Asian Journal of Information Technology 15 (21): 4377-4391, 2016
ISSN: 1682-3915


# New Criteria for Estimating the Hidden Layer Neuron Numbers for Recursive Radial Basis Function Networks and its Application in Wind Speed Forecasting

M. Madhiarasan and S.N. Deepa
Anna University, Regional Campus, Coimbatore, Tamil Nadu, India

**Abstract:** This study is intended to propose new criteria to decide appropriate hidden layer neuron numbers in Recursive Radial Basis Function Networks (RRBFN) and successfully applied to the wind speed forecasting application in renewable energy system. Purpose of the proposed methodology eliminate both either over fitting or under fitting issues. The proper hidden layer neuron numbers is evolved through the presented 150 various criteria. Exact modeling of recursive radial basis function networks possess with three input variables using the proposed new determining criteria are validated by means of the convergence theorem. In order to verify effectiveness and generalization capability of the proposed methodology, computer simulation is carried out on two real-time data sets and selection of data influence on the results are analyzed with various training and testing data. Experiment results confirmed that the proposed criteria result better framework for recursive radial basis function networks with reduced statistical errors compared with other previous methodologies.

**Key words:** Artificial intelligence, new criteria, hidden layer neuron numbers, recursive radial basis function networks, wind speed forecasting

## INTRODUCTION

Artificial neural networks have wide spread familiarity in lot of fields due to the real-time operation capability, self-learning capability, adaptability and cost effectiveness. An artificial neural network is an information processing structure developed based on the inspiration of biological system (brain, process information), it is designed from interconnected elementary processing elements is called neurons. Selection of proper hidden layer neuron numbers are play active role in neural networks output. But selection of proper hidden layer neuron numbers is one of major problem in neural networks development process. Neural networks with few hidden neuron units may not have the sufficient power to satisfy the requirements such as accuracy, error precision and capacity. The problem of over fitting data is caused by over training. The over training issue has occurred in the neural networks design. Therefore, selection of hidden neuron numbers in neural networks design is one of the important problems.

Feed-forward networks and feedback (recurrent) networks are classification of artificial neural network. Feed-forward networks is a network with no feedback path e.g. Radial basis function networks (RBFN), Back Propagation Networks (BPN), etc., whereas feedback networks is a network with feedback path that occur between the layers (or) within the layers, e.g. Recurrent Neural Networks (RNN), Hop-field, etc. In late 80's Radial basis function networks is employed as one of the important variant of artificial neural networks, generally applied for prediction, function approximation and pattern recognition applications.

Due to the need for enhanced and advanced power system, wind farm planning, scheduling and control operation several researcher turned their attention towards wind speed forecasting. This paper propose novel recursive radial basis function networks for wind speed forecasting application and right hidden layer neuron numbers are evolved based on different 150 criteria, suggested all 150 criteria are verified based on the convergence theorem.

**Literature review:** This study analyzes the proper choice of hidden layer neuron numbers for recursive radial basis function neural networks. If the hidden layer neuron number is large the hidden output connection weights become so less, learn easily and also the trade-off in stability between input and hidden output connection exists. The output neurons become unstable for large hidden layer neuron numbers. While if the hidden layer neuron number is small it may fall in to local mini ma due to slow learning of the network and hidden neurons become unstable. Therefore, deciding the proper hidden layer neuron numbers is an important critical problem for neural networks design and development. Several

**Corresponding Author:** M. Madhiarasan, Anna University, Regional Campus, Coimbatore, Tamil Nadu, India

research experiments tried and suggested determining proper hidden layer neuron numbers in neural networks by many researchers. A comprehensive survey is made to determine the hidden layer neuron numbers in neural networks design and illustrations are given as below.

Arai (1993) performed selection of hidden neurons in network design based on Two Parallel Hyper plane Methods (TPHM), appropriate hidden neurons for network development is stated as $2^n/3$. Li *et al.* (1995) carried out investigation to found the optimal number of hidden units in the higher order feed-forward neural network based on estimation theory. Tamura and Tateishi (1997) presented finite number of hidden neurons based neural network capability studies. From the results it can be observed that N-1 hidden neurons for three-layered feed-forward network and (N/2)+3 hidden neurons for four-layered feed-forward network get exact input and target relation with minimum error. Four-layered network get better performance than that of three-layered network. Fujita (1998) pointed out proper number of hidden neurons for feed-forward neural network using statistical estimation. In addition to the number of inputs, the non-linearity of hidden neurons has an effect on output error reduction and stated that $N_h = K\log(\|P_C^{(0)}Z\|/C)/\log S$ are required number of hidden neurons where: K = data sets, C = allowable output error, S = total amount of candidates that are randomly found optimal hidden neurons.

Zhang *et al.* (2003) developed three layer binary neural networks using set covering algorithm. The sufficient and required number of hidden neurons is determined as 3L/2 where: L = Number of unit sphere contained in the selected unit sphere covering of N-dimensional Hamming space. Compared with two-parallel hyper plane approach, SCA based approach estimating much reduced number of hidden neurons. Mao and Huang (2005) performed data structure preserving criterion based determination of hidden neurons for RBF neural network classifier. Ke and Liu (2008) pointed out neural network development for prediction of stock price and performed neural network sensitivity investigation based on optimal number of hidden neurons and hidden layers. Based on the computed generalization and training errors for forty cases with different hidden neurons using formula $N = (N_{in} + \sqrt{N_p})/L$ where, $N_{in}$ is the number of input neuron, $N_p$ is the number of input sample, L is the number of hidden layer; the minimum error is evolved as the optimal number of hidden neurons. Xu and Chen (2008) suggested feed-forward neural network for financial data mining application and number of hidden neurons estimated using novel approach is:

$$N_h = C_f(N/(d\log N))^{1/2}$$

Where:

$C_f$ = The first absolute moment of the fourier magnitude distribution of the target function
d = The input dimension of target function
N = The number of training pair

**Remarks:** In this case local mini ma issue is not addressed, suggested formulation based $N_h$ get the lowest RMS error. Trenn (2008) pointed out the necessary number of hidden neurons and approximation order for multilayer perceptron network. Necessary number of hidden neurons is $N_h = (n+n_o-1)/2$ where, $n_o$ number of inputs; which is formulated based on the desired approximation order and number of input variables. Shibata and Ikeda (2009) investigated the impact of learning rate and number of hidden neurons on stable learning in large scale layered neural network. Formulate the learning rate $\eta = 32/(\sqrt{N^{(i)}}N^{(0)})$ as and number of hidden neurons $N_h\sqrt{N^{(i)}}N^{(0)}$:

Where:

$N^{(i)}$ = No. of input neurons
$N^{(o)}$ = No. of output neurons

Hunter *et al.* (2012) analyzed appropriate choice of neural network size and architecture. Different learning algorithm (EBP, LM and NBN), various network topologies (MLP, BMLP and FCC) efficiency and generalization issues are investigated in order to select the proper neural network size and architecture. For MLP, BMLP and FCC the required number of hidden neurons is stated as $N_h = N+1$, $N_h = 2N+1$ and $N_h = 2^n-1$ a, respectively.

**Remarks:** Compared to MLP, BMLP network trained easily. Numerous trials are required for selection of hidden neurons. Qian and Yong (2013) evolved rural per ca pita living consumption prediction using BP neural network, better accuracy is achieved with:

$$N_h = \sqrt{n+m+a}$$

Where:

n = No. of nodes in input layer
m = No. of nodes in output layer
a = Integer among 0-10

Sheela and Deepa (2013) implemented wind speed prediction based on radial basis function network and hidden neurons are searched by means of the new algorithm. Best hidden neurons are selected from the suggested 101 criteria based on the minimal error value. Vora and Yagnik (2014) presented solution for local mini

ma issue caused due to the large hidden neurons by means of the new algorithm having structural changes in feed-forward neural network. Madhiarasan and Deepa (2016) proposed wind speed forecasting model using IBPN (improved back propagation network) and analyzed network hidden neuron number selection with existing and proposed 151 criterion, finally the best result achieved based on the proposed novel criterion. According to the previous related works the following points are observed.

Numerous researcher emphases to achieve the better performance by handling this issue. But in neural network there is no other way to determine the hidden layer neuron numbers without attempting and evaluating during the training process and evolving the generalization error. The neural network properties such as stability and convergence are checked by means of the performance analysis. Much research developed different methodologies for searching the proper hidden neuron numbers in neural networks design in order to achieve the quick convergence, better accuracy and efficiency with minimal statistical errors.

In neural networks design process the fixation of hidden layer neurons play vital role, it helps to achieve the better performance and stability. Hence, this study proposes new criteria for proper fixation of hidden layer neuron numbers in recursive radial basis function networks for wind speed forecasting application to make better accuracy, stability and generalization ability.

## MATERIALS AND METHODS

**Scope of estimating hidden layer neuron numbers:** Determination of correct hidden layer neuron numbers and parameters is crucial and important task for neural networks design. Major issue is to decide right hidden layer neuron numbers in order to get exact solution for particular task.

**Issues in neural networks modeling:** The issues involved in neural networks modeling and training for a specific application are given as follows:

- Selecting what architecture to be used in the neural networks design
- Selection of how many hidden layers to be used in the neural networks
- Deciding the hidden neurons in each hidden layers
- Fixing how many training pair to be used in the neural networks
- Deciding which training algorithm to be used in the neural networks
- Evaluating the network to check for over fitting (or) under fitting issues
- Searching a global optimal solution that prevents local mini ma issue by Hunter *et al.* (2012) and Panchal *et al.* (2011)

Neural networks is designed with less hidden neurons produce large training errors and generalization errors because of the under fitting problem while too many hidden neurons used in neural networks cause low training error and high generalization error due to over fitting issue stated by Xu and Chen (2008). Hence, choice of proper hidden neurons has profound impact on the errors. The errors are used to evolve the neural network stability. Ke and Liu (2008) found that neural network with minimum error has the better stability while the network with higher error has the poor stability.

The process of deciding number of hidden layers and number of hidden neurons in each hidden layers is still confusing and challenging task stated by Karsoliya (2012). Strategies to select the hidden neuron numbers in artificial neural networks can be classified in to pruning and constructive strategies. In the pruning strategy the network starts with oversized and then prunes the minimum relevant neuron and weights search the minimum size, while in the constructive strategy network starts with undersized network and then supplementary hidden neuron are included to the network stated by Li *et al.* (1995) and Zhang and Morris (1998). Neural networks training accuracy is estimated by the parameters such as neural networks architecture, inputs, number of hidden layers, hidden neuron numbers in each hidden layers, activation function, outputs and weights updating process.

**Proposed determining criteria:** The small size networks offers simple structure and better generalization ability but it may not learn the issue well. While in the large size network learn easily but slow and poor generalization performance due to over fitting stated by Urolagin *et al.* (2011). In neural network design different heuristics are exists, there is no hard and fast accurate rule for estimation of hidden neuron numbers in the neural networks design. Therefore, new criteria are proposed for proper selection of hidden layer neuron numbers in recursive radial basis function networks. The considered 150 different criteria are function of input layer neurons and are noted to be satisfied the convergence theorem. Among the 150 different criteria, the best criteria are estimated based on the least statistical errors. Proposed new criteria estimate the right hidden layer neuron numbers in recursive radial basis function networks with less computational complexity.

**Proposed wind speed forecasting model:** The ultimate aim of the proposed wind speed forecasting model is to

estimate the hidden neuron numbers in the recursive radial basis function neural network for wind speed forecasting with reduced statistical errors and better accuracy.

**Impact of wind speed:** The dual aims of global reduction of $CO_2$ emission and improving energy security (energy policy goals in many countries) coincides in the increasing use of wind energy for electricity generation. Due to the time varying and changeability of wind, wind speed forecasting is very important hot topic in research. Wind power caused bad impact on electrical system is eliminated by accurate wind speed forecasting. Small fraction deviations of wind speed will lead to a large error output of wind driving systems. Need for wind speed forecasting is given below:

- Reliable and better quality operation of power system
- To assist planning, scheduling and control of wind farm and power system
- Effective integration of wind power to the electrical power grid
- Minimizes the operating cost of the wind power generation
- To meet low spinning reserve

**Purpose neural networks for wind speed forecasting:** Wind has uncertain and irregularity characteristic. In order to achieve the better generalization capabilities for the wind speed forecasting the input and output is to be modeled 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. Many researchers were tried to developed different strategies to select the proper hidden neuron numbers in the neural network but yet none was effective and accurate. In the current scenario lot of forecasting research fields have been heuristic in nature. An accurate wind speed forecasting is one of the important problems in renewable energy systems because of dilute and fluctuating nature of wind.

This study ultimate aim is developed the recursive radial basis function networks model for application of wind speed forecasting and to estimate the optimal hidden neuron numbers in recursive radial basis function networks based on the different 150 criteria, where these criteria are framed as a function of n (input layer neurons). The single appropriate topology is utilized. The proposed model confirms that even though large hidden neuron numbers in the recursive radial basis function networks get stable performance on training. Features of the proposed approach are discussed here. The recursive radial basis function networks superior to radial basis function networks and back propagation feed-forward networks because it does not require much training time, faster rate of convergence, avoid local mini ma problem, simple and compact. The proposed novel recursive radial basis function network with single hidden layer is sufficient to solve any continuous function with reduced complexity and suitable for various applications such as prediction, pattern recognition and function approximation.

**Description of recursive radial basis function networks:** The proposed novel recursive radial basis function network is a multi-layer feed-forward networks. Recursive radial basis function networks consist of three layer such as input layer, hidden layer and output layer. In the case of novel recursive radial basis function networks weights are recursively updated in order to make the least output error. Input layer outputs are evolved based on the distance between the inputs and hidden layer centers. Input layer outputs are transformed to the hidden layer as a nonlinear form. The hidden layer has a large dimension because all input layer neurons are linked directly to the hidden layer. Each hidden neurons of hidden layer has parameters such as width and center place. In recursive radial basis function networks each and every hidden neuron in hidden layer has Gaussian activation function. The Gaussian recursive radial basis function is fine-tuned by means of the spread value adjustment, i.e., shape the Gaussian recursive radial basis function curve. At zero distance the Gaussian activation function curve has a peak value and further minimizes as distance from the center increase. The outputs of the hidden layer are weighted form of the input layer output which is transformed to the output layer as a linear form. Weights are updated recursively in order to achieve the minimum output error. The weight updating process uses the gradient descent rule. The objective of the suggested approach is to select the appropriate hidden layer neuron numbers in order to obtain the faster convergence and better accuracy with minimal statistical error.

**Design structure of Recursive radial basis function network:** Neural network designing process plays a key role to get better network performance. Proposed recursive radial basis function networks based wind speed forecasting constructed with three inputs neurons such as wind speed ($N_w$), wind direction ($WD_w$) and temperature ($T_w$), one hidden layer, one output neuron (forecast wind speed) and hidden neuron numbers in hidden layer is decided from the proposed 150 various criteria based on the lowest minimal error. The design

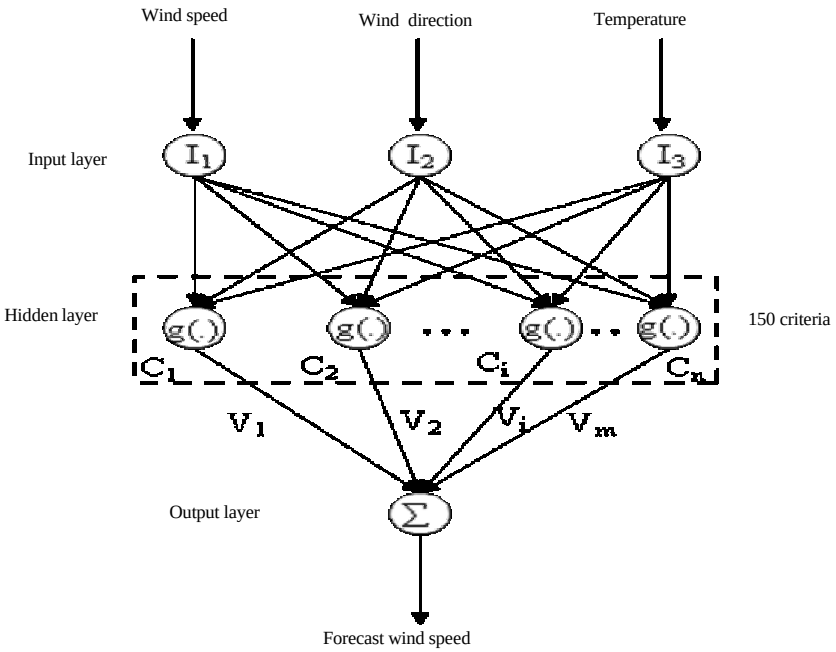


Fig. 1: Proposed recursive radial basis function networks design structure

Table 1: Input and output variables of proposed forecasting model

| Input variables | Description | Output variable | Description |
|---|---|---|---|
| $I_1$ | Wind Speed ($N_w$) | O | Forecast wind speed ($N_{fw}$) |
| $I_2$ | Wind Direction ($WD_w$) | | |
| $I_3$ | Temperature ($T_W$) | | |

Table 2: Recursive radial basis function networks design parameters

| Recursive radial basis function networks | |
|---|---|
| Parameters | Parametric values |
| Input neurons | $3[N_W, WD_W, T_W]$ |
| Number of hidden layer | 1 |
| Hidden neuron numbers | Estimated based on 150 criteria |
| Output neuron | $1[N_{fw}]$ |
| Number of epochs | 2000 |
| Spread | 2.5 |

structure of the proposed recursive radial basis function networks model for estimating the hidden neuron numbers is shown in Fig. 1. Figure 1 inferred input and output target vector pairs are described in Table 1.

In neural network each layer performs the independent computations on received data and the computed results are transferred to the next layer and lastly network output is computed this information is inferred from Fig. 1. The new criteria are used to estimate hidden layer neuron numbers in network design and the proposed model is adopted for wind speed forecasting application.

The proposed recursive radial basis function networks designed parameters includes dimensions and epochs shown in Table 2. The dimensions such as input neuron numbers, hidden neuron numbers, output neuron numbers are defined in the network design. Input layer and hidden layer are interlinked by means of the hypothetical connection. Hidden layer has Gaussian function. Hidden layer and output layer are interlinked by means of the weighted connections. In recursive radial basis function networks weights updating process performed recursively in order to improve the convergence rate and achieve minimal error. The output layer has linear function. Training learns from the normalized data. Error coming to a very negligible value is a stopping condition for testing process.

**Mathematical modeling:**

Input vector:

$$I = [N_w, WD_w, T_w] \quad (1)$$

Output vector:

$$O = [N_{fw}] \quad (2)$$

Weight vectors of hidden to output vector:

$$V = [V_1, V_2, ..., V_m] \quad (3)$$

Gaussian activation function:

Table 3: Collected actual input data samples (from Suzlon Energy Pvt. Ltd)

| Temperature (°C) | Wind direction (Degree) | Wind speed (m $sec^{-1}$) | Temperature (°C) | Wind direction (Degree) | Windspeed(m $sec^{-1}$) |
|---|---|---|---|---|---|
| 26.4 | 285.5 | 8.90 | 24.1 | 77.30 | 2.9 |
| 25.9 | 285.5 | 8.60 | 24.1 | 83.00 | 1.1 |
| 25.8 | 279.8 | 7.70 | 28.4 | 83.00 | 6.4 |
| 26.1 | 286.9 | 6.90 | 26.8 | 105.5 | 7.6 |
| 30.4 | 298.1 | 6.80 | 26.1 | 101.2 | 5.6 |
| 32.4 | 277.0 | 5.90 | 25.8 | 122.3 | 4.0 |
| 27.5 | 315.0 | 3.80 | 24.1 | 184.2 | 0.4 |
| 26.6 | 299.5 | 1.90 | 25.8 | 336.1 | 2.4 |
| 25.2 | 112.5 | 9.20 | 24.5 | 26.70 | 5.3 |
| 26.4 | 111.1 | 15.9 | 23.2 | 157.5 | 2.4 |

$$f(Z_{in}) = e^{(-Z_{in}^2)} \quad (4)$$

where, $Z_{in}$ = net input. Output of recursive radial basis function networks:

$$Z_{in} = \sum_{i=1}^{n} f\left(\left\|I - C_i\right\| * V_{ik}\right), \quad k = 1, 2, ..., m \quad (5)$$

Where:

n = Number of hidden neurons.
I = Input vector
$C_i$ = ith center node in hidden layer
$\|I-C_i\|$ = Euclidean distance between $C_i$ and I
f = Gaussian activation function
$V_{ik}$ = Weights between hidden and output layer

**Numerical implementation of proposed networks:** The wind speed forecasting using neural networks associated with modeling, training and testing process. Accurate neural networks model development is a complex and a challenging task. The input parameters used for the recursive radial basis function networks are real-time data. Hence, normalization process is performed due to overcome training process problem of large valued input data tend to reduce the effect of smaller value input data. The min-max normalization technique is adapted to normalize the different range of real-time data in the range of 0 to 1. The normalization process helps to improve the numerical computational accuracy, so the recursive radial basis function networks model accuracy is also enhanced.

Estimation of proper hidden layer neuron numbers in neural networks is a challenging task, random selection of hidden neuron numbers lead to over fitting or under fitting problem. Therefore, new 150 criteria are proposed in order to select right hidden layer neuron numbers in neural networks. Presented 150 criteria are verified by means of the convergence theorem. The appropriate hidden neuron numbers in hidden layer is estimated by means of the lowest minimal error.

**Data collection and site description:** The real-time data was collected from Coimbatore site with various windmill heights such as 65 and 50 m height at 10 sec time intervals from Suzlon Energy Private Limited from January 2012 to December 2014. First real-time data set is collected from Coimbatore site wind farm with 65 m windmill height, which contains 10,000 data samples. Second real-time data set is collected from Coimbatore site wind farm with 50 m windmill height which consists of 10,000 data samples. Table 3 depicts collected actual data sample inputs. The proposed recursive radial basis function networks based wind speed forecasting model is developed by means of two real-time data sets; each data set contain 10,000 numbers of samples.

**Normalization:** Scaling (or) Normalization is very important for dealing with real-time data samples; the real-time data samples have different range and different units. Hence, the normalization is used to normalize the real-time data within the range of 0-1. The proposed model utilizes the min-max normalization method. Following transformation equation is used to normalize the real-time data samples.

**Normalized input:**

$$I_i' = \left(\frac{I_i - I_{min}}{I_{max} - I_{min}}\right)\left(I'_{max} - I'_{min}\right) + I'_{min} \quad (6)$$

Where:

$I_i$ = Actual input data
$I_{min}$ = Minimum input data
$I_{max}$ = Maximum input data
$I'_{min}$ = Minimum target value
$I'_{max}$ = Maximum target value

**Criteria for determining hidden layer neuron numbers:** Suggested 150 different criteria are satisfied the convergence theorem requirement. All chosen criteria were examined in recursive radial basis function networks in order to estimate the networks training process and statistical errors.

The development of the proposed model is initiated by employing the chosen criteria to the recursive radial basis function networks. After the development process train the recursive radial basis function networks and compute the statistical errors. Estimation of right hidden layer neuron numbers is fixed based on the reduced statistical error.

**Training and testing of the network performance:** The wind speed forecasting model based on recursive radial basis function networks is developed using training data set while the performance of the proposed model is evaluated by using the testing data set. The collected real-time data are initially classified into the training and testing set. Training set is used in neural network learning and testing set is used to compute the error. Collected 10,000 real-time data is classified in to training and testing sets. Collected 70% of data samples (7000) are used for training phase and 30% of the collected data samples (3000) are used for testing phase of the proposed networks. The considered 150 various criteria are applied to recursive radial basis function networks one by one without changing networks design parameters and check performance is accepted (or) not. The network performance is calculated based on the statistical error criteria such as MSE, MAE and MRE. Equation (7-9) represents the statistical error formulas. During training process of the network the generalization performance differ overtime.

## RESULTS AND DISCUSSION

The suggested criteria are considered in this study is used to building three layer neural networks. The proposed recursive radial basis function networks is developed to run on an Acer laptop computer with Pentium (R) Dual Core processor running at 2.30 GHZ with 2 GB of RAM. Networks performance is evaluated based on the statistical error calculation. The experimental results confirm with minimum error is determined as the best solution for selecting hidden layer neuron numbers in recursive radial basis function networks.

**Performance metric for forecast accuracy:** The implemented recursive radial basis function networks based wind speed forecasting model performance is analyzed based on the performance metric such as Mean Absolute Error (MAE), Mean Relative Error (MRE) and Mean Square Error (MSE). From the different 150 criteria, the best hidden neuron numbers is selected based on the minimal error performance. The statistical errors are used to measure the quality of forecast wind speed obtained by neural network. Statistical error criteria formulas are given as follows:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(T_i' - T_i)^2 \quad (7)$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}(T_i' - T_i) \quad (8)$$

$$MRE = \frac{1}{N}\sum_{i=1}^{N}\left|(T_i' - T_i)/\overline{T_i}\right| \quad (9)$$

Where:

$N$ = The number of samples
$T_i'$ = The actual output
$\overline{T_i}$ = The average actual output
$T_i$ = The forecast output

**Statistical analysis of computed wind speed and error factors:** Evaluating proper hidden layer neuron numbers in novel recursive radial basis function networks among from considered 150 different criteria, the experiments have performed on two real-time data sets. Computed statistical errors for the proposed forecasting model are established in Table 4. From the Table 4 it can be noted that the appropriate hidden layer neuron numbers for recursive radial basis function networks is estimated as 97 based on the proposed new criteria $N_h = [(2(5n^2+3)+1)/(n^2-8)]$. Proposed recursive radial basis function network with 97 hidden neuron numbers is achieved minimal statistical errors compared to other considered criteria it can be inferred from experiment on two real-time data sets. The experimentation on data samples collected from 65 m windmill height confirms that the 97 hidden neurons estimated by means of the criteria $N_h = [(2(5n^2+3)+1)/(n^2-8)]$ achieve minimal statistical errors MSE of 7.6315e-13, MAE of 5.6554e-07 and MRE of 6.9806e-08. The experimentation on data samples collected from 50 m windmill height revealed that the 97 hidden neurons estimated by means of the criteria $N_h = [(2(5n^2+3)+1)/(n^2-8)]$ achieve much minimal statistical errors MSE of 5.1959e-13, MAE of 2.5817-07 and MRE of 3.4229e-08. Comparison with experimentation on two real-time data sets, the experimentation on data samples collected from 50 m windmill height proves with the least statistical errors. Therefore, the proposed criteria improve the effectiveness and accuracy for recursive radial basis function network based wind speed forecasting.

Based on the proposed forecasting model experimental result comparison between the actual and forecast wind speed (50 m height windmill) for 3,000 samples is shown in Fig. 2, relationship between

Table 4: Recursive radial basis function networks statistical analysis of different criteria for estimating hidden layer neuron numbers

| Hidden neuron numbers | Considered criteria for determining 3 inputs networks hidden neurons | Data set 1 | | | Data set 2 | | |
|---|---|---|---|---|---|---|---|
| | | MSE | MAE | MRE | MSE | MAE | MRE |
| 6 | 2n/(n-2) | 1.2083e-10 | 5.3726e-06 | 6.9102e-07 | 2.2742e-10 | 9.8362e-06 | 1.3614e-06 |
| 134 | $(8(n^2+7)+6)/(n^2-8)$ | 9.2971e-07 | 4.5566e-04 | 5.8607e-05 | 4.4276e-07 | 1.5972e-04 | 2.1176e-05 |
| 18 | (4n+6)/(n-2) | 1.0501e-10 | 5.0433e-06 | 6.4867e-07 | 1.6281e-10 | 8.2770e-06 | 1.1456e-06 |
| 113 | $(8(n^2+5)+2)/(n^2-8)$ | 1.2952e-10 | 5.7588e-06 | 7.4069e-07 | 2.3545e-10 | 1.0240e-05 | 1.4172e-06 |
| 24 | (7n+3)/(n-2) | 1.2602e-10 | 5.3665e-06 | 6.9024e-07 | 1.6361e-10 | 8.2986e-06 | 1.1486e-06 |
| 145 | $(8(n^2+9)+1)/(n^2-8)$ | 0.0273 | 0.1264 | 0.0153 | 0.0280 | 0.0974 | 0.0129 |
| 39 | (11n+6)/(n-2) | 6.8661e-06 | 0.0012 | 1.5461e-04 | 5.6724e-06 | 0.0013 | 1.8597e-04 |
| 128 | $(8(n^2+7))/(n^2-8)$ | 0.0024 | 0.0275 | 0.0033 | 0.0010 | 0.0220 | 0.0029 |
| 82 | $(6(n^2+4)+4)/(n^2-8)$ | 7.2482e-05 | 0.0053 | 6.3833e-04 | 5.0668e-07 | 3.9226e-04 | 5.4290e-05 |
| 100 | $(8(n^2+3)+4)/(n^2-8)$ | 1.4254e-10 | 5.2661e-06 | 6.7732e-07 | 4.5577e-10 | 1.3801e-05 | 1.9101e-06 |
| 46 | $(4n^2+10)/(n^2-8)$ | 3.6197e-08 | 2.9994e-05 | 3.7022e-06 | 1.3275e-08 | 2.1791e-05 | 2.8892e-06 |
| 73 | $(4(n^2+7)+9)/(n^2-8)$ | 1.1979e-10 | 5.1958e-06 | 6.6828e-07 | 1.1837e-10 | 8.9164e-06 | 1.2341e-06 |
| 107 | $(7(n^2+6)+2)/(n^2-8)$ | 1.1444e-10 | 4.9672e-06 | 6.3887e-05 | 8.9551e-10 | 1.7558e-05 | 2.4300e-06 |
| 55 | $(4(n^2+5)-1)/(n^2-8)$ | 4.6430e-10 | 9.2212e-06 | 1.1860e-06 | 4.9342e-10 | 1.4367e-05 | 1.9884e-06 |
| 85 | $(6(n^2+5)+1)/(n^2-8)$ | 1.1866e-08 | 5.5245e-05 | 7.1056e-06 | 3.9394e-08 | 1.2522e-04 | 1.7331e-05 |
| 119 | $(7(n^2+8))/(n^2-8)$ | 1.2083e-10 | 5.3728e-06 | 6.9104e-07 | 3.8460e-09 | 1.7759e-05 | 2.3546e-06 |
| 50 | $(4(n^2+4)-2)/(n^2-8)$ | 4.7313e-04 | 0.0142 | 0.0018 | 2.5660e-04 | 0.0112 | 0.0015 |
| 65 | $(4(n^2+6)+5)/(n^2-8)$ | 1.0678e-07 | 1.8021e-04 | 2.3179e-05 | 1.6209e-07 | 2.3875e-04 | 3.3044e-05 |
| 26 | (7n+5)/(n-2) | 8.2372e-11 | 4.6618e-06 | 5.9960e-07 | 8.6253e-11 | 5.7373e-06 | 7.9407e-07 |
| 96 | $(6(n^2+7))/(n^2-8)$ | 9.0080e-10 | 1.1002e-05 | 1.4150e-06 | 8.1856e-10 | 1.7042e-05 | 2.3586e-06 |
| 124 | $(8(n^2+6)+4)/(n^2-8)$ | 1.0501e-10 | 5.0429e-06 | 6.4861e-07 | 6.8711e-10 | 1.8326e-05 | 2.4298e-06 |
| 59 | $(4(n^2+4)+7)/(n^2-8)$ | 1.5100e-10 | 6.1244e-06 | 7.8772e-07 | 1.9687e-10 | 9.3461e-06 | 1.2935e-06 |
| 3 | 4n/(n+1) | 0.0314 | 0.1277 | 0.0164 | 0.0558 | 0.1538 | 0.0200 |
| 140 | $(8(n^2+8)+4)/(n^2-8)$ | 9.5274e-11 | 4.9085e-06 | 6.3133e-07 | 8.6253e-11 | 5.7379e-06 | 7.9415e-07 |
| 88 | $(7(n^2+3)+4)/(n^2-8)$ | 9.6024e-07 | 4.8437e-04 | 6.2299e-05 | 4.8989e-07 | 4.0330e-04 | 5.5818e-05 |
| 132 | $(8(n^2+7)+4)/(n^2-8)$ | 1.3299e-10 | 5.9495e-06 | 7.6522e-07 | 4.6496e-10 | 7.8083e-06 | 1.0353e-06 |
| 89 | $(6(n^2+5)+5)/(n^2-8)$ | 3.5069e-07 | 2.9791e-04 | 3.8317e-05 | 2.3969e-07 | 2.8864e-04 | 3.9948e-05 |
| 118 | $(8(n^2+5)+6)/(n^2-8)$ | 4.6142e-07 | 3.6076e-04 | 4.6401e-05 | 3.0617e-07 | 3.1836e-04 | 4.4063e-05 |
| 5 | (3n+1)/(n-1) | 0.0173 | 0.0971 | 0.0125 | 0.0186 | 0.1051 | 0.0136 |
| 48 | $(4(n^2+2)+4)/(n^2-8)$ | 6.8959e-08 | 1.4881e-04 | 1.9139e-05 | 6.2725e-08 | 1.4537e-04 | 2.0120e-05 |
| 74 | $(4(n^2+8)+6)/(n^2-8)$ | 1.3867e-08 | 6.8005e-05 | 8.7468e-06 | 3.6147e-08 | 1.1250e-04 | 1.5571e-05 |
| 126 | $(8(n^2+7)-2)/(n^2-8)$ | 0.0026 | 0.0310 | 0.0037 | 0.0032 | 0.0427 | 0.0057 |
| 98 | $(7(n^2+5))/(n^2-8)$ | 1.4308e-10 | 4.9484e-06 | 5.8589e-07 | 4.7272e-10 | 1.3909e-05 | 1.9251e-06 |
| 20 | (6n+2)/(n-2) | 1.1231e-06 | 5.3271e-04 | 6.8517e-05 | 2.2145e-06 | 7.9545e-04 | 1.1009e-04 |
| 149 | $(8(n^2+9)+5)/(n^2-8)$ | 0.0047 | 0.0444 | 0.0054 | 0.0050 | 0.0544 | 0.0072 |
| 57 | $(4(n^2+4)+5)/(n^2-8)$ | 4.7505e-09 | 3.2718e-05 | 4.2082e-06 | 4.8799e-08 | 1.4525e-05 | 2.0104e-05 |
| 105 | $(7(n^2+6))/(n^2-8)$ | 0.0287 | 0.1318 | 0.0169 | 0.0166 | 0.1002 | 0.0130 |
| 69 | $(4(n^2+7)+5)/(n^2-8)$ | 1.4886e-08 | 6.6641e-05 | 8.5713e-06 | 5.6481e-08 | 1.4938e-04 | 2.0674e-05 |
| 52 | $(4(n^2+3)+4)/(n^2-8)$ | 2.1381e-05 | 0.0034 | 4.4306e-04 | 2.8881e-06 | 9.7266e-04 | 1.3462e-04 |
| 75 | $(6(n^2+3)+3)/(n^2-8)$ | 4.9467e-08 | 1.3160e-04 | 1.6927e-05 | 4.5780e-08 | 1.3790e-04 | 1.9086e-05 |
| 14 | (9n+1)/(n-1) | 1.4257e-10 | 6.0070e-06 | 7.7262e-07 | 1.9804e-10 | 9.4212e-06 | 1.3039e-06 |
| 139 | $(9(n^2+6)+4)/(n^2-8)$ | 3.5571 | 1.6104 | 0.2071 | 2.4931 | 1.3558 | 0.1760 |
| 1 | 2n/(n+3) | 3.5571 | 1.6104 | 0.2071 | 2.4931 | 1.3558 | 0.1760 |
| 67 | $(4(n^2+6)+7)/(n^2-8)$ | 1.5828e-08 | 6.8276e-05 | 8.7816e-06 | 5.7936e-08 | 1.4583e-04 | 2.0183e-05 |
| 36 | $(3n^2+9)/(n^2-8)$ | 7.8889e-10 | 1.3704e-05 | 1.7626e-06 | 8.1857e-10 | 1.7042e-05 | 2.3586e-06 |
| 121 | $(7(n^2+8)+2)/(n^2-8)$ | 1.0314e-04 | 0.0068 | 8.1939e-04 | 3.1774e-04 | 0.0013 | 1.6581e-04 |
| 28 | $(2n^2+10)/(n^2-8)$ | 8.2778e-06 | 0.0016 | 2.0466e-04 | 4.0868e-06 | 0.0013 | 1.7521e-04 |
| 9 | 3n/(n-2) | 1.2871e-10 | 5.6239e-06 | 7.2335e-07 | 2.1342e-10 | 9.2840e-06 | 1.2849e-06 |
| 144 | $(9(n^2+7))/(n^2-8)$ | 6.2045e-07 | 3.9838e-04 | 5.1240e-05 | 2.8439e-07 | 3.0260e-04 | 4.1880e-05 |
| 80 | $(8(n^2+1))/(n^2-8)$ | 1.6859e-10 | 6.9499e-06 | 8.9389e-07 | 2.4975e-10 | 1.0486e-05 | 1.4512e-06 |
| 133 | $(7(n^2+9)+7)/(n^2-8)$ | 9.3267e-07 | 4.8635e-04 | 6.2554e-05 | 1.1293e-06 | 2.8098e-04 | 3.7254e-05 |
| 84 | $(7(n^2+3))/(n^2-8)$ | 1.3931e-08 | 6.8755e-05 | 8.8433e-06 | 1.3436e-07 | 2.1238e-04 | 2.9394e-05 |
| 150 | $(9(n^2+7)+6)/(n^2-8)$ | 1.4413e-10 | 5.8859e-06 | 7.5704e-07 | 4.9600e-10 | 1.5439e-05 | 2.0470e-06 |
| 25 | (9n-2)/(n-2) | 1.7786e-10 | 6.6619e-06 | 8.5685e-07 | 1.7154e-10 | 8.4518e-06 | 1.1698e-06 |
| 104 | $(8(n^2+4))/(n^2-8)$ | 3.5571 | 1.6104 | 0.2071 | 2.4931 | 1.3558 | 0.1760 |
| 37 | (9n+10)/(n-2) | 5.9654e-10 | 1.0283e-05 | 1.3226e-06 | 6.9907e-10 | 1.5342e-05 | 2.1234e-06 |
| 60 | $(5(n^2+2)+5)/(n^2-8)$ | 6.5638e-09 | 4.1820e-05 | 5.3788e-06 | 2.0324e-08 | 7.4693e-05 | 1.0338e-05 |
| 127 | $(7(n^2+9)+1)/(n^2-8)$ | 4.6537e-07 | 3.7539e-04 | 4.8282e-05 | 3.3559e-07 | 1.5400e-04 | 2.0419e-05 |
| 19 | (7n-2)/(n-2) | 7.6032e-11 | 4.4057e-06 | 5.6666e-07 | 8.6250e-11 | 5.7381e-06 | 7.9417e-07 |
| 95 | $(6(n^2+6)+5)/(n^2-8)$ | 4.2535e-07 | 3.5483e-04 | 4.5638e-05 | 4.6891e-07 | 4.0692e-04 | 5.6319e-05 |
| 49 | $(4(n^2+3)+1)/(n^2-8)$ | 0.0036 | 0.0367 | 0.0047 | 0.0031 | 0.0385 | 0.0050 |
| 64 | $(4(n^2+7))/(n^2-8)$ | 7.1052e-06 | 0.0010 | 1.3107e-04 | 1.8724e-06 | 7.4890e-04 | 1.0365e-04 |
| 56 | $(4(n^2+5))/(n^2-8)$ | 1.1940e-10 | 5.3462e-06 | 6.8762e-07 | 1.6726e-10 | 8.4364e-06 | 1.1676e-06 |
| 8 | (3n-1)/(n-2) | 6.6538e-05 | 0.0016 | 1.9415e-04 | 1.0728e-05 | 6.4206e-04 | 7.9252e-05 |

Table 4: Continue

| Hidden neuron numbers | Considered criteria for determining 3 inputs networks hidden neurons | Data set 1 | | | Data set 2 | | |
|---|---|---|---|---|---|---|---|
| | | MSE | MAE | MRE | MSE | MAE | MRE |
| 143 | $(8(n^2+8)+7)/(n^2-8)$ | 0.0066 | 0.0506 | 0.0061 | 0.0080 | 0.0678 | 0.0090 |
| 91 | $(6(n^2+6)+1)/(n^2-8)$ | 1.5602e-09 | 2.2331e-05 | 2.8722e-06 | 2.2779e-09 | 2.7563e-05 | 3.8148e-06 |
| 30 | $(3n^2+3)/(n^2-8)$ | 8.7292e-11 | 4.6900e-06 | 6.0322e-07 | 9.9821e-11 | 6.3262e-06 | 8.7557e-07 |
| 77 | $(6(n^2+3)+5)/(n^2-8)$ | 1.1240e-06 | 4.9024e-04 | 6.3055e-05 | 5.4781e-07 | 3.8869e-04 | 5.3796e-05 |
| 122 | $(8(n^2+6)+2)/(n^2-8)$ | 1.1697e-10 | 5.3566e-06 | 6.8896e-07 | 1.3423e-09 | 2.1810e-05 | 3.0185e-06 |
| 42 | $(5n^2-3)/(n^2-8)$ | 1.4049e-05 | 0.0023 | 2.9711e-04 | 4.9671e-05 | 0.0052 | 6.7384e-04 |
| 110 | $(7(n^2+7)-2)/(n^2-8)$ | 3.2454e-05 | 0.0038 | 4.6311e-04 | 9.5261e-05 | 0.0014 | 1.8447e-04 |
| 13 | (8n+2)/(n-1) | 9.8711e-11 | 5.0858e-06 | 6.5413e-07 | 9.7176e-11 | 6.2378e-06 | 8.6334e-07 |
| 135 | $(7(n^2+9)+9)/(n^2-8)$ | 1.0468e-10 | 5.3874e-06 | 6.9293e-07 | 1.2402e-09 | 2.1362e-05 | 2.9566e-06 |
| 58 | $(4(n^2+5)+2)/(n^2-8)$ | 1.3299e-10 | 5.9489e-06 | 7.6514e-07 | 1.8419e-10 | 9.0026e-06 | 1.2460e-06 |
| 116 | $(8(n^2+5)+4)/(n^2-8)$ | 9.6423e-11 | 5.2674e-06 | 6.7749e-07 | 9.9819e-11 | 6.3262e-06 | 8.7557e-05 |
| 93 | $(6(n^2+6)+3)/(n^2-8)$ | 7.6612e-11 | 4.4852e-06 | 5.7688e-07 | 5.7067e-11 | 4.5017e-06 | 5.8438e-07 |
| 142 | $(8(n^2+9)-2)/(n^2-8)$ | 0.0042 | 0.0395 | 0.0048 | 0.0062 | 0.0537 | 0.0071 |
| 17 | (6n-1)/(n-2) | 7.7547e-11 | 4.5303e-06 | 5.8269e-07 | 8.6249e-11 | 5.7378e-06 | 7.9414e-07 |
| 32 | $(3n^2+5)/(n^2-8)$ | 9.3178e-10 | 1.4103e-05 | 1.8139e-06 | 2.0631e-10 | 4.0038e-06 | 5.3086e-07 |
| 129 | $(7(n^2+9)+3)/(n^2-8)$ | 1.2752e-10 | 5.4716e-06 | 7.0376e-07 | 4.5854e-10 | 9.7168e-06 | 1.2883e-06 |
| 76 | $(6(n^2+4)-2)/(n^2-8)$ | 1.3761e-08 | 6.6826e-05 | 8.5951e-06 | 1.7968e-08 | 7.1201e-05 | 9.8545e-06 |
| 108 | $(8(n^2+4)+4)/(n^2-8)$ | 8.4532e-11 | 4.7974e-06 | 6.1704e-07 | 8.6252e-11 | 5.7376e-06 | 7.9411e-07 |
| 23 | (6n+5)/(n-2) | 8.3579e-06 | 0.0016 | 2.0417e-04 | 1.8340e-05 | 0.0029 | 3.6998e-04 |
| 81 | $(7(n^2+2)+4)/(n^2-8)$ | 3.6066e-10 | 8.7042e-06 | 1.1195e-06 | 4.5578e-10 | 1.3801e-05 | 1.9101e-06 |
| 4 | (5n+1)/(n+1) | 1.2896e-10 | 5.6447e-06 | 7.2602e-07 | 2.9784e-10 | 1.1327e-05 | 1.5678e-06 |
| 61 | $(4(n^2+5)+5)/(n^2-8)$ | 2.4017e-09 | 2.4971e-05 | 3.2117e-06 | 3.4706e-09 | 3.2703e-05 | 4.5262e-06 |
| 53 | $(4(n^2+4)+1)/(n^2-8)$ | 6.6840e-06 | 0.0011 | 1.3923e-04 | 2.4713e-06 | 8.9012e-04 | 1.2320e-04 |
| 45 | $(9n^2+9)/(n^2-7)$ | 1.4960e-09 | 2.1341e-05 | 2.7449e-06 | 1.7917e-09 | 2.4952e-05 | 3.4534e-06 |
| 148 | $(9(n^2+7)+4)/(n^2-8)$ | 6.0190e-07 | 3.9979e-04 | 5.1421e-05 | 1.3898e-07 | 2.1342e-04 | 2.9538e-05 |
| 21 | 7n/(n-2) | 8.9046e-11 | 4.7974e-06 | 6.1704e-07 | 9.7423e-11 | 6.2253e-06 | 8.6161e-07 |
| 125 | $(7(n^2+9)-1)/(n^2-8)$ | 3.5571 | 1.6104 | 0.2071 | 2.4931 | 1.3558 | 0.1760 |
| 87 | $(6(n^2+5)+3)/(n^2-8)$ | 1.5178e-10 | 6.1521e-06 | 7.9129e-07 | 4.3200e-10 | 1.3324e-05 | 1.8441e-06 |
| 114 | $(7(n^2+6)+9)/(n^2-8)$ | 0.0016 | 0.0258 | 0.0031 | 0.0014 | 0.0286 | 0.0037 |
| 35 | $(4n^2-1)/(n^2-8)$ | 9.5319e-09 | 4.8073e-05 | 6.1831e-06 | 6.9161e-09 | 4.1699e-05 | 5.7713e-06 |
| 72 | $(4(n^2+8)+4)/(n^2-8)$ | 2.0472e-06 | 7.3737e-04 | 9.4840e-05 | 9.8844e-07 | 5.6531e-04 | 7.8242e-05 |
| 44 | $(5n^2-1)/(n^2-8)$ | 1.4061e-12 | 7.6798e-07 | 9.4794e-08 | 8.9927e-12 | 2.0200e-06 | 2.6782e-07 |
| 138 | $(8(n^2+8)+2)/(n^2-8)$ | 0.0053 | 0.0454 | 0.0055 | 0.0074 | 0.0047 | 6.2236e-04 |
| 11 | (4n-1)/(n-2) | 0.0022 | 0.0240 | 0.0031 | 0.0149 | 0.0936 | 0.0121 |
| 102 | $(6(n^2+8))/(n^2-8)$ | 1.2691e-10 | 5.5813e-06 | 7.1787e-07 | 4.6572e-10 | 1.3100e-05 | 1.8132e-06 |
| 92 | $(7(n^2+3)+8)/(n^2-8)$ | 9.7861e-07 | 4.8396e-04 | 6.2247e-05 | 2.4658e-07 | 2.8304e-04 | 3.9174e-05 |
| 12 | (7n+3)/(n-1) | 1.0753e-10 | 5.2254e-06 | 6.7208e-07 | 1.5811e-10 | 8.1855e-06 | 1.1329e-06 |
| 147 | $(8(n^2+9)+3)/(n^2-8)$ | 5.8067e-07 | 3.9410e-04 | 5.0688e-05 | 1.3690e-07 | 2.0988e-04 | 2.9048e-05 |
| 94 | $(6(n^2+7)-2)/(n^2-8)$ | 8.1461e-10 | 1.5709e-05 | 1.9390e-06 | 2.0328e-10 | 9.4973e-06 | 1.3145e-06 |
| 117 | $(7(n^2+7)+5)/(n^2-8)$ | 4.3138e-10 | 1.1579e-05 | 1.4010e-06 | 1.3953e-09 | 9.3098e-06 | 1.2344e-06 |
| 31 | (9n+4)/(n-2) | 1.1859e-10 | 5.3855e-06 | 6.9268e-07 | 1.8419e-10 | 9.0024e-06 | 1.2460e-06 |
| 71 | $(4(n^2+7)+7)/(n^2-8)$ | 1.6637e-05 | 0.0030 | 3.8774e-04 | 1.8714e-06 | 7.4827e-04 | 1.0356e-04 |
| 7 | (2n+1)/(n-2) | 0.0034 | 0.0390 | 0.0050 | 0.0036 | 0.0397 | 0.0052 |
| 83 | $(5(n^2+7)+3)/(n^2-8)$ | 1.0564e-04 | 0.0070 | 8.4422e-04 | 0.0012 | 0.0258 | 0.0033 |
| 136 | $(8(n^2+8))/(n^2-8)$ | 0.0032 | 0.0317 | 0.0038 | 0.0031 | 0.0327 | 0.0043 |
| 15 | (9n+3)/(n-1) | 7.5771e-11 | 4.4571e-06 | 5.7328e-07 | 1.7349e-10 | 8.3037e-06 | 1.1010e-06 |
| 109 | $(7(n^2+6)+4)/(n^2-8)$ | 5.4350e-07 | 3.8364e-04 | 4.9344e-05 | 4.7698e-07 | 4.1426e-04 | 5.7336e-05 |
| 66 | $(4(n^2+7)+2)/(n^2-8)$ | 1.4958e-08 | 6.6814e-05 | 8.5936e-06 | 5.7122e-08 | 1.5105e-04 | 2.0906e-05 |
| 40 | $(5n^2-5)/(n^2-8)$ | 1.0568e-09 | 1.5928e-05 | 2.0486e-06 | 2.8293e-09 | 2.9811e-05 | 4.1259e-06 |
| 54 | $(4(n^2+4)+2)/(n^2-8)$ | 1.5213e-10 | 6.3538e-06 | 8.1723e-07 | 1.8137e-10 | 8.9160e-06 | 1.2340e-06 |
| 123 | $(7(n^2+8)+4)/(n^2-8)$ | 1.0748e-10 | 5.2865e-06 | 6.7995e-07 | 1.1889e-09 | 2.0552e-04 | 2.8445e-06 |
| 78 | $(5(n^2+7)-2)/(n^2-8)$ | 0.0028 | 0.0327 | 0.0040 | 0.0014 | 0.0275 | 0.0036 |
| 112 | $(7(n^2+7))/(n^2-8)$ | 9.3003e-11 | 4.9095e-06 | 6.3146e-07 | 9.7424e-11 | 6.2254e-06 | 8.6162e-07 |
| 41 | (11n+8)/(n-2) | 5.8167e-10 | 9.8489e-06 | 1.2668e-06 | 5.3190e-10 | 1.4969e-05 | 2.0718e-06 |
| 27 | (8n+3)/(n-2) | 1.0920e-05 | 0.0019 | 2.4305e-04 | 1.9728e-05 | 0.0026 | 3.5977e-04 |
| 101 | $(6(n^2+8)-1)/(n^2-8)$ | 1.1053e-10 | 5.2106e-06 | 6.7019e-07 | 1.8418e-10 | 9.0021e-06 | 1.2459e-06 |
| 62 | $(4(n^2+6)+2)/(n^2-8)$ | 1.2098e-08 | 5.5756e-05 | 7.1713e-06 | 2.0940e-08 | 7.8072e-05 | 1.0805e-05 |
| 130 | $(8(n^2+7)+2)/(n^2-8)$ | 0.0025 | 0.0313 | 0.0040 | 0.0012 | 0.0259 | 0.0034 |
| 99 | $(6(n^2+7)+3)/(n^2-8)$ | 0.0029 | 0.0331 | 0.0043 | 0.0154 | 0.0946 | 0.0123 |
| 34 | (9n+7)/(n-2) | 1.2418e-10 | 5.4934e-06 | 7.0656e-07 | 2.8658e-10 | 5.3006e-06 | 7.0280e-07 |
| 141 | $(9(n^2+6)+6)/(n^2-8)$ | 9.8718e-11 | 5.0865e-06 | 6.5422e-07 | 9.9822e-11 | 6.3262e-06 | 8.7558e-07 |
| 146 | $(9(n^2+7)+2)/(n^2-8)$ | 0.0032 | 0.0339 | 0.0044 | 0.0037 | 0.0271 | 0.0036 |
| 33 | (8n+9)/(n-2) | 0.0172 | 0.0971 | 0.0125 | 0.0235 | 0.1221 | 0.0158 |

Table 4: Continue

| Hidden neuron numbers | Considered criteria for determining 3 inputs networks hidden neurons | Data set 1 | | | Data set 2 | | |
|---|---|---|---|---|---|---|---|
| | | MSE | MAE | MRE | MSE | MAE | MRE |
| 137 | $(9(n^2+6)+2)/(n^2-8)$ | 0.0045 | 0.0398 | 0.0048 | 0.0035 | 0.0340 | 0.0045 |
| 79 | $(4(n^2+9)+7)/(n^2-8)$ | 1.6292e-08 | 7.2830e-05 | 9.3674e-06 | 1.2467e-07 | 2.0933e-04 | 2.8972e-05 |
| 106 | $(8(n^2+4)+2)/(n^2-8)$ | 1.2129e-10 | 5.1435e-06 | 6.6156e-07 | 1.0350e-09 | 1.8980e-05 | 2.6270e-06 |
| 90 | $(7(n^2+4)-1)/(n^2-8)$ | 9.1207E-07 | 4.8173E-04 | 6.1960e-05 | 4.7620e-07 | 3.7718e-04 | 5.2203e-05 |
| 63 | $(4(n^2+7)-1)/(n^2-8)$ | 2.4080e-09 | 2.5976e-05 | 3.3410e-06 | 2.4822e-09 | 2.9129e-05 | 4.0316e-06 |
| 16 | (10n+2)/(n-1) | 0.0018 | 0.0262 | 0.0034 | 0.0017 | 0.0285 | 0.0039 |
| 120 | $(8(n^2+6))/(n^2-8)$ | 0.0024 | 0.0277 | 0.0033 | 0.0063 | 0.0395 | 0.0052 |
| 2 | 4n/(n+3) | 0.1375 | 0.2978 | 0.0383 | 0.144 | 0.2631 | 0.0342 |
| 111 | $(8(n^2+5)-1)/(n^2-8)$ | 1.2602e-10 | 5.3664e-06 | 6.9022e-07 | 5.6137e-10 | 6.4726e-06 | 8.5819e-07 |
| 70 | $(4(n^2+8)+2)/(n^2-8)$ | 1.0729e-06 | 4.9339e-04 | 6.3460e-05 | 1.2201e-06 | 5.6691e-04 | 7.8462e-05 |
| 29 | (9n+2)/(n-2) | 3.5571 | 1.6104 | 0.2071 | 2.4931 | 1.3558 | 0.176 |
| 43 | (11n+10)/(n-2) | 8.7200e-10 | 1.5122e-05 | 1.9450e-06 | 8.9532e-10 | 1.7020e-05 | 2.3556e-06 |
| 51 | $(4(n^2+3)+3)/(n^2-8)$ | 6.4465e-06 | 0.0011 | 1.4207e-04 | 2.9611e-06 | 9.3694e-04 | 1.2968e-04 |
| 38 | (10n+8)/(n-2) | 5.0684e-10 | 9.4646e-06 | 1.2173e-06 | 5.5484e-10 | 1.4630e-05 | 2.0249e-06 |
| 22 | (7n+1)/(n-2) | 8.2422e-11 | 4.6612e-06 | 5.9952e-07 | 9.7421e-11 | 6.2253e-06 | 8.6161e-07 |
| 68 | $(4(n^2+8))/(n^2-8)$ | 3.6261e-08 | 1.1061e-04 | 1.4226e-05 | 8.8446e-08 | 1.7980e-04 | 2.4884e-05 |
| 97 | $(2(5n^2+3)+1)/(n^2-8)$ | 7.6315e-13 | 5.6554e-07 | 6.9806e-08 | 5.1959e-13 | 2.5817e-07 | 3.4229e-08 |
| 115 | $(7(n^2+7)+3)/(n^2-8)$ | 0.0016 | 0.0251 | 0.003 | 0.0011 | 0.0058 | 7.1050e-04 |
| 47 | $(4(n^2+1)+7)/(n^2-8)$ | 1.4530e-05 | 0.0025 | 3.2647e-04 | 1.5506e-04 | 0.0087 | 0.0011 |
| 131 | $(7(n^2+9)+5)/(n^2-8)$ | 0.0032 | 0.0346 | 0.0041 | 0.002 | 0.0256 | 0.0034 |
| 10 | (3n+1)/(n-2) | 0.0059 | 0.0551 | 0.0068 | 0.0093 | 0.0481 | 0.0064 |
| 103 | $(6(n^2+7)+7)/(n^2-8)$ | 1.4446e-10 | 5.2961e-06 | 6.8118e-07 | 6.9908e-10 | 1.5343e-05 | 2.1235e-06 |
| 86 | $(7(n^2+3)+2)/(n^2-8)$ | 2.2585e-07 | 2.8276e-04 | 3.6368e-05 | 4.5820e-07 | 4.1012e-04 | 5.6763e-05 |

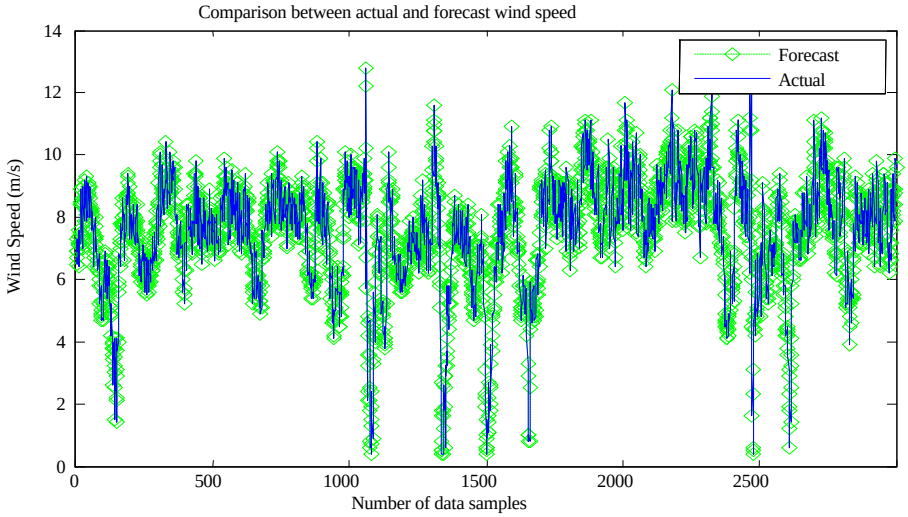


Fig. 2: Comparison between actual and forecast wind speed (50 m height windmill)

actual and forecast wind speed (50 m height windmill) is noticed from Fig. 3 and forecasting error (50 m height windmill) is shown in Fig. 4. Similarly comparison between the actual and forecast wind speed (65 m height windmill) for 3,000 samples is shown in Fig. 5, relationship between actual and forecast wind speed (65 m height windmill) is shown in Fig. 6 and forecasting error (65 m height windmill) is depicted in Fig. 7. Comparison of proposed recursive radial basis function networks based on statistical error such as MSE, MAE and MRE vs. hidden layer neuron numbers is shown in Fig. 8. Presented approach merits are very effective, minimum statistical error and simple implementation for wind speed forecasting. Problems of appropriate hidden neuron numbers for a particular problem are to be selected. The suggested approach was

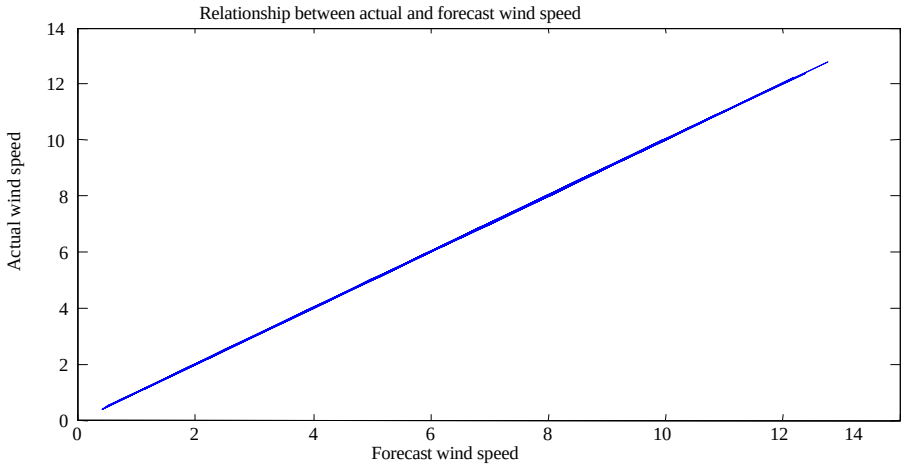


Fig. 3: Relationship between actual and forecast wind speed (50 m height windmill)

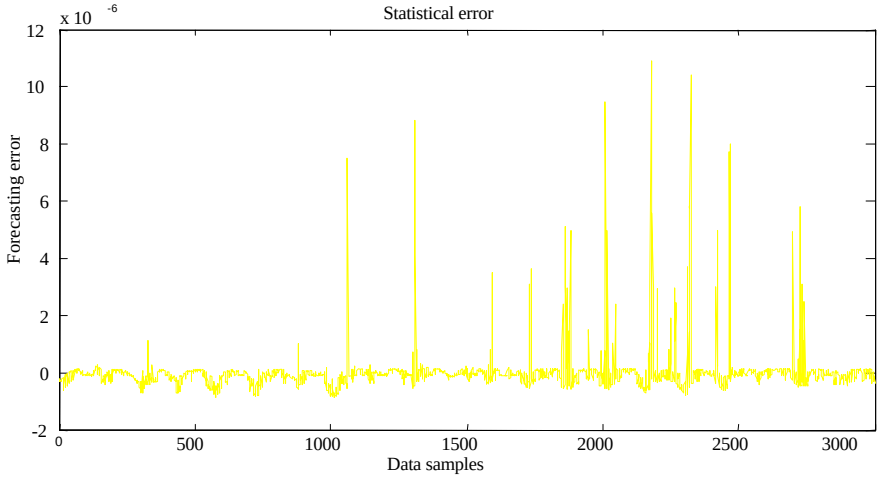


Fig. 4: Forecasting error (50 m height windmill)

simulated using MATLAB and results prove with minimal statistical errors such as MSE of 5.1959e-13, MAE of 2.5817e-07 and MRE of 3.4229e-08 for real-time data samples (50 m height windmill).

Earlier approaches use trial and error rule to determine hidden neuron numbers in neural networks. This starts network with undersized hidden neuron numbers and neurons are added to $N_h$. Demerits of earlier approach are there are no guarantees of selecting the number of hidden neurons and it consumes high computational time. Therefore, this study presented new criteria to avoid both either under fitting and over fitting problems and chosen criteria adopted to recursive radial basis function network for wind speed forecasting application is $[2(5n^2+3)+1)/(n^2-8)$ which utilized 97 hidden layer neuron numbers and achieved a reduced Mean Square Error (MSE) value of 5.1959e-13 in comparison with other criteria.

The simulation results are proved that the forecast wind speed is in the best agreement with the experimental measured values. Comparative analysis is performed on the earlier approaches with presented new criteria and results revealed that the proposed approach made the best results than that of other existing approaches. Table 5 depicts that compared to the other existing model the suggested model achieves better minimal statistical errors.

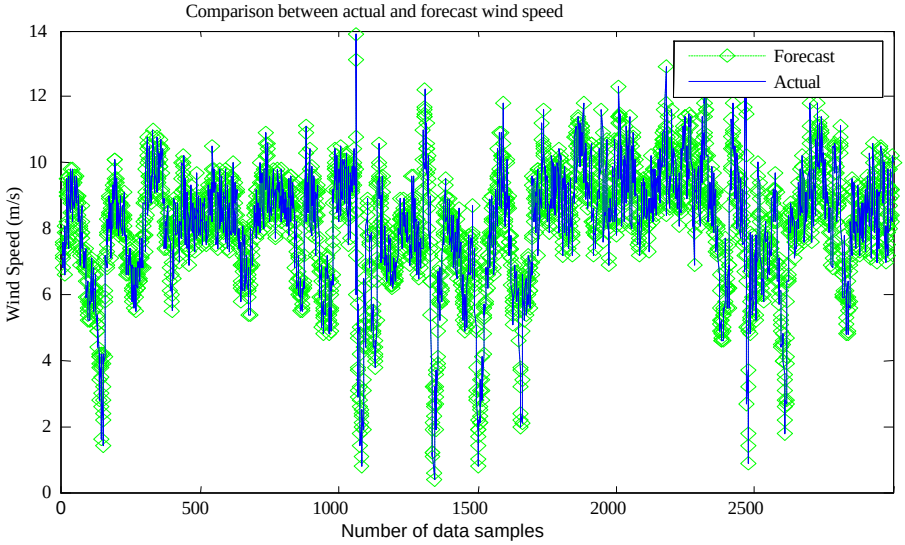


Fig. 5: Comparison between actual and forecast wind speed (65 m height windmill)

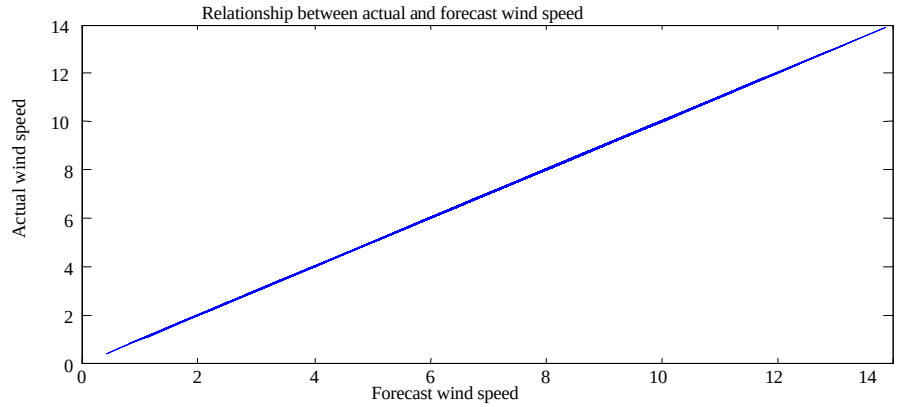


Fig. 6: Relationship between actual and forecast wind speed (65 m height windmill)

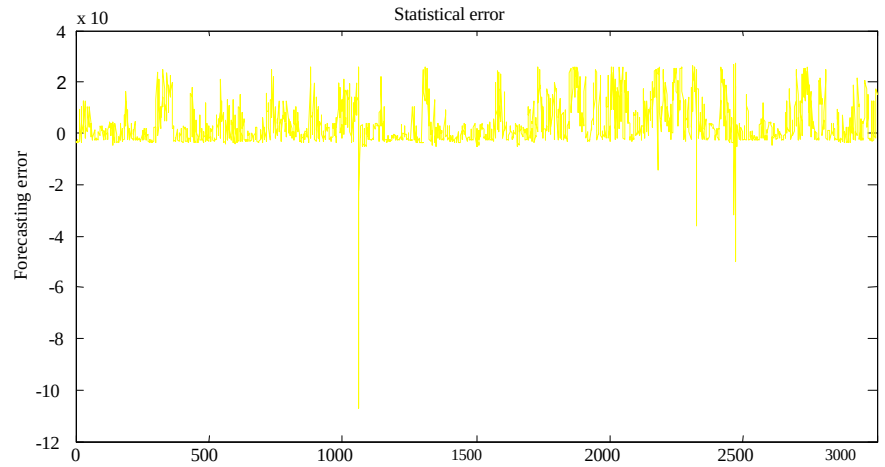


Fig. 7: Forecasting error (65 m height windmill)

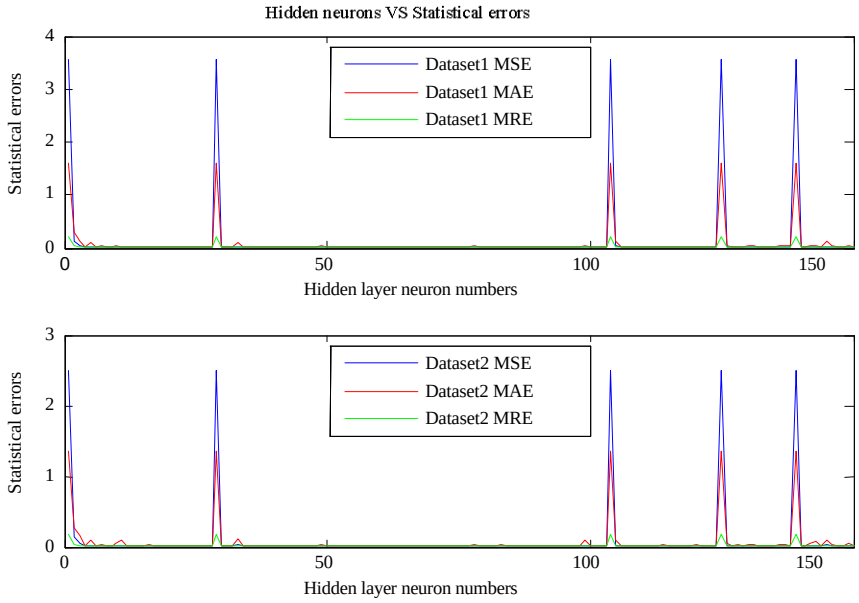


Fig. 8: Comparison of statistical errors vs. hidden layer neuron numbers

Table 5: Comparative analysis of different existing approaches performance with proposed approach

| Hidden neuron numbers | Years | Different approaches | Statistical Error (MSE) |
|---|---|---|---|
| $N_h = 2^n/3$ | 1993 | M. Arai Approach | 0.0314 |
| $N_h = \left(\sqrt{1+8n-1}\right)/2$ | 1995 | Jin-Yan Li *et al.* Approach | 0.1375 |
| $\mathbf{N_h = N-1}$ | 1997 | S. Tamura, M. Tateishi Approach | 0.1375 |
| $N_h = K\log\left(\left\|P_v^{(0)}Z\right\|/C\right)/\log S$ | 1998 | Osamu Fujita Approach | 1.2896e-10 |
| $N_h = 2^n/(n+1)$ | 2003 | Zhaozhi Zhang *et al.* Approach | 0.1375 |
| $N_h = \left(N_{in} + \sqrt{N_p}/L\right)$ | 2008 | Jinchauan Ke, Xinzhe Lie Approach | 1.7786e-10 |
| $N_h = C_f\left(N/\left(d\log N\right)\right)^{1/2}$ | 2008 | Shuxiang Xu, Ling Chen Approach | 0.0034 |
| $\mathbf{N_h = (n+n_o-1)/2}$ | 2008 | Stephen Trenn Approach | 0.1375 |
| $N_h = \sqrt{N^{(0)}N^{(0)}}$ | 2009 | Katsunari Shibata,Yusuke Ikeda Approach | 0.1375 |
| $N_h = 2^n - 1$ | 2012 | David Hunter *et al.* Approach | 1.4915e-08 |
| $N_h = \sqrt{n+m+a}$ | 2013 | Gue Qian, Hao Yong Approach | 1.2896e-10 |
| $N_h = (4^n+18)/(n-1)$ | 2013 | K. Gnana Sheela, S.N. Deepa Approach | 5.8167e-10 |
| $N_h = \left[\left(2\left(5n^2+3\right)+1\right)/n^2-8\right]$ | | Proposed Approach | 5.1959e-13 |

Table 6: Variations of training and testing data samples

| Variations of training and testing data samples | | Statistical errors | | | | | |
|---|---|---|---|---|---|---|---|
| | | Data set 1 | | | Data set 2 | | |
| Training data (%) | Testing data (%) | MSE | MAE | MRE | MSE | MAE | MRE |
| 1000 (10) | 9000 (90) | 3.3783e-10 | 1.2028e.05 | 1.7737e-06 | 1.4282e-08 | 3.4033e-05 | 5.4554e-06 |
| 2000 (20) | 8000 (80) | 1.0872e-10 | 7.3804e-06 | 1.0622e-06 | 7.3771e-11 | 2.5613e-06 | 3.9991e-07 |
| 3000 (30) | 7000 (70) | 2.4856e-11 | 3.0573e-06 | 4.1603e-07 | 5.7354e-07 | 5.0474e-05 | 7.9505e-06 |
| 4000 (40) | 6000 (60) | 1.9712e-12 | 8.6414e-07 | 1.1039e-07 | 1.6502e-09 | 8.0838e-06 | 1.1111e-06 |
| 5000 (50) | 5000 (50) | 2.0527e-09 | 2.6952e-05 | 3.2737e-06 | 2.6808e-11 | 2.2128e-06 | 2.8856e-07 |
| 6000 (60) | 4000 (40) | 5.0830e-11 | 5.2097e-06 | 6.3651e-07 | 2.1233e-09 | 2.4538e-05 | 3.2208e-06 |
| 7000 (70) | 3000 (30) | 7.6315e-13 | 5.6554e-07 | 6.9806e-08 | 5.1959e-13 | 2.5817e-07 | 3.4229e-08 |
| 8000 (80) | 2000 (20) | 8.5922e-10 | 1.8222e-05 | 2.2351e-06 | 4.6908e-09 | 3.5496e-05 | 4.6890e-06 |
| 9000 (90) | 1000 (10) | 1.3035e-06 | 6.5636e-04 | 7.6540e-05 | 2.6169e-08 | 8.1286e-05 | 1.0181e-05 |

**Validation of proposed best criteria:** The proof for the estimation of hidden layer neuron numbers is initiated based on the convergence theorem discussion in the Appendix. Lemma 1 is proves the convergence of the presented estimation criteria.

**Lemma 1:**
The sequence $S_n = [(2(5n^2+3)+1)/(n^2-8)]$ is converged and $S_n \geq 0$. The sequence tends to have a finite limit u, if there exists constant $\varepsilon>0$ such that $|S_n-u|<\varepsilon$, then $\lim_{n\to\infty} S_n = u$.

**Proof:** Lemma 1 based proof defined as follows. Regarding to convergence theorem, the selected parameter (or) sequence converges to finite limit value:

$$S_n = \frac{(2(5n^2+3)+1)}{n^2-8} \tag{10}$$

$$\begin{aligned}&\lim_{n\to\infty}\frac{(2(5n^2+3)+1)}{n^2-8}\\ &=\lim_{n\to\infty}\frac{n^2\left[\left(10+6/n^2\right)+1/n^2\right]}{n^2(1-8/n^2)}\\ &=10, \text{ finite limit value}\end{aligned} \tag{11}$$

Here, 10 is limit value of the selected sequence as $\lim_{n\to\infty}$. Hence, above sequence is convergent sequence because it has the finite limit where n number of input parameters.

**Analysis of data samples influence on the results:** The existence of the neural networks output errors are depends on the proper division of training and testing sets. Neural networks with smaller number of training data samples may not respond effectively while neural networks with larger number of data samples also not respond properly.

Therefore, the proposed recursive radial basis function networks with 97 hidden layer neuron numbers based forecasting model influence of different training and testing sets analyzed and variations made in the training and testing data samples randomly are established in Table 6. Based on the Table 6 observed that neural networks with 70% randomly chosen data samples for training process and 30% randomly selected data samples for testing process achieve the best results with minimal statistical errors.

## CONCLUSION

This study investigates the various previous work related to selection of hidden layer neuron numbers in neural networks and performed comparative analysis with the proposed criteria. Impacts of data selection on the results are investigated and best training and testing data samples are selected. From the experimental evaluation on two real-time data sets with various windmill height concluded that the proposed criteria is effectively decided the proper hidden layer neuron numbers in Recursive radial basis function networks and outperform than that of other previous hidden neuron selection methodologies. Hence, the implemented recursive radial basis function network adapted for wind speed forecasting outperform in terms of better accuracy with minimal error, stability, generalization ability and improving the convergence speed.

## ACKNOWLEDGEMENTS

The researchers would like to thank the Suzlon Energy Private Limited for providing real-time data to perform the research work.

## APPENDIX

Considering different criteria with 'n' as input variables. All considered criteria are noted satisfied the convergence theorem. If the limit of sequence is getting to finite value, the sequence is called convergent sequence. If the limit of a sequence does not tend to a get finite value, the sequence is called divergent defined by Dass (2009). The convergence theorem possesses the following characteristics: A convergent sequence has a finite limit. All convergent sequences are bounded. 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. In the neural network model most important property is it is always converges to a stable state. The infinite sequence is changed into finite sequence is called convergence. In real-time optimization problem the convergence play a major role, risk of getting stuck at some local mini ma problem in a network is prevented by the convergence. The real-time neural optimization solvers are designed by the use of convergence properties. For the research work. Taking the sequence:

$$S_n = \frac{3n+1}{n-2} \tag{1}$$

Applying convergence theorem, it has finite value:

$$\lim_{n\to\infty} S_n = \lim_{n\to\infty}\frac{3n+1}{n-2} = \lim_{n\to\infty}\frac{n(3+1/n)}{n(1-2/n)} = 3 \neq 0 \tag{2}$$

Hence, the terms of sequence has a finite limit value and is bounded so the considered sequence is convergent in nature. Take the sequence:

$$S_n = \frac{2n^2+10}{n^2-8} \tag{3}$$

Apply convergence theorem,

$$\lim_{n\to\infty} S_n = \lim_{n\to\infty}\frac{2n^2+10}{n^2-8} = \lim_{n\to\infty}\left(\frac{n^2(2+10/n^2)}{n^2(1-8/n^2)}\right) = 2 \neq 0 \tag{4}$$

it has finite value. Thus, the terms sequence has a finite limit value and is bounded so the considered sequence is convergent in nature.

## REFERENCES


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

Dass, H.K., 2009. Advanced Engineering Mathematics. S. Chand, Delhi India, ISBN-81-219-0345-9,.

Fujita, O., 1998. Statistical estimation of the number of hidden units for feedforward neural networks. Neural Networks, 11: 851-859.

Hunter, D., H. Yu, III.M.S. Pukish, J. Kolbusz and B.M. Wilamowski, 2012. Selection of proper neural network sizes and architectures-a comparative study. IEEE Trans. Ind. Inf., 8: 228-240.

Karsoliya, S., 2012. Approximating number of hidden layer neurons in multiple hidden layer BPNN architecture. Int. J. Eng. Trends Technol., 3: 713-717.

Ke, J. and X. Liu, 2008. Empirical analysis of optimal hidden neurons in neural network modeling for stock prediction. Proceeding of the 2008 PACIIA'08 Pacific-Asia Workshop on Computational Intelligence and Industrial Application, December 19-20, 2008, IEEE, Beijing, China, ISBN: 978-0-7695-3490-9, pp: 828-832.

Li, J.Y., T.W. Chow and Y.L. Yu, 1995. The estimation theory and optimization algorithm for the number of hidden units in the higher-order feedforward neural network. Proceedings of the IEEE International Conference on Neural Networks, November 27-1December 1995, IEEE, Kowloon, Hong Kong, ISBN: 0-7803-2768-3, pp: 1229-1233.

Madhiarasan, M. and S.N. Deepa, 2016. A novel criterion to select hidden neuron numbers in improved back propagation networks for wind speed forecasting. Appl. Intell., 44: 878-893.

Mao, K.Z. and G.B. Huang, 2005. Neuron selection for RBF neural network classifier based on data structure preserving criterion. IEEE Trans. Neural Networks, 16: 1531-1540.

Panchal, G., A. Ganatra, Y.P. Kosta and D. Panchal, 2011. Behaviour analysis of multilayer perceptronswith multiple hidden neurons and hidden layers. Int. J. Comput. Theory Eng., 3: 332-337.

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

Sheela, K.G. and S.N. Deepa, 2013. A new algorithm to find number of hidden neurons in Radial Basis Function Networks for wind speed prediction in renewable energy systems. J. Control Eng. Appl. Inf., 15: 30-37.

Shibata, K. and Y. Ikeda, 2009. Effect of number of hidden neurons on learning in large-scale layered neural networks. Proceedings of the Conference on ICCAS-SICE, August 18-21 2009, IEEE, New York, USA., ISBN: 978-4-907764-34-0, pp: 5008-5013.

Tamura, S.I. and M. Tateishi, 1997. Capabilities of a four-layered feedforward neural network: four layers versus three. IEEE Trans. Neural Networks, 8: 251-255.

Trenn, S., 2008. Multilayer perceptrons: Approximation order and necessary number of hidden units. IEEE Trans. Neural Networks, 19: 836-844.

Urolagin, S., K.V. Prema and N.S. Reddy, 2011. Generalization Capability of Artificial Neural Network Incorporated with Pruning Method. In: Advanced Computing, Networking and Security, Santhi, P.T., R.P. Alwyn, K. Chandrasekaran and N. Balakrishnan (Eds.). Springer Berlin Heidelberg, Berlin, Germany, ISBN:978-3-642-29279-8, pp: 171-178.

Vora, K. and S. Yagnik, 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.

Xu, S. and L. Chen, 2008. A novel approach for determining the optimal number of hidden layer neurons for FNN's and its application in data mining. Proceeding of the 5th International Conference on Information Technology and Application, June 23-26, 2008, ICITA, Cairns, Queensland, Australia, pp: 683-686.

Zhang, J. and A.J. Morris, 1998. A sequential learning approach for single hidden layer neural networks. Neural Networks, 11: 65-80.

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