Artif Intell Rev (2017) 48:449–471
DOI 10.1007/s10462-016-9506-6

Comparative analysis on hidden neurons estimation in multi layer perceptron neural networks for wind speed forecasting

M. Madhiarasan[1] · **S. N. Deepa**[1]

Published online: 19 August 2016

Abstract In this paper methodologies are proposed to estimate the number of hidden neurons that are to be placed numbers in the hidden layer of artificial neural networks (ANN) and certain new criteria are evolved for fixing this hidden neuron in multilayer perceptron neural networks. On the computation of the number of hidden neurons, the developed neural network model is applied for wind speed forecasting application. There is a possibility of over fitting or under fitting occurrence due to the random selection of hidden neurons in ANN model and this is addressed in this paper. Contribution is done in developing various 151 different criteria and the evolved criteria are tested for their validity employing various statistical error means. Simulation results prove that the proposed methodology minimized the computational error and enhanced the prediction accuracy. Convergence theorem is employed over the developed criterion to validate its applicability for fixing the number of hidden neurons. To evaluate the effectiveness of the proposed approach simulations were carried out on collected real-time wind data. Simulated results confirm that with minimum errors the presented approach can be utilized for wind speed forecasting. Comparative analysis has been performed for the estimation of the number of hidden neurons in multilayer perceptron neural networks. The presented approach is compact, enhances the accuracy rate with reduced error and faster convergence.

Keywords Hidden neurons · Multilayer perceptron networks · Wind speed forecasting · Convergence theorem

1 Introduction

Artificial neural network (ANN) is a nonlinear information processing system, which is developed from interconnected elementary processing devices called neurons. ANN is an

✉ M. Madhiarasan
mmadhiarasan89@gmail.com

S. N. Deepa
deepapsg@gmail.com

[1] Anna University, Regional Campus, Coimbatore, India

information processing paradigm that is inspired by the biological nervous system (Sivanandam et al. 2008). ANN is widely used for various applications because of certain special features such as the good self learning ability, adaptability, real-time operation, fault tolerance ability, easy implementation and cost effectiveness; as a result this network can be classified as feed forward and feedback (recurrent) networks. A network arranged into layer with no feedback path is called a feed forward network. The arranging is from input, through one (or) more hidden layers, to the output layer, example: multilayer perceptron (MLP), back propagation neural network (BPN), radial basis function neural network (RBFN) and so on. A network with feedback path can occur between the layers (or) within the layer forms the feedback network, example: Elman, Hopfield, Boltzmann machine, and so on. The multilayer perceptron networks effectively solve the nonlinear complex task and it is generally applied for pattern recognition and classification. The various problems that occur in the design of neural network model and its training process include the following:

(1) Fixing the number of hidden layers in the network.
(2) Selecting the number of hidden neurons in each hidden layer of the network.
(3) Determining how many training pair to be employed in the network.
(4) Applicability of suitable training algorithm for the network considered.
(5) Selection of architecture to be used in the neural network.
(6) Finding a global optimal solution that prevents local minima problem.
(7) Testing the network to check for over fitting (or) under fitting problem (Panchal et al. 2011; Hunter et al. 2012).

Basically, neural network when designed with very few hidden neurons results in large training errors and generalization errors because of the under fitting problem, while too many hidden neurons used in network results in low training errors and high generalization errors due to over fitting issue (Xu and Chen 2008). Henceforth, the presence of hidden neurons has profound impact on the errors. The computed errors determine the neural network stability. The network with minimum error has the better stability while the network with higher error has instability occurrence (Ke and Liu 2008). Process of selecting the number of hidden layer and number of neurons in each hidden layer is still confusing and challenging task (Karsoliya 2012). Numerous researchers design neural network models and evaluate the amount of hidden neurons in the hidden layers but still it's a major task in the design process. The neural network training accuracy is estimated by the parameters such as neural network architecture, number of hidden layers, amount of hidden neurons in the hidden layer, activation function, inputs and outputs and weight updation phenomenon.

Due to the environmental deteriorate and exhaustion of nonrenewable energy, more attention is focused on wind energy. Since wind is one of the most attractive, flexible and tractable of all energy sources, the mechanical energy derived directly from wind can be readily and efficiently converted into other form of energy. The two fold aims of global minimization of CO_2 emission and enhancing energy security (energy policy objectives in many countries) coincides with the increasing use of wind power for electricity production. Due to changeability and the existence of unpredictability, the wind speed forecasting is a very important topic in wind research area.

If small fractional deviations of wind speed occur, it will lead to a large error output of wind driving systems. The basic need for performing wind speed forecasting is as follows:

(1) On time and high quality operation of power system and wind farm.
(2) Effective integration of wind power to the electricity grid.
(3) Reducing the operating cost of the wind power generation.

(4) For achieving low spinning reserve.
(5) To assist planning and control of power system and wind farm.

The statistical errors are employed to measure the quality of forecasted wind speed that is been computed using the proposed neural network model. During the training process of the network the generalization performance differs overtime. At this juncture, the stability of neural network model gets introduced. For a feed forward neural network, the stability aspect defines its nature to withstand the change in weight values during the training process and sustaining the achieved minimal error value without getting trapped in local and global minima problems.

In this paper different 151 criteria were considered for appropriate selection of hidden neurons in the hidden layer of multilayer perceptron (MLP) networks, suggested all 151 criteria are satisfied based on the convergence theorem. The proposed multilayer perceptron network design is adapted for application of wind speed forecasting. The main aim is to achieve the minimal error, improve the network stability and better accuracy compared to other existing approaches in order to assist planning and control of power system and wind farm.

2 Necessity for neural network design

Numerous research experiments were attempted and the approaches were suggested by the researchers for selecting the appropriate number of hidden neurons in neural network modeling. A comprehensive review has been made in this section with regard to the various methodologies carried out in the early literature for fixing the number of hidden neurons in neural network design and are presented as given below:

Huang and Huang (1991) investigates the issues associated for multilayer perceptron with one hidden layer and pointed out a method to find the sufficient hidden neurons in multilayer perceptron. Arai (1993), pointed out $2^n/3$ as sufficient number of hidden neurons based on two parallel hyper plane methods (TPHM). Hagiwara (1994), presented consuming energy method and weight power method in order to obtain the size minimization. Murata et al. (1994), pointed out a new method to estimate the number of hidden neurons in an artificial network model based on network information criterion (NIC). Li et al. (1995), evolved required number of hidden neurons in higher order feed forward neural network by means of the estimation theory, based on this approach determined that 4 and 7 hidden neurons are sufficient for second order and first order networks respectively. Remark: Second order neural network is superior to first order neural network in terms of training convergence. Limitation: Much training and testing time is required. Onoda (1995), performed construction of optimal architectures of multi layered neural network based on the Neural network information criterion (NNIC) in order to solve the model selection and selection of hidden neurons issues. Tamura and Tateishi (1997), carried out work on an Akaike's information criteria (AIC) based hidden neurons selection, stated that $N - 1$ and $(N/2) + 3$ are the required number of hidden neurons for three-layered and four-layered neural network respectively, where 'N' represents the number of input neurons. Remark: Four layered network outperform than three layered networks. Fujita (1998), estimated sufficient number of hidden neurons for feed forward neural network by means of the statistical estimation and stated that $N_h = K \log(\| P_c^{(0)} Z \| / C) / \log S$ is the required number of hidden neurons, where, N_h—number of hidden neurons, S—total number of candidates that are randomly found hidden neurons, K—data set, C—allowable output error. Keeni et al. (1999), evolved number of

hidden neurons and initial weights automatically for fast learning multilayer neural networks and suggested method adopted for cancer cell prediction. Limitation: Poor generalization ability.

Huang (2003), pointed out a model for learning capability and storage capacity of two-hidden layer feed forward neural networks. Required number of hidden neurons in single and two hidden layers networks are determined as $N_h = \sqrt{(m+2)N} + 2\sqrt{N/(m+2)}$ and $N_h = m\sqrt{N/(m+2)}$ respectively, where 'm' specifies the number of layers. Yuan et al. (2003), determined feed forward neural network required number of hidden neurons by means of the information entropy. Zhang et al. (2003), developed unit sphere covering (USC) of the hamming space (HS) based set covering algorithm for three layer binary neural networks and found that $3L/2$ hidden neurons are the sufficient hidden neurons, where, L is the number of unit sphere contained in N-dimensional Hamming space. Mao and Huang (2005), performed a selection of hidden neurons for RBF neural network classifier by means of the data structure preserving criterion. Teoh et al. (2006), carried out work on estimation of hidden neurons in a feed forward neural network by means of the singular value decomposition. Zeng and Yeung (2006), presented quantified sensitivity measures based pruning technique, to remove as many neurons to obtain the lowest relevance hidden neuron for MLP.

Choi et al. (2008), pointed out a separate learning algorithm for two layered feed forward neural network in order to solve the local minima problem with large number of hidden neurons. Ke and Liu (2008), performed neural network based stock price prediction and analyzed the proper number of hidden layers and hidden neurons in neural network design. Pointed out formula $N_h = (N_{in} + \sqrt{N_p})/L$ is evaluated on forty cases, where, N_{in} is the number of input neurons, N_p is the number of input sample, L is the number of hidden layer. The best number of hidden neurons is selected based on the least generalization error. Han and Yin (2008), suggested support vector machine and ridge regression based hidden neurons selection for wavelet network. Jiang et al. (2008), carried out the implementation of an arbitrary function by using the lower bound on the number of hidden units in three-layer multi-valued multi threshold neural networks. The presented approach defined number of hidden neurons as $N_h = q^n$ where, q is valued upper bound function. Xu and Chen (2008), pointed out feed forward neural network optimal number of hidden neurons based on a novel approach, which are applied for data mining. The suggested number of hidden neurons is $N_h = C_f(N/(d \log N))^{1/2}$ where, C_f—first absolute moment of the Fourier magnitude distribution of the target function, d—input dimension of target function, N—number of training pair. Remark: Suggested number of hidden neurons N_h leads to the least root mean square error. Limitation: Local minima problem is not addressed in this work. Trenn (2008), evolved multilayer perceptron neural network required number of hidden neurons as $N_h = (n + n_o - 1)/2$ where, n_o—number of inputs. Stated formula depends on input variables and desired approximation order.

Shibata and Ikeda (2009), analyzed the impact of hidden neurons and learning stability in neural networks. The number of hidden neuron and learning rate are stated as $N_h = \sqrt{N^{(i)}N^{(0)}}$ and $\eta = 32/(\sqrt{N^{(i)}N^{(o)}})$ respectively, where, $N^{(i)}$—number of input neurons, $N^{(o)}$—number of output neurons. Remark: Increasing number of hidden neurons leads to small output connection weights. Doukin et al. (2010), presented a coarse to fine search method in order to find the number of hidden neurons in a multi layer perceptron neural network for skin detection application. Li et al. (2010), pointed out Elman neural network based one step ahead wind speed prediction. The result proves that the suggested method is suitable for various numbers of hidden neurons and various tested input data. Hunter et al. (2012), analyzed appropriate selection of neural network architecture and size.

The generalization problem, various network topologies efficiency and learning algorithm are analyzed in order to choose the suitable neural network architecture and size. Remark: Trial and error approach is avoided. Sun (2012), suggested hidden nodes determination strategies and learning algorithm for newly developed local coupled feed forward neural network classifier. Ramadevi et al. (2012), analyzed the hidden neurons role in the Elman recurrent neural network in classification of cavitation signals and by means of the trial and error approach the optimal number of hidden neurons are fixed. Karsoliya (2012), investigate BPN architecture modeling issue of approximation of required number of hidden layers and the number of neurons and also resolved issue of determination of a sufficient number of hidden neurons and number of hidden layers.

Gnana Sheela and Deepa (2013), investigate the methods to fix the number of hidden neurons in neural network and suggest the $(4n^2 + 3)/(n^2 - 8)$ criteria among the various 101 criteria in order to decide the number of hidden neurons in the Elman neural network for wind speed prediction. Qian and Yong (2013), suggested rural per capita living consumption prediction by means of the back propagation neural network. The number of hidden neurons is evolved as $N_h = \sqrt{n + m + a}$ in order to get better accuracy where, n—number of nodes in the input layer, m—number of nodes in output layer, a—integer among 0–10. Vora and Yagnik (2014), presented a new algorithm having structural changes in feed forward neural network in order to solve the local minima problem with issue of larger hidden neurons. Madhiarasan and Deepa (2016), carried out work on improved back propagation network for wind speed forecasting and novel criterion is proposed for hidden neurons selection.

Based on the literature review carried out for the work related to fixing the number of hidden neurons, the following points are observed:

- Many researches implemented different methodologies for searching the proper hidden neurons in network design in order to achieve the fastest convergence, better efficiency, minimal errors and improved accuracy.
- In the neural network construction process the estimation of hidden neurons plays a vital role and it helps to get better performance with the least error.

In design of neural network model, the fixation of number of layers also plays a vital role. Generally, a 3-layer neural network model is adopted—1 input layer, 1 hidden layer and 1 output layer. Any neural network will comprise an input layer and output layer, hidden layer is introduced in order to achieve faster convergence of the network. More number of hidden layers will increase and cost more on the computational complexity of the network. Thus, it is always feasible to introduce one hidden layer in the neural network design which enhances the convergence of the network.

As well, hidden neurons selection and its placement in the hidden layer is a generic problem in artificial neural network design and in this paper also it is addressed in a general manner by proposing suitable criterion based on the number of input neurons. The application chosen for applying the proposed criterion for fixing the number of hidden neurons is wind speed forecasting. The wind speed forecasting application is chosen in this paper because of the growth of the energy sector and the need to meet the existing power demand.

3 Problem description

In this paper, appropriate estimation of number of hidden neurons for multilayer perceptron networks is carried out. A neural network with few hidden neuron units does not possess sufficient power to satisfy the requirements such as accuracy, error precision and capacity.

The problem of over fitting data is caused by over training. Over training issue is noted to occur in the neural network design. Therefore, the estimation of hidden neuron units in neural network design is one of the most important problems. The determination of optimal hidden neuron numbers and its parameters are crucial and it requires determining the divergence between artificial neural network and an actual system. So many researchers' emphasize to achieve better performance by tackling this problem. But in neural network there is no methodology to fix the hidden neuron numbers without attempting and evaluating during the training process and calculating the generalization error. If the hidden neuron number is large the hidden output connection weights become so small and also the trade off in stability between input and hidden output connection exists. The output neurons become unstable for large hidden neuron numbers. If the hidden neuron number is small it may fall into local minima because of slow learning of the network and the neuron model becomes unstable. Hence, the fixation of hidden neuron numbers is a major critical problem for neural network design and is considered for determining solution in this research contribution.

3.1 Wind speed forecasting

An accurate wind speed forecasting is one of the important issues in renewable energy systems because of diurnal diversity and changing nature of wind. The wind has the uncertain irregularity characteristic. In order to achieve the better generalization capabilities for the wind speed, forecasting the input and output is to be done and the hidden neuron numbers should be appropriately selected for the neural network design. The dynamic system may not achieve a feasible solution due to the insufficient hidden neuron numbers in the neural network. In the current scenario lot of forecasting research fields are heuristic in nature.

3.2 Evaluation metrics employed for validating the proposed approach

The small size networks offer simple structure and better generalization ability, but it may not be easy to learn the issue. While in the large size network learning can be done easily, but it is slow and achieves poor generalization performance due to over fitting (Urolagin et al. 2012). The proposed model confirms that even though more number of hidden neurons is employed in the multilayer perceptron network, it obtains stable performance on training. The ultimate aim is to fix the hidden neuron numbers in the multilayer perceptron network for wind speed forecasting application with better accuracy and minimal statistical error. The below given evaluation metrics are used to select the optimal hidden neuron numbers in neural networks. The proposed approach performance is analyzed based on the mean absolute error (MAE), mean relative error (MRE) and mean square error (MSE) evaluation metrics. Among the proposed different 151 criteria, a suitable number of placements of hidden neurons are selected based on the minimum error performance. The statistical error metrics employed for selecting the number of hidden neurons are given by,

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}(Y_i' - Y_i), \tag{1}$$

$$\mathrm{MRE} = \frac{1}{N}\sum_{i=1}^{N}\left|(Y_i' - Y_i)/\bar{Y}_i\right| \tag{2}$$

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(Y_i' - Y_i)^2 \tag{3}$$

where, N is the number of samples, Y_i' is actual output, $\bar{Y}_i$ is average actual output and Y_i is forecasted output. The neural network designing process plays a vital role in the network performance involving both the training and testing process.

4 Proposed multilayer perceptron networks architecture

In the literature review carried out, it is well noted that different heuristics exist and there is no hard and fast, accurate rule for estimation of number of hidden neurons in the neural network design. Based on the knowledge gained from the existing approach several novel criteria are proposed for estimating the optimal number of hidden neurons in multilayer perceptron networks. A single appropriate topology is utilized for the purpose.

4.1 Multilayer perceptron networks: an overview

The Multilayer perceptron network is a feed forward neural network with supervised learning rule to search for weight values employing linear activation function and it tends to solve complex problems. The multilayer perceptron network comprises an input layer, hidden layer and an output layer. Multilayer perceptron is superior to the single layer perceptron due to its ability of increasing the computational efficiency. These networks learn linear and nonlinear relationships between the input and output vectors because of the presence of hidden layer neurons and their nonlinear transfer function. Most commonly used nonlinear transfer function is hyperbolic tangent sigmoid activation function which is applied over the net input of the hidden layer to obtain the respective output. Back propagation gradient descent learning rule is employed to train the multilayer perceptron networks. These networks are fully connected networks and these induce the faster convergence of the network.

The proposed multilayer perceptron networks designed for wind speed forecasting possess the inputs as Temperature (TD_w), Wind direction (WD_w), Relative Humidity (RH_w) and Wind speed (N_w) and these four inputs acts as four input neurons of the network. The output layer has a single output neuron (i.e.) forecasted wind speed. The objective of the proposed approach is to estimate the appropriate hidden neuron numbers in order to obtain the faster convergence and better accuracy with the least statistical error for the considered application. The architecture of the proposed multilayer perceptron network model is as shown in Fig. 1. Table 1 presents the parameters employed in the proposed neural network model. Each of the layers performs independent computations on received data and the computed results are passed to the next layer and they lastly determines the network output and this information are inferred from Fig. 1. The various computation flow involved during the training process is as given below:

$$\text{Input vector, } X = [TD_w, WD_w, RH_w, N_w] \tag{4}$$

$$\text{Output vector, } Y = \left[N_{fw}\right] \tag{5}$$

Weight vectors of input to hidden vector

$$W = [W_{11}, W_{12}, \ldots, W_{1n}, W_{21}, W_{22}, \ldots, W_{2n}, W_{31}, W_{32}, \ldots, W_{3n}, W_{41}, W_{42}, \ldots, W_{4n}] \tag{6}$$

Compute the Net input of hidden layer and subsequently the output of the hidden layer,

$$Z_j = f\left(\sum_{i=1}^{4}\sum_{j=1}^{n} X_i W_{ij}\right) \tag{7}$$

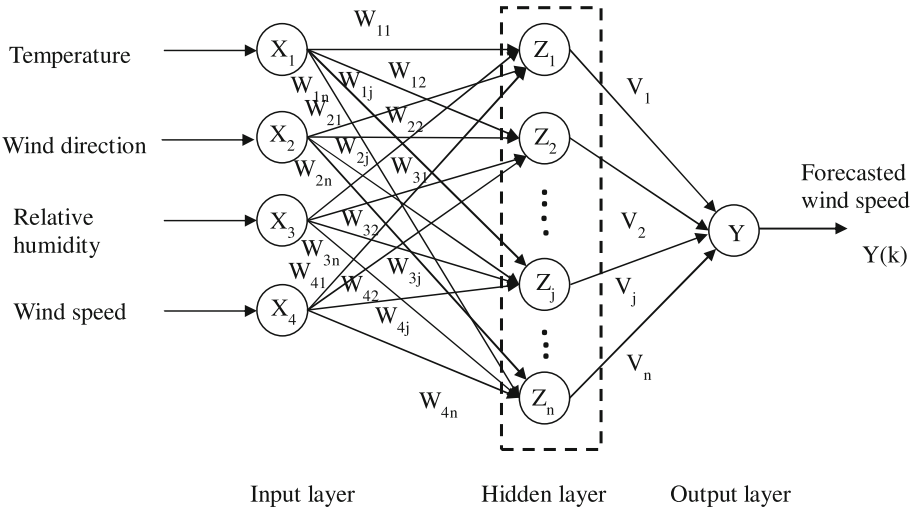

Fig. 1 Proposed multilayer perceptron networks architecture for wind speed forecasting application

Table 1 Input and output variables of proposed neural network model

Input variables	Description	Output variable	Description
X_1	Temperature (TD_w)	Y	Forecasted wind speed (N_{fw})
X_2	Wind direction (WD_w)		
X_3	Relative humidity (RH_w)		
X_4	Wind speed (N_w)		

where, X refers to the input vector, W specifies the weights between input and hidden layer and n indicates the number of hidden neurons.

$$\text{Weight vectors of hidden to output vector, } V = [V_1, V_2, \ldots\ldots, V_n] \tag{8}$$

Compute the net input of output layer and subsequently its output,

$$Y = f\left(\sum_{j=1}^{n}\left(Z_j V_j\right)\right), \quad j = 1, 2, \ldots, n \tag{9}$$

where, V is the weight between hidden and output layer and f is the tangent sigmoidal activation function.

4.2 Numerical experimentation of proposed approach

The proposed approach aims to fix suitable hidden neuron numbers in multilayer perceptron networks for wind speed forecasting application with better accuracy and improved convergence speed. The wind speed forecasting using multilayer perceptron neural network involves the modeling, training and testing process. The considered input data are real-time data and

henceforth, normalization process is carried out to overcome the occurrence of various training problems of large valued input data and thus tend to reduce the effect of smaller value of the input data. In this work, min–max scaling technique is utilized to normalize the variable range of real time data between the range of 0–1. The normalization process helps to improve the numerical computational accuracy; as a result the multilayer perceptron networks model accuracy is also enhanced.

4.2.1 Data collection

The real-time data were collected from the National Oceanic and Atmospheric Administration, United States for a period from January 1948 to December 2013. The temperature (°C), wind direction (°), relative humidity (%) and wind speed (m/s) are inputs to the proposed multilayer perceptron network and network output is forecasted wind speed (m/s). The proposed multilayer perceptron network model is developed by 80,000 numbers of samples. Table 2 shows the input parameters of the developed multilayer perceptron neural network model. Table 3 shows the wind data samples employed in this research contribution.

4.2.2 Data normalization

Normalization (or) scaling is very important for dealing with real time data; the real time data has different range and different units. Hence, the normalization is used to scale the real time data within the range of 0–1. The proposed approach utilizes the min–max normalization technique. The following transformation is employed for normalization of the real time data.

Table 2 Proposed model input parameters

S. no	Input parameter	Units	Range of parameter
1	Temperature	°C	23–30
2	Wind direction	°	0.1–360
3	Relative humidity	%	53–94
4	Wind speed	m/s	1–8

Table 3 Collected real time input data samples

Temperature (°C)	Wind direction (°)	Relative humidity (%)	Wind speed (m/s)
26.9490	193.3204	82.3222	5.7405
25.9666	191.9673	82.425	7.4599
26.9539	357.9629	54.1871	3.4599
23.51	193.5835	91.16	5.9100
28.6538	235.2403	58.1148	2.3899
25.4973	190.8898	85.8516	4.75
28.6087	258.5871	61.9775	2.4100
25.8013	190.0067	85.9016	6.2099
28.5903	292.6652	61.1471	1.8600
24.6519	173.7271	86.9213	5

Table 4 Proposed multilayer perceptron network designed parameters

Multilayer perceptron network
Input neuron = 4 [TD_w, WD_w, RH_w, N_w]
Number of hidden layer = 1
Output neuron = 1 N_{fw}
Number of epochs = 2000
Learning rate = 0.1
Threshold = 1

$$\text{Scaled input, } X'_i = \left(\frac{X_i - X_{\min}}{X_{\max} - X_{\min}}\right)\left(X'_{\max} - X'_{\min}\right) + X'_{\min} \tag{10}$$

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

4.2.3 Proposed neural network design

The design parameters for the proposed multilayer perceptron network are as shown in Table 4. The dimensions such as number of input neurons, the number of output neurons and so on are defined in the network design. The presented network design has four input neurons (temperature, wind direction, relative humidity and wind speed), one hidden layer, one output neuron (forecasted wind speed) and the hidden neuron numbers in the hidden layer is estimated from the proposed new criteria based on the lowest minimal error.

The inputs are transferred to hidden layer that multiplies weight W using hyperbolic tangent sigmoid activation function and the output from the hidden layer is transferred to the output layer that multiplies with weight V using purelin activation function. The training is carried out based on the normalized data. Test for the stopping condition is the error achieving a near negligible value.

4.2.4 Fixation of hidden neuron numbers

In order to estimate and fix the number of hidden neurons in the hidden layer, this paper developed specific new criteria based on the number of input neurons and all the developed novel criteria are verified on the mathematical foundation of convergence theorem. The infinite sequence is changed into finite sequence and this is called convergence. The developed criteria in this paper for estimating and fixing the number of hidden neurons seem to satisfy the convergence theorem. All chosen criteria were examined in multilayer perceptron networks in order to estimate the network training process and statistical errors. The estimation of appropriate number of hidden neurons in the hidden layer is identified and fixed based on the evolved minimal statistical error as can be evaluated from the Eq. (1) to (3).

4.2.5 Training and testing the network performance

A wind speed forecasting model is developed based on the training data while the performance of the proposed model is evaluated by using the testing data. The collected 80,000 real-time data is classified into training and testing. The collected 70 % of data (56,000) are used for training phase and 30 % of the collected data (24,000) are utilized for the testing phase

of the network. Developed new criteria are applied to multilayer perceptron networks and the error values are computed to verify whether the performance is accepted (or) not. The network performance is calculated based on the statistical errors, namely MSE, MAE and MRE. Criterion with the lowest statistical error is the best criterion for estimation of hidden neurons in multilayer perceptron networks.

4.2.6 Denormalization

Denormalization process is a post processing of the data. The real-time output values are computed employing the denormalization process.

5 Experimental results and discussion

Based on the extensive survey made, it is noted that numerous strategies are employed to select the hidden neuron numbers in artificial neural network. The strategies are classified into pruning and constructive strategies. In the pruning strategy the network starts with over sized and then prunes the minimum relevant neurons and weights and it searches for the minimum size, while in the constructive strategy network starts with undersized network and then supplementary hidden neurons are included to the network (Li et al. 1995; Morris and Zhang 1998).

The developed new criteria in this paper are used to build three layer neural networks. Real-time data are initially classified into training and testing set. The training set is used in neural network learning, and testing set is used to calculate the error. Presented multilayer perceptron network design were run on an Acer laptop computer with Pentium (R) Dual Core processor running at 2.30 GHZ with 2 GB of RAM. The analysis of wind speed forecasting is in this paper is based on the presented new criteria.

5.1 Validation for the estimated proposed criterion

The validation for the estimation and fixation of number of hidden neurons is initiated based on the convergence theorem as presented in the "Appendix". Lemma 1 confirms the convergence of the chosen estimation criterion with the least error in this paper.

Lemma 1 *The sequence* $u_n = (4n-2)/(n-3)$ *is converged and* $u_n \geq 0$. *The sequence tends to have a finite limit* l, *if there exists constant* $\varepsilon > 0$ *such that* $|u_n - l| < \varepsilon$, *then* $n \overset{\lim}{\to} \infty \;\; u_n = l$. *The proof for the derived lemma is as given below:*

Regarding to convergence theorem, the selected parameter (or) sequence converges to a finite limit value.

$$u_n = \frac{(4n-2)}{(n-3)}, \tag{11}$$

$$n \overset{\lim}{\to} \infty \;\; \frac{4n-2}{n-3} = n \overset{\lim}{\to} \infty \;\; \frac{n\,(4-2/n)}{n\,(1-3/n)} = 4, \quad \text{finite limit value.} \tag{12}$$

Here 4 is the limit value of the selected sequence as $n \overset{\lim}{\to} \infty$. Hence, the above sequence is convergent sequence because it has the finite limit, where n—the number of input parameters.

Table 5 Statistical analysis of multilayer perceptron networks for developed criteria in fixing the number of hidden neurons

Hidden neuron numbers	Considered criteria for estimating hidden neuron numbers	MSE	MAE	MRE
42	$(5(n+4)+2)/(n-3)$	1.0643e−10	5.2029e−06	1.5160e−06
123	$(6(n^2+4)+3)/(n^2-15)$	1.1627e−08	4.1083e−05	1.1970e−05
75	$5(n^2-1)/(n^2-15)$	2.7480e−09	2.4994e−05	7.2824e−06
108	$(5(n^2+5)+3)/(n^2-15)$	5.6557e−10	7.3519e−06	2.1421e−06
31	$(8n-1)/(n-3)$	5.3579e−12	9.7847e−07	2.8510e−07
82	$(5n^2+2)/(n^2-15)$	6.5117e−08	5.1740e−05	1.5076e−05
136	$(7(n^2+3)+3)/(n^2-15)$	1.2211e−05	8.3025e−04	2.4191e−04
94	$(4(n^2+7)+2)/(n^2-15)$	1.5415e−09	1.2807e−05	3.7316e−06
113	$(7n^2+1)/(n^2-15)$	3.0525e−06	2.4356e−04	7.0966e−05
18	$(n^2+2)/(n^2-15)$	7.3295e−11	5.2840e−06	1.5396e−06
85	$(4(n^2+4)+5)/(n^2-15)$	7.7990e−10	1.3334e−05	3.8852e−06
145	$(9n^2+1)/(n^2-15)$	1.3153e−06	3.3408e−04	9.7339e−05
95	$(6n^2-1)/(n^2-15)$	3.3438e−09	1.2410e−05	3.6160e−06
129	$(8n^2+1)/(n^2-15)$	7.5708e−07	1.7821e−04	5.1924e−05
87	$(5(n^2+1)+2)/(n^2-15)$	6.1538e−09	2.2885e−05	6.6681e−06
131	$(6(n^2+5)+5)/(n^2-15)$	2.5795e−05	0.0015	4.3479e−04
11	$(n^2-5)/(n^2-15)$	1.8814e−10	9.8631e−06	2.8738e−06
55	$11(n+1)/(n-3)$	6.3463e−09	2.5821e−05	7.5233e−06
117	$(7n^2+5)/(n^2-15)$	4.9969e−08	3.8582e−05	1.1242e−05
32	$4(n+4)/(n-3)$	3.6366e−11	4.7464e−06	1.3829e−06
83	$(4(n^2+4)+3)/(n^2-15)$	3.5700e−09	1.2066e−05	3.5155e−06
142	$(8(n^2+1)+6)/(n^2-15)$	1.5833e−05	6.2708e−04	1.8271e−04
97	$(5(n^2+3)+2)/(n^2-15)$	6.3104e−09	1.8791e−05	5.4752e−06
122	$(7(n^2+1)+3)/(n^2-15)$	5.1288e−07	1.2684e−04	3.6958e−05
79	$(5n^2-1)/(n^2-15)$	2.1866e−08	3.4735e−05	1.0121e−05
91	$(5(n^2+3)-4)/(n^2-15)$	2.3943e−10	7.2422e−06	2.1101e−06
120	$(7(n^2+1)+1)/(n^2-15)$	5.9913e−06	7.5792e−04	2.2084e−04
12	$3n/(n-3)$	6.5737e−12	2.0074e−06	5.8490e−07
34	$6(n+2)/(n-3)$	2.9778e−11	3.9949e−06	1.1640e−06
147	$(9n^2+3)/(n^2-15)$	8.7650e−06	0.0013	3.8008e−04
89	$(5(n^2+1)+4)/(n^2-15)$	4.7469e−09	2.4536e−05	7.1491e−06
102	$(5(n^2+4)+2)/(n^2-15)$	1.6710e−08	4.5321e−05	1.3205e−05
35	$(2(n^2+1)+1)/(n^2-15)$	4.9542e−10	7.7716e−06	2.2644e−06
59	$(3(n^2+4)-1)/(n^2-15)$	1.2210e−09	4.4553e−06	1.2981e−06
150	$(7(n^2+5)+3)/(n^2-15)$	1.5983e−06	3.1898e−04	9.2942e−05
7	$(n+3)/(n-3)$	8.3577e−11	5.6418e−06	1.6438e−06
130	$(7(n^2+2)+4)/(n^2-15)$	1.6802e−07	1.1368e−04	3.3122e−05

Table 5 continued

Hidden neuron numbers	Considered criteria for estimating hidden neuron numbers	MSE	MAE	MRE
67	$(3(n^2+6)+1)/(n^2-15)$	2.7797e−08	3.2765e−05	9.5466e−06
26	$(6n+2)/(n-3)$	4.1959e−12	1.4017e−06	4.0840e−07
86	$(5n^2+6)/(n^2-15)$	2.4403e−06	2.7366e−04	7.9735e−05
78	$(4(n^2+3)+2)/(n^2-15)$	2.6156e−08	4.0134e−05	1.1694e−05
63	$9(n+3)/(n-3)$	1.6675e−10	6.8448e−06	1.9944e−06
15	$(n^2-1)/(n^2-15)$	1.8391e−10	7.4496e−06	2.1706e−06
128	$(6(n^2+5)+2)/(n^2-15)$	3.3888e−08	4.6706e−05	1.3609e−05
43	$(10n+3)/(n-3)$	3.9465e−10	8.9524e−06	2.6084e−06
109	$(6(n^2+2)+1)/(n^2-15)$	3.2675e−08	2.7802e−05	8.1006e−06
50	$10(n+1)/(n-3)$	8.7496e−10	1.3490e−05	3.9304e−06
135	$(8n^2+7)/(n^2-15)$	8.7440e−08	9.9479e−05	2.8985e−05
5	$(2n-3)/(n-3)$	7.5806e−11	5.1612e−06	1.5038e−06
41	$(2(n^2+4)+1)/(n^2-15)$	1.9255e−10	6.4819e−06	1.8886e−06
33	$(7n+5)/(n-3)$	2.0965e−11	2.6781e−06	7.8033e−07
151	$(8(n^2+2)+7)/(n^2-15)$	8.2197e−06	0.0011	3.2839e−04
69	$3(n^2+7)/(n^2-15)$	1.5473e−08	2.1580e−05	6.2877e−06
133	$(6(n^2+6)+1)/(n^2-15)$	3.1235e−07	1.8687e−04	5.4448e−05
16	$4n/(n-3)$	3.9063e−11	4.8803e−06	1.4220e−06
121	$(6(n^2+4)+1)/(n^2-15)$	9.5984e−08	8.6073e−05	2.5079e−05
47	$(11n+3)/(n-3)$	3.1541e−10	8.2048e−06	2.3906e−06
140	$(8(n^2+1)+4)/(n^2-15)$	5.3300e−07	2.0834e−04	6.0704e−05
51	$(2(n^2+6)+7)/(n^2-15)$	8.1451e−09	4.0164e−05	1.1703e−05
92	$(6n^2-4)/(n^2-15)$	6.8381e−10	7.2270e−06	2.1057e−06
76	$4n^2+3)/(n^2-15)$	2.2783e−09	1.4672e−05	4.2751e−04
80	$10(n+4)/(n-3)$	3.4750e−09	2.1949e−05	6.3951e−06
37	$(9n+1)/(n-3)$	9.4903e−11	6.1802e−06	1.8007e−06
61	$(3(n^2+3)+4)/(n^2-15)$	1.5444e−08	4.3937e−05	1.2802e−05
100	$4(n^2+9)/(n^2-15)$	1.2725e−09	1.4284e−05	4.1619e−06
1	$(2n-7)/(n-3)$	1.9817e−09	3.4459e−05	1.0040e−05
125	$(6(n^2+4)+5)/(n^2-15)$	1.9591e−09	1.3318e−05	3.8806e−06
65	$(3(n^2+5)+2)/(n^2-15)$	1.0442e−10	4.6428e−06	1.3528e−06
28	$4(n+3)/(n-3)$	1.5243e−10	5.1083e−06	1.4884e−06
138	$(7(n^2+3)+5)/(n^2-15)$	1.3769e−07	1.0308e−04	3.0033e−05
58	$(2(n^2+9)+8)/(n^2-15)$	1.2532e−10	6.3858e−06	1.8606e−06
103	$(6(n^2+1)+1)/(n^2-15)$	4.4690e−06	2.0142e−04	5.8686e−05
74	$(3(n^2+8)+2)/(n^2-15)$	1.9785e−09	1.8687e−05	5.4449e−06
118	$(6(n^2+3)+4)/(n^2-15)$	6.7286e−09	2.0547e−05	5.9868e−06
20	$((n^2+3)+1)/(n^2-15)$	1.1352e−12	6.0310e−07	1.7572e−07

Table 5 continued

Hidden neuron numbers	Considered criteria for estimating hidden neuron numbers	MSE	MAE	MRE
104	$(5(n^2+4)+4)/(n^2-15)$	5.5420e−08	7.8876e−05	2.2982e−05
99	$(5(n^2+4)-1)/(n^2-15)$	1.9357e−08	2.3871e−05	6.9552e−06
143	$(7(n^2+4)+3)/(n^2-15)$	2.9101e−06	5.5663e−04	1.6218e−04
57	$(3n^2+9)/(n^2-15)$	8.3250e−10	1.2418e−05	3.6183e−06
119	$(7n^2+7)/(n^2-15)$	1.3179e−09	1.1124e−05	3.2413e−06
17	$(3n+5)/(n-3)$	2.3851e−10	1.0524e−05	3.0664e−06
29	$(7n+1)/(n-3)$	2.7112e−10	7.4118e−06	2.1596e−06
44	$(6(n+3)+2)/(n-3)$	9.3439e−10	9.7645e−06	2.8451e−06
72	$8(n+5)/(n-3)$	1.0155e−08	3.1246e−05	9.1042e−06
110	$(5(n^2+5)+5)/(n^2-15)$	4.3391e−08	7.4771e−05	2.1786e−05
53	$(3(n^2+1)+2)/(n^2-15)$	1.5529e−10	7.6468e−06	2.2280e−06
21	$(5n+1)/(n-3)$	1.3959e−10	7.4493e−06	2.1705e−06
3	$(2n-5)/(n-3)$	4.2697e−11	4.9396e−06	1.4392e−06
45	$9(n+1)/(n-3)$	6.8438e−10	9.0065e−06	2.6242e−06
62	$(4n^2-2)/(n^2-15)$	1.1008e−09	1.2242e−05	3.5671e−06
24	$(3(n+3)+3)/(n-3)$	5.5362e−11	4.9016e−06	1.4282e−06
137	$(8(n^2+1)+1)/(n^2-15)$	4.2137e−05	0.0022	6.5216e−04
6	$(n+2)/(n-3)$	1.4224e−10	9.2186e−06	2.6860e−06
149	$(8(n^2+2)+5)/(n^2-15)$	1.1638e−05	0.0011	3.2818e−04
106	$(5(n^2+5)+1)/(n^2-15)$	6.3517e−10	8.4495e−06	2.4619e−06
93	$(4(n^2+8)-3)/(n^2-15)$	5.3735e−09	9.9356e−06	2.8949e−06
124	$(7(n^2+1)+5)/(n^2-15)$	1.3037e−09	9.0830e−06	2.6465e−06
68	$4(n^2+1)/(n^2-15)$	1.8148e−08	4.1631e−05	1.2130e−05
52	$(6(n+4)+4)/(n-3)$	1.2566e−09	1.0150e−05	2.9575e−06
10	$(4n-6)/(n-3)$	1.4980e−10	8.9756e−06	2.6152e−06
4	$n/(n-3)$	2.2739e−10	1.1684e−05	3.4044e−04
148	$(7(n^2+5)+1)/(n^2-15)$	4.8707e−06	7.1638e−04	2.0873e−04
98	$(4(n^2+8)+2)/(n^2-15)$	2.5609e−08	5.1780e−05	1.5087e−05
105	$(6(n^2+1)+3)/(n^2-15)$	8.3273e−07	1.1712e−04	3.4125e−05
70	$(4n^2+6)/(n^2-15)$	3.1924e−09	2.6116e−05	7.6095e−06
56	$7(n+4)/(n-3)$	3.7724e−08	3.2158e−05	9.3698e−06
9	$(2n+1)/(n-3)$	2.3592e−11	3.2729e−06	9.5361e−07
23	$(5n+3)/(n-3)$	4.0519e−11	4.8398e−06	1.4102e−06
111	$(5(n^2+6)+1)/(n^2-15)$	7.3510e−07	2.1662e−04	6.3115e−05
88	$4(n^2+6)/(n^2-15)$	4.4673e−08	3.0195e−05	8.7978e−06
36	$(5(n+3)+1)/(n-3)$	3.9980e−11	3.3129e−06	9.6528e−07
49	$(8(n+2)+1)/(n-3)$	1.8303e−10	6.1238e−06	1.7843e−06
73	$(4(n^2+3)-3)/(n^2-15)$	9.4147e−09	2.4686e−05	7.1927e−06
139	$(6(n^2+6)+7)/(n^2-15)$	1.0592e−05	3.9411e−04	1.1483e−04

Table 5 continued

Hidden neuron numbers	Considered criteria for estimating hidden neuron numbers	MSE	MAE	MRE
13	$(4n-3)/(n-3)$	1.6315e−11	2.9227e−06	8.5157e−07
144	$(6(n^2+7)+6)/(n^2-15)$	1.9093e−07	1.1212e−04	3.2669e−05
90	$(4(n^2+6)+2)/(n^2-15)$	5.9789e−10	1.0618e−05	3.0938e−06
116	$(5(n^2+7)+1)/(n^2-15)$	1.2226e−08	4.9695e−05	1.4479e−05
64	$8(n+4)/(n-3)$	3.5130e−11	3.3256e−06	9.6897e−07
132	$(7(n^2+2)+6)/(n^2-15)$	7.6054e−06	8.3015e−04	2.4188e−04
8	$(3n-4)/(n-3)$	3.3484e−11	4.1694e−06	1.2148e−06
39	$(6(n+2)+3)/(n-3)$	2.4465e−10	6.6075e−06	1.9252e−06
127	$(7(n^2+2)+1)/(n^2-15)$	4.8326e−07	1.9636e−04	5.7214e−05
25	$5(n+1)/(n-3)$	3.0892e−11	4.1200e−06	1.2004e−06
112	$(6(n^2+2)+4)/(n^2-15)$	2.2182e−07	1.9133e−04	5.5747e−05
146	$(7(n^2+4)+6)/(n^2-15)$	1.6619e−05	0.0013	3.7609e−04
96	$4(n^2+8)/(n^2-15)$	1.3488e−09	1.3660e−05	3.9802e−06
114	$(5(n^2+6)+4)/(n^2-15)$	6.5597e−11	2.5987e−06	7.5717e−07
60	$10(n+2)/(n-3)$	5.0169e−08	7.2703e−05	2.1183e−05
30	$5(n+2)/(n-3)$	7.7234e−11	4.8685e−06	1.4185e−06
2	$(n-2)/(n-3)$	1.6196e−10	9.9044e−06	2.8858e−06
141	$(9n^2-3)/(n^2-15)$	7.8572e−06	9.3683e−04	2.7296e−04
27	$3(n+5)/(n-3)$	3.0230e−11	3.6702e−06	1.0694e−06
81	$(4(n^2+4)+1)/(n^2-15)$	1.0288e−09	1.6110e−05	4.6941e−06
48	$6(n+4)/(n-3)$	4.8769e−10	1.1510e−05	3.3535e−06
77	$(3(n^2+9)+2)/(n^2-15)$	6.5267e−09	2.7482e−05	8.0073e−06
107	$(6(n^2+1)+5)/(n^2-15)$	6.3742e−08	7.3299e−05	2.1357e−05
14	$(4n-2)/(n-3)$	**4.9865e−14**	**1.6387e−07**	**4.7746e−08**
54	$9(n+2)/(n-3)$	2.8120e−10	1.0184e−05	2.9672e−06
134	$(7(n^2+3)+1)/(n^2-15)$	8.1559e−05	0.0021	6.2095e−04
46	$(2(n^2+5)+4)/(n^2-15)$	1.7390e−09	1.0689e−05	3.1143e−06
71	$(3(n^2+8)-1)/(n^2-15)$	3.0543e−08	2.6819e−05	7.8144e−06
38	$(7(n+1)+3)/(n-3)$	1.9068e−10	4.2388e−06	1.2351e−06
22	$(n^2+6)/(n^2-15)$	1.1801e−11	2.5374e−06	7.3932e−07
101	$(6n^2+5)/(n^2-15)$	1.4660e−07	9.2800e−05	2.7039e−05
84	$(5(n^2+1)-1)/(n^2-15)$	1.2486e−07	1.2171e−04	3.5461e−05
115	$(6(n^2+3)+1)/(n^2-15)$	1.6444e−06	2.0650e−04	6.0169e−05
40	$8(n+1)/(n-3)$	2.7283e−10	1.0038e−05	2.9246e−06
126	$(8n^2-2)/(n^2-15)$	5.2265e−07	2.0893e−04	6.0875e−05
66	$11(n+2)/(n-3)$	6.9399e−13	3.3750e−07	9.8337e−08
19	$(3n+7)/(n-3)$	1.5179e−11	2.3923e−06	6.9706e−07

The bold values imply the best results

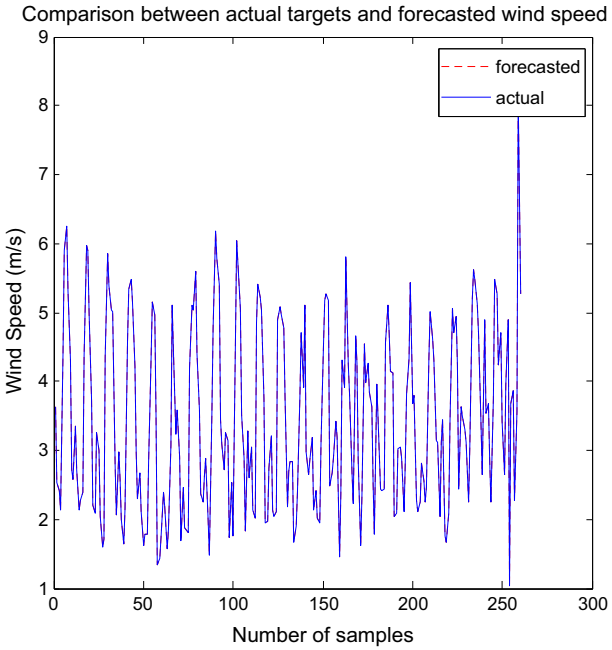

Fig. 2 Comparison between actual and forecasted wind speed

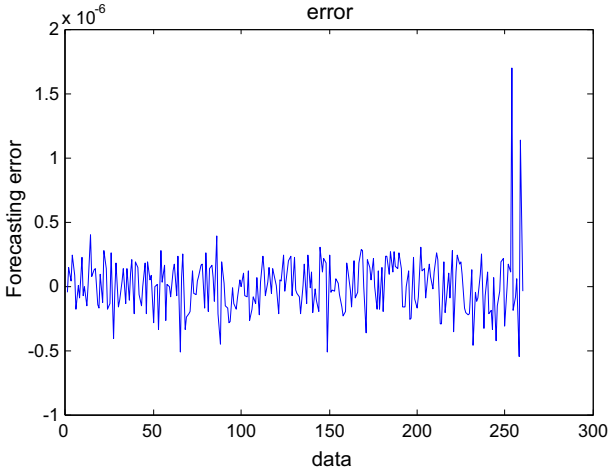

Fig. 3 Forecasting error

5.2 Analysis of error factor and computed wind speed based on proposed criteria

The developed new criteria for estimating and fixing the number of hidden neuron numbers in multilayer perceptron networks based on the statistical error are established in the Table 5. The selected criterion for multilayer perceptron network design is $u_n = [(4n-2)/(n-3)]$. In Table 5, it is noted that for the developed criterion $(4n-2)/(n-3)$, the error values MSE,

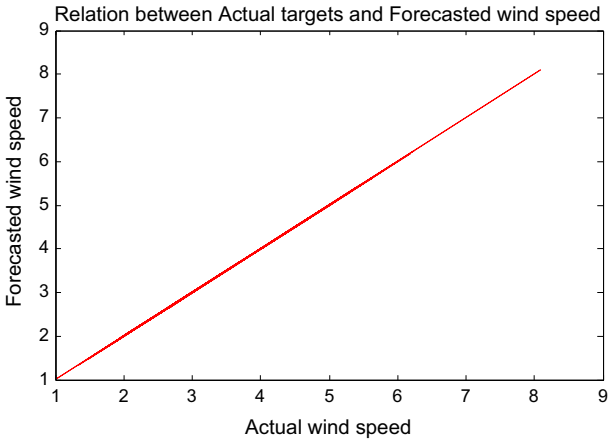

Fig. 4 Relation between actual and forecasted wind speed

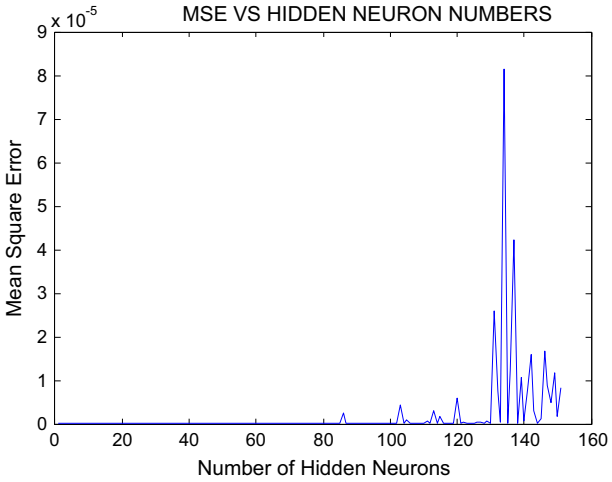

Fig. 5 MSE versus number of hidden neurons

MAE and MRE computed are 4.9865e−14, 1.6387e−07 and 4.7746e−08 respectively. On comparison of the error values computed employing the proposed other criteria in Table 5, the said criterion $(4n-2)/(n-3)$ has resulted in minimal error values. This elucidates that the proposed criterion $(4n-2)/(n-3)$ employs 14 hidden neurons and achieved minimal MSE, MAE and MRE values than that of the other proposed criteria in Table 5. Therefore, the proposed criterion improves the effectiveness and accuracy for wind speed forecasting application.

Based on the proposed model the comparison between the actual and forecasted wind speed shown in Fig. 2. For the clarity of actual and forecasted wind speed, 240 samples of actual and forecasted data are depicted in the result is noticed from Fig. 2. According to Fig. 2 it is well noted that the curve plotted for the computed forecast wind speed converges with that of the existing actual wind speed. This proves that the forecasted wind speed employing

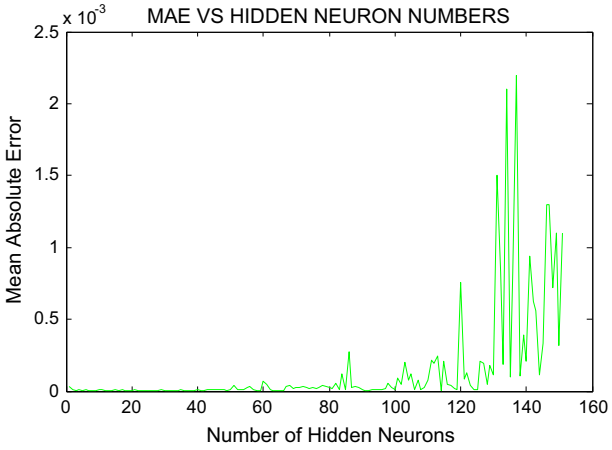

Fig. 6 MAE versus number of hidden neurons

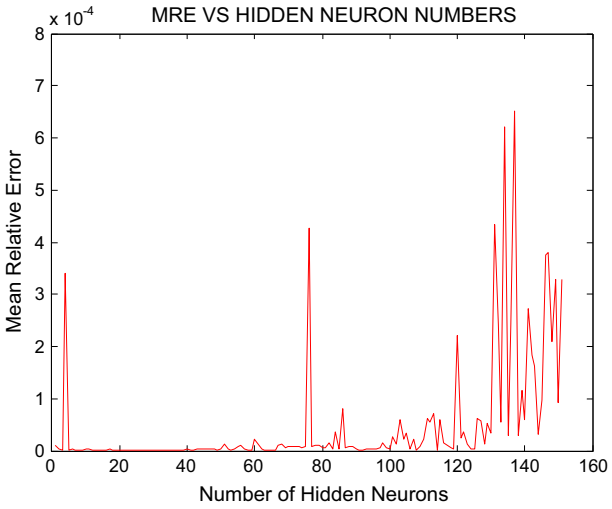

Fig. 7 MRE versus number of hidden neurons

the proposed methodology is highly accurate in nature due to exact convergence of the plot given in Fig. 2.

The forecasting error based on the data is shown in Fig. 3. Figure 3 infers that the error variation for the number of sample data is fluctuating nearer to zero and it is noted that at the end of the presentation of the last data sample it has met zero point. This convergence proves the efficacy of the proposed model in concurrence with that of the error rate. The relation between actual and forecasted wind speed is noticed from Fig. 4. There exists a linear variation between actual and forecasted wind speed which is observed in Fig. 4 and this proves that the computed wind speed is directly proportional to that of the actual wind speed confirming the better solution achieved using the proposed methodology.

Comparisons of statistical errors such MSE, MAE and MRE with that of the number of hidden neurons are depicted respectively in Figs. 5, 6 and 7 respectively. When the number

Table 6 Comparative analysis of different approaches performance in existing and proposed method

S. no	Different approaches	Year	Hidden neuron numbers	Statistical error (MSE)
1	Arai M approach	1993	$N_h = 2^n/3$	7.5806e−11
2	Jin-Yan Li et al. approach	1995	$N_h = (\sqrt{1+8n-1})/2$	4.2697e−11
3	Tamura S and Tateishi M approach	1997	$N_h = N-1$	4.2697e−11
4	Osamu Fujita approach	1998	$N_h = K\log(\|P_v^{(0)}Z\|/C)/\log S$	1.6315e−11
5	Zhaozhi Zhang et al. approach	2003	$N_h = 2^n/(n+1)$	4.2697e−11
6	Herbert Jaeger and Harald Haas approach	2004	$N_h = 3^n - 4n$	3.1265e−11
7	Jinchauan Ke and Xinzhe Lie approach	2008	$N_h = (N_{in} + \sqrt{N}_p/L)$	3.6366e−11
8	Shuxiang Xu and Ling Chen approach	2008	$N_h = C_f(N/(d\log N))^{1/2}$	1.8391e−10
9	Stephen Trenn approach	2008	$N_h = (n+n_0-1)/2$	1.6196e−10
10	Katsunari Shibata and Yusuke Ikeda approach	2009	$N_h = \sqrt{N^{(i)}N^{(0)}}$	1.6196e−10
11	David Hunter et al. approach	2012	$N_h = 2^n - 1$	1.8391e−10
12	Gnana Sheela K and Deepa SN approach	2013	$N_h = (4n^2+3)/(n^2-8)$	3.3484e−11
13	Gue Qian and Hao Yong approach	2013	$N_h = \sqrt{n+m+a}$	2.2739e−10
14	Peter SE et al. approach	2013	$N_h = 6n/(n+1)$	9.1430e−12
15	Wang J and Hu J approach	2015	$N_h = \sqrt{(n+m)}/2$	1.9834e−12
16	Meng A et al. approach	2016	$N_h = 3^n/(n+6)$	6.7821e−12
17	Proposed approach		$N_h = (4n-2)/(n-3)$	**4.9865e−14**

The bold values imply the best results

of hidden neurons increases that the mean square error value also increases, it is noted in Fig. 5, Fig. 5 depicts that when the number of hidden neurons are minimal, MSE value is also minimal. In Figs. 6 and 7 also it is noted that for increased number of neurons, the error values also increase. As the number of hidden neurons decrease, MAE and MRE values also get decreased. On simulation runs, the proposed approach achieved a minimum MSE of 4.9865e−14, MAE of 1.6387e−07 and MRE of 4.7746e−08 when the number of hidden neurons is 14 as given in Table 5.

The identified problem in this research paper is to fix an appropriate number of hidden neurons in the multilayer perceptron neural networks as applicable for wind speed forecast-

ing application. The earlier approaches employ trial and error method to determine number of hidden neurons in MLP network model. This starts the network with undersized hidden neuron numbers and neurons are added to N_h. The demerits are there is no guarantee of selecting the number of hidden neurons and it consumes more time. The developed criterion as adapted to the multilayer perceptron network model for wind speed forecasting is $(4n-2)/(n-3)$ which employed 14 hidden neuron numbers and achieved a reduced mean square error (MSE) value of 4.9865e−14 in comparison with that of the other developed criteria as shown in Table 5. Experimental results confirm that the developed criterion with minimum error is determined as the best solution for estimating the number of hidden neurons in a multilayer perceptron network model. Simulation results prove that the forecasted wind speed is in the best agreement with that of the experimental measured values. The analysis of wind speed forecasting is performed with respect to the presented new criterion. Table 6 infers that compared to the other existing model in the literature the proposed model achieves better accuracy with minimal error value in comparison with that of the extreme learning machine model (Wang and Hu 2015), echo state network model (Jaeger and Haas 2004), wavelet neural model (Meng et al. 2016) and spiking neural network model (Peter et al. 2013).

6 Conclusion

In this paper a comparative analysis is performed for estimating the number of hidden neurons in artificial neural network design. The developed approach was adapted and evaluated with real-time wind data. The main objective of the proposed approach is to assist the planning; integration and control of power system and wind farm by estimating the appropriate number of hidden neurons in multilayer perceptron networks for wind speed forecasting application and this as well attains a better performance in comparison with that of the other existing approaches. Statistical errors are used to analyze the performance of the network. The developed multilayer perceptron network as applicable for wind speed forecasting achieves better accuracy with reduced error, improved stability and faster convergence.

Appendix

Considers the applicability of different criteria with 'n' as input parameters. All developed criteria should satisfy the convergence theorem. If the limit of a sequence is finite, the sequence is called a convergent sequence. If the limit of a sequence does not tend to be a finite number, the sequence is called divergent (Dass 2009).

The convergence theorem characteristics are given below.

1. A convergent sequence has a finite limit.
2. All convergent sequences are bounded sequence.
3. All bounded point has a finite limit.
4. Convergent sequence needed condition is that it has finite limit and is bounded.
5. An oscillatory sequence does not tend to have a unique limit.

In a network there is no change occurring in the state of the network, regardless of the operation is called the stable network. For neural network model most important property is it always converges to a stable state. In real-time optimization problem the convergence plays a major role, the risk of getting stuck at some local minima problem in a network is prevented by the convergence. The convergence of sequence infinite has been established

in convergence theorem because of the discontinuities in the model. The real-time neural optimization solvers are designed with the use of convergence properties.

Presenting the convergence of the considered sequence as follows,

$$\text{Taking the sequence } u_n = \frac{11\,(n+1)}{n-3} \tag{13}$$

Apply convergence theorem,

$$n \overset{\lim}{\to} \infty u_n = n \overset{\lim}{\to} \infty \frac{11\,(n+1)}{n-3} = n \overset{\lim}{\to} \infty \frac{n\,(11+1/n)}{n\,(1-3/n)} = 11 \neq 0, \quad \text{it has a finite value.} \tag{14}$$

Hence, the terms of a sequence have a finite limit value and are bounded so the considered sequence is convergent sequence.

$$\text{Take the sequence } u_n = \frac{8n^2-2}{n^2-15} \tag{15}$$

Apply convergence theorem,

$$n \overset{\lim}{\to} \infty u_n = n \overset{\lim}{\to} \infty \frac{8n^2-2}{n^2-15} = n \overset{\lim}{\to} \infty \left(\frac{n^2}{n^2} \left[\frac{8-2/n^2}{1-15/n^2} \right] \right) = 8 \neq 0, \quad \text{it has a finite value.} \tag{16}$$

Hence, the terms of a sequence have a finite limit value and are bounded so the considered sequence is convergent sequence.

References

Arai M (1993) Bounds on the number of hidden units in binary-valued three-layer neural networks. Neural Netw 6:855–860

Choi B, Lee J-H, Kim D-H (2008) Solving local minima problem with large number of hidden nodes on two layered feed forward artificial neural networks. Neurocomputing 71:3640–3643

Dass HK (2009) Advanced engineering mathematics, 1st edn 1988. S. CHAND & Company Ltd, New Delhi

Doukin CA, Dargham JA, Chekima A (2010) Finding the number of hidden neurons for an MLP neural network using coarse to fine search technique. In: 10th International conference on information sciences signal processing and their applications (ISSPA), pp 606–609

Fujita O (1998) Statistical estimation of the number of hidden units for feed forward neural network. Neural Netw 11:851–859

Gnana Sheela K, Deepa SN (2013) Review on methods to fix number of hidden neurons in neural networks. Math Probl Eng 2013:1–11

Hagiwara M (1994) A simple and effective method for removal of hidden units and weights. Neuro Comput 6:207–218

Han M, Yin J (2008) The hidden neurons selection of the wavelet networks using support vector machines and ridge regression. Neuro Comput 72:471–479

Huang G-B (2003) Learning capability and storage capacity of two-hidden layer feed forward networks. IEEE Trans Neural Netw 14:274–281

Huang S-C, Huang Y-F (1991) Bounds on the number of hidden neurons in multilayer perceptrons. IEEE Trans Neural Netw 2:47–55

Hunter D, Hao Y, Pukish III MS, Kolbusz J, Wilamowski BM (2012) Selection of proper neural network sizes and architecture—a comparative study. IEEE Trans Ind Inf 8:228–240

Jaeger H, Haas H (2004) Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. Science 304(5667):78–80

Jiang N, Zhang Z, Ma X, Wang J (2008) The lower bound on the number of hidden neurons in multi-valued multi threshold neural networks. In: Second international symposium on intelligent information technology application, vol 1, pp 103–107

Karsoliya S (2012) Approximating number of hidden layer neuron in multiple hidden layer BPNN architecture. Int J Eng Trends Technol 31:714–717

Keeni K, Nakayama K, Shimodaira H (1999) Estimation of initial weights and hidden units for fast learning of multilayer neural networks for pattern classification. In: International joint conference on neural networks, vol 3, pp 1652–1656

Ke J, Liu X (2008) Empirical analysis of optimal hidden neurons in neural network modeling for stock prediction. In: Pacific-Asia workshop on computational intelligence and industrial application, vol 2, pp 828–832

Li J-Y, Chow TWS, Yu Y-L (1995) The estimation theory and optimization algorithm for the number of hidden units in the higher-order feed forward neural network. In: Proceeding IEEE international conference on neural networks, vol 3, pp 1229–1233

Li J, Zhang B, Mao C, Xie G, Li Y, Lu J (2010) Wind speed prediction based on the Elman recursion neural networks. In: International conference on modelling, identification and control, pp 728–732

Madhiarasan M, Deepa SN (2016) A novel criterion to select hidden neuron numbers in improved back propagation networks for wind speed forecasting. Appl Intell 44(4):878–893

Mao KZ, Huang G-B (2005) Neuron selection for RBF neural network classifier based on data structure preserving criterion. IEEE Trans Neural Netw 16:1531–1540

Meng A, Ge J, Yin H, Chen S (2016) Wind speed forecasting based on wavelet packet decomposition and artificial neural networks trained by crisscross optimization algorithm. Energy Convers Manag 114:75–88

Morris AJ, Zhang J (1998) A sequential learning approach for single hidden layer neural network. Neural Netw 11:65–80

Murata N, Yoshizawa S, Amari S-I (1994) Network information criterion determining the number of hidden units for an artificial neural network model. IEEE Trans Neural Netw 5:865–872

Onoda T (1995) Neural network information criterion for the optimal number of hidden units. In: Proceeding IEEE international conference on neural networks, vol 1, pp 275–280

Panchal G, Ganatra A, Kosta YP, Panchal D (2011) Behaviour analysis of multilayer perceptrons with multiple hidden neurons and hidden layers. Int J Comput Theory Eng 3:332–337

Peter SE, Kulkarni S, Raglend IJ, Simon SP (2013) Wavelet based spike propagation neural network (WSPNN) for wind power forecasting. Int Rev Model Simul (IREMOS) 6(5):1513–1522

Qian G, Yong H (2013) Forecasting the rural per capita living consumption based on Matlab BP neural network. Int J Bus Soc Sci 4:131–137

Ramadevi R, Sheela Rani B, Prakash V (2012) Role of hidden neurons in an Elman recurrent neural network in classification of cavitation signals. Int J Comput Appl 37:9–13

Shibata K, Ikeda Y (2009) Effect of number of hidden neurons on learning in large-scale layered neural networks. In: ICROS-SICE international joint conference, pp 5008–5013

Sivanandam SN, Sumathi S, Deepa SN (2008) Introduction to neural networks using Matlab 6.0, 1st edn. Tata McGraw Hill, New Delhi

Sun J (2012) Learning algorithm and hidden node selection scheme for local coupled feed forward neural network classifier. Neuro Comput 79:158–163

Tamura S, Tateishi M (1997) Capabilities of a four-layered feed forward neural network: four layer versus three. IEEE Trans Neural Netw 8:251–255

Teoh EJ, Tan KC, Xiang C (2006) Estimating the number of hidden neurons in a feed forward network using the singular value decomposition. IEEE Trans Neural Netw 17:1623–1629

Trenn S (2008) Multilayer perceptrons: approximation order and necessary number of hidden units. IEEE Trans Neural Netw 19:836–844

Urolagin S, Prema KV, Subba Reddy NV (2012) Generalization capability of artificial neural network incorporated with pruning method. Lect Notes Comput Sci 7135:171–178

Vora K, Yagnik S (2014) A new technique to solve local minima problem with large number of hidden nodes on feed forward neural network. Int J Eng Dev Res 2:1978–1981

Wang J, Hu J (2015) A robust combination approach for short-term wind speed forecasting and analysis—combination of the ARIMA (autoregressive integrated moving average), ELM (extreme learning machine), SVM (support vector machine) and LSSVM (least square SVM) forecasts using a GPR (Gaussian process regression) model. Energy 93:41–56

Xu S, Chen L (2008) A novel approach for determining the optimal number of hidden layer neurons for FNN's and its application in data mining. In: 5th International conference on information technology and application (ICITA), pp 683–686

Yuan HC, Xiong FL, Huai XY (2003) A method for estimating the number of hidden neurons in feed-forward neural networks based on information entropy. Comput Electron Agric 40:57–64

Zeng X, Yeung DS (2006) Hidden neuron purning of multilayer perceptrons using a quantified sensitivity measure. Neuro Comput 69:825–837

Zhang Z, Ma X, Yang Y (2003) Bounds on the number of hidden neurons in three-layer binary neural networks. Neural Netw 16:995–1002

M. Madhiarasan has completed his B.E (EEE) in the year 2010 from Jaya Engineering College, Thiruninravur, M.E. (Electrical Drives & Embedded Control) from Anna University, Regional Centre, Coimbatore, in the year 2013. He is currently doing Research (Ph.D) under Anna University, TamilNadu, India. His Research areas include Neural Networks, Modeling and simulation, Renewable Energy System and Soft Computing.

S. N. Deepa has completed her B.E (EEE) in the year 1999 from Government College of Technology, Coimbatore, M.E. (Control Systems) from PSG College of Technology in the year 2004 and Ph.D.(Electrical Engineering) in the year 2008 from PSG College of Technology under Anna University, TamilNadu, India. Her Research areas include Linear and Non-linear control system design and analysis, Modeling and simulation, Soft Computing and Adaptive Control Systems.