Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2013, Article ID 425740, 11 pages
http://dx.doi.org/10.1155/2013/425740

Research Article

Review on Methods to Fix Number of Hidden Neurons in Neural Networks

K. Gnana Sheela and S. N. Deepa

Anna University, Regional Centre, Coimbatore 641047, India

Correspondence should be addressed to K. Gnana Sheela; sheelabijins@gmail.com

Received 18 March 2013; Revised 16 May 2013; Accepted 26 May 2013

Academic Editor: Matjaz Perc

This paper reviews methods to fix a number of hidden neurons in neural networks for the past 20 years. And it also proposes a new method to fix the hidden neurons in Elman networks for wind speed prediction in renewable energy systems. The random selection of a number of hidden neurons might cause either overfitting or underfitting problems. This paper proposes the solution of these problems. To fix hidden neurons, 101 various criteria are tested based on the statistical errors. The results show that proposed model improves the accuracy and minimal error. The perfect design of the neural network based on the selection criteria is substantiated using convergence theorem. To verify the effectiveness of the model, simulations were conducted on real-time wind data. The experimental results show that with minimum errors the proposed approach can be used for wind speed prediction. The survey has been made for the fixation of hidden neurons in neural networks. The proposed model is simple, with minimal error, and efficient for fixation of hidden neurons in Elman networks.

1. Introduction

One of the major problems facing researchers is the selection of hidden neurons using neural networks (NN). This is very important while the neural network is trained to get very small errors which may not respond properly in wind speed prediction. There exists an overtraining issue in the design of NN training process. Over training is akin to the issue of overfitting data. The issue arises because the network matches the data so closely as to lose its generalization ability over the test data.

Artificial neural networks (ANN) is an information processing system which is inspired by the models of biological neural networks [1]. It is an adaptive system that changes its structure or internal information that flows through the network during the training phase. ANN is widely used in many areas because of its features such as strong capacity of nonlinear mapping, high accuracy for learning, and good robustness. ANN can be classified into feedforward and feedback network. Back propagation network and radial basis function network are the examples of feedforward network, and Elman network is an example of feedback network. The feedback has a profound impact on the learning capacity and its performance in modeling nonlinear dynamical phenomena.

From the development of NN model, the researcher can face the following problems for a particular application [2].

(i) How many hidden neurons can be used?

(ii) How many training pairs should be used?

(iii) Which training algorithm can be used?

(iv) What neural network architecture should be used?

The hidden neuron can influence the error on the nodes to which their output is connected. The stability of neural network is estimated by error. The minimal error reflects better stability, and higher error reflects worst stability. The excessive hidden neurons will cause over fitting; that is, the neural networks have overestimate the complexity of the target problem [3]. It greatly degrades the generalization capability to lead with significant deviation in prediction. In this sense, determining the proper number of hidden neurons to prevent over fitting is critical in prediction problem. The modeling process involves creating a model and developing the model with the proper values [4]. One of the major

challenges in the design of neural network is the fixation of hidden neurons with minimal error and highest accuracy. The training set and generalization error are likely to be high before learning begins. During training, the network adapts to decrease the error on the training patterns. The accuracy of training is determined by the parameters underconsideration. The parameters include NN architecture, number of hidden neurons in hidden layer, activation function, inputs, and updating of weights.

Prediction plays a major role in planning today's competitive environment, especially in the areas characterized by high concentration of wind generation. Due to the fluctuation and intermittent nature of wind, prediction result varies rapidly. Thus this increases the importance of accurate wind speed prediction. The proposed model is to be implemented in Elman network for the accurate wind speed prediction model. The need for wind speed prediction is to assist with operational control of wind farm and planning development of power station. The quality of prediction made by the network is measured in terms of error. Generalization performance varies over time as the network adapts during training.

Thus various criteria were proposed for fixing hidden neuron by researchers during the last couple of decades. Most of researchers have fixed number of hidden neurons based on trial rule. In this paper, new method is proposed and is applied for Elman network for wind speed prediction. And the survey has been made for the fixation of hidden neuron in neural networks for the past 20 years. All proposed criteria are tested using convergence theorem which converges infinite sequences into finite sequences. The main objective is to minimize error, improve accuracy and stability of network. This review is to be useful for researchers working in this field and selects proper number of hidden neurons in neural networks.

2. Literature Survey

Several researchers tried and proposed many methodologies to fix the number of hidden neurons. The survey has been made to find the number of hidden neurons in neural network is and described in a chronological manner. In 1991, Sartori and Antsaklis [5] proposed a method to find the number of hidden neurons in multilayer neural network for an arbitrary training set with P training patterns. Several existing methods are optimized to find selection of hidden neurons in neural networks. In 1993, Arai [6] proposed two parallel hyperplane methods for finding the number of hidden neurons. The $2^n/3$ hidden neurons are sufficient for this design of the network.

In 1995, Li et al. [7] investigated the estimation theory to find the number of hidden units in the higher order feedforward neural network. This theory is applied to the time series prediction. The determination of an optimal number of hidden neurons is obtained when the sufficient number of hidden neurons is assumed. According to the estimation theory, the sufficient number of hidden units in the second-order neural network and the first-order neural networks are 4 and 7, respectively. The simulation results show that the second-order neural network is better than the first-order in training convergence. According to that, the network with few nodes in the hidden layer will not be powerful for most applications. The drawback is long training and testing time.

In 1996, Hagiwara [8] presented another method to find an optimal number of hidden units. The drawback is that there is no guarantee that the network with a given number of hidden units will find the correct weights. According to the statistical behaviors of the output of the hidden units, if a network has large number of hidden nodes, a linear relationship is obtained in the hidden nodes.

In 1997, Tamura and Tateishi [9] developed a method to fix hidden neuron with negligible error based on Akaike's information criteria. The number of hidden neurons in three layer neural network is $N - 1$ and four-layer neural network is $N/2 + 3$ where N is the input-target relation. In 1998, Fujita [10] proposed a statistical estimation of number of hidden neurons. The merits are speed learning. The number of hidden neurons mainly depends on the output error. The estimation theory is constructed by adding hidden neurons one by one. The number of hidden neurons is formulated as $N_h = K \log \|P_c Z\| / \log S$, where S is total number of candidates that are randomly searched for optimum hidden unit c is allowable error.

In 1999, Keeni et al. [11] presented a method to determine the number of hidden units which are applied in the prediction of cancer cells. Normally, training starts with many hidden units and then prune the network once it has trained. However pruning does not always improve generalization. The initial weights for input to hidden layer and the number of hidden units are determined automatically. The demerit is no optimal solution.

In 2001, Onoda [12] presented a statistical approach to find the optimal number of hidden units in prediction applications. The minimal errors are obtained by the increase of number of hidden units. Md. Islam and Murase [13] proposed a large number of hidden nodes in weight freezing of single hidden layer networks. The generalization ability of network may be degraded when the number of hidden nodes (N_h) is large because Nth hidden node may have some spurious connections.

In 2003, Zhang et al. [14] implemented a set covering algorithm (SCA) in three-layer neural network. The SCA is based on unit sphere covering (USC) of hamming space. This methodology is based on the number of inputs. Theoretically the number of hidden neurons is estimated by random search. The output error decreases with N_h being added. The N_h is significant in characterizing the performance of the network. The number of hidden neurons should not be too large for heuristic learning system. The N_h found in set covering algorithm is $3L/2$ hidden neurons where L is the number of unit spheres contained in N dimensional hidden space. In the same year Huang [15] developed the model for learning and storage capacity of two-hidden-layer feedforward network. In 2003, Huang [15] formulated the following: $N_h = \sqrt{(m+2)N} + 2\sqrt{N/(m+2)}$ in single hidden layer and $N_h = m\sqrt{N/(m+2)}$ in two hidden layer. The main features are the following.

(i) It has high first hidden layer and small second hidden layer.

(ii) Weights connecting the input to first hidden layer can be prefixed with most of the weights connecting the first hidden layer and second hidden layer that can be determined analytically.

(iii) It may be trained only by adjusting weights and quantization factors to optimize the generalization performance.

(iv) It may be able to overfit the sample with any arbitrary small error.

In 2006, Choi et al. [16] developed a separate learning algorithm which includes a deterministic and heuristic approach. In this algorithm, hidden-to-output and input-to-hidden nodes are separately trained. It solved the local minima in two-layered feedforward network. The achievement is best convergence speed. In 2008, Jiang et al. [17] presented the lower bound on the number of hidden neurons. The number of hidden neurons is $N_h = q^n$ where q is valued upper bound function. The calculated values represent that the lower bound is tighter than the ones that has existed. The lower and upper bound on the number of hidden neurons help to design constructive learning algorithms. The lower bound can accelerate the learning speed, and the upper bound gives the stopping condition of constructive learning algorithms. It can be applied to the design of constructive learning algorithm with training set N numbers. In the same year, Jinchuan and Xinzhe [3] investigated a formula tested on 40 cases: $N_h = (N_{\text{in}} + \sqrt{N_p})/L$ where L is the number of hidden layer, N_{in} is the number of input neuron and N_p is the number of input sample. The optimum number of hidden layers and hidden units depends on the complexity of network architecture, the number of input and output units, the number of training samples, the degree of the noise in the sample data set, and the training algorithm.

The quality of prediction made by the network is measured in terms of the generalization error. Generalization performance varies over time as the network adapts during training. The necessary numbers of hidden neurons approximated in hidden layer using multilayer perceptron (MLP) were found by Trenn [18]. The key points are simplicity, scalability, and adaptivity. The number of hidden neurons is $N_h = n + n_0 - 1/2$ where n is the number of inputs and n_0 is the number of outputs. In 2008, Xu and Chen [19] developed a novel approach for determining optimum number of hidden neurons in data mining. The best number of hidden neurons leads to minimum root means Squared Error. The implemented formula is $N_h = C_f(N/d \log N)^{1/2}$, where N is number of training pairs, d is input dimension, and C_f is first absolute moment of Fourier magnitude distribution of target function.

In 2009, Shibata and Ikeda [20] investigated the effect of learning stability and hidden neuron in neural network. The simulation results show that the hidden output connection weight becomes small as number of hidden neurons N_h becomes large. This is implemented in random number mapping problems. The formula for hidden nodes is $N_h = \sqrt{N_i N_0}$ where N_i is the input neuron and N_0 is the output neuron. Since neural network has several assumptions which are given before starting the discussion to prevent divergent. In unstable models, number of hidden neurons becomes too large or too small. A tradeoff is formed that if the number of hidden neurons becomes too large, output of neurons becomes unstable, and if the number of hidden neurons becomes too small, the hidden neurons becomes unstable again.

In 2010, Doukim et al. [21] proposed a technique to find the number of hidden neurons in MLP network using coarse-to-fine search technique which is applied in skin detection. This technique includes binary search and sequential search. This implementation is trained by 30 networks and searched for lowest mean squared error. The sequential search is performed in order to find the best number of hidden neurons. Yuan et al. [22] proposed a method for estimation of hidden neuron based on information entropy. This method is based on decision tree algorithm. The goal is to avoid the overlearning problem because of exceeding numbers of the hidden neurons and to avoid the shortage of capacity because of few hidden neurons. The number of hidden neurons of feedforward neural network is generally decided on the basis of experience. In 2010, Wu and Hong [23] proposed the learning algorithms for determination of number of hidden neurons. In 2011, Panchal et al. [24] proposed a methodology to analyze the behavior of MLP. The number of hidden layers is inversely proportional to the minimal error.

In 2012, Hunter et al. [2] developed a method used in proper NN architectures. The advantages are the absence of trial-and-error method and preservation of the generalization ability. The three networks MLP, bridged MLP, and fully connected cascaded network are used. The implemented formula: as follows, $N_h = N + 1$ for MLP Network, $N_h = 2N + 1$ for bridged MLP Network and $N_h = 2^n - 1$ for fully connected cascade NN. The experimental results show that the successive rate decreases with increasing parity number. The successive rate increases with number of neurons used. The result is obtained with 85% accuracy.

The other algorithm used to fix the hidden neuron is the data structure preserving (DSP) algorithm [25]. It is an unsupervised neuron selection algorithm. The data structure denotes relative location of samples in high dimensional space. The key point is retaining the separate margin underlying the full set of neuron. The optimum number of hidden nodes is found out by trial-and-error approach [26]. The advantages are the improvement in learning and classification, cavitations signal using Elman neural network. The simulation results show that error gradient and N_h selection scheme work well. Another approach is to fix hidden neuron based on information entropy which uses decision tree algorithm [27]. The N_h is generally decided based on the experience. Initially N_h should be trained. The activation values of hidden neuron should be calculated by inputting the training sample. Finally information is calculated. To select the hidden neurons, SVM stepwise algorithm is used. In this algorithm, linear programming SVM is employed to preselect the number of hidden neurons [28]. The performance is evaluated by RMSE (root means square error). The advantage

is improved computation time. The hidden neuron is selected empirically such as 2, 4, 6, 12, 24, and it is applied in sonar target classification problem [29]. It is close to Bayes classifier. The analysis of variance is done on the result of the aspect angle dependent test experiments. The improvement of performance is by 10%.

Another approach to fix hidden neuron is the sequential orthogonal approach (SOA). This approach [30] is about adding hidden neurons one by one. Initially, increase N_h sequentially until error is sufficiently small. This selection problem can be approached statistically by generalizing Akaike's information criterion (AIC) to be applied in unrealizable model under general loss function including regularization. The other existing methods are trial and error, thump rule, and so forth. In thump rule, N_h is between size of number of input neurons and number of output neurons. Another rule is equal to 2/3 size of input layer and output layer [19]. The other N_h is less than twice the size of input layer. The number of hidden neuron depends on number of inputs, outputs, architectures, activations, training sets, algorithms, and noises. The demerit is higher training time. The other existing techniques are network growing and network pruning [31, 32]. The growing algorithm allows the adaptation of network structure. This starts with undersized N_h and adds neurons to number of hidden neurons. The disadvantages are time consuming and no guarantee of fixing the hidden neuron.

The researchers have been implemented various methods for selecting the hidden neuron. The researchers are aimed at improving factors like faster computing process, more efficiency and accuracy and less errors. The proper selection of hidden neuron is important for the design of neural network.

3. Problem Description

The proper selection of number of hidden neurons has been analyzed for Elman neural network. To select hidden neurons in order to to solve a specific task has been an important problem. With few hidden neurons, the network may not be powerful enough to meet the desired requirements including capacity and error precision. In the design of neural network, an issue called overtraining has occurred. Over training is akin to the problem of overfitting data. So fixing the number of a hidden neuron is important for a given problem. An important but difficult task is to determine the optimal number of parameters. In other words, it needs to measure the discrepancy between neural network and an actual system. In order to tackle this, most researches have mainly focused on improving the performance. There is no way to find hidden neuron in neural network without trying and testing during the training and computing the generalization error. The hidden output connection weights becomes small as number of hidden neurons become large, and also the tradeoff in stability between input and hidden output connection exists. A tradeoff is formed that if the N_h becomes too large, the output neurons becomes unstable, and if the number of hidden neuron becomes too small, the hidden neuron becomes unstable again. The problems in the fixation of hidden neurons still exist. The properties of neural networks are convergence and stability to be verified by the performance analysis. The problem of wind speed prediction is closely linked to intermittency nature of wind. The characteristics of wind involve uncertainty.

The input and output neuron is to be modeled, while N_h should be fixed properly in order to provide good generalization capabilities for the prediction. ANN comprising deficient number of hidden neuron may not be feasible to dynamic system. During the last couple decades, various methods were developed to fix hidden neurons. Nowadays most predicting research fields have been heuristic in nature. There is no generally accepted theory to determine how many hidden neurons are needed to approximate any given function in single hidden layer. If it has few numbers of hidden neurons, it might have a large training error due to underfitting. If it has more numbers of hidden neurons, might have a large training error due to overfitting. An exceeding number of hidden neurons made on the network deepen the local minima problem [30]. The proposed method shows that there is a stable performance on training despite of the large number of hidden neurons in Elman network. The objective is to select hidden neurons to design the Elman network and minimize the error for wind speed prediction in renewable energy systems. Thus, research is being carried out for fixing hidden neuron in neural networks. The optimal number of hidden neurons based on the following error criteria. The error criteria such as mean square error (MSE), mean relative error (MRE), and Mean Absolute Error (MAE) are assessed on the proposed model performance. The fixing of the number of hidden neurons in Elman network is based on minimal error performance. The formulas of error criteria are as follows;

$$\begin{aligned} \mathrm{MSE} &= \sum_{i=1}^{N} \frac{\left(Y_i' - Y_i\right)^2}{N}, \\ \mathrm{MRE} &= \frac{1}{N} \sum_{i=1}^{N} \left| \frac{\left(Y_i' - Y_i\right)}{Y_i} \right|, \\ \mathrm{MAE} &= \frac{1}{N} \sum_{i=1}^{N} \left(Y_i' - Y_i\right), \end{aligned} \tag{1}$$

where Y_i is predicted output, Y_i' is actual output, $\overline{Y_i}$ is average actual output, and N is number of samples. The process of designing the network plays an important role in the performance of network.

4. Proposed Architecture

There exists various heuristics in the literature; amalgamating the knowledge gained from previous experiments where a near optimal topology might exist [33–35]. The objective is to devise the criteria that estimate the number of hidden neurons as a function of input neurons (n) and to develop the model for wind speed prediction in renewable energy

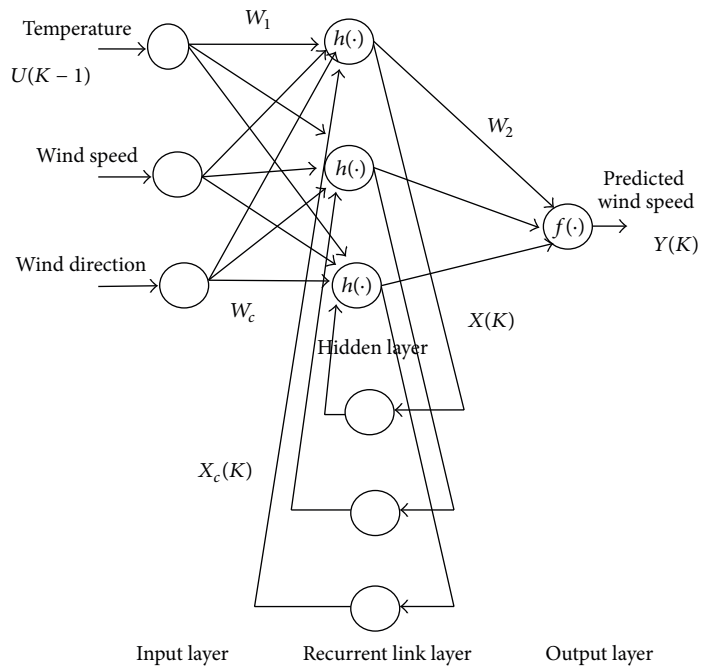

Figure 1: Architecture of the proposed model for fixing the number of hidden neurons in Elman Network.

systems. The estimate can take the form of a single exact topology to be adopted.

4.1. Overview of Elman Network. Elman network has been successfully applied in many fields, regarding prediction, modeling, and control. The Elman network is a recurrent neural network (RNN) adding recurrent links into hidden layer as a feedback connection [36–38]. It consists of input, hidden, recurrent link and output layer. The recurrent layer copies one step delay of hidden layer. It reflects both input and output layers' information by intervening fedback between output of input and hidden layers. The output is taken from the hidden layer. The feedback is stored in another layer called recurrent link layer which retains the memory. It is chosen due to hidden layer being wider than output layer. This wider layer allows more values to be feedback to input, thus allowing more information to be available to the network. The hidden layer has the hyperbolic tangent sigmoid activation function, and output layer has purelin activation function.

For the considered wind speed prediction model, the inputs are temperature (T_w), wind direction (D_w), and wind speed (N_w). As a result, three input neurons were built in the output layer. The wind speed to be predicted forms the single output neuron in output layer. The proposed approach aims to fix the number of neurons so as to achieve better accuracy and faster convergence. From Figure 1, the input and the output target vector pairs are as foloows.

$(X_1, X_2, X_3 : Y)$ = (temperature, wind direction, and wind speed : predicted Wind speed).

$(X_1, X_2, X_3 : Y) = (T_w, D_W, N_w : N_{wp})$, where N_{wp} is the predicted wind speed.

Let W_c be the weight between context layer and input layer.

Let W_1 be the weight between input and hidden layer.

Let W_2 be the weight between hidden and recurrent link layer.

The neurons are connected one to one between hidden and recurrent link layer:

$f(\cdot)$ is purelin activation function,

$h(\cdot)$ is hyperbolic sigmoid activation function.

From Figure 1, it can be observed that the layers make independent computation on the data that they receive and pass the results to another layer and finally determine the output of the network. The input $U(K-1)$ is transmitted through the hidden layer that multiplies W_1 by hyperbolic sigmoid function. The network learns the function based on current input $W_1U(K-1)$ plus record of previous state output $W_cX_c(K)$. Further, the value $X(K)$ is transmitted through second connection multiplied with W_2 by purelin function. As a result of training the network, past information reflects to the Elman network. The number of hidden neuron is fixed based on new criteria. The proposed model is used for estimation and prediction. The key of the proposed method is to select the number of neurons in hidden layer. The proposed architecture for fixing number of hidden neurons in Elman Network is shown in Figure 1:

$$\begin{aligned} &\text{Input vector,} \quad X = \left[T_w, D_w, N_w\right] \\ &\text{Output vector,} \quad Y = \left[N_{wp}\right]. \end{aligned} \tag{2}$$

Table 1: Input parameters that applied to the proposed model.

S. no.	Input parameters	Units	Range of the parameters
1	Temperature	Degree. Celsius	24–36
2	Wind direction	Degree	1–350
3	Wind speed	m/s	1–16

Weight vector of input to hidden vector, $W_1 = [W_{11}, W_{12}, \ldots W_{1n}, W_{21}, W_{22}, \ldots W_{2n}, W_{31}, W_{32}, W_{3n}]$.

Weight vector of hidden to recurrent link vector, $W_2 = [W_{21}, W_{22}, \ldots W_{2n}]$.

Weight vector of recurrent link layer to input vector, $W_c = [W_{c11}, W_{c12}, \ldots W_{c1n}, W_{c21}, W_{c22}, \ldots W_{c2n}, W_{c31}, W_{c32}, \ldots W_{c3n}]$.

$$\text{Output,} \quad Y(K) = f\left(W_2 X(K)\right)$$

$$\text{Input,} \quad X(K) = h\left(W_c X_c(K) + W_1 U(K-1)\right) \tag{3}$$

$$\text{Input of recurrent link layer,} \quad X_c(K) = X(K-1).$$

4.2. Proposed Methodology. Generally, neural network involves the process of training, testing, and developing a model at end stage in wind farms. The perfect design of NN model is important for challenging other not so accurate models. The data required for inputs are wind speed, wind direction, and temperature. The higher valued collected data tend to suppress the influence of smaller variable during training. To overcome this problem, the min-max normalization technique which enhances the accuracy of ANN model is used. Therefore, data are scaled within the range [0 1]. The scaling is carried out to improve accuracy of subsequent numeric computation. The selection criteria to fix hidden neuron are important in prediction of wind speed. The perfect design of NN model based on the selection criteria is substantiated using convergence theorem. The training can be learned from previous data after normalization. The performance of trained network is evaluated by two ways: First the actual and predicted wind speeds comparison and second computation of statistical errors of the network. Finally wind speed is predicted which is the output of the proposed NN model.

4.2.1. Data Collection. The real-time data was collected from Suzlon Energy Ltd., India Wind Farm for a period from April 2011 to December 2012. The inputs are temperature, wind vane direction from true north, and wind speed in anemometer. The height of wind farm tower is 65 m. The predicted wind speed is considered as an output of the model. The number of samples taken to develop a proposed model is 10000.

The parameters considered as input to the NN model are shown in Table 1. The sample inputs are collected from wind farm are as shown in Table 2.

4.2.2. Data Normalization. The normalization of data is essential as the variables of different units. The data are scaled within the range of 0 to 1. The scaling is carried out to improve accuracy of subsequent numeric computation and obtain better output. The min-max technique is used. The advantage is preserving exactly all relationships in the data, and it does not introduce bias. The normalization of data is obtained by the following transformation (4).

Table 2: Collected sample inputs from wind farm.

Temp Degree Celsius	Wind vane direction from true north (degree)	Wind speed (m/sec)
26.4	285.5	8.9
26.4	286.9	7.6
25.9	285.5	8.6
25.9	284.1	8.9
31.9	302.7	3
25.9	285.5	8.1
25.8	282.7	7.9
33.8	307.4	6.7
25.8	281.2	7.9
25.9	282.7	7.9
25.9	282.7	8.4
25.8	282.7	7.9

Table 3: Designed parameters of Elman network.

Elman network
Output neuron = 1 (N_{wp})
No. of hidden layers = 1
Input neurons = 3 (T_w, D_w, N_w)
No. of epochs = 2000
Threshold = 1

Normalized input,

$$X_i' = \left(\frac{X_i - X_{\min}}{X_{\max} - X_{\min}}\right)\left(X'_{\max} - X'_{\min}\right) + X'_{\min}, \tag{4}$$

where X_i, $X_{\min}$, $X_{\max}$ are the actual input data, minimum and maximum input data. $X'_{\min}$, $X'_{\max}$ be the minimum and maximum target value.

4.2.3. Designing the Network. Set-up parameter includes epochs and dimensions. The training can be learned from the past data after normalization. The dimensions like number of input, hidden, and output neuron are to be designed. The three input parameters are temperature, wind direction and wind speed. The number of hidden layer is one. The number of hidden neurons is to be fixed based on proposed criteria. The input is transmitted through the hidden layer that multiplies weight by hyperbolic sigmoid function. The network learns function based on current input plus record of previous state. Further, this output is transmitted through second connection multiplied with weight by purelin function. As a result of training the network, past information is reflected to Elman network. The stopping criteria are reached

Table 4: Statistical analysis of various criteria for fixing number of hidden neurons in Elman network.

Considered criteria for fixing number of hidden neurons	no. of hidden neurons	MSE	MRE	MAE
$4n/(n-1)$	6	0.1329	0.0158	0.1279
$5n^2+1/n^2-8$	46	0.0473	0.0106	0.0858
$4n/n-2$	12	0.0783	0.0122	0.099
$8n+7/n-2$	31	0.0399	0.0117	0.0944
$5(n^2+1)+2/n^2-8$	52	0.0661	0.0106	0.0862
$4n^2+7/n^2-8$	43	0.0526	0.0113	0.0917
$7n^2+13/n^2-8$	76	0.0093	0.0084	0.0684
$8n+2/n-2$	26	0.083	0.0135	0.1097
$7(n^2+2)/n^2-8$	77	0.0351	0.008	0.065
$3n+5/n-2$	14	0.0361	0.0116	0.094
$9n/(n-2)$	27	0.0388	0.086	0.0701
$5(n^2+3)+2/n^2-8$	62	0.0282	0.0055	0.0444
$9n+6/n-2$	33	0.0446	0.0096	0.0778
$5(n^2+1)+3/n^2-8$	53	0.0447	0.0097	0.0782
$n/(n+1)$	1	0.1812	0.0239	0.1937
$5(n^2+6)/n^2-8$	75	0.017	0.0057	0.0462
$8n+1/n-2$	25	0.0299	0.0105	0.0853
$(10n+1)/n$	10	0.1549	0.0153	0.1238
$3n^2+7/n^2-8$	34	0.0414	0.0105	0.0854
$6(n^2+4)+3/n^2-8$	81	0.0285	0.0065	0.0526
$7(n^2+5)+3/n^2-8$	101	0.0697	0.0069	0.0561
$5n^2/n^2-8$	45	0.0541	0.01	0.0813
$9n+1/n-2$	28	0.038	0.0112	0.0904
$8n+8/n-2$	32	0.028	0.0105	0.0848
$8n/(n-2)$	24	0.0268	0.009	0.0727
$5(n^2+1)+4/n^2-8$	54	0.0155	0.0086	0.0695
$4n^2+8/n^2-8$	44	0.0599	0.0112	0.0907
$5(n^2+3)+3/n^2-8$	63	0.0324	0.0094	0.0758
$5(n^2+6)-1/n^2-8$	74	0.0208	0.0109	0.0881
$4n+1/n-2$	13	0.1688	0.0167	0.1349
$5(n^2+4)-1/n^2-8$	64	0.0359	0.0133	0.1079
$7(n^2+5)+2/n^2-8$	100	0.0194	0.0058	0.0472
$5(n^2+2)/n^2-8$	55	0.0618	0.0166	0.1342
$2n/(n+1)$	2	0.217	0.028	0.2272
$8n+5/n-2$	29	0.0457	0.008	0.0651
$(11n+1)/n$	11	0.0547	0.0118	0.0959
$5(n^2+4)/n^2-8$	65	0.0232	0.0098	0.0798
$5(n^2+4)+1/n^2-8$	66	0.0155	0.006	0.0487
$6(n^2+4)/n^2-8$	78	0.0171	0.0086	0.0697
$3n^2+8/n^2-8$	35	0.1005	0.0138	0.116
$5n/(n-2)$	15	0.0838	0.0109	0.088
$5(n^2+4)+1/n^2-8$	82	0.0293	0.0061	0.0497
$7(n^2+5)-2/n^2-8$	96	0.0295	0.0106	0.0859
$6(n^2+4)+2/n^2-8$	79	0.072	0.009	0.0731
$3n+7/n-2$	16	0.0588	0.0115	0.0934
$8n+6/n-2$	30	0.0831	0.0115	0.0935
$5(n^2+4)+2/n^2-8$	67	0.037	0.0079	0.0636
$5n^2+2/n^2-8$	47	0.0527	0.0097	0.0784
$5(n^2+2)+1/n^2-8$	56	0.0157	0.0093	0.0755
$8n^2/n^2-7$	36	0.0188	0.0117	0.0948
$5(n^2+8)+1/n^2-8$	86	0.0236	0.0132	0.1066
$2n/(n-1)$	3	0.1426	0.0218	0.1763
$5(n^2+4)-3/n^2-8$	87	0.0118	0.0064	0.0519
$7(n^2+5)-1/n^2-8$	97	0.0319	0.01	0.0813
$3n/(n-1)$	5	0.1339	0.0208	0.1685
$5(n^2+2)+2/n^2-8$	57	0.021	0.0078	0.0629
$5(n^2+4)+3/n^2-8$	68	0.0249	0.0081	0.0654
$3n+8/n-2$	17	0.039	0.0114	0.0925
$5(n^2+7)/n^2-8$	80	0.0558	0.0098	0.0797
$5(n^2+2)+3/n^2-8$	58	0.0283	0.0079	0.064
$5(n^2+5)-1/n^2-8$	69	0.1152	0.0127	0.1028
$6n/(n-2)$	18	0.0437	0.0094	0.0759
$6(n^2+7)+2/n^2-8$	98	0.0283	0.0079	0.064
$4n^2+6/n^2-8$	42	0.0136	0.0099	0.0806
$5(n^2+3)+2/n^2-8$	61	0.0249	0.0101	0.082
$5(n^2+5)+1/n^2-8$	71	0.0162	0.0077	0.0623
$7n+2/(n-2)$	23	0.1142	0.0156	0.1264
$5n/(n-1)$	8	0.0894	0.0142	0.1149
$5(n^2+9)-1/n^2-8$	89	0.0202	0.01	0.0808
$4n^2+4/n^2-8$	40	0.0163	0.0102	0.083
$5(n^2+1)/n^2-8$	50	0.0184	0.0092	0.0744
$5(n^2+9)/n^2-8$	90	0.0188	0.0049	0.0395
$4n^2+5/n^2-8$	41	0.0329	0.0118	0.0957
$7(n^2+4)/n^2-8$	91	0.0405	0.0071	0.0573
$7(n^2+4)+3/n^2-8$	94	0.0258	0.0127	0.1028
$5n+7/n-2$	22	0.049	0.0088	0.0713
$3(n+1)/n$	4	0.0723	0.0118	0.0954
$5(n^2+1)+1/n^2-8$	51	0.0278	0.009	0.0728
$5(n^2+3)/n^2-8$	60	0.035	0.0106	0.0861
$4n^2+1/n^2-8$	37	0.0198	0.0078	0.0633
$5(n^2+9)-2/n^2-8$	88	0.0336	0.008	0.0648
$5n+6/n-2$	21	0.1008	0.0134	0.1089
$4n^2+3/n^2-8$	**39**	**0.018**	**0.0049**	**0.04**
$5n^2+4/n^2-8$	49	0.0636	0.0095	0.077
$7(n^2+4)+1/n^2-8$	92	0.0332	0.011	0.089
$(9n+1)/n$	9	0.1091	0.0123	0.1
$6(n^2+5)/n^2-8$	84	0.0169	0.009	0.0727
$7(n^2+5)-3/n^2-8$	95	0.0103	0.0064	0.0517
$5(n^2+8)/n^2-8$	85	0.0293	0.0104	0.0842
$5(n^2+5)+2/n^2-8$	72	0.0209	0.0091	0.0739
$4n^2+2/n^2-8$	38	0.0735	0.0101	0.0815

Table 4: Continued.

Considered criteria for fixing number of hidden neurons	no. of hidden neurons	MSE	MRE	MAE
$5n + 4/n - 2$	19	0.0771	0.0125	0.101
$5(n^2 + 2) + 4/n^2 - 8$	59	0.0422	0.0101	0.0815
$5(n^2 + 5)/n^2 - 8$	70	0.0366	0.0083	0.067
$4.5n/(n - 1)$	7	0.0727	0.0161	0.1301
$5n^2 + 3/n^2 - 8$	48	0.0183	0.0118	0.0958
$6(n^2 + 5) - 1/n^2 - 8$	83	0.0407	0.0092	0.0747
$5(n^2 + 6) - 2/n^2 - 8$	73	0.0493	0.0089	0.0721
$5n + 5/n - 2$	20	0.142	0.0157	0.1271
$7(n^2 + 4) + 2/n^2 - 8$	93	0.0133	0.0093	0.0753
$7(n^2 + 5) + 1/n^2 - 8$	99	0.0377	0.012	0.0971

until the minimum error. The parameters used for design of Elman network are shown in Table 3.

4.2.4. Selection of Proposed Criteria. For the proposed model, 101 various criteria were examined to estimate training process and errors in Elman network. The input neuron is taken into account for all criteria. It is tested on convergence theorem. Convergence is changing infinite into finite sequence. All chosen criteria are satisfied with the convergence theorem. Initially, apply the chosen criteria to the Elman network for the development of proposed model. Then, train the neural network and compute statistical errors. The result with the minimum estimated error is determined as the fixation of hidden neuron. The statistical errors are formulated in (1).

4.2.5. Training and Evaluation Performance of Network. The collected data is divided into training and testing of network. The training data was used to develop models of wind speed prediction, while testing data was used to validate performance of models from training data. 7000 data was used for training, and 3000 data was used for testing data. The training can be learned from past data after normalization. The testing data was used to evaluate the performance of network. MSE, MRE, and MAE are used as the criteria for measuring performance. Based on these criteria, the results show that proposed model can give better performance. The selection of proposed criteria is based on the lowest errors. These proposed criteria are applied to Elman network for wind speed prediction. The network checks whether performance is accepted, otherwise goes to next criteria, then train and test performance of network. The errors, value is calculated for each criterion. To perform analysis of NN model, 101 cases with various hidden neuron are examined to estimate learning and generalization errors. The result with minimal error is determined as the best for selection of neurons in hidden layer.

4.3. Proof for the Chosen Proposed Strategy. Based on the discussion on convergence theorem in the Appendix, the proof for the selection criteria is established henceforth.

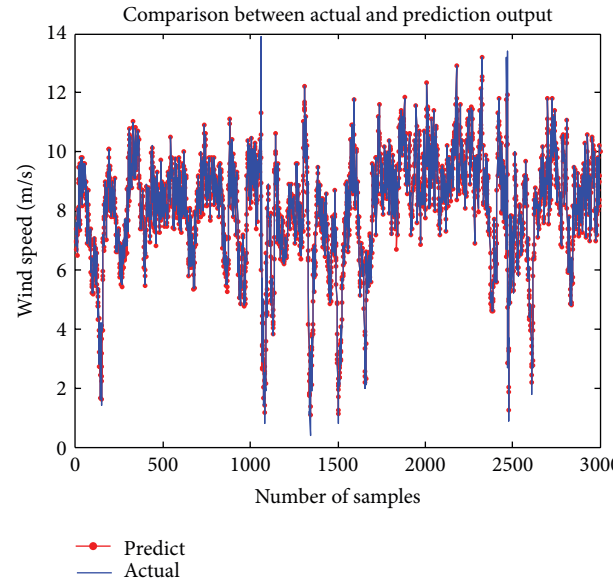

Figure 2: Actual/Predicted output waveform obtained from proposed model.

Lemma 1 is an estimate of sequence which proves the convergence of the proposed criteria.

Lemma 1. *Suppose a sequence* $a_n = (4n^2 + 3)/(n^2 - 8)$ *is converged and* $a_n \geq 0$.

It has limit l. *If there exists constant* $\varepsilon > 0$ *such that* $|a_n - l| < \varepsilon$, *then* $\lim_{n\to\infty} a_n = l$.

Proof. The proof based on Lemma 1.

According to convergence theorem, parameter converges to finite value:

$$a_n = \frac{4n^2 + 3}{n^2 - 8},$$

$$\lim_{n\to\infty} \frac{4n^2 + 3}{n^2 - 8} = \lim_{n\to\infty} \frac{4n^2\left(1 + 3/4n^2\right)}{n^2\left(1 - 8/n^2\right)} = 4, \quad \text{finite value.} \tag{5}$$

Here 4 is limit of sequence as $n \to \infty$. □

If sequence has limit, then it is a convergent sequence. n is the number of input parameters.

The considered 101 various criteria for fixing the number of hidden neuron with statistical errors are established in Table 4. The selected criteria for NN model is $(4n^2 + 3)/(n^2 - 8)$ it has been observed that the error values are less compared to other criteria. So this proposed criterion is very effective for wind speed prediction in renewable energy systems.

The actual and predicted wind speeds observed based on proposed model is shown in Figure 2. The advantages of

Table 5: Performance analysis of various approaches in existing and proposed models.

S. no.	Various methods	Year	Number of hidden neurons	MSE
1	Li et al. method [7]	1995	$N_h = \left(\sqrt{1+8n}-1\right)/2$	0.0399
2	Tamura and Tateishi method [9]	1997	$N_h = N-1$	0.217
3	Fujita method [10]	1998	$N_h = K\log\left\|P_c Z\right\|/\log S$	0.0723
4	Zhang et al. method [14]	2003	$N_h = 2^n/n+1$	0.217
5	Jinchuan and Xinzhe method [3]	2008	$N_h = \left(N_{\text{in}} + \sqrt{N_p}\right)/L$	0.0299
6	Xu and Chen method [19]	2008	$N_h = C_f\left(N/d\log N\right)^{0.5}$	0.0727
7	Shibata and Ikeda method [20]	2009	$N_h = \sqrt{N_i N_0}$	0.1812
8	Hunter et al. method [2]	2012	$N_h = 2^n-1$	0.0727
9	Proposed approach		$N_h = \left(4n^2+3\right)/\left(n^2-8\right)$	**0.018**

the proposed approach are minimal error, effective, and easy implementation for wind speed prediction. This proposed algorithm was simulated and obtained a minimal MSE of 0.018, MRE of 0.0049, and MAE of 0.04.

5. Discussion and Results

Several researchers proposed many approaches to fix the number of hidden neurons in neural network. The approaches can be classified into constructive and pruning approaches. The constructive approach starts with undersized network and then adds additional hidden neuron [7, 39]. The pruning approach starts with oversized network and then prunes the less relevant neuron and weights to find the smallest size. The problems of proper number of hidden neurons for a particular problem are to be fixed. The existing method to determine number of hidden neurons is trial-and-error rule. This starts with undersized number of hidden neurons and adds neurons to N_h. The disadvantage is that it is time consuming and there is no guarantee of fixing the hidden neuron. The selected criteria for NN model is $(4n^2+3)/(n^2-8)$ which used 39 numbers of hidden neurons and obtained a minimal MSE value of 0.018 in comparison with other criteria.

The salient points of the proposed approach are discussed here. The result with minimum error is determined as best solution for fixing hidden neurons. Simulation results are showing that predicted wind speed is in good agreement with the experimental measured values. Initially real-time data are divided into training and testing set. The training set performs in neural network learning, and testing set performs to estimate the error. The testing performance stops improving as the N_h continues to increase; training has begun to fit the noise in the training data, and overfitting occurs. From the results, it is observed that the proposed methodology gives better results than the other approaches. In this paper, proposed criteria are considered for designing a three-layer neural networks. The proposed models were run on a Lenova laptop computer with Pentium III processor running at 1.3 GHz with 240 MB of RAM. The statistical errors are calculated to evaluate the performance of network. It is known that certain approaches produce large size network that is unnecessary whereas others are expensive. The analysis of wind speed prediction is carried out by the proposed new criteria. Table 5 shows that the proposed model gives better value for statistical errors in comparison with other existing models.

6. Conclusion

In this paper, a survey has been made on the design of neural networks for fixing the number of hidden neurons. The proposed model was introduced and tested with real-time wind data. The results are compared with various statistical errors. The proposed approach aimed at implementing the selection of proper number of hidden neurons in Elman network for wind speed prediction in renewable energy systems. The better performance is also analyzed using statistical errors. The following conclusions were obtained.

(1) Reviewing the methods to fix hidden neurons in neural networks for the past 20 years.

(2) Selecting number of hidden neurons thus providing better framework for designing proposed Elman network.

(3) Reduction of errors made by Elman network.

(4) Predicting accurate wind speed in renewable energy systems.

(5) Improving stability and accuracy of network.

Appendix

Consider various criteria n as number of input parameters. All criteria are satisfied with convergence theorem. Some illustrations are shown below. The convergence sequence have a finite limit. Another sequence is called divergent sequence [40]. The characteristics of convergence theorem are as follows.

(i) A convergent sequence has a limit.

(ii) Every convergent sequence is bounded.

(iii) Every bounded point has a limit point.

(iv) A necessary condition for the convergence sequence is that it is bounded and has limit.

(v) An oscillatory sequence is not convergent, that is, divergent sequence.

A network is stable meaning no there is change occurring in the state of the network regardless of the operation. An important property of the NN model is that it always converges to a stable state. The convergence is important in optimization problem in real time since it prevents a network from the risk of getting stuck at some local minima. Due to the presence of discontinuities in model, the convergence of sequence infinite has been established in convergence theorem. The properties of convergence are used in the design of realtime neural optimization solvers.

Discuss convergence of the following sequence.

$$\text{Consider the sequence} \quad a_n = \frac{4n}{n-1}. \tag{A.1}$$

Apply convergence theorem

$$\lim_{n\to\infty} \frac{4n}{n-1} = \lim_{n\to\infty} \frac{4n}{n(1-1/n)} = 4, \quad \text{finite value.} \tag{A.2}$$

Therefore, the terms of sequence are bounded, and the sequence has a limit value. Hence the sequence is convergent.

$$\text{Consider the sequence,} \quad a_n = \frac{2n}{n-1}. \tag{A.3}$$

Apply convergence theorem,

$$\lim_{n\to\infty} \frac{2n}{n-1} = \lim_{n\to\infty} \frac{2n}{n(1-1/n)} = 2, \quad \text{finite value.} \tag{A.4}$$

Therefore, the terms of sequence are bounded and the sequence has a limit value. Hence the sequence is convergent.

References

[1] S. N. Sivanandam, S. Sumathi, and S. N. Deepa, *Introduction to Neural Networks Using Matlab 6.0*, Tata McGraw Hill, 1st edition, 2008.

[2] D. Hunter, H. Yu, M. S. Pukish III, J. Kolbusz, and B. M. Wilamowski, "Selection of proper neural network sizes and architectures: a comparative study," *IEEE Transactions on Industrial Informatics*, vol. 8, no. 2, pp. 228–240, 2012.

[3] K. Jinchuan and L. Xinzhe, "Empirical analysis of optimal hidden neurons in neural network modeling for stock prediction," in *Proceedings of the Pacific-Asia Workshop on Computational Intelligence and Industrial Application*, vol. 2, pp. 828–832, December 2008.

[4] B. Curry and P. H. Morgan, "Model selection in neural networks: some difficulties," *European Journal of Operational Research*, vol. 170, no. 2, pp. 567–577, 2006.

[5] M. A. Sartori and P. J. Antsaklis, "A simple method to derive bounds on the size and to train multilayer neural networks," *IEEE Transactions on Neural Networks*, vol. 2, no. 4, pp. 467–471, 1991.

[6] M. Arai, "Bounds on the number of hidden units in binary-valued three-layer neural networks," *Neural Networks*, vol. 6, no. 6, pp. 855–860, 1993.

[7] J. Y. Li, T. W. S. Chow, and Y. L. Yu, "Estimation theory and optimization algorithm for the number of hidden units in the higher-order feedforward neural network," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 3, pp. 1229–1233, December 1995.

[8] M. Hagiwara, "A simple and effective method for removal of hidden units and weights," *Neurocomputing*, vol. 6, no. 2, pp. 207–218, 1994.

[9] S. Tamura and M. Tateishi, "Capabilities of a four-layered feedforward neural network: four layers versus three," *IEEE Transactions on Neural Networks*, vol. 8, no. 2, pp. 251–255, 1997.

[10] O. Fujita, "Statistical estimation of the number of hidden units for feedforward neural networks," *Neural Networks*, vol. 11, no. 5, pp. 851–859, 1998.

[11] K. Keeni, K. Nakayama, and H. Shimodaira, "Estimation of initial weights and hidden units for fast learning of multi-layer neural networks for pattern classification," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '99)*, vol. 3, pp. 1652–1656, IEEE, July 1999.

[12] T. Onoda, "Neural network information criterion for the optimal number of hidden units," in *Proceedings of the 1995 IEEE International Conference on Neural Networks*, vol. 1, pp. 275–280, December 1995.

[13] M. M. Islam and K. Murase, "A new algorithm to design compact two-hidden-layer artificial neural networks," *Neural Networks*, vol. 14, no. 9, pp. 1265–1278, 2001.

[14] Z. Zhang, X. Ma, and Y. Yang, "Bounds on the number of hidden neurons in three-layer binary neural networks," *Neural Networks*, vol. 16, no. 7, pp. 995–1002, 2003.

[15] G. B. Huang, "Learning capability and storage capacity of two-hidden-layer feedforward networks," *IEEE Transactions on Neural Networks*, vol. 14, no. 2, pp. 274–281, 2003.

[16] B. Choi, J. H. Lee, and D. H. Kim, "Solving local minima problem with large number of hidden nodes on two-layered feed-forward artificial neural networks," *Neurocomputing*, vol. 71, no. 16–18, pp. 3640–3643, 2008.

[17] N. Jiang, Z. Zhang, X. Ma, and J. Wang, "The lower bound on the number of hidden neurons in multi-valued multi-threshold neural networks," in *Proceedings of the 2nd International Symposium on Intelligent Information Technology Application (IITA '08)*, pp. 103–107, December 2008.

[18] S. Trenn, "Multilayer perceptrons: approximation order and necessary number of hidden units," *IEEE Transactions on Neural Networks*, vol. 19, no. 5, pp. 836–844, 2008.

[19] S. Xu and L. Chen, "A novel approach for determining the optimal number of hidden layer neurons for FNN's and its application in data mining," in *Proceedings of the 5th International Conference on Information Technology and Applications (ICITA '08)*, pp. 683–686, June 2008.

[20] K. Shibata and Y. Ikeda, "Effect of number of hidden neurons on learning in large-scale layered neural networks," in *Proceedings of the ICROS-SICE International Joint Conference 2009 (ICCAS-SICE '09)*, pp. 5008–5013, August 2009.

[21] C. A. Doukim, J. A. Dargham, and A. Chekima, "Finding the number of hidden neurons for an MLP neural network using coarse to fine search technique," in *Proceedings of the 10th International Conference on Information Sciences, Signal Processing and Their Applications (ISSPA '10)*, pp. 606–609, May 2010.

[22] H. C. Yuan, F. L. Xiong, and X. Y. Huai, "A method for estimating the number of hidden neurons in feed-forward neural networks

based on information entropy," *Computers and Electronics in Agriculture*, vol. 40, no. 1–3, pp. 57–64, 2003.

[23] Y. K. Wu and J. S. Hong, "A literature review of wind forecasting technology in the world," in *Proceedings of the IEEE Lausanne Power Tech*, pp. 504–509, July 2007.

[24] G. Panchal, A. Ganatra, Y. P. Kosta, and D. Panchal, "Behaviour analysis of multilayer perceptrons with multiple hidden neurons and hidden layers," *International Journal of Computer Theory and Engineeringvol*, vol. 3, no. 2, pp. 332–337, 2011.

[25] K. Z. Mao and G. B. Huang, "Neuron selection for RBF neural network classifier based on data structure preserving criterion," *IEEE Transactions on Neural Networks*, vol. 16, no. 6, pp. 1531–1540, 2005.

[26] R. Devi, B. S. Rani, and V. Prakash, "Role of hidden neurons in an elman recurrent neural network in classification of cavitation signals," *International Journal of Computer Applications*, vol. 37, no. 7, pp. 9–13, 2012.

[27] H. Beigy and M. R. Meybodi, "Backpropagation algorithm adaptation parameters using learning automata," *International Journal of Neural Systems*, vol. 11, no. 3, pp. 219–228, 2001.

[28] M. Han and J. Yin, "The hidden neurons selection of the wavelet networks using support vector machines and ridge regression," *Neurocomputing*, vol. 72, no. 1–3, pp. 471–479, 2008.

[29] N. Murata, S. Yoshizawa, and S. I. Amari, "Network information criterion-determining the number of hidden units for an artificial neural network model," *IEEE Transactions on Neural Networks*, vol. 5, no. 6, pp. 865–872, 1994.

[30] J. Sun, "Learning algorithm and hidden node selection scheme for local coupled feedforward neural network classifier," *Neurocomputing*, vol. 79, pp. 158–163, 2012.

[31] X. Zeng and D. S. Yeung, "Hidden neuron pruning of multilayer perceptrons using a quantified sensitivity measure," *Neurocomputing*, vol. 69, no. 7–9, pp. 825–837, 2006.

[32] X. Wang and Y. Huang, "Convergence study in extended Kalman filter-based training of recurrent neural networks," *IEEE Transactions on Neural Networks*, vol. 22, no. 4, pp. 588–600, 2011.

[33] V. Kůrková, P. C. Kainen, and V. Kreinovich, "Estimates of the number of hidden units and variation with respect to half-spaces," *Neural Networks*, vol. 10, no. 6, pp. 1061–1068, 1997.

[34] Y. Liu, J. A. Starzyk, and Z. Zhu, "Optimizing number of hidden neurons in neural networks," in *Proceedings of the IASTED International Conference on Artificial Intelligence and Applications (AIA '07)*, pp. 121–126, February 2007.

[35] Y. Lan, Y. C. Soh, and G. B. Huang, "Constructive hidden nodes selection of extreme learning machine for regression," *Neurocomputing*, vol. 73, no. 16–18, pp. 3191–3199, 2010.

[36] J. Li, B. Zhang, C. Mao, G. Xie, Y. Li, and J. Lu, "Wind speed prediction based on the Elman recursion neural networks," in *Proceedings of the International Conference on Modelling, Identification and Control (ICMIC '10)*, pp. 728–732, July 2010.

[37] Q. Cao, B. T. Ewing, and M. A. Thompson, "Forecasting wind speed with recurrent neural networks," *European Journal of Operational Research*, vol. 221, no. 1, pp. 148–154, 2012.

[38] W. M. Lin and C. M. Hong, "A new Elman neural network-based control algorithm for adjustable-pitch variable-speed wind-energy conversion systems," *IEEE Transactions on Power Electronics*, vol. 26, no. 2, pp. 473–481, 2011.

[39] J. Zhang and A. J. Morris, "A sequential learning approach for single hidden layer neural networks," *Neural Networks*, vol. 11, no. 1, pp. 65–80, 1998.

[40] B. S. Grewal, *Higher Engineering Mathematics*, Khanna Publishers, 40th edition, 2007.