IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Optimal Wheat Yield Prediction Using Feed Forward Neural Networks

Vikas Lamba

Associate Professor

Chitkara University

Abstract: Agriculture crop predication is a challenging and interesting problem domain for the computer community. The computer machine helps to improve the productivity of the crop with such predication. This paper illustrates a research model which provides the predications of the wheat crop in the Rajasthan region of India. This paper uses feed forward neural networks models for accomplish the task of predication for the wheat crop. These models are trained with three different feed forward neural networks architectures namely multi layer perception feed forward neural network, generalized regression neural network and radial basis neural network. There is feature scaling method was used for normalization of large previous year's crop detail data which has been obtained by agriculture department of Rajasthan government. The simulation results indicates the better predication with feed forward neural network model trained with Back propagation learning rule in comparison of generalized regression neural network and radial basis neural network.

***Index Terms*: Predication, Feed forward neural network, generalized regression neural network, radial basis neural network, Approximation.**

1. Introduction

Agriculture segment plays a foremost role in Indian economy and wheat is a key crop product which sound effects the agricultural area growth. Wheat yield production effected by many factors, by domineering those factors we can improve the wheat production. The optimal set of these values can improve the production of wheat. The major factors those are effecting the production of wheat yield are cultivated land area, total rain fall in that year, seed distribution, fertilizer distribution (N, P, and P), minimum selling price (MSP) and temperature (T). There are also many more factors, but these given factors are major's factors which affect the wheat production. The correct estimation for the Prediction of wheat based on the previous data of these factors can improve the production of wheat by suggesting the optimal values of these parameters.

There are many various prediction tools available in literature, but artificial neural network is found the most effective and efficient tool for forecasting or predication. Specifically the Multilayer feed forward neural network

has established the dominance over other predicating tools since last 15 years. The dominance of multilayer feed forward neural network for the forecasting or predication can exhibit in literature. Eveleen Darbey et al.(2005) have used head and neck cancer metastasis prediction via artificial neural networks. Zhen Zhang and Nan Jing (2008) have used radial basis function method for prediction of protein secondary structure. Christopher Gen et al. (2005) used artificial neural networks (ANN) to analyze consumer behaviour and to model a feed forward multi layer perception (MLP) and an Elman recurrent network to predict a company's stock value based on its stock share value history. Mahdi Pakaman Naeini et al. (2010) also used the stock market value prediction using neural networks. Jyothi Patil et al.(2013) used the pest population dynamics by applying analytical and other techniques on pest surveillance data sets. There are many more research done in prediction in different areas (Qeethara Shayea 2013; Reza Ghodsi et al. 2012; J.C. Neves 2006; P.V.M.D. Prasad Readdy et al. 2010; V.S.Dhaka and Vikas Lamba 2015 and Nidhi Gupta, M.P.Singh 2005) but the predication for wheat yield specifically in Rajasthan region has not been considered and mentioned.

This paper illustrates a research model which provides the predications of the wheat crop in the Rajasthan region of India. This paper uses feed forward neural networks models for accomplish the task of predication for the wheat crop. These models are trained with three different feed forward neural networks architectures namely multi layer perception feed forward neural network, generalized regression neural network and radial basis neural network. There are two methods were used for normalization of large previous year's crop detail data which has been obtained by agriculture department of Rajasthan government. The simulation results indicates the better predication with feed forward neural network model trained with Back propagation learning rule in comparison of generalized regression neural network and radial basis neural network. In our research work we are using eight different parameters which affect the wheat yield production. Those parameters are cultivated area, rainfall, temperature, seed distribution, Nitrogen, Phosphor, Potassium consumption in the form of fertilizer and MSP. The simulation results indicated that the Feed forward neural network presents Optimal Wheat Yield Prediction (OWYP) model for next (upcoming) six years (2014 to 2019) with 97% accuracy in comparison to generalized regression neural network and Radial basis neural network model. Thus, the multilayer feed forward neural network with Back propagation learning rule (Levenberg-Marguardt method) appears more effective tool for predication of wheat yield.

2. Data Acquisition and Pattern Formation

The predication of wheat yield with neural networks requires training set of sample data. Therefore, the data acquisition and pattern formation becomes the pre processing step for the proposed method. Hence to construct the training set patterns we studied the factors from the collected data. This data collection is performed 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. The main factors discovered from the available data in the growth of the wheat are cultivated area, temperature, rainfall, fertilizer (Nitrogen, Phosphor, and Potassium consumption), seeds used and minimum selling price. These eight factors are described as:

Cultivated Area: The major area used for wheat cultivation for the given year. Area is given in hectors.

Temperature: Minimum and maximum temperature of that year and then find the average of both.

Annual Rain fall: Annual average rainfall in taken for that year from January to December.

Seed Distribution: Seed distributed by the Rajasthan govt. In that particular session (winter) for wheat.

Fertilizer Consumption: Total N, P and K (Nitrogen, Phosphor, Potassium) consumption in that session for wheat.

MSP: Minimum selling price decided by the government of wheat in that particular year.

We understood that there are many more other factors like soil type, disease of pests or diseases of plant and so on those affect wheat yield growth. These parameters are very difficult to get hold, so we have not used such factors for our prediction model. The collected data from agriculture department were in different form so that the data should arrange in a fixed form or in a defined range for efficient data processing. Therefore it is necessary to normalize the data in the definite operating and representative way though it could use as the training set for neural networks. Hence here we used feature scaling method for data normalization as:

$$X' = (X - X_{\min})/(X_{\max} - X_{\min}) \quad (2.1)$$

Where:

X' is Normalized data in a range,

X is Actual value in that year,

X_{min} is Minimum value in that Colum or of that factor in all years,

X_{max} is Maximum value in that Colum or of that factor in all years.

Thus from this formula all the data presents in the range of 0 to 1.

As an example the table 1 is showing the sample data in original form and table 2 is presenting the data after normalization.

Table 1: Shows the original samples of data for 1993-94 and 1994-95 years.

PARAMETER	SAMPLE 1	SAMPLE 2
Year	1993-94	1994-1995
Area	147297	143671
Temperature (MIN, MAX, AVG.)	3, 47, 25	3, 46, 25
Rainfall	669	709
Seed Distribution	10908	11502
Fertilizer Distribution (N, P, K)	12987, 3780, 391	13738, 3858, 381
MSP (Minimum Selling Price)	500	510

SOURCE: Library, Rajasthan agriculture department, Jaipur, India.

Table 2: Shows the normalized samples of data for 1993-94 and 1994-95 years.

PARAMETER	SAMPLE 1	SAMPLE 2
Year	1993-94	1994-95
Area	0.2406201	0.1835537
Temperature (MIN, MAX, AVG.)	0.666666667	0.5
Rainfall	0.8189655	0.9339080
Seed Distribution	0.004015751	0.01559515
Fertilizer Distribution (N, PH, PO)	0.00866441	0.088017751
	0.22047244	0.232755910
	0.10640200	0.101894000
MSP (Minimum Selling Price)	0	0.011

SOURCE: All these values are calculated by using formula 2.1

Therefore we have used all these normalized eight (08) factors as input to the neural networks for the training. The training set will contain the feature vectors for these eight factors with the target output pattern as $((X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8), T)$ .Thus we have a large training set which consist with samples from year 1993-2012 and each sample consists with input pattern of size 8 X 1 and a target output pattern vector of size 1 X 1. Among these sample patterns 5 samples are selected as the test pattern samples. Thus the training set consists of size 8 X 15 and test pattern consists of 8 X 5.

3. Neural Networks Model

Neural network architectures can be classified as, feed- forward and feedback (recurrent) networks. One of the interesting characteristics of multi layer feed forward neural network is that in addition to classifying an input pattern, they also provide a confidence in the classification (A.K.Jain et al. 2000). These confidence values may be used for rejecting a test pattern in case of doubt (S. Arora et al. 2010 and U.Bhattacharya et al. 2004). The Feed forward neural network architecture with supervised learning rule has been considered as one of the efficient classifier, which can exhibit generalization in classification process (Daniel Svozil et al. 1997). It consists with number of simple neuron-like processing units, organized in layers. Every unit in a layer is connected with all the units in the previous layer. These connections are not equal instead of this each connection may have a different strength or weight. The weights on these connections encode the knowledge of a network. Input pattern vector enters at the inputs and passes through the network, layer by layer, until it arrives at the outputs. This neural network consists of an input layer of units, one or more hidden layers, and an output layer. Each node in the layer has one corresponding node in the next layer, thus creating the stacking effect.

The input layer's nodes have output functions that deliver signal to the first hidden layer nodes. The hidden layer(s) are the processing layer, where all of the actual computation takes place. Each node in hidden layer computes a sum based on its input from the previous layer (either the input layer or another hidden layer). The sum is then "compacted" by a sigmoid function (a logistic transfer function), which changes the sum to a limited and manageable range. The output sum from the hidden layers is passed on to the output layer, which produces the final network. The feed-forward networks may contain any number of hidden layers, network with a single hidden layer can learn any set of training data that a network with multiple layers can learn, depends upon the complexity of the problem (S.B.Cho 2002).

An input may be either a raw/pre-processed signal or image. Alternatively, some specific features can also be used. If specific features are used as input, there number and selection is crucial and application dependent. Weights are connected between an input and a summing node and affect to the summing operation. The Bias or threshold value is considered as a weight with constant input 1 i.e. x0=1 and w0=θ, usually the weight are randomized in the beginning (W. Finnoff 1994). The neuron is the basic information processing unit of a NN. It consists of: A set of links, describing the neuron inputs, with weights, w1,w2,w3….wn , An adder function (linear combiner) for computing the weighted sum as:

$$v = \sum_{j=1}^{m} w_j x_j \qquad (3.1)$$

And activation function (squashing function) for limiting the amplitude of the neuron output as shown in figure 1

$$y = \varphi(v + b) \text{ ;where, } v = \sum_{j=0}^{m} w_j x_j \quad b = w_o \qquad (3.3)$$

The output at every node can finally calculates by using sigmoid function

$$y = f(x) = \frac{1}{1 + e^{-Kx}} \text{ ; where K is the adaption constant} \qquad (3.4)$$

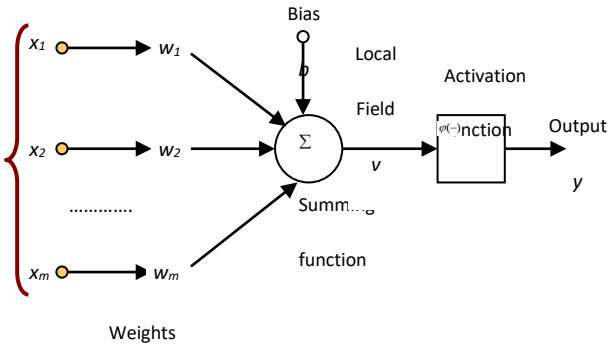

Figure 1: The Functioning of neural network architecture.

3.1 Back Propagation Learning Rules

The Back Propagation (BP) learning algorithm is currently the most popular supervised learning rule for performing pattern classification tasks. It is not only used to train feed forward neural networks such as the multilayer perceptron, it has also been adapted to recurring neural networks (F.J.pineda 1987). The BP algorithm is a generalization of the delta rule, known as the least mean square algorithm (H.Z.Yahya 2005). This algorithm propagates backward the error between the desired signal and the network output through the network. After providing an input pattern, the output of the network is then compared with a given target pattern and the error of each output unit calculated. This error signal is propagated backward, and a closed-loop control system is thus established. The weights can be adjusted by a gradient-descent approach. In order to implement the BP algorithm, a continuous, nonlinear, monotonically increasing, differentiable activation function is required as Logistic Sigmoid function or hyperbolic tangent function.

So that, to provide the training for multi-layer feed forward network to approximate an unknown function, based on some training data consisting of pairs $(x, z) \in S$, the input pattern vector x represents a pattern of input to the network, with desired output pattern vector z from the training set S. The objective function for optimization or minimization is defined in the sum of instantaneously squared error as:

$$E^P = \frac{1}{2}\sum_{j=1}^{J}(T_j - S_j)^2 \qquad (3.1.1)$$

where $(T_j - S_j)^2$ is the squared difference between the actual output of the network on the output layer for the presented input pattern P and the target output pattern vector for the pattern P.

all the network parameters $W^{(m-1)}$ and θ^m, m = 2 · · · M, can be combined and represented by the matrix $W = \lfloor w_{ij} \rfloor$. So that, the error function E can be minimized by applying the gradient-descent procedure as:

$$\Delta W = -\eta \frac{\partial E}{\partial W} \qquad (3.1.2)$$

where η is a learning rate or step size, provided that it is a sufficiently small positive number.

applying the chain rule the equation (3.1.2) can express as:

$$\frac{\partial E}{\partial w_{ij}^{(m)}} = \frac{\partial E}{\partial u_j^{(m+1)}} \frac{\partial u_j^{(m+1)}}{\partial w_{ij}^{(m)}} \qquad (3.1.3)$$

while

$$\frac{\partial u_j^{(m+1)}}{\partial w_{ij}^{(m)}} = \frac{\partial}{\partial w_{ij}^{(m)}}\left(\sum w_j^{(m)} o^{(m)} + \theta_j^{(m+1)}\right) = o_i^{(m)} \qquad (3.1.4)$$

$$\frac{\partial E}{\partial u_j^{(m+1)}} = \frac{\partial E}{\partial o_j^{(m+1)}} \frac{\partial o_j^{(m+1)}}{\partial u_j^{(m+i)}} = \frac{\partial E}{\partial o_j^{(m+1)}} \phi_j^{(m+1)}\left(u_j^{(m+1)}\right)$$

and (3.1.5)

For the output unit m=M-1

$$\frac{\partial E}{\partial o_j^{(m+1)}} = e_j \tag{3.1.6}$$

For the hidden units, m = 1,2,3……,M − 2,

$$\frac{\partial E}{\partial o_j^{(m+1)}} = \sum_{\omega=2}^{j_{m+2}} \frac{\partial E}{\partial u_\omega^{m+2}} \omega_{j\omega}^{m+1} \tag{3.1.7}$$

Define the delta function by

$$\delta_j^{(m)} = \frac{\partial E}{\partial u_p^{(m)}} \tag{3.1.8}$$

for m = m = 2,3……,M. By substituting (3.12), (3.17), and (3.18) into (3.13), we finally obtain the following. For the output units, m = M − 1,

$$\delta_j^{(M)} = -e_j \phi_j^{(M)}\left(u_j^{(M)}\right) \tag{3.1.9}$$

For hidden units, m = 1,……..,M − 2,

$$\delta_j^{(M)} = -e_j \phi_j^{(M)}\left(u_j^{(M)}\right) \sum_{\omega=1}^{J_{m+2}} \delta_\omega^{(m+2)} \omega_\omega^{m+1} \tag{3.1.10}$$

Equations (3.19) and (3.20) provide a recursive method to solve $\delta_j^{(m+1)}$ for the whole network. Thus, W can be adjusted by

$$\frac{\partial E}{\partial \omega_{ij}^{(m)}} = -\delta_j^{(m+1)} o_i^{(m)} \tag{3.1.11}$$

For the activation transfer functions, we have the following relations

For the sigmoid function

$$\phi(u) = \beta\phi(u)[1-\phi(u)] \tag{3.1.12}$$

For the tangent hyperbolic function

$$\phi(u) = \beta\left[1-\phi^2(u)\right] \tag{3.1.13}$$

The update for the biases can be done in two ways. The biases in the (m+1)th layer i.e. θ(m+1) can be expressed as the expansion of the weight W(m), that is, $\theta^{(m+1)} = \left(\omega_{0,1}^{(m)}, \ldots\ldots\ldots\ldots\ldots \omega_{0,J_{m+1}}^{(m)}\right)$. Accordingly, the output o(m) is expanded into $o^{(m)} = \left(1, o_1^{(m)}, \ldots\ldots\ldots\ldots, o_{J_m}^{(m)}\right)$. Another way is to use a gradient-descent method with regard to θ(m), by following the above procedure. Since the biases can be treated as special weights, these are usually omitted in practical applications. The algorithm is convergent in the mean if $0 < \eta < \frac{2}{\lambda_{\max}}$, where λmax is the largest eigenvalue of the autocorrelation of the vector x, denoted as C (R.Battiti 1990).

When η is too small, the possibility of getting stuck at a local minimum of the error function is increased. In contrast, the possibility of falling into oscillatory traps is high when η is too large. By statistically pre-processing the input patterns, namely, de-correlating the input patterns, the excessively large eigenvalues of C can be avoided and thus, increasing η can effectively speed up the convergence. The BP algorithm can be extended or improved by adding a momentum term (M.mangal and M.P.Singh 2007) and known as Gradient Descent with momentum term. As per this learning rule the weight update between output layer and hidden layer is represented by following weight updating equations as:

$$\Delta w_{ho}(s+1) = -\eta \sum_{i=1}^{H} \frac{\partial E}{\partial w_{ho}} + \alpha \Delta w_{ho}(s) + \frac{1}{1-(\alpha \Delta w_{ho}(s))} \quad (3.1.14)$$

Whereas the weight update between hidden layer and input layer can be represent as:

$$\Delta w_{ih}(s+1) = -\eta \sum_{i=1}^{N} \frac{\partial E}{\partial w_{ih}} + \alpha \Delta w_{ih}(s) + \frac{1}{1-(\alpha \Delta w_{ho}(s))} \quad (3.1.15)$$

Where α is the momentum factor, usually $0 < \alpha \leq 1$.

The BP algorithm is a supervised gradient-descent technique, wherein the MSE between the actual output of the network and the desired output is minimized. It is prone to local minima in the cost function. The performance can be improved and the occurrence of local minima reduced by allowing extra hidden units, lowering the gain term, and with modified training with different initial random weights.

3.2 Radial Basis Function

The multi layer feed forward neural networks trained with BP algorithm usually suffers with the convergence problem of local error surface and the use of second order gradient descent term in weight update has significantly improved the performance of multi layer feed forward neural networks (S.Becker 1988) but still there is no guarantee to find the global optimum.

Global optimization for neural nets is especially difficult because the number of distinct local optima can be astronomical. Another important consideration in the choice of optimization algorithms is that neural nets are often ill-condition (S.Saarinen 1993), especially when there are many hidden units. Due to the ill-conditioned nature of multi layer feed forward neural networks it has also been proposed to evaluate the performance of the proposed network with the introduction of gradient descent of RBF. The Gradient Descent RBF methods have been proven to be most effective and fast convergence methods in the problem domain of supervised learning. In this section, we investigate a network structure related to the multi layer feed forward neural network (FFNN), implemented using the Radial Basis Function (RBF-MLP) which suffices the need of locally responsive neurons.

The architecture and training methods of the RBF network are well known (J.Moody, C.J.Darken 1989; T.Poggio 1990; M.T.Musavi 1992; D.Wettschereck 1992; W.P.Vogt 1993 and S.Haykin 1994) & well established. The Radial basis function network (RBFN) is a universal approximator with a solid foundation in the conventional approximation theory. The RBFN is a popular alternative to the multi layer feed forward neural networks, since it has a simple structure and a much faster training process. The RBFN has its origin in performing exact interpolation of a set of data points in a multidimensional space (D.S.Broomhead 1988).

An RBFN is a three layer feed forward network that consists of one input layer, one hidden layer and one output layer as shown in Figure 2, each input neuron corresponds to a component of an input vector x. The hidden layer consists of K neurons and one bias neuron. Each node in the hidden layer uses an RBF denoted with $\phi(r),$ as its non-linear activation function.

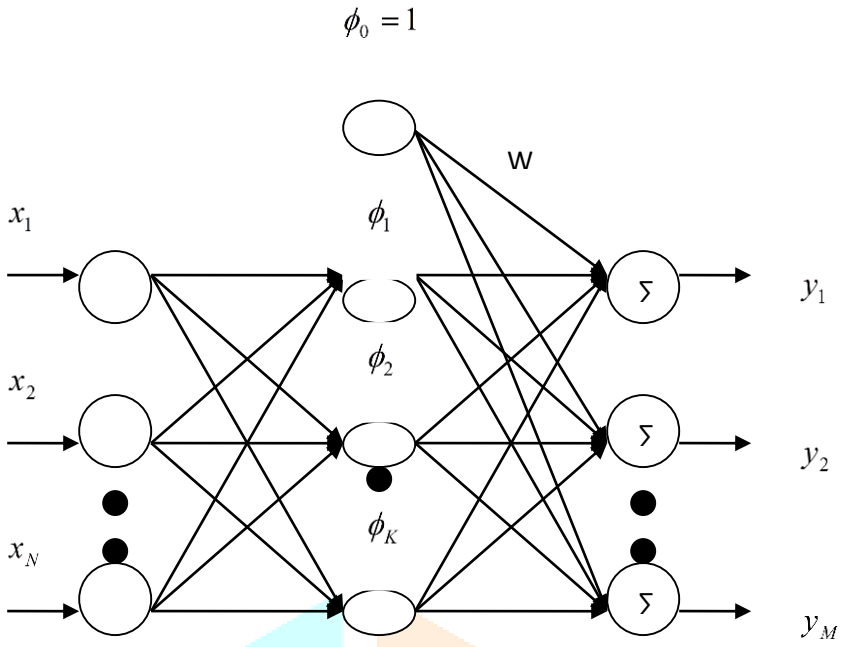

Figure 2: Architecture of the RBFN. The input layer has N nodes; the hidden and the output layer have K and M neurons, respectively. $\phi_0(x) = 1$, corresponds to the bias.

The hidden layer performs a non-linear transform of the input and the output layer this layer is a linear combiner which maps the nonlinearity into a new space. The biases of the output layer neurons can be modeled by an additional neuron in the hidden layer, which has a constant activation function $\phi_0(r) = 1$. The RBFN can achieve a global optimal solution to the adjustable weights in the minimum MSE range by using the linear optimization method. Thus, for an input pattern x, the output of the jth node of the output layer can define as:

$$y_j(x) = \sum_{k=1}^{K} w_{kj}\phi_k(\|x_i - \mu_k\|) + w_{0j} \qquad (3.2.1)$$

for $j = (1,2,.....,M)$ where $y_j(x)$ is the output of the j^{th} processing element of the output layer for the RBFN , w_{kj} is the connection weight from the kth hidden unit to the jth output unit , μ_k is the prototype or centre of the kth hidden unit. The Radial Basis Function $\phi(.)$ is typically selected as the Gaussian function that can be represented as:

$$\phi_k(x_l) = \exp\left(-\frac{\|x_l - \mu_k\|^2}{2\sigma_k^2}\right) \quad \text{for } k = (1,2,.....,K) \qquad (3.2.2)$$

and 1 for $k = 0$ (bias neuron)

Where x is the N- dimensional input vector, μ_k is the vector determining the centre of the basis function ϕ_k and σ_k represents the width of the neuron. The weight vector between the input layer and the kth hidden layer neuron can consider as the centre μ_k for the feed forward RBF neural network.

Hence, for a set of L pattern pairs $\{(x_l, y_l)\}$, (3.2.1) can be expressed in the matrix form as

$$Y = w^T\phi \qquad (3.2.3)$$

where $W = [w_1,.....w_m]$ is a KxM weight matrix, $w_j = (w_{0j},.....w_{kj})^T$, $\phi = [\phi_0,.....\phi_k]$ is a K x L matrix, $\phi_{l,k} = [\phi_{l,1},.....\phi_{l,k}]^T$ is the output of the hidden layer for the lth sample, $\phi_{l,k} = \phi(\|x_l - c_k\|)$, $Y = [y_1, y_{2,}.....y_m]$ is a M X L matrix and $y_{lj} = (y_{l1,}.....y_{lm})^T$.

The important aspect of the RBFN is the distinction between the rules of the first and second layers weights. It can be seen (C.M.Bishop 1995) that, the basis functions can be interpreted in a way, which allows the first layer weights (the parameters governing the basis function), to be determined by unsupervised learning. This leads to the two stage training procedure for RBFN. In the first stage the input data set {xn} is used to determine the parameters of the basis functions. The basis functions are then keep fixed while the second – layer weights are found in the second phase of training. There are various techniques have been proposed in the literature for optimizing the basis functions such as unsupervised methods like selection of subsets of data points (M.A.Kraaijved 1991), orthogonal least square method (S.Chen et al. 1991), clustering algorithm, Gaussian mixture models (C.M.Bishop 1994) and with the supervised learning method.

It has been observed that the use of unsupervised techniques to determine the basis function parameters is not in general an optimal procedure so far as the subsequent supervised training is concerned. The difficulty with the unsupervised techniques arises due to the setting up of the basis functions, using density estimation on the input data and takes no consideration for the target labels associated with the data. Thus, it is obvious that to set the parameters of the basis functions for the optimal performance, the target data should include in the training procedure and it reflects the supervised training. Hence, the basis function parameters for regression can be found by treating the basis function centers and widths along with the second layer weights, as adaptive parameters to be determined by minimization of an error function as shown in equation (3.1.1). This error will minimize along the decent gradient of error surface in the weight space between hidden layer and the output layer. The same error will minimize with respect to the Gaussian basis function's parameter as defined in equation (3.2.2). Thus, we obtain the expressions for the derivatives of the error function with respect to the weights and basis function parameters for the set of L pattern pairs (x^l, y^l) for $l = 1$ to L as:

$$\Delta w_{jk} = -\eta_1 \frac{\partial E^l}{\partial w_{jk}} \tag{3.2.4}$$

$$\Delta \mu_k = -\eta_2 \frac{\partial E^l}{\partial \mu_k} \tag{3.2.5}$$

and

$$\Delta \sigma_k = -\eta_3 \frac{\partial E^l}{\partial \sigma_k} \tag{3.2.6}$$

and

$$y_j^l = \sum_{k=1}^{K} w_{jk} \phi_k(\|x^l - \mu_k^l\|) \tag{3.2.7}$$

and

$$\phi_k(\|x^l - \mu_k^l\|) = \exp(-\frac{\|x^l - \mu_k^l\|^2}{2\sigma_k^2})$$

Hence, from the equation (3.2.4) we have,

$$\Delta w_{jk} = -\eta_1 \frac{\partial E^l}{\partial w_{jk}} = -\eta_1 \frac{\partial E^l}{\partial y_j^l} . \frac{\partial y_j^l}{\partial w_{jk}} = -\eta_1 \frac{\partial E^l}{\partial y_j^l} . \phi_k \|x^l - \mu_k^l\|$$

$$\Delta w_{jk} = -\eta_1 \frac{\partial E^l}{\partial s_j^l(y_j^l)} . \frac{\partial s_j^l(y_j^l)}{\partial y_j^l} . \exp(-\frac{(\|x^l - \mu_k^l\|)^2}{2\sigma_k^2}) = \eta_1 \sum_{j=1}^{M} (d_j^l - y_j^l) . \dot{s}_j^l(y_j^l) . \sum_{k=1}^{K} \exp(-\frac{(\|x^l - \mu_k^l\|)^2}{2\sigma_k^2}$$

or

$$\Delta w_{jk} = \eta_1 \sum_{j=1}^{M} \sum_{k=1}^{K} (d_j^l - y_j^l) \dot{s}_j^l(y_j^l) \exp(-\frac{(\|x^l - \mu_k^l\|)^2}{2\sigma_k^2})$$

So, that (3.2.8)

Now, from the equation (3.2.6) we have

$$\Delta\mu_{ki} = -\eta_2 \frac{\partial E^l}{\partial \mu_{ki}} = -\eta_2 \frac{\partial E^l}{\partial y_j^l} \cdot \frac{\partial y_j^l}{\partial \mu_{ki}}$$

$$= -\eta_2 \frac{\partial E^l}{\partial y_j^l} . w_{jk} . \exp(-\frac{(\|x_i^l - \mu_{ki}^l\|^2)}{2\sigma_k^2}) . (\frac{x_i^l - \mu_{ki}^l}{\sigma_k^2})$$

or $$\Delta\mu_{ki} = \eta_2 \sum_{j=1}^{M} \sum_{k=1}^{K} (d_j^l - y_j^l) \dot{s}_j^l (y_j^l) w_{jk} . \exp(-\frac{(\|x_i^l - \mu_{ki}^l\|)^2}{2\sigma_k^2}) \ (\frac{x_i^l - \mu_{ki}^l}{\sigma_k^2}) \quad (3.2.9)$$

Now, from the equation (3.2.6) we have

$$\Delta\sigma_k = -\eta_3 \frac{\partial E^l}{\partial \sigma_k} = -\eta_3 \frac{\partial E^l}{\partial y_j^l} \cdot \frac{\partial y_j^l}{\partial \sigma_k} = -\eta_3 \frac{\partial E^l}{\partial y_j^l} . w_{jk} \exp(-\frac{(\|x_i^l - \mu_{ki}^l\|)^2}{2\sigma_k^2}) \frac{\|x_i^l - \mu_{ki}^l\|^2}{\sigma_k^3}$$

or, $$\Delta\sigma_k = \eta_3 \sum_{j=1}^{M} \sum_{k=1}^{K} (d_j^l - y_j^l) \dot{s}_j^l (y_j^l) . w_{jk} . \exp(-\frac{(\|x_i^l - \mu_{ki}^l\|)^2}{2\sigma_k^2}) \frac{\|x_i^l - \mu_{ki}^l\|^2}{\sigma_k^3} \quad (3.2.10)$$

So that, we have from equations (3.2.8), (3.2.9) & (3.2.10) the expressions for change in weight vector & basis function parameters to accomplish the learning in supervised way. The adjustment of the basis function parameters with supervised learning represents a non-linear optimization problem, which will typically be computationally intensive and may be prove to finding local minima of the error function.

Thus, for reasonable well-localized RBF, an input will generate a significant activation in a small region and the opportunity of getting stuck at a local minimum is small. Hence, the training of the network for L pattern pair i.e. (x^l, y^l) will accomplish in iterative manner with the modification of weight vector and basis function parameters corresponding to each presented pattern vector. The parameters of the network at the mth step of iteration can express as;

$$w_{jk}(m) = w_{jk}(m-1) + \eta_1 \sum_{j=1}^{M} \sum_{k=1}^{K} (d_j^l - y_j^l) \dot{s}_j^l (y_j^l) . \exp(-\frac{(\|x_i^l - \mu_{ki}^l\|)^2}{2\sigma_k^2}) \quad (3.2.11)$$

$$\mu_{ki}(m) = \mu_{ki}(m-1) + \eta_2 \sum_{j=1}^{M} \sum_{k=1}^{K} (d_j^l - y_j^l) \dot{s}_j^l (y_j^l) . w_{jk} . \phi_k (x_i^l) . (\frac{x_i^l - \mu_{ki}^l}{\sigma_k^2}) \quad (3.2.12)$$

$$\sigma_k(m) = \sigma_k(m-1) + \eta_3 \sum_{j=1}^{M} \sum_{k=1}^{K} (d_j^l - y_j^l) \dot{s}_j^l (y_j^l) . w_{jk} . \phi_k (x_i^l) . \frac{\|x_i^l - \mu_{ki}^l\|^2}{\sigma_k^3} \quad (3.2.13)$$

where $\eta_1, \eta_2 \& \eta_3$ are the coefficient of learning rate.

The discussed gradient decent approach for implementation of RBFNNs system is incremental learning algorithm in which the parameters update for each example (x^l, y^l). The RBFNNs trained by the gradient-decent method is capable of providing the equivalent or better performance compared to that of the multi layer feed forward neural network trained with the back propagation. The gradient decent method is slow in convergence since it cannot efficiently use the locally tuned representation of the hidden layer units. Thus, there is no guarantee that the RBFNN remains localized after the supervised learning. As a result the computational advantage of locality is not utilized. Indeed, in numerical simulations it is found that the subset of the basis functions may evolve to have very broad responses. It has been realized that some of the main advantages of the radial basis function network is fast two-stage training and interpretability of the hidden unit representation.

3.3 Generalized Regression Neural Network

Generalized regression neural networks are a type of radial basis network which is based on the statistical technique. It is frequently used for function approximation and use to solve different type of generalization problems. The predication is one of the generalization problems. It is similar to RBFN network expect the regression part. In this type neural network structure we consider the regression between the given pattern vector of training set and the best estimation of its target output. Its regression process finds the estimated value of y that minimizes the MSE as mentioned in equation (3.1.1). It calculates the joint density function for the input pattern vectors in the feature space and correlated them with the target output pattern vectors because the probability density function is find out from the data with no preconceptions in relation to its form, the system is absolutely general. The generalized regression neural network considers the input pattern vector X and produces the estimated output Y of the target output pattern vector T as (M.T.Hegan 1996):

$$E[Y \mid X] = \frac{\sum_{1}^{n} yP(X,Y)}{\sum_{1}^{n} P(X,Y)} \tag{3.3.1}$$

Where Y is the resultant output of the estimator, X is the input vector, $E[Y \mid X]$ is the likely output value and input pattern vector X and $P(X,Y)$ is joint probability density function of X and Y. Since, the function value is calculated as:

$$Y_j = \frac{\sum_{i=1}^{n} h_i T_{ij}}{\sum_{i=1}^{n} h_i} \tag{3.3.2}$$

Where T_{ij} is the target output of the jth unit of output layer corresponding to input training vector X_i. The output of the ith unit of the hidden layer is define as:

$$h_i = \frac{(\Delta)^2}{e^{2\varphi}} \tag{3.3.3}$$

The square distance between input pattern vector X and the ith training set pattern vector μ_i can define as:

$$\Delta^2 = (X - \mu_i)^T (X - \mu_i) \tag{3.3.4}$$

And,

$$\varphi = \exp\left(-\frac{\|x - \mu_i\|^2}{2\sigma_i^2}\right) \tag{3.3.5}$$

Thus, in this model of neural networks there is no need of training or updating of weights. The substitution for adjustment of weight is with the assignment of weight between jth unit of hidden layer and ith unit of output i.e. w_{ij} to the target T_{ij} from the training set associated with input training vector i and component j of its resultant output vector.

4. Design and Implementation

In our proposed models of predication for weight yield there are three different neural networks models are used to train the normalized input data of last 20 year. The first architecture which we considered is feed forward neural network with Backpropagation learning rule (Levenberg-Marguardt algorithm). The simulation design for this model contained three hidden layers with one output layer. The size of input pattern vector is of 8 X 15 with target pattern is of size 1 X 15 constitutes the training set. Thus the input layer contains the 8 neurons and output layer contains 1 neuron. There are 38, 19 and 13 neurons are considered in the first, second and third hidden layers respectively.

The tangent hyperbolic transfer functions are used in hidden layers and liner function is used for output layer because the tangent hyperbolic function is most efficient transfer function for non-linear real life application data. To measure the performance of the model we are using mean square error (MSE) function. Generalized regression neural network is considered as second architecture for the proposed model. It is basically used for function approximation application but it can be used in prediction application also to provide better results. It is a two layered network where the second layer is output layer and first one is the hidden layer. The output layer contains a single neuron whereas single input pattern is of size 8 X 1. Neurons of the first layer i.e. hidden layer use Radial basis function and the output layer neuron uses the liner function. There are 15 neurons are used for the hidden layer because the size of training set is 8 X 15. In this model the bias is used only for the hidden layer not for the output layer.

Radial basis neural network is considered as third model to accomplish the task of predication based on the data samples in training set. In this model we consider the two layers. First one is the hidden layer which contains the 15 neurons with radial basis function and basis values for each neuron. The second layer is the output layer which contains the single neuron with linear output function with a basis value. Again the training set contains the size of 8 X 15 i.e. 8 inputs for a pattern.

5. Results and Discussion

Multi layer feed forward neural network of topology 8-38-19-13-1 with BP algorithm (Levenberg-Marguardt training function) gives the accuracy of 97.75% in training, testing and validation data. The features for the network topology with best trained network are given below in table-3 which gives the information about the various parameters used in the training and the performance of the neural network.

Table 3: Shows the features and performance index of FFNN model.

FEATURE	**FFNN MODEL**
Training Algorithm	Levenberg-Marquardt
Performance	Mean square error (mse)
Data division	Random
Performance	0.0008
Validation check	4
Best validation performance	*8.47E-05*
MU	0.001
Training (T)	100%
Validation (V)	97%

Testing (T)	93%
Over all (T, V & T)	98.47%
Regression (T v/s O)	97.75%
Network function	newffnn
Weight function	dist
Transfer function	tansig
Hidden layers	3
Network Topology	38-19-13

Here over all training result is 98.47%, which is the best outcomes after training the network and regression in between network output and target values is 97.75%, it means the network output and target values are matches with 97.75% accuracy. So we can say that proposed FFNN model provides 97.75% accuracy in results or in prediction as shown in figure 3 (a and b)

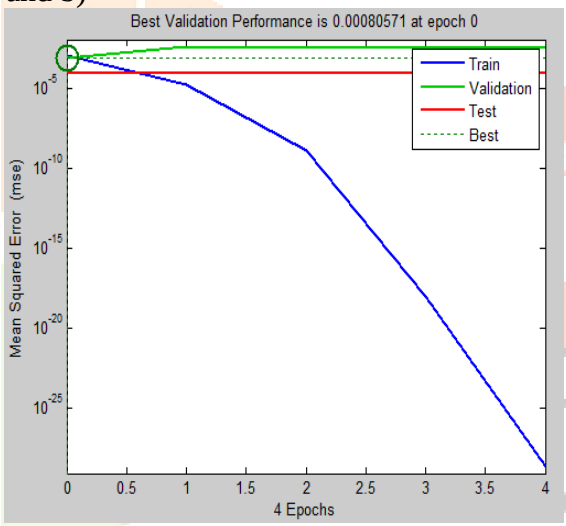

Figure 3 (a): Performance of FFNN for training set

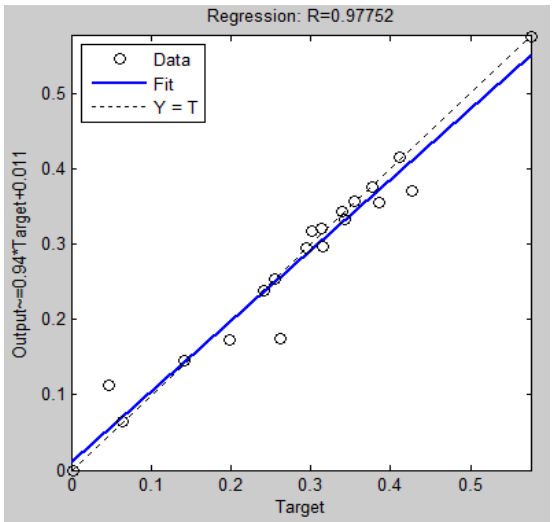

Figure 3 (b): Regression plot in between actual and target output for FFNN

The figures 3 (a) shows that best validation performance is reached 0.00080571 at 0 epoch and the graph gradually falls down it shown the best performance. The figure 3 (b) shows the relationship in between output value and the target value and it is nearly reached 0.97752. It means, the matching of values between target and output values is achieved up to 97%. Generalized regression neural networks with two layers can be created very rapidly with the inputs pattern vectors of size 8 X 15 with 1 X 15 target output pattern with spread $\varphi = 1$. It is obvious that to fit the data more smoothly the larger spread is required. This network gives the accuracy of 99% in training, testing and validation data as shown in figure 4.

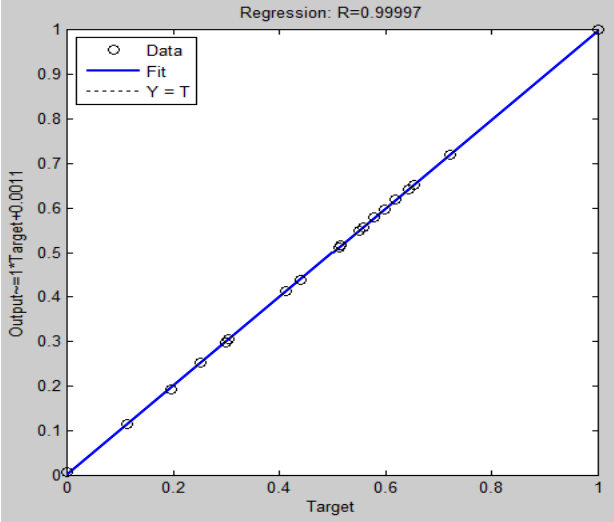

Figure 4: Regression plot in between output and target values for GRNN.

Figure 4 represents the regression plot with R=0.99997 for generalized regression neural network. It shows that generalized regression provides data matching accuracy as 99% and it is more accurate than FFNN. Radial Basis function neural networks with two layers can be created very rapidly with zero error for the inputs pattern vectors of size 8 X 15, target output pattern with 1 X 15 and spread $\varphi = 1$. It is obvious that to fit the data more smoothly the larger spread is required too large spread can cause the problem of non convergence. This network gives the accuracy of 100% in training, testing and validation data as shown in figure 5.

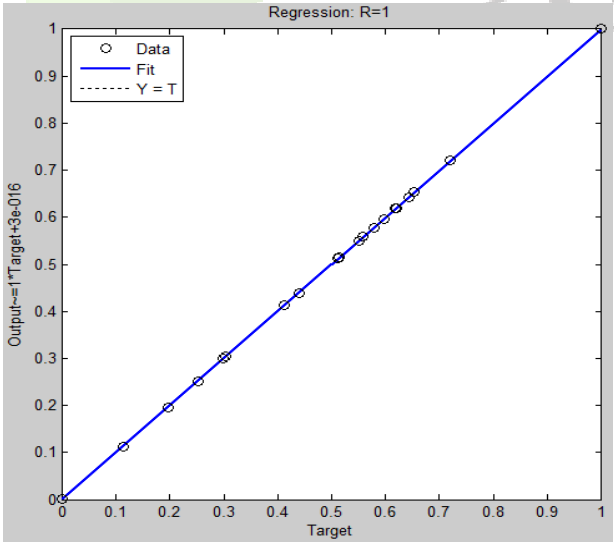

Figure 5: Regression plot in between output and target values for RBFNN model

Figure 5 provides the regression plot with R= 1 for Radial Basis Function neural network. Thus it 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. It can see from the simulation results of all three models of neural networks have different performance for the raining set. The feed forward neural network exhibits 97% accuracy while generalized regression neural network and Radial basis neural network exhibit 99% and 100 % accuracy respectively. These accuracies are for the data of training set. Now, we considered these trained and tuned neural networks to predicate the wheat yield for next six years. The different combinations of 8 input parameters are applied to these trained neural networks to obtain simulated outputs. These simulated outputs exhibit the predication of wheat yield as shown in table 4 and 5. The simulation results are indicating that the best optimal predication for next six year production is obtained by feed forward neural network model with respect to generalized regression neural network and radial basis function neural network model. Feed forward neural network performs as a better estimator of predicator because FFNN network filters the input data with back propagation of the intermediate output and finds the efficient final output. Thus feed forward neural network appears as efficient model for predication of wheat yield in comparison to other two networks GRNN and RBNN.

Table 4: Shows the predicted data, given by all three models in normalize form, for next upcoming six years by using feature scaling method.

YEAR	FFNN MODEL	GRNN MODEL	RBNN MODEL
2013-14	0.40460	0.33670	0.49200
2014-15	0.39680	0.57850	0.54040
2015-16	0.38840	0.65290	0.52730
2016-17	0.40440	0.65270	0.51800
2017-18	0.46580	0.57850	0.57990
2018-19	0.49710	0.57850	0.51800

Table 5: Shows the predicted data, given by all three models in original form, for next upcoming six years (in tonnes).

YEAR	FFNN MODEL	GRNN MODEL	RBNN MODEL
2013-14	489731.75160	365219.04030	418069.02800
2014-15	485122.37280	447505.75650	434539.98360
2015-16	480158.42640	472824.74610	430081.93570
2016-17	489613.56240	472756.68430	426917.06200
2017-18	525897.64680	447505.75650	447982.18910
2018-19	544394.25660	447505.75650	426917.06200

6. 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 yield. Hence to accomplish this objective, we used neural networks as a prediction tool and chose 8 significant factors as input parameters to predict the wheat production. Data is collected from Rajasthan Agricultural Krishi department, Jaipur. The considered eight input factors are such as area under cultivation, annual average temperature, annual rainfall, seed distribution, fertilizer distribution as N, P and K and minimum selling price (MSP). These data are annual and contain of 20 years from 1993 to 2012. In order to get the best model of neural network for this study, three distinct neural network models are tested to find the best model for the optimum predication of wheat production for next 6 years from 2013 to 2018. The simulation results exhibits that feed forward neural network with BP algorithm is found best and suitable neural network model for producing the more optimal predication. As shown in table 5 FFNN model for wheat production has the lowest MSE and consequently is chosen as the preferred or optimum model in comparison of generalized regression neural network and radial basis neural network. All three models are run 100 times to take care of possible bias or noise.

7. References

1. Eveleen Darby, Tezeswari Nettimi, Shilpa Kodali and Liwen Shih (2005) Head and neck cancer metastasis prediction via artificial neural networks. Computational systems bioinformatics conference workshops, IEEE
2. Zhen Zhang, Nan Jing (2008) Radial basis function method for prediction of protein secondary structure, seventh international conference on machine learning and cybernetics. Kunming IEEE
3. Christopher Gan, Visit Limsombunchai, Mike Clemes, Amy Weng (2005) Consumer choice prediction: Artificial neural networks versus logistic models. Commerce division discussion 104: pp. 1-25
4. Mahdi Pakdaman Naeini, Hamidreza Taremian, Homa Baradaran Hashemi (2010) Stock market value prediction using neural network. International conference on computer information systems and industrial management applications (CISIM) IEEE pp. 132-136
5. Jyothi Patil and V.D.Mytri (2013) A prediction model for population dynamics of cotton pest (Trips' tobacco lined) using multilayer-perception neural network. International journal of computer applications 67:pp. 19-26
6. Qeethara Shayea and Ghaleb El-Refea (2013) Predicting the effects of medical waste in the environment using artificial neural networks: A case study. International journal of computer science 10 (1)
7. Reza Ghodsi, Ruzbeh Mirabdollah Yani, Rana Jalali, Mahsa Ruzbahman (2012) Predicting wheat production in Iran using an artificial neural networks approach. International journal of academic research in business and social sciences 2 (2): pp. 34-47
8. J. C. Neves and A. Vieir (2006) Improving bankruptcy prediction with hidden layer learning vector quantization. European accounting review 15 (2): pp. 253-271

9. P. V. G. D. Prasad Reddy, K. R. Sudha, P. Rama Sree, and S. N. S. V. S. C. Ramesh (2010) Software effort estimation using radial basis and generalized regression neural network. Journal of computing 2 (5): pp. 87-93
10. Nidhi Gupta and Manu Paratap Singh (2005) Estimation of Software Reliability with Execution time Model Using the Pattern Mapping Technique of Artificial Neural Network. Journal of Elsevier in Computer and Operation Research 32: pp.187-199
11. A. K. Jain, R. P. W. Duin, and J. Mao (2000) Statistical Pattern Recognition: A Revie. IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (1): pp. 4-37
12. S. Arora, et al. (2010) Performance Comparison of SVM and ANN for Handwritten Devnagari Character Recognition. IJCSI International Journal of Computer Science 7 (3): pp. 1-7
13. U. Bhattacharya, S. Vajda, A. Mallick, B. B. Chaudhuri, and A. Belaid (2004) On the Choice of Training Set, Architecture and Combination Rule of Multiple MLP Classifiers for Multi resolution Recognition of Handwritten Characters. 9th Int'l Workshop on Frontiers in Handwriting Recognition (IWFHR-9).
14. Daniel Svozil, Vladimir KvasniEka, JiE Pospichal (1997) Introduction to multi-layer feed-forward neural networks. Chemometrics and intelligent laboratory systems Elsevier 39: pp. 43-62
15. S. B. Cho (2002) Fusion of neural networks with fuzzy logic and genetic algorithm. IOS Press pp. 363–372
16. W. Finnoff (1994) Diffusion approximations for the constant learning rate Backpropagation algorithm and resistance to local minima. Neural Comp. 6 (2): pp. 285–295
17. O. Sharma, V. Lamba, G.G.S.Pradeep Ghatasala, S.Mohapatra, " Analysing optimal environment for the text classification in deep learning", AIP Conference Proceedings, 2023, doi.org/10.1063/5.0150678
18. V. Solanki, R. K. Sachdeva, V. Lamba, U. Solanki and B. K. Sahoo, "Performance Evaluation of Speed Sensitive Handover in Two-Tier Wireless Networks," IEEE, 2023 6th International Conference on Information Systems and Computer Networks (ISCON), Mathura, India, 2023, pp. 1-8, doi: 10.1109/ISCON57294.2023.10111957.
19. V. Bhardwaj, Virender, M. Kumar, D. Thakur and V. Lamba, "Robotic Process Automation for Automating Business Processes: A use case," IEEE, 2023 7th International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 2023, pp. 762-766, doi: 10.1109/ICCMC56507.2023.10083859.
20. V. Khullar, R. Chhabra, M. Angurala, M. Veeramanickam, K. Singh and V. Lamba, "Indian National Stock Exchange Crude Oil (CL=F) Close Price Forecasting Using LSTM and Bi-LSTM Multivariate Deep Learning Approaches," IEEE, 2023 International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES), Greater Noida, India, 2023, pp. 839-842, doi: 10.1109/CISES58720.2023.10183624.

21. Vikas Lamba, Susheela Hooda, Vikas Solanki, Vivek Bhardwaj, Umesh Kumar Lilhore, Vikas Khullar, " Nifty Junior (CNX Nifty) Value Prediction by Applying Depth Psychology Approach in Machine Learning", IEEE, 10th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), 2022.
22. Vivek Bhardwaj, K. Venkata Rahul, Mukesh Kumar, Vikas Lamba, " Analysis and Prediction of Stock Market Movements using Machine learning", IEEE, 4th International Conference on Inventive Research in Computing Applications (ICIRCA), 2022.
23. Susheela Hooda, Vikas Lamba, Sharanpreet Kaur, Vidhu Kiran Sharma, Raju Kumar, Vandana Sood, " Impact of Software Quality on "GA-FC" Software Testing Technique", IEEE, International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES), 2022.
24. Vikas Lamba, Tarun Agarwal, "A Review on Role of machine learning models on Coronary Heart Disease Detection Accuracy", IJCRT, Vol. 10 (2), 2022.
25. Vikas Lamba, Susheela Hooda, Rakesh Ahuja, " Wheat Yield Prediction Using Feedforward Neural Networks", IEEE, International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), November, 2021.
26. Vikas Lamba, Susheela Hooda, "Comparative Analysis of Artificial Neural Networks for Predicting Nifty 50 value in The Indian Stock Market", IEEE, 5th International Conference on Information Systems and Computer Networks (ISCON), 2021.
27. Vikas Solanki, Vikas Lamba, "Importance of Artificial Intelligence and Machine Learning in fighting with COVID-19 Epidemic", IEEE, 5th International Conference on Information Systems and Computer Networks (ISCON), 2021.
28. Susheela Hooda, Vikas Lamba, "AI and Soft Computing Techniques for Securing Cloud and Edge Computing: A Systematic Review", IEEE, ISCON, 2021.
29. Rakesh Ahuja, Amandeep Kaur, Vikas Lamba, "Securing Copyright of Digital Video by Exploiting Quantized DC coefficients", IEEE, International Conference on Signal Processing, Computing and Control (ISPCC), November, 2021.
30. Susheela Hooda, Vikas Lamba, Amandeep Kaur, "Performance Evaluation of "Ga-Fc"
31. Technique for Aspect-Oriented Software System", Indian Journal of Science and Technology (INDJST), 14(41), November, 2021.
32. Vikas Lamba, Zuber Ahmad Mir and Tanveer Ahmad Dar, "Application of Big Data using Wireless Sensor Networks in imparting Smart Nation", IJERA, September, 2020.
33. Vikas Lamba, Tanveer Ahmad Dar, "Study and Analysis of Privacy Preservation and Security Concerns in Big Data", IJERA, September, 2020.

34. Vikas Lamba and Shaina Arora, "A Study of Technologies to Further Research In Health Care Data Security In Medical Report Using Block Chain" International Journal of Advanced engineering Research and Science (IJAERS), June, 2020.
35. Vikas Lamba and Anooja A., "A Survey: To Explore the Factors Behind Inaccuracy in Cost Effort Estimation and Providing Improvements and Suggestions", Studies in Indian Place Names, March, 2020.
36. V.S.Dhaka, Vikas Lamba and Anubhav Nandan, "Application Layer proxy detection, prevention with predicted load optimization", IEEE, International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2016), December 23-25, 2016.
37. Vikas Lamba and V.S. Dhaka published paper on Compressive Neural Network Techniques Application in Wheat Yield Prediction in IJSETR, August, 2015.
38. Vikas Lamba and V.S.Dhaka Published paper on Wheat Yield Prediction Using Artificial Neural Network and Crop Prediction Techniques (A Survey) in IJRASET, September, 2014.
39. Vikas Lamba and Sunita Lamba Published paper on GMDH for Wheat Yield Prediction in IJEECS, February, 2014.
40. Vikas Lamba and Sunita Lamba Published paper on An Adapted Group Method of Data Handling for Abrupt Data Analysis in International Journal of Engineering and Computer Science, February, 2014.
41. F. J. Pineda (1987) Generalization of back-propagation to recurrent neural networks. Physical Rev Letter 59: pp. 2229–2232
42. H. Z. Yahya, D. Lakmal, and K. A. Seneviratne (2005) Stability analysis of a three-term back propagation algorithm. Neural Networks 18 (10): 1341-1347
43. R. Battiti, and F. Masulli (1990) BFGS optimization for faster automated supervised learning. In: Proc. Int. Neural Network Conf. France 2:pp. 757-760
44. M. Mangal, Manu Pratap Singh (2007) Analysis of pattern classification for the multidimensional parity-bit-checking problem with hybrid evolutionary feed-forward neural network. Neurocomputing, Elsevier 70: pp. 1511-1524
45. S. Becker, & Y. Le Cun (1988) Improving the convergence of the back-propagation learning with second order methods. In D. S. Touretzky, G. E. Hinton, & T. J. Sejnowski (1988 Eds.) Proceedings of the Connectionist Models Summer School. San Mateo, CA: Morgan Kaufmann, pp. 29-37
46. S. Saarinen, R. Bramley and G. Cybenko (1993) Ill-conditioning in neural network training problems. Siam J. of Scientific Computing, 14:pp. 693-714
47. J. Moody and C.J. Darken (1989) Fast learning in networks of locally-tuned processing units. Neural Computation, 1(2): pp. 281-294
48. T. Poggio, and F. Girosi (1990) Regularization algorithms for learning that are equivalent to multilayer networks. SCIENCE 247: pp. 978-982

49. M. T. Musavi , W. Ahmed , K. H. Chan , K. B. Faris , *D. M.* Hummels (1992) On the training of radial basis function classifiers. Neural Networks 5(4): pp. 595-603
50. D. Wettschereck, T. G. Dietterich (1992) Improving the performance of radial basis function networks by learning center locations. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, (1992 Eds.) Advances in Neural Information Processing Systems San Mateo CA: Morgan Kaufmann, 4*:* pp. 1133-1140
51. W. P. Vogt (1993) Dictionary of Statistics and Methodology: A Nontechnical Guide for the Social Sciences. Thousand Oaks: Sage
52. S. Haykin (1994) NEURAL NETWORKS. Macmillan College Publishing Company Inc. New York
53. D. S. Broomhead and D. Lowe (1988) Multivariable functional interpolation and adaptive networks.Complex System 2: pp. 321-355
54. C. M. Bishop (1995) Neural Networks for Pattern Recognition. Oxford University Press
55. M.A. Kraaijveld and R.P.W. Duin (1991) Generalization capabilities of minimal kernel-based networks. Proc. Int. Joint Conf. on Neural Networks (Seattle, WA, July 8-12) IEEE, Piscataway U.S.A. I-843 - I-848
56. S. Chen, C. F. N. Cowan, and P. M. Grant (1991) Orthogonal least squares algorithm for radial basis function networks. IEEE Transactions on Neural Network ISSN 1045-9227 2 (2) : pp. 302-309
57. C. M. Bishop, "Novelty detection and neural network validation (1994) IEEE Proceedings: Vision, Image and Signal Processing. Special issue on applications of neural networks 141 (4) : pp. 217–222
58. V. S. Dhaka and Vikas Lamba, "Comprehensive Neural Network Techniques Application in Wheat yield Prediction", Published in International Journal of Science, Engineering and Technology Research (IJSETR), Vol. 4, Issue 8, August 2015, pp. 2936-2944.