vol. 50, 2003, pp. 686-698.
15. Y. C. Huang and C. M. Huang, "Evolving wavelet networks for power transformer condition monitoring," *IEEE Transactions on Power Delivery*, Vol. 17, 2002, pp. 412-416.
 16. M. Meyer and G. Pfeiffer, "Multilayer perceptron based decision feedback equalizers for channels with intersymbol interference," in *IEE Proceedings-I*, Vol. 140, 1993, pp. 420-424.
 17. C. T. Lin and C. S. G. Lee, *Neural Fuzzy Systems: A Neural-Fuzzy Synergism to Intelligent Systems*, Prentice-Hall, Englewood Cliffs, New Jersey, 1996.
 18. R. S. Cowder, "Predicting the Mackey-glass time series with cascade-correlation learning," in *Proceedings of the Connectionist Models Summer School*, in D. Touretzky, G. Hinton, and T. Sejnowski (eds.), 1990, pp. 117-123.
 19. A. S. Lapedes and R. Farber, "Nonlinear signal processing using neural network: prediction and system modelling," Technical Report No. LA-UR-87-2662, Los Alamos National Laboratory, Los Alamos, New Mexico, 1987.
 20. C. J. Lin and C. T. Lin, "An ART-based fuzzy adaptive learning control network," *IEEE Transactions on Fuzzy Systems*, Vol. 5, 1997, pp. 477-496.
 21. C. F. Juang and C. T. Lin, "An on-line self-constructing neural fuzzy inference network and its applications," *IEEE Transactions on Fuzzy Systems*, Vol. 6, 1998, pp. 12-31.
 22. F. J. Lin, C. H. Lin, and P. H. Shen, "Self-constructing fuzzy neural network speed controller for permanent-magnet synchronous motor drive," *IEEE Transactions on Fuzzy Systems*, Vol. 9, 2001, pp. 751-759.
 23. J. S. R. Jang, "ANFIS: adaptive-network-based fuzzy inference system," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 23, 1993, pp. 665-685.
 24. M. A. Lee and H. Takagi, "Integrating design stages of fuzzy systems using genetic algorithms," in *Proceedings of the IEEE International Conference on Fuzzy Systems*, Vol. 1, 1993, pp. 612-617.
 25. C. F. Juang, "A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithms," *IEEE Transactions on Fuzzy Systems*, Vol. 10, 2002, pp. 155-170.
 26. M. C. Hung and D. L. Yang, "The efficient fuzzy c-means clustering technique," in *Proceedings of the IEEE International Conference on Data Mining*, 2001, pp. 225-232.
 27. J. He, L. Liu, and G. Palm, "Speaker identification using hybrid LVQ-SLP networks," in *Proceedings of the IEEE International Conference on Neural Networks*, Vol. 4, 1995, pp. 2052-2055.
 28. K. K. Ang, C. Quek, and M. Pasquier, "POPFNN-CRI(S): pseudo outer product based fuzzy neural network using the compositional rule of inference and singleton fuzzifier," *IEEE Transactions on Systems, Man, and Cybernetics – Part B*, Vol. 33, 2003, pp. 838-849.
 29. W. L. Tung and C. Quek, "GenSoFNN: a generic self-organizing fuzzy neural network," *IEEE Transactions on Neural Networks*, Vol. 13, 2002, pp. 1075-1086.



Cheng-Jian Lin (林正堅) received the B.S. degree in Electrical Engineering from Tatung University, Taiwan, R.O.C., in 1986 and the M.S. and Ph.D. degrees in Electrical and Control Engineering from the National Chiao Tung University, Taiwan, R.O.C., in 1991 and 1996. From April 1996 to July 1999, he was an Associate Professor in the Department of Electronic Engineering, Nan Kai College, Nantou, Taiwan, R.O.C. Since August 1999, he has been with the Department of Computer Science and Information Engineering, Chaoyang University of Technology. Currently, he is a Professor of Computer Science and Information Engineering Department, Chaoyang University of Technology, Taichung, Taiwan, R.O.C. He served as the chairman of Computer Science and Information Engineering Department from 2001 to 2005. His current research interests are neural networks, fuzzy systems, pattern recognition, intelligence control, bioinformatics, and FPGA design. He has published more than 90 papers in the referred journals and conference proceedings. Dr. Lin is a member of the Phi Tau Phi. He is also a member of the Chinese Fuzzy Systems Association (CFSA), the Chinese Automation Association, the Taiwanese Association for Artificial Intelligence (TAAI), the IEICE (The Institute of Electronics, Information and Communication Engineers), and the IEEE Computational Intelligence Society. He is an executive committee member of the Taiwanese Association for Artificial Intelligence (TAAI). Dr. Lin currently serves as the Associate Editor of International Journal of Applied Science and Engineering.