Comprehensive Neural Network Techniques Application in Wheat Yield Prediction

V. S. Dhaka, Vikas Lamba

Computer Science and Engineering Department

Jaipur National University, Jagatpura, Jaipur, India.

Abstract: - Agricultural sector area plays major role in Indian economy. This paper shows research comparison in between MLP Feed Forward Neural Network, Generalized Regression Neural Network and Radial-Basis Function Neural Network in the field of Wheat yield prediction using Z-score Normalization method. The outcome represents that GRNN present better prediction results as compared to FFNN and RBNN. Eight different parameters are used in all these models which effect the wheat yield production like Area, Rainfall, Temperature, Seed Distribution, Fertilizer (P, N, and K) and Minimum selling price (MSP). GRNN presents better filtered result for next six years for wheat yield by applying varying input parameter vectors.

Keywords: - Minimum Selling Price (MSP), Yield, Nitrogen, Phosphorus, Potassium, Production, Feed Forward Neural Network (FFNN), Generalized Regression Neural Network (GRNN), Radial Basis Function Neural Network (RBFNN),

1. INTRODUCTION

Agriculture sector plays a major role in Indian economy and wheat is major crop product which effects the agricultural sector growth. Wheat yield production effected by many factors those effect the wheat production by controlling those factors we can improve the wheat production. The major factors are area of cultivated land, total rain fall in that year, seed distribution, fertilizer distribution (N, P, and K), minimum selling price and Temperature. There are also many more factors but these are majors which effect the wheat production. So prediction of wheat in advance can improve the production of wheat for that year. There are many tools for prediction but Artificial Neural Network is most efficient prediction tool. The major Neural Networks for prediction are Multi Layers Perception Feed Forward Neural, Generalized Regression Neural Network and Radial Basis Function Neural Network. These are used in this paper. Here we describe the comparison in between MLP Feed Forward Neural Network, Generalized Regression Neural Network and Radial-Basis Function Neural Network in the field of Agricultural for Wheat yield prediction. Heping Pan, et al., presented a computational approach for predicting the Australian stock market index – AORD using multi-layer feed-forward neural networks from the time series data of AORD and various interrelated markets. Christopher Gan, et al., presented interest in applying artificial neural networks (ANN) to analyze consumer behaviour and to model the consumer decision-making process [1]. Mahdi Pakdaman Naeini, et al., presented two kinds of neural networks, a feed forward multilayer Perception (MLP) and an Elman recurrent network, are used to predict a company's stock value based on its stock share value history [2]. Nekoukar et al., have used radial basis function neural network for financial time-series forecasting, and the result of their experiment shows the feasibility and effectiveness. Geetha, et al., have predicted Rainfall in Chennai using back propagation neural network model, and in their research the mean monthly rainfall is predicted using ANN model. Jyothi Patil, have attempted to comprehend the pest population dynamics by applying analytical and other techniques on pest surveillance data sets. [3]

2. NEURAL NETWORK MODEL

Artificial neural networks (ANNs) are networks of simple processing elements (called 'neurons') operating on their local data and communicating with other elements. The design of ANNs was motivated by the structure of a real brain, but the processing elements and the architectures used in ANN have gone far from their biological inspiration. There exist many types of neural networks. But the basic principles are very similar.
Each neuron in the network is able to receive input signals, to process them and to send an output signal. Each neuron is connected at least with one neuron, and each connection is evaluated by a real

ISSN: 2278 – 7798

number, called the weight coefficient, that reflects the degree of importance of the given connection in the neural network. In principle, neural network has the power of a universal approximation, i.e. it can realise an arbitrary mapping of one vector space onto another vector space. The main advantage of neural networks is the fact, that they are able to use some a priori un- known information hidden in data (but they are not able to extract it). Process of 'capturing' the un- known information is called 'learning of neural network' or 'training of neural network'. In mathematical formalism to learn means to adjust the weight coefficients in such a way that some conditions are fulfilled. The basic diagram of neural network is given below with inputs and weights and those are forwarded next in the figure by neurons, at last output is generated.

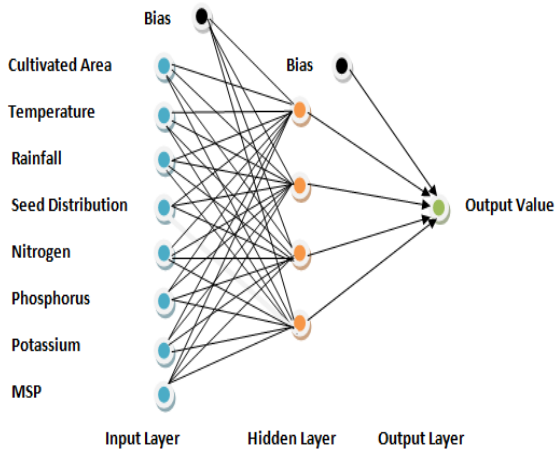

Fig-1 Basic structure of Neurons in Network Model

There exist two main types of training process: supervised and unsupervised training. Supervised training (e.g. multi-layer feed-forward (MLF) neural network) means, that neural network knows the de- sired output and adjusting of weight coefficients is done in such way, that the calculated and desired outputs are as close as possible. Unsupervised training means, that the desired output is not known, the system is provided with a group of facts (patterns) and then left to itself to settle down (or not) to a stable state in some number of iterations[4]. Here we are presenting the three different neural networks for prediction the wheat yield. That basic information is given in next sub sections.

2.1 Feed Forward Neural Network: Among the different networks, the feed forward neural networks or multi layer perception (MLP) are the most commonly used in engineering science. MLP networks are normally arranged in three layers of neurons, the so called multilayer structure:

•Input layer: its neurons also called nodes or processing units) introduce the model inputs.

•Hidden layer(s) (one or more layers): Its nodes combine the inputs with weights that are adapted during the learning process.

•Output layer: This layer provides the estimations of the network [5].In these networks, the output is function of the linear combination of hidden units' activations; each one is a non-linear function of the weighted sum of inputs:

$$Y = f(x, \theta) + \varepsilon \qquad (1)$$

Where x is the vector of explanatory variables, ε is the random error component. f(x, θ) = ŷ is the unknown function for estimation and prediction from the available data. Consider a MLP with three layers and one output. The network consists of the following form:

$$\hat{Y} = F\left(u_0 + \sum_{j=1}^{m} H(\lambda_j + \sum_{i=1}^{n} x_i\theta_{ij})\, u_j\right) \qquad (2)$$

Where:

Ŷ: Network output,

F: output unit activation function,

H: hidden unit activation function,

n: number of input units, m: number of hidden units,

x_j: input vector for unit j ($x_{ij} = i^{th}$ input to the j unit),

θ_{ij}: weight from input layer i to hidden unit j,

u0: output bias,

λ_j: hidden units biases (j = 1 . . . m),

u_j: weights from hidden unit j to output (j = 1 . . . m)

Another critical issue in ANNs is the neural learning or model estimation based upon searching the weights that minimize some cost function such as square error:

$$\text{Min}\,[E\,(Y - f\,(x, \theta))^2\,] \quad \theta\, \varepsilon\, \Theta \qquad (3)$$

The most popular learning algorithm is the Back Proportion (BP).BP learning is a kind of supervised learning introduced by Werbos (1974) and later developed by Rumelhart and McClelland (1986)[5]. Desirable output for input set is made by this algorithm. Error in each neuron is the difference between ANN output and real output. The interconnections weight and threshold value in each neuron is adjusted to minimize the error. Let E denote error function. In this algorithm for reducing error, the weights vector (u) is adjusted. For this ∂E/∂θ is calculated.

Let: o_j = output of unit j,

Θ_{ij}: weight from input layer i to hidden unit j,

Z_j: $\theta_j . x_j$ the weighted sum of inputs for unit j,

By applying chain rule, we have:

$$\frac{\partial E}{\partial \theta ji} = \left(\frac{\partial E}{\partial Zj}\right)\left(\frac{\partial Zj}{\partial \theta ji}\right)$$

ISSN: 2278 – 7798

It is proof that

- $\frac{\partial E}{\partial \theta ji} = \delta_i . O_j$ That : $\frac{\partial E}{\partial Zj} = \delta_i$

Thus:

$\Delta\theta_{ji} = \eta\delta_i O_j$

$\theta_{ji}(k+1) = \theta_{ji}(k) + \eta\delta_i O_j$

This algorithm process can be as follows:

$$\theta (k+1) = \theta(K) - \eta \frac{\partial E}{\partial \theta}(k) \quad (4)$$

Or

$$\theta (k+1) = \theta(K) + \eta \Delta f(x, \theta) [Y - f(x, \theta)] \quad (5)$$

BP is an iterative process (k indicators iteration). Parameters are revised from the error function (E) gradient by the learning rate η, constant or variable. The error propagates backwards to correct the weights until some stoppage criterion – epoch, error goals – is reached [5]. Adding a term called momentum can improve this algorithm:

$$\theta (k+1) = \theta(K) + \eta \frac{\partial E}{\partial \theta}(k) + \mu\Delta \theta (k-1)$$

After neural training (training set), new observations (validation and/or test sets) are presented to the network to verify the so-called generalization capability. ANNs have advantages, but logically they also have several drawbacks. Therefore, ANNs can learn from experience and can generalize, estimate, predict, with few assumptions about data and relationships between variables. These attributes have made the ANN approach fairly efficient for problem solving. Hence, ANNs have an important role when these relationships are unknown (non-parametric method) or non-linear (non-linear method), provided there are enough observations with flexible form and universal approximation property. However, the flexibility could cause learning of the noise. Finally, algorithm convergence and trial and error process are also some relevant drawbacks [5].

2.2 Generalized Regression Neural Network: The GRNN was applied to solve a variety of problems like prediction, control, plant process modelling or general mapping problems [6][7][8].General regression neural network does not require an iterative training procedure as in back-propagation method[9][10][11]. The GRNN is used for estimation of continuous variables, as in standard regression techniques. It is related to the radial basis function network and is based on a standard statistical technique called kernel regression. By definition, the regression of a dependent variable y on an independent x estimates the most probable value for y, given x and a training set. The regression method will produce the estimated value of y, which minimizes the mean-squared error. GRNN is a method for estimating the joint probability density function (PDF) of x and y, given only a training set. Because the PDF is derived from the data with no preconceptions about its form, the system is perfectly general. Furthermore, it is consistent; that is, as the training set size becomes large, the estimation error approaches zero, with only mild restrictions on the function. In GRNN, instead of training the weights, one simply assigns to w_{ij} the target value directly from the training set associated with input training vector i and component j of its corresponding output vector [12]. GRNN architecture is given in Fig. 2. GRNN is based on the following formula [13].

$$E[y \mid x] = \frac{\int_{-\infty}^{\infty} yf(x,y).dy}{\int_{-\infty}^{\infty} .f(x,y).dy} \quad (6)$$

Where y is the output of the estimator, x is the estimator input vector, E[y | x] is the expected output value, given the input vector x and f(x, y) is the joint probability density function (pdf) of x and y. The function value is estimated optimally as follows:

$$y_j = \frac{\sum_{i=1}^{n} hi.wij}{\sum_{i=1}^{n} hi} \quad (7)$$

Where w_{ij}= the target output corresponding to input training vector xi, $h_i = e^{\frac{-D^2}{2.Sprea\ d^2}}$, the output of the hidden layer neuron, for i^{th} neuron $D^2 = (x-u_i)^T.(x-u_i)$, the square distance between the input vector x and the training vector u, x= the input vector, u_i=training vector i, the center of neuron i, spread=a constant controlling the size of the receptive region.

2.3 Radial Basis Neural Network: Radial Basis Neural Network (RBNN) consists of two layers: a hidden radial basis layer of S1 neurons, and an output linear layer of S2 neurons [14]. A Radial Basis neuron model with R inputs is shown in Fig Radial Basis Neuron uses the radbas transfer Function The net input to the radbas transfer function is the vector distance between its weight vector w and the input vector p, multiplied by the bias b. (The ||dist|| box in this figure accepts the input vector p and the single row input weight matrix, and produces the dot product of the two.)The transfer function for a radial basis neuron is given in equation 8.

$$\text{radbas}\ (n) = e^{-n^2} \quad (8)$$

ISSN: 2278 – 7798

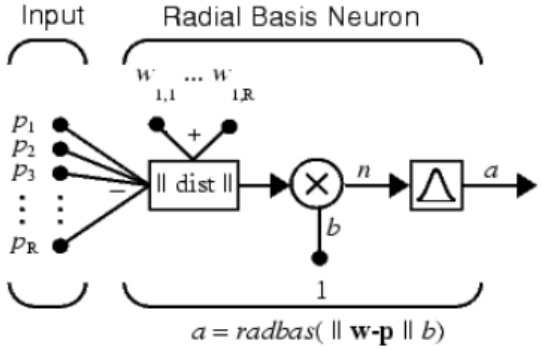

Fig-2 Radial Basis Neuron Model

A plot of the radbas transfer function is shown in Fig given below

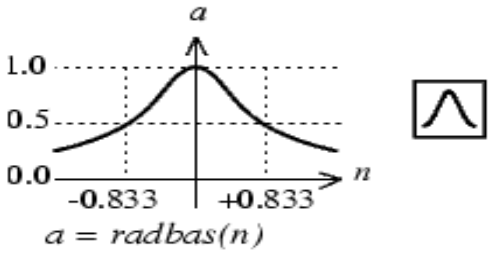

Fig-3: radbas transfer function

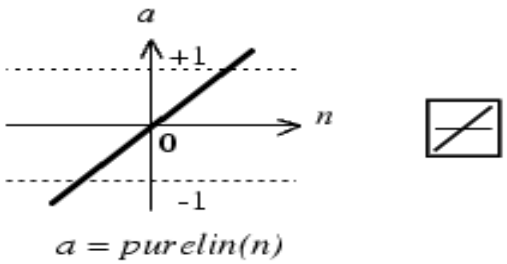

Fig-4: purelin transfer function

The radial basis function has a maximum of 1 when its input is 0and it is first layer neuron transfer function. As the distance between w and decreases, the output increases. Thus, a radial basis neuron acts as a detector that produces 1 whenever the input p is identical to its weight vector w. The bias b allows the sensitivity of the radbas neuron to be adjusted. The purelin transfer function provides output along with n values that given in fig 4. purelin is a second layer neural transfer function. Transfer functions calculate a layer's output from its net input. Radial basis networks can be designed in a fraction of the time that it takes to train standard feed forward networks. They work best when many training vectors are available. Radial Basis Networks are created with zero error on training vectors [15].

3. EXPERIMANTAL RESULTS

3.1 Data Collection and Analysis

For this study, we have used 8 main factors as inputs. We have got all data from Rajasthan Agricultural Krishi Department, C-scheme, Jaipur, Rajasthan, India. These data's are in annual form and contain of 20 years from 1993 to 2012. There are 8 main factors which used in this research are given below:

a. Cultivated Area: The major area used for wheat cultivation for the given year. Area is given in hectors.
b. Temperature: Minimum and maximum temperature of that year.
c. Annual Rain fall: Annual rainfall in taken for that year from January to December.
d. Seed Distribution: Seed distributed by the rajasthan govt. In that particular session (winter) for wheat.
e. Fertilizer Consumption: Total N, P and K consumption in that session for wheat (Three parameters).
f. MSP: Minimum selling price decided by the government of wheat in that year.

3.2 Data Normalization

The collected data from agriculture department were in different form so that we should have to arrange those data in a fixed form or in a defined range for efficient data processing. So here we used Z-score method for data normalization. The basic formula used in normalization is given below:

$$X' = (X - A') / SA \quad (9)$$

And $SA = \sqrt{\frac{\sum_{i=1}^{i=n}(Xi-A')^2}{(n-1)}}$

Where:
X': Normalized data in a range (Z-score(x)),
X: Actual value in that year,
A': Mean of the distribution X, it is also called mean of x or average of all values of that parameter of distribution 'x',
SA: Standard deviation applied on x parameter,
X_i: Actual i^{th} value.

3.3 Result Analysis

To develop all neural network models we are using MATLAB tool because NN Tool is very effective for non linear data and it provides all types of functions in build by using those functions we can develop any type of artificial neural network for any real life application in very less time and in a very effective manner.

ISSN: 2278 – 7798

We are describing all three neural network models with experimental result below.

3.3.1 Feed forward Neural Network

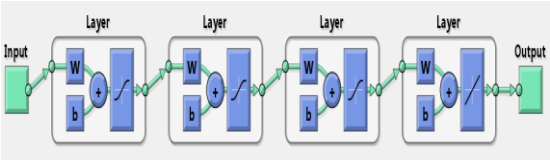

Fig-5 Feed Forward Neural Network Structure

To create feed forward neural network using MATLAB we use newff function and in that function we are using three hidden layers. In input layer we are applying 8 input parameters as input vectors and output comes in output layer. Here tansig function has been used as transfer function in all three hidden layers because it is most efficient transfer function for non-linear real life application data. In all three hidden layers we are using 30, 16 and 8 neurons and according to those neurons we will adjust the weight matrices and fix the weights with bias to find efficient results. The Levenberg-Marguardt(trainlm) training algorithm is used for training because this algorithm is most effective than others training algorithm in respect to time and memory uses in execution. To measure the performance of the model we are using Mean Square Error (MSE) function. Data division in done randomly by dividerand function in 60, 20 and 20 sets as training sets, validation sets and testing sets respectively. There are given below some experimental results by graphs and charts.

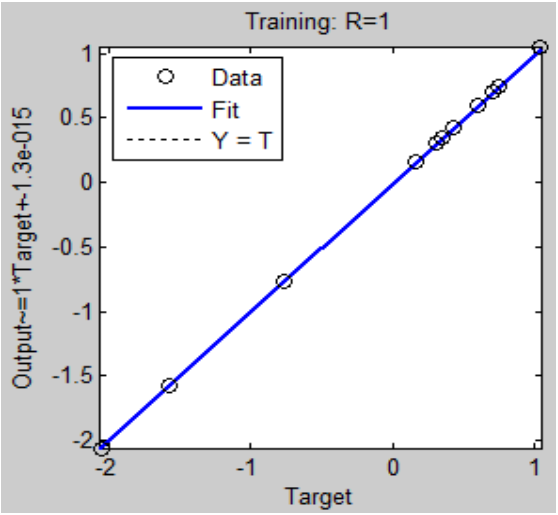

Fig-6 Training Set

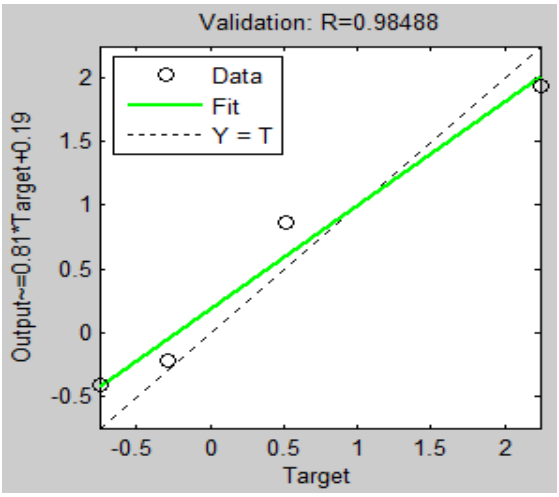

Fig-7 Validation Set

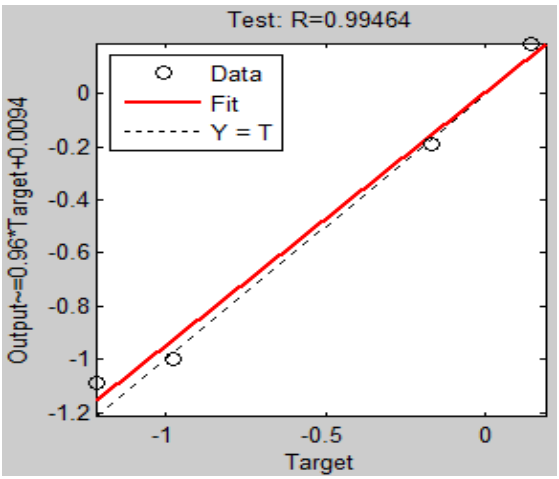

Fig-8 Testing Set

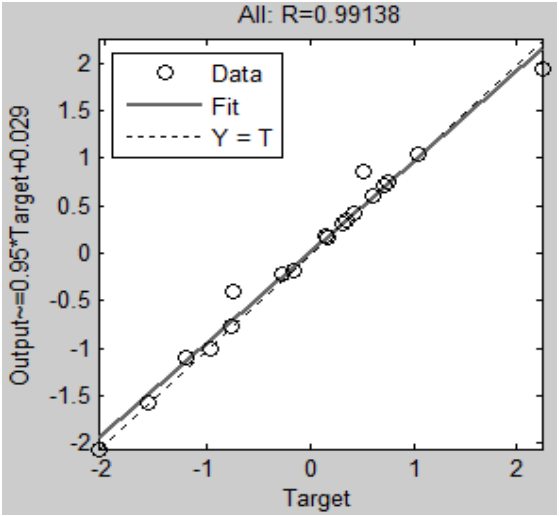

Fig-9 All Common Set

All these figures fig-3, fig-4, fig-5 and fig-6 shows the training set, validation set, testing set and all common set figures respectively.

ISSN: 2278 – 7798

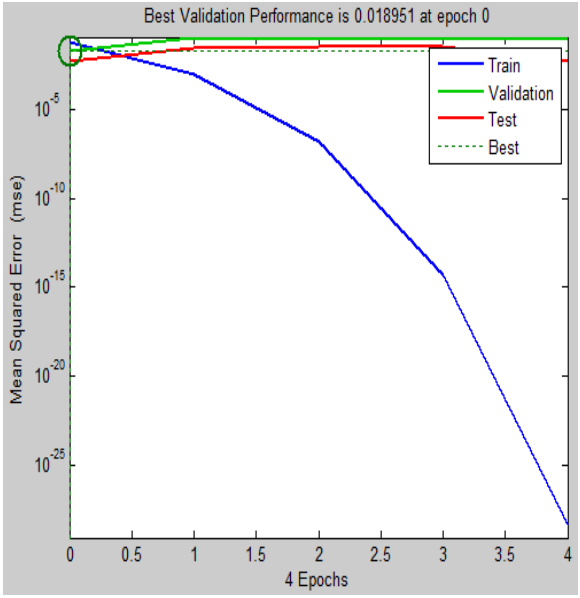

Fig-10 Validation performance Index Graph in MSE

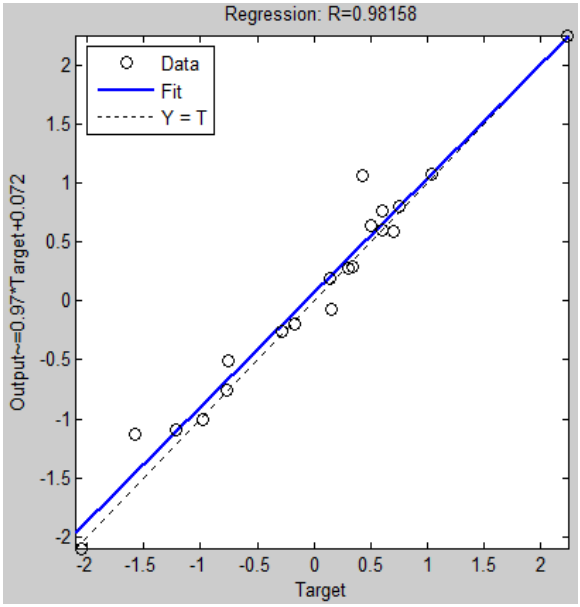

Fig-11 Regression plot in between Output and Target values

The figures 10 shows that best validation performance is reached 0.018951 at 0 epoch and the graph gradually falls down it shown the best performance. The figure 11 shows the relationship in between output value and the target value and it is nearly reached 0.98158. It means it is best reached at 98% with target value.

Comparision In-Between Output and Target value by FFNN Model

YEAR	TARGET	OUTPUT	DIFFERENCE
1993-94	-2.05948384	-2.09700000	-0.03751616
1994-95	-1.21496499	-1.09020000	0.12476499
1995-96	0.34058269	0.28510000	-0.05548269
1996-97	-0.97922275	-1.00150000	-0.02227725
1997-98	0.30582071	0.28370000	-0.02212071
1998-99	-0.28462753	-0.25410000	0.03052753
1999-2000	-0.17011008	-0.19070000	-0.02058992
2000-01	-0.75263836	-0.51210000	0.24053836
2001-02	1.04004104	1.07440000	0.03435896
2002-03	0.15435604	-0.06700000	-0.22135604
2003-04	0.70507812	0.59230000	-0.11277812
2004-05	0.50773572	0.63810000	0.13036428
2005-06	0.59755854	0.59690000	-0.00065854
2006-07	-0.77441512	-0.75400000	0.02041512
2007-08	-1.57553202	-1.12580000	0.44973202
2008-09	0.14397292	0.18810000	0.04412708
2009-10	0.42722235	1.06280000	0.63557765
2010-11	2.23913957	2.24240000	0.00326043
2011-12	0.60234589	0.76330000	0.16095411
2012-13	0.74714112	0.80820000	0.06105888

Table-1

Table-1 shows the comparisons in between output and target values the output values are generated by the feed Forward Neural Network Model and target values (Actual values) are the year wise production values. All these values are in normalized form, those are generated by the normalization formula. We can see that there is very little difference in between both of these values correspondence to the particular year. It means FFNN model find the best result for prediction that is with 98% accuracy.

3.3.2 Generalized Regression Neural Network

Generalized regression neural networks are a kind of radial basis network that is often used for function approximation. GRNNs can be designed very quickly NEWGRNN (P, T, SPREAD) takes these inputs, P- R*Q1 matrix of Q1 input vectors. T- S*Q2 matrix of Q2 target class vectors. SPREAD - Spread of radial basis functions, default = 1.0 and returns a new generalized regression neural network. The larger SPREAD is, the smoother the function. Approximation will be. To fit data closely, use a SPREAD smaller than the typical distance between input vectors. To fit the data more smoothly use a larger SPREAD.NEWGRNN creates a two layer network.

ISSN: 2278 – 7798

The first layer has RADBAS neurons, calculates weighted inputs with DIST and net input with NETPROD. The second layer has PURELIN neurons, calculates weighted input with NORMPROD and net inputs with NETSUM. Only the first layer has biases.

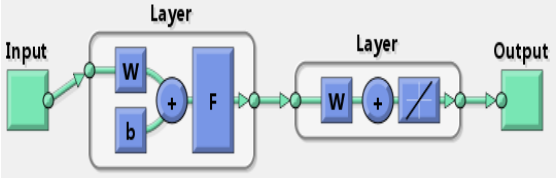

Fig-11 Generalized Regression Neural Network Structure

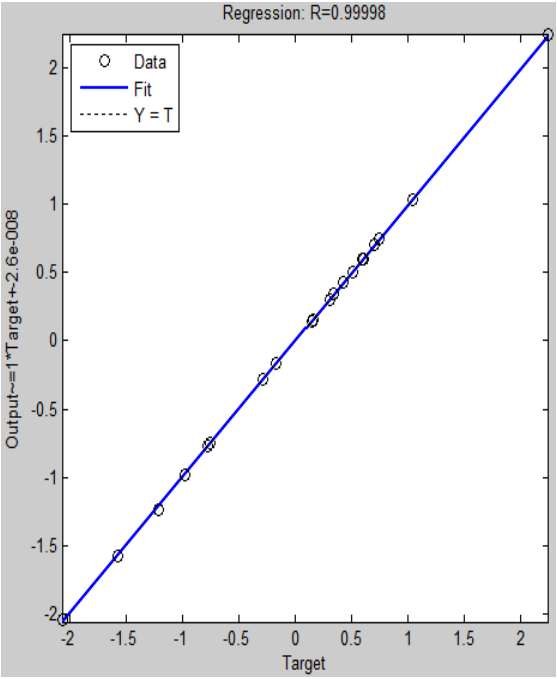

Fig-12 Regression plot in between Output and Target values by GRNN Model

Given regression plot with R=0.99998 by Generalized Regression Neural Network shows that generalized regression provides data matching accuracy as 99% and it is more accurate than FFNN

Comparision In-Between Output and Target value by GRNN Model

YEAR	TARGET	OUTPUT	DIFFERENCE
1993-94	-2.05948384	-2.04180000	0.01768384
1994-95	-1.21496499	-1.23250000	-0.01753501
1995-96	0.34058269	0.34040000	-0.00018269
1996-97	-0.97922275	-0.97920000	0.00002275
1997-98	0.30582071	0.30580000	-0.00002071
1998-99	-0.28462753	-0.28460000	0.00002753
1999-2000	-0.17011008	-0.17010000	0.00001008
2000-01	-0.75263836	-0.75260000	0.00003836
2001-02	1.04004104	1.03780000	-0.00224104
2002-03	0.15435604	0.15660000	0.00224396
2003-04	0.70507812	0.70420000	-0.00087812
2004-05	0.50773572	0.50830000	0.00056428
2005-06	0.59755854	0.59790000	0.00034146
2006-07	-0.77441512	-0.77440000	0.00001512
2007-08	-1.57553202	-1.57220000	0.00333202
2008-09	0.14397292	0.14060000	-0.00337292
2009-10	0.42722235	0.42720000	-0.00002235
2010-11	2.23913957	2.23910000	-0.00003957
2011-12	0.60234589	0.60230000	-0.00004589
2012-13	0.74714112	0.74710000	-0.00004112

Table-2

3.3.3 Radial Basis Neural Network

Radial basis networks can be used to approximate functions. NEWRBE very quickly designs a radial basis network with zero error on the design vectors. NEWRBE (P, T, and SPREAD) takes two or three arguments, P-R*Q matrix of Q input vectors. T-S*Q matrix of Q target class vectors. SPREAD - of radial basis functions, default = 1.0 and returns a new exact radial basis network. The larger that SPREAD, is the smoother the function approximation will be. Too large a spread can cause numerical problems. NEWRBE creates a two layer network. The first layer has RADBAS neurons, and calculates its weighted inputs with DIST, and its net input with NETPROD. The second layer has PURELIN neurons, and calculates its weighted input with DOTPROD and its net inputs with NETSUM. Both layer's have biases.

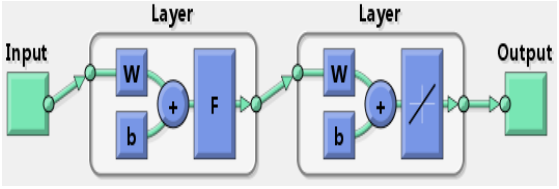

Fig-13 Radial Basis Neural Network Structure

ISSN: 2278 – 7798

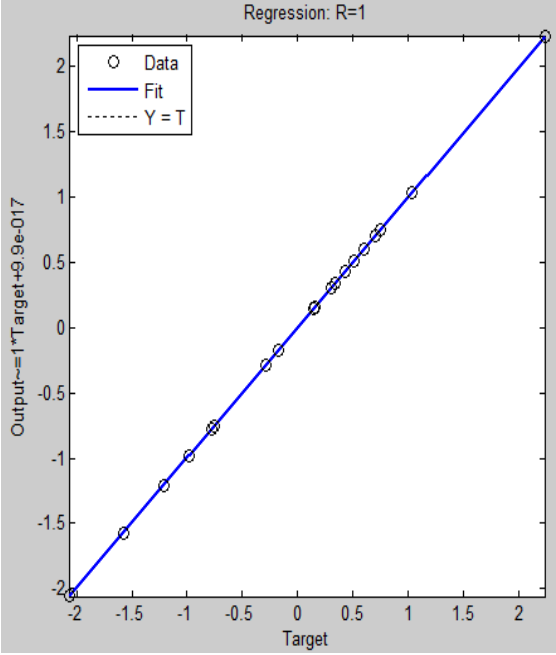

Fig-14 Regression plot in between Output and Target values by BRNN Model

This regression plot provides the exact regression in between output and target values with 100%. It means RBNN provides more data matching accuracy than FFNN and GRNN without error.

Comparision In-Between Output and Target value by RBNN Model			
YEAR	TARGET	OUTPUT	DIFFERENCE
1993-94	-2.05948384	-2.05950000	-0.00001616
1994-95	-1.21496499	-1.21500000	-0.00003501
1995-96	0.34058269	0.34060000	0.00001731
1996-97	-0.97922275	-0.97920000	0.00002275
1997-98	0.30582071	0.30580000	-0.00002071
1998-99	-0.28462753	-0.28460000	0.00002753
1999-2000	-0.17011008	-0.17010000	0.00001008
2000-01	-0.75263836	-0.75260000	0.00003836
2001-02	1.04004104	1.04000000	-0.00004104
2002-03	0.15435604	0.15440000	0.00004396
2003-04	0.70507812	0.70510000	0.00002188
2004-05	0.50773572	0.50770000	-0.00003572
2005-06	0.59755854	0.59760000	0.00004146
2006-07	-0.77441512	-0.77440000	0.00001512
2007-08	-1.57553202	-1.57550000	0.00003202
2008-09	0.14397292	0.14400000	0.00002708
2009-10	0.42722235	0.42720000	-0.00002235
2010-11	2.23913957	2.23910000	-0.00003957
2011-12	0.60234589	0.60230000	-0.00004589
2012-13	0.74714112	0.74710000	-0.00004112

Table-3

3.3.4 Comparison Analysis In Between All Three Models

All three models find different regressions Feed Forward Neural Network finds 98%, Generalized Regression Neural Network finds 99% and Radial Basis Neural Network finds 100%. Using all these models we find next 6 years productions by applying 8 parameters varying inputs and we find that Generalized Regression Neural Network provide best result than FFNN and BRNN network because GRNN network provides better result when the data set are large or in a wide range, hare data range is in between -3 to +3 range (normalization data) So we can say that GRNN provides more batter result when data will in wide range or on large samples data. Or we can also say that GRNN provides batter function approximation than RBNN. The next 6 years Wheat Prediction outs are given in the table by each model. The production outputs are in normalized form and in original form in table 4 and table 5 respectively.

Prediction Comparision for Next 6 Years of Wheat Production		
FFNN	GRNN	RBE
0.3803	0.5293	0.0365
0.9134	0.5976	0.0365
0.7409	0.7471	0.0469
0.9801	0.7471	0.0397
1.9984	2.2391	0.0463
1.6163	2.2363	0.0365

Table-4 Prediction Data is in Normalize form

Prediction Comparision for Next 6 Years of Wheat Production		
FFNN	GRNN	RBE
443787.2996	455583.1784	416569.6945
485991.2125	460990.2826	416569.6945
472334.9098	472825.745	417393.031
491271.6495	472825.745	416823.0288
571887.3774	590942.8676	417345.5308
541637.6773	590721.2001	416569.6945

Table-5 Data is in simple form (In Tonnes)

4. CONCLUSION

India is one of the largest countries that produce wheat in Asia and use of wheat in many part of this country is seen widely. For this reason, we have decided to predict production of wheat in India. In order to conduct this review, we have used NN as a prediction tool and chose 8 significant factors in wheat producing. We gathered data's from Rajasthan Agricultural Krishi department, Jaipur and used them as an input variables. These factors are such as area under cultivation, annual average temperature, annual rainfall, seed distribution,

ISSN: 2278 – 7798

fertilizer distribution as N, P and K and minimum selling price (MSP) These data's are annual and contain of 20 years from 1993 to 2012.In other words we have considered 8 factors as input factors. In order to get the best NN for this study, three distinct NN models are tested to find the optimum model for wheat production. In this way, we have found best NN model is GRNN for the output. Max number of neurons in the FFNN model are first hidden layer is set to 30, second layer is set to 16 and in third hidden layer is set to 8. All three models are run 100 times to take care of possible bias or noise. As shown in table 5 GRNN model for wheat production provides better result and consequently is chosen as the preferred or optimum model. By using the best NN model for our review, we predict production of wheat for next 6 years as for 2013 to 2018. The results show that the proposed GRNN model is a suitable way of predicting wheat production in comparison to other FFNN and RBNN models when input data sets are very large or data sets are in wide range.

REFERENCES

[1] Christopher Gan, Visit Limsombunchai, Mike Clemes, Amy Weng," Consumer Choice Prediction: Artificial Neural Networks versus Logistic Models" Commerce Division Discussion Paper No. 104, July 2005.

[2] Mahdi Pakdaman Naeini, Hamidreza Taremian, Homa Baradaran Hashemi, "Stock Market Value Prediction Using Neural Networks" IEEE 2010.

[3] Jyothi Patil and V.D.Mytri, A Prediction Model for Population Dynamics of Cotton Pest (Trips' tobacco Lined) Using Multilayer-Perception Neural Network, International Journal of Computer Applications, Vol. 67, April 2013.

[4] Daniel Svozil, Vladimir KvasniEka, JiE Pospichal, Introduction to multi-layer feed-forward neural networks, Chemometrics and Intelligent Laboratory Systems, Elsevier, 39 (1997) 43-62.

[5] Reza Ghodsi, Ruzbeh Mirabdollah Yani, Rana Jalali, Mahsa Ruzbahman " Predicting Wheat Production in Iran Using an Artificial Neural Networks Approach" International Journal of Academic Research in Business and Social Sciences February 2012, Vol. 2, No. 2 ISSN: 2222-6990.

[6] Qeethara Shayea and Ghaleb El-Refea, "Predicting the Effects of Medical Waste in the Environment Using Artificial Neural Networks: A Case Study" IJCSI International Journal of Computer Science Issues, Vol. 10, Issue 1, No 3, January 2013 ISSN (Print): 1694-0784 | ISSN (Online): 1694-0814.

[7] J. C. Neves and A. Vieira, "Improving Bankruptcy Prediction with Hidden Layer Learning Vector Quantization", European Accounting Review, Vol. 15, No. 2, 2006, p.p. 253-271.

[8] D. W. Patterson, Artificial Neural Networks, Theory, and Applications, Englewood Cliffs, NJ: Prentice-Hall, 1995.

[9] D. F. Specht, "A general regression neural network", IEEE Trans. Neural Networks, vol. 2, 1991, pp.568–576.

[10] E. A. Nadaraya, "On estimating regression", Theory of Probability Applicant, vol. 9, 1964, pp. 141–142.

[11] G. S. Watson, "Smooth regression analysis", Sankhya Series A, vol. 26, 1964, pp. 359–372.

[12] M. T. Hagan, H. B. Demuth, M. Beale, Neural network design, PWS Publishing Company, Boston, 1996.

[13] K. Kayaer and T. Yildirim, "Medical Diagnosis on Pima Indian Diabetes Using General Regression Neural Networks", web page available at: www.yildiz.edu.tr/~tulay/publications/Icann-Iconip2003-2.pdf

[14] Ali Idri, Alain Abran, Samir Mbarki,Validating and Understanding Software Cost Estimation Models based on Neural Networks, 2004 IEEE, 0-7803-8482-2/04.

[15] Prasad Reddy P.V.G.D, Sudha K.R, Rama Sree P and Ramesh S N .S.V.S.C "Software Effort Estimation using Radial Basis and Generalized Regression Neural Networks" JOURNAL OF COMPUTING, VOLUME 2, ISSUE 5, MAY 2010, ISSN 2151-9617.

[16] Karunanithi, N., et al. "Using neural networks in reliability prediction", IEEE Software, pp. 53-59, 1992

[17] LiMin Fu, "Neural Networks in Computer Intelligence," TataMcGraw-Hill Edition 2003, pp.94-97

[18] Adesh Kumar Pandey, A.K Sinha, and V.K Srivastava "A Comparative Study of Neural-Network & Fuzzy Time Series Forecasting Techniques – Case Study: Wheat Production Forecasting", IJCSNS International Journal of Computer Science and Network Security Vol. 8, September 2008.

[19] Muhammad Awais, Shafay Shamail, Nisar Ahmed and Saman Shahid, A Comparative Study of Feed Forward Neural Networks and Radial Basis Neural Networks for Modeling Tokamak Fusion Process Proceedings IEEE, INMlC 2003.

[20] Kanna Bhaskar, S. N. Singh "AWNN-Assisted Wind Power Forecasting Using Feed-Forward Neural Network" IEEE Transactions on Sustainable Energy, Vol. 3, no. 2, April 2012.

[21] Zhongbo Liu, Zhaosheng Yang, Peng Gao "Research on the Short-term Traffic Flow Prediction Method Based on BP Neural Networks" IEEE 2009.

[22] Klaus P. Hochheim, David G. Barber, Paul R. Bullock "Spring Wheat Yield Prediction for Western Canada Using Weekly NOAA AVHRR Composites" IEEE 1996.

ISSN: 2278 – 7798