JOURNAL OF INFORMATION, KNOWLEDGE AND RESEARCH IN COMPUTER SCIENCE AND APPLICATIONS

PERFORMANCE ANALYSIS OF THE FEED FORWARD NEURAL NETWORKS WITH GRADIENT DESCENT LEARNING ALGORITHMS AND STRUCTURE SIMILARITY ALGORITHM FOR RECOGNITION OF HANDWRITTEN *'MARATHI'* CHARACTERS

[1]*GIRISH S. KATKAR*, [2]*DR. M. P. SINGH*

[1]Head, Dept. of Computer Science, Arts,Comm & Sci. College, Koradi, Nagpur(M.S.),
[2]Inst. of Computer and Information Sc., Dr. B. R. Ambedkar University, AGRA

girishkatkar2007@rediffmail.com

ABSTRACT : *In this paper we are evaluating the performance of multilayer feedforward neural network trained with various form of gradient descent approach & structure similarity algorithm for the generalized classification of Handwritten 'Marathi' language in the different classes. In this performance analysis, five Handwritten 'Marathi' characters from different people are collected and stored them as image. The MATLAB function is used to determine the densities of these scanned images after partitioning the image into 16 portions. These 16 densities for each character are used as an input pattern for the feed forward Multilayer Neural Network and structure similarity algorithm. The training of feed forward Multilayer Neural Network is conducted with Resilient Back propagation (RP) algorithm and Gradient Descent with Momentum & Adaptive Learning Rate algorithm (GDX). We consider five trials of each sample for network architecture with both the learning algorithms so, in total we conducted 120 experiments and found that the performance of GDX is better than RP for correct classification. The results of experiments also indicate that the structure similarity approach is better than gradient descent approaches for off-line pattern recognition. The rate of correct classification is found 88 % for structure similarity algorithm and this is most promising than other two methods.*

KEYWORDS : *Character recognition, ANN, GDX, MLP, SSIM*

1. Introduction

In general the recognition is a basic property of all human beings when a person sees an object, he or she first gathers all information about the object and compares its properties and behaviors with the existing knowledge stored in the mind. If we find a proper match, we recognize it [1]. The concept of recognition is simple in the real world environment, but in the world of computer science, recognizing any object is an amazing feat. Pattern Recognition is the field concerned with machine recognition of regularities in noisy or complex environment [2]. Alternatively, pattern recognition is the search for structure in data [3]. Ultimately, it involves placing an input pattern into one of possibly many decision classes. Pattern belonging to the same class shares similar properties [4].

The functionality of the human brain is amazing it is not comparable with any machines or software. The act of recognition can be divided into two broad categories namely:

1) Concrete item recognition, it involves the recognition of spatial samples such as fingerprints, weather maps, pictures and physical objects and the recognition of temporal samples such as, waveforms and signatures.

2) Abstract item recognition, it involves the recognition of a solution to a problem, an old conversation or argument.

More generally the pattern recognition spans a number of scientific disciplines, uniting them in search for a solution to the common problem of recognizing the pattern of a given class and assigning the name of identified class. Pattern recognition is the categorization of input data into identifiable classes through the extraction of significant attributes of the data from irrelevant background detail. A pattern class is a category determined by some common attributes. Therefore, a pattern is the description of a category member representing a pattern class. Pattern recognition by machine involves techniques for assigning patterns to their classes automatically with as little human interventions as possible. Pattern classification aims to classify data (patterns) based on either a priori knowledge or on statistical information extracted from the patterns. The patterns to be classified are usually groups of measurements or observations, defining points in an appropriate multidimensional space [5]. In the literature the complete pattern recognition system is defined with the components like; sensor that gathers the observations to be classified or described, a feature extraction mechanism that computes numeric or symbolic information from the observations; and a classification or description scheme that does the actual job of classifying or describing observations which relying on the extracted features. A very common domain in computer science for the pattern recognition is identified in the terms of character recognition or classification in precise manner.

Character recognition plays an important role in today's life [6]. It can solve many complex problems of real life. An example of character recognition is classification of Handwritten English alphabets. The classic difficulty of being able to correctly recognize even typed optical language symbols is the complex irregularity among pictorial representations of the same character due to variations in fonts, styles and size. This irregularity undoubtedly widens when one deals with handwritten characters [7]. The analysis of handwritten and printed documents has been a subject of intensive research for the last decades. The interest devoted to this field is not only explained from the scientific point of view, but also in terms of the social benefits that convey those systems. Two examples of interesting applications are the analysis of old handwritten archive manuscripts and sketching or calligraphic interfaces. The analysis of ancient documents is a growing interest in Europe and its main concern is not only the digitization but the extraction of knowledge from ancient documents to convert them to digital libraries, so that these documents can be edited and published, contributing to the diffusion and preservation of artistic and cultural heritage. Concerning to sketching interfaces, it is a joint interest between the fields of Pattern Recognition and Human Computer Interaction, which allows computers to integrate a natural way of interaction based on handwritten strokes which are interpreted as textual annotations or graphical gestures. Although the analysis of textual handwritten documents has an intensive activity, the analysis of hand-drawn documents with graphical alphabets is an emerging subfield. Due to the fact that architectural, cartographic and musical documents use their own alphabets of symbols (corresponding to the domain dependent graphic notations used in these documents), the automatic interpretation of such documents requires specific processes, within the field of Graphics Recognition, more than the field of Cursive Script Recognition. Two major differences between the two problems can be stated. Cursive script recognition has the context information in one dimensional way, but graphical alphabets usually are bi-dimensional. In addition, the use of syntactical knowledge, and lexicons, is more effective in text recognition than in diagrammatic notations because of the variability of structures and alphabets of the latter.

Even though numerous commercial OCR system vendors, claim near-perfect text recognition (close to 99.9%), these accuracy rates are rarely achieved in practice. Several methods for recognition of Arabic, Chinese, English, Japanese, Korean, Russian, Spanish scripts have been proposed. Among Indian script, some pioneering work has been done on Bengali, Oriya, Telgu, Urdu scripts and character recognition systems for this script are ready for commercialization. Still there is lot scope do develop commercial character recognition systems for several Indian regional languages.

In this paper we are evaluating the performance of multilayer feedforward neural network trained with various form of gradient descent approach & structure similarity algorithm for the generalized classification of Handwritten '*Marathi*' language in the different classes. The training of feed forward Multilayer Neural Network is conducted with Resilient Back propagation (RP) algorithm and Gradient Descent with Momentum & Adaptive Learning Rate algorithm (GDX) for the given training set of handwritten '*Marathi*' characters. The training set consists with samples of hand written '*Marathi*' characters. These samples are collected from five different people and store them as images. The density function of MATLAB has used to determine the features from these sample images. Each training samples is considered as a pattern of 16 features i.e. 16*1 vectors. These pattern vectors are used to train the feed forward neural network with both the training algorithms. The performances of these learning techniques are evaluated and analyzed on the neural network architecture. Therefore, to accomplish this analyses and performance evaluation, five trails for each sample is considered with both the training algorithms on the neural network so, in total we conducted 120 experiments and found that the performance of GDX is better than RP to produce the correct classification for the presented input pattern samples. The major drawbacks those are identified during this performance analysis with both the gradient descent learning algorithm is that both the learning algorithm explore the problem of local minima's and slow convergence rate(some time non-convergence), which limit the scope of real time application [8]. A local minimum is defined as a point such that all points in a neighborhood have an error value greater than or equal to the error value in that point [9]. These samples of training sets are presented to the structure similarity algorithm and the performance of this algorithm is also analysised for off-line character recognition for the given samples of the training set. The analysis procedure indicates that the performance of GDX algorithm is better than RP algorithm in terms of epochs and accuracy in the classification rate. The results of experiments also indicate that the structure similarity approach is better than gradient descent approaches for off-line pattern recognition. The rate of correct classification is found 88 % for structure similarity algorithm and this is most promising than other two methods.

Section 2 of this paper describes the literature review for handwritten character recognition. The Section 3 of this paper describes the approach which is used for the feature extraction for the '*Marathi'* characters and preparation of the input output patterns pairs for the training set. Section 4 for this paper describes the generalized gradient descent learning rule in detail. The description of this method represents the implementation details of this method. The section 5 describes the structure similarity algorithm and its implementation. The section 6 describes the architecture and design of the network used and the simulation design for structure similarity algorithm. Section 7 shows the results of the experiments with the discussion. Section 8 describes the conclusion followed by the references.

2. Literature Review

Pattern recognition is an emerging area of the machine learning and intelligence. The problem of pattern recognition has been considered in many ways. The one of the most popular way is in the form of the pattern classification. Pattern classification is a problem in which the machine can distinguish the different input stimuli in meaningful categorization according to the present features in these inputs. This meaningful categorization can exhibit with some already predefined classes depending upon the nature of problem. Pattern recognition and its application have been studied from very long period of time and there are various methods have been proposed to accomplish the task of pattern classification [10-16].

3. Feature Extraction for the *'Marathi'* Characters

India is a multi-lingual and multi-script country comprising of eighteen official languages. One of the defining aspects of Indian script is the repertoire of sounds it has to support. Because there is typically a letter for each of the phonemes in Indian languages, the alphabet set tends to be quite large. Most of the Indian languages originated from *BRAMHI* script. These scripts are used for two distinct major linguistic groups, Indo-European languages in the north, and Dravidian languages in the south [59]. *DEVNAGARI* is the most popular script in India. It has 11 vowels and 33 consonants. They are called basic characters. Vowels can be written as independent letters, or by using a variety of diacritical marks which are written above, below, before or after the consonant they belong to. So that when vowels are written in this way they are known as modifiers and the characters so formed are called conjuncts. Sometimes two or more consonants can combine and take new shapes. These new shape clusters are known as compound characters. These types of basic characters, compound characters and modifiers are present not only in *DEVNAGARI* but also in other scripts. In this paper we are concerning about the classification for hand written characters of *'Marathi'* language. A sample of *'Marathi'* characters set which is obtained from five different people for each character in their handwriting.

In this paper we are considering the feature extraction from the input stimuli by using density function of MATLAB. In our approach we have considered the input data in the form of five different set of each hand written *'Marathi'* characters by five different peoples. It is quite natural that the five different people considered the different hand writing and different writing style for every character. So, in this way we have total 240 samples of the character sets. Each character set is containing different examples of same sample in different hand writing. Now to prepare our training set of input output pattern pairs, we consider each scanned hand written character as a bit map image. This bitmap image of a character is now partitioned in 16 equal parts. After this the row wise & column wise mean of each partition is obtained. Thus, we obtained 16 real number values for each scanned image. Hence, every scanned image is now considered in the form of input pattern vector of 16 dimensions. Thus, we consider each input pattern vector in 16*1 row matrix form. The example of one such character & its representation in input pattern vector form can be shown in figure 1.

	[2.514091; 2.135156; 2.292104; 2.433057; 2.016629; 2.177502; 2.102489; 2.065017; 1.924126; 2.103291; 2.158773; 2.246736; 1.893306; 2.180586; 2.324606; 2.457668]

Figure 1: The input pattern vector of order (16*1) for character अ from MATLAB function

Therefore in this way we can determine the input pattern vector for every scanned image. Thus we have the training set which consist with 240 input pattern vectors of size 16*1. It means that if we consider the entire training set as matrix of training pattern than it will be the order of 16X240. To distinguish each character set from other character set for classification during the training the target output is needed. Therefore to classify these hands written *'Marathi'* characters there must be 48 different classes. As we know that the single neuron can differentiate between two classes, so to differentiate 48 different classes we required minimum six output neurons. Hence, the target output pattern for each input pattern will be of dimension six. In this proposed method we are considering the target output pattern for each character in six bit binary form as shown in figure 2.

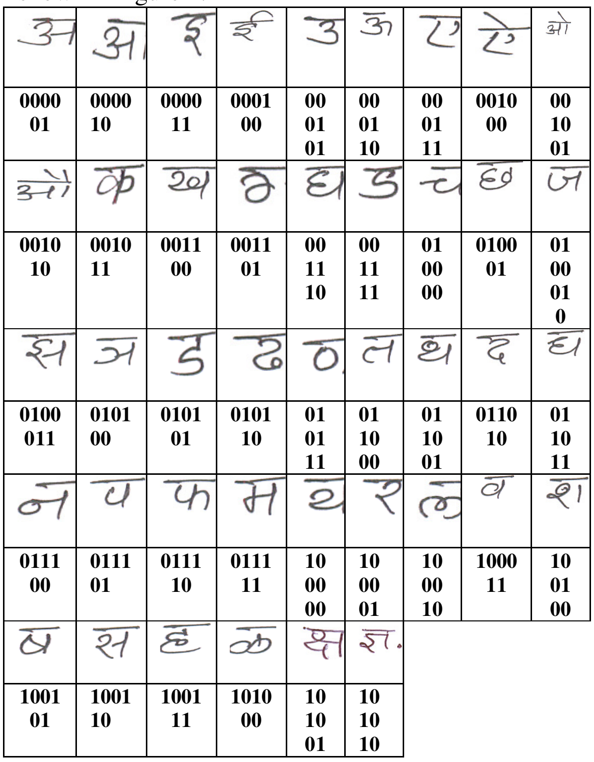

Figure 2: Target output pattern for different handwritten *'Marathi'* Characters

Thus, we have constructed the training set of input output patterns pairs to analyze the performance of multilayer feed forward neural networks with extended back propagation learning algorithms and structure similarity approach. We have also constructed our test pattern set to verify the performance of networks. Our test pattern set consist with another set of hand written characters i.e. Two set each by two different people. Thus, our test pattern set consist with 82 samples. Input pattern for these test character set are constructed in same manner as we did for training set pattern.

The experiments described in this study were designed to evaluate the performance of feed forward neural network when evolved with the back propagation algorithm for training algorithm Resilient Back propagation (RP) and Gradient Descent with Momentum & Adaptive Learning Rate Back propagation (GDX). To accomplish this task two neural networks are considered namely NN1 and NN2. NN1 is using the GDX learning rule and NN2 is using the RP learning rule.

4. Generalized gradient descent learning

Neural network architectures can be classified as feed- forward and feedback neural network architectures. The most common neural networks used in the OCR systems are the multilayer perceptron (MLP) of the feed forward networks and the Kohonen's Self Organizing Map (SOM) of the feedback networks. One of the interesting characteristics of MLP is that in addition to classifying an input pattern, they also provide a confidence in the classification [60]. These confidence values may be used for rejecting a test pattern in case of doubt [61, 62]. A detailed comparison of various NN classifiers is made by M. Egmont-Petersen [63, 64]. Various other works have also reported for classification of handwritten English alphabet classification with the use of back propagation type NN classifier [65]. Genetic algorithm based feature selection and classification along with fusion of NN and Fuzzy logic is reported [66, 67] for English alphabet classification, but no any work is reported for Indian languages. 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. 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; each connection may have a different strength or weight. The weights on these connections encode the knowledge of a network. Data enters at the inputs and passes through the network, layer by layer, until it arrives at the outputs. During normal operation, that is when acts as a classifier, there is no feedback between layers. This is why they are called feed forward neural networks. The 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 data 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 [68]. However, neural network with a single hidden layer may take longer to train.

An input may be either a raw/preprocessed 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. $x_0=1$ *and* $w_0=\theta$, usually the weight are randomized in the beginning [69, 70]. The neuron is the basic information processing unit of a NN. It consists of: A set of links, describing the neuron inputs, with weights, $w_1, w_2, w_3 \ldots w_n$, An adder function (linear combiner) for computing the weighted sum as:

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

And activation function (squashing function) for limiting the amplitude of the neuron output as shown in figure4

$$y = \varphi(v + b) \qquad (2)$$

where,

$$v = \sum_{j=0}^{m} w_j x_j \qquad (3)$$

$$b = w_o$$

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

$$y = f(x) = \frac{1}{1 + e^{-Kx}} \quad ;$$

where K is the adaption constant (4)

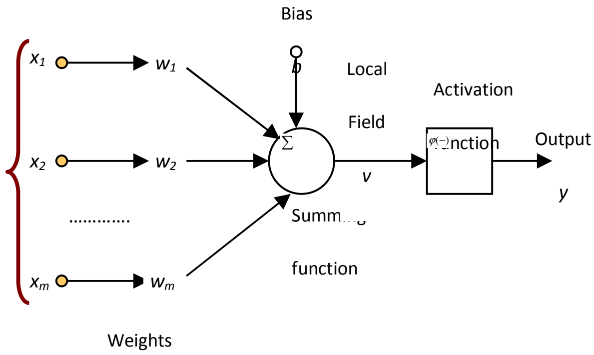

Figure 3: The Functioning of neural network architecture.

The supervised learning mechanism is commonly used to train the feed forward multilayer neural network architecture. In this learning process a pattern is presented at the input layer. The pattern will be transformed in its passage through the layers (hidden) of the network until it reaches the output layer. The units in the output layer all belong to a different category. The outputs of the network as they are now compared with the outputs as they ideally would have been if this pattern were correctly classified, in the later case unit with the correct category would have had the largest output value and the output values of the other output units would have been very small. On the basis of this comparison all the connection weights are modified a little bit to guarantee that, the next time this same pattern is presented at the inputs, the value of the output unit that corresponds with the correct category is a little bit higher than it is now and that, at same time, the output values of all the other incorrect outputs are little bit lower than they are now. The differences between the actual outputs and the idealized outputs are propagated back from the top layer to lower layers to be used at these layers to modify connection weights. Thus it is consider as back propagation learning algorithm.

The backpropagation (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 [71, 72]. The BP algorithm is also called as the generalization of the delta rule because it can extend for any number of successive hidden layers. The BP overcomes the limitations of the perceptron learning enumerated by Minsky and Papert [73]. 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.
Hence, 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 $\boldsymbol{x}$ represents a pattern of input to the network, with desired output pattern vector $\boldsymbol{z}$ from the training set $\boldsymbol{S}$. The objective function for optimization or minimization is defined in the sum of instantaneously squared error as:

$$E^P = \frac{1}{2}\sum_{j=1}^{J}(T_j - S_j)^2 \tag{5}$$

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, $m = 2 \cdot \cdot \cdot M$, can be combined and represented by the matrix $W = [w_{ij}]$. So that, the error function E can be minimized by applying the gradient-descent procedure as:

$$\Delta\ W = -\eta\frac{\partial E}{\partial W} \tag{6}$$

where η is a learning rate or step size, provided that it is a sufficiently small positive number.
Applying the chain rule the equation (6) can express as

$$\frac{\partial E}{\partial w_{ij}^{(m)}} = \frac{\partial E}{\partial u_j^{(m+1)}}\frac{\partial u_j^{(m+1)}}{\partial w_{ij}^{(m)}} \tag{7}$$

while

$$\frac{\partial u_j^{(m+1)}}{\partial w_{ij}^{(m)}} = \frac{\partial}{\partial w_{ij}^{(m)}}\left(\sum w_j^{(m)} o^{(m)} + \theta_j^{(m+1)}\right) = o_i^{(m)} \tag{8}$$

and

$$\frac{\partial E}{\partial u_j^{(m+1)}} = \frac{\partial E}{\partial o_j^{(m+1)}}\frac{\partial o_j^{(m+1)}}{\partial u_j^{(m+i)}} = \frac{\partial E}{\partial o_j^{(m+1)}}\phi_j^{(m+1)}\left(u_j^{(m+1)}\right) \tag{9}$$

For the output unit $m=M-1$

$$\frac{\partial E}{\partial o_j^{(m+1)}} = e_j \tag{10}$$

For the hidden units, $m = 1,2,3.........,M - 2$,

$$\frac{\partial E}{\partial o_j^{(m+1)}} = \sum_{\omega=2}^{j_{m+2}}\frac{\partial E}{\partial u_\omega^{m+2}}\omega_{j\omega}^{m+1} \tag{11}$$

Define the delta function by

$$\delta_j^{(m)} = \frac{\partial E}{\partial u_p^{(m)}} \tag{12}$$

for $m = m = 2,3.........,M$. By substituting (7), (11), and (12) into (9), we finally obtain the following.
For the output units, $m = M - 1$,

$$\delta_j^{(M)} = -e_j\phi_j^{(M)}\left(u_j^{(M)}\right) \tag{13}$$

For hidden units, $m = 1,........,M - 2$,

$$\delta_j^{(M)} = -e_j\phi_j^{(M)}\left(u_j^{(M)}\right)\sum_{\omega=1}^{J_{m+2}}\delta_\omega^{(m+2)}\omega_\omega^{m+1} \tag{14}$$

Equations (13) and (14) provide a recursive method to solve $\delta_j^{(m+1)}$ for the whole network. Thus, **W** can be adjusted by:

$$\frac{\partial E}{\partial \omega_{ij}^{(m)}} = -\delta_j^{(m+1)} o_i^{(m)} \tag{15}$$

For the activation transfer functions, we have the following relations
For the logistic function

$$\phi(u) = \beta\phi(u)[1 - \phi(u)] \tag{16}$$

For the *tanh* function

$$\phi(u) = \beta\left[1 - \phi^2(u)\right] \tag{17}$$

The update for the biases can be done in two ways. The biases in the $(m+1)^{th}$ layer i.e. $\theta^{(m+1)}$ can be expressed as the expansion of the weight $W^{(m)}$, that is, $\theta^{(m+1)} = \left(\omega_{0,1}^{(m)},...................\omega_{0,J_{m+1}}^{(m)}\right)$. Accordingly, the output $\mathbf{o}(m)$ is expanded into $o^{(m)} = \left(1, o_1^{(m)},............., o_{J_m}^{(m)}\right)$. Another way is to use a gradient-descent method with regard to $\boldsymbol{\theta}^{(m)}$, 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 $\lambda_{\max}$ is the largest eigen value of the autocorrelation of the vector **x**, denoted as **C** [74]. 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 preprocessing 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. PCA preconditioning speeds up the BP in most cases, except when the pattern set consists of sparse vectors. In practice, η is usually chosen to be $0 < \eta < 1$ so that successive weight changes do not overshoot the minimum of the error surface. The BP algorithm can be extended or improved by adding a momentum term [75] and known as Gradient Descent with momentum term. Now we are considering this extended form of Back propagation learning algorithm to accomplish the proposed analysis. This learning procedure is defined in neural network toolbox of MATLAB as TRAINGDX (Gradient descent with momentum & adaptive back propagation) function. As per this learning rule the weight update between output layer and hidden layer is represented by following weight updating equations as:

$$\Delta w_{ho}(s+1) = -\eta \sum_{i=1}^{H} \frac{\partial E}{\partial w_{ho}} + \alpha \Delta w_{ho}(s) + \frac{1}{1-(\alpha \Delta w_{ho}(s))} \quad (18)$$

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

$$\Delta w_{ih}(s+1) = -\eta \sum_{i=1}^{N} \frac{\partial E}{\partial w_{ih}} + \alpha \Delta w_{ih}(s) + \frac{1}{1-(\alpha \Delta w_{ho}(s))} \quad (19)$$

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 by eliminating the influence of the magnitude of the partial derivative on the step size of the weight update. The update of each weight is according to the sequence of signs of the partial derivatives in each dimension of the weight space. This extended form of BP algorithm is known as Resilient propagation (RP) learning algorithm for the multilayer feed-forward neural networks [76]. In this learning procedure the update for each weight or bias $w_{ij}(m)$ is given according to the following procedure:

$$C = g_{ij}^{(m)}(t-1).g_{ij}^{(m)}(t) \quad (20)$$

$$\Delta_{ij}^{(m)}(t) = \begin{Bmatrix} \min\{\eta_0^{+}\Delta_{ij}^{(m)}(t-1), \Delta_{\max}\}, C > 0 \\ \max\{\eta_0^{-}\Delta_{ij}^{(m)}(t-1), \Delta_{\min}\}, C < 0 \\ \Delta_{ij}^{(m)}(t-1), C = 0 \end{Bmatrix} \quad (21)$$

$$\Delta w_{ij}^{(m)}(t) = \begin{Bmatrix} -sign(g_{ij}^{(m)}(t)).\Delta_{ij}^{(m)}(t), C \geq 0 \\ -\Delta w_{ij}^{(m)}(t-1), C < 0 \end{Bmatrix} \quad (22)$$

$$\begin{aligned} & g_{ij}^{(m)}(t) = 0, C < 0 \\ & w_{ij}^{(m)}(t+1) = w_{ij}^{(m)}(t) + \Delta w_{ij}^{(m)}(t) \end{aligned} \quad (23)$$

Here, $0 < \eta_0^{-} < 1 < \eta_0^{+}$, and typically $\eta_0^{+} = 1.2$ and $\eta_0^{-} = 0.5$. The value of $\Delta_{ij}^{(m)}(0)$ is not critical to the algorithm, and is selected as a positive constant Δ_0. The upper and lower bounds, denoted by $\Delta_{\max}$ and $\Delta_{\min}$, respectively, are used to restrict overflow / underflow problems of floating – point variables. A smaller value of $\Delta_{\max}$ such as 1.0 may result in a smoothened behavior of the decrease in error. The RP algorithm is robust against the choice of its initial parameters. In comparison with BP algorithm the number of learning steps is significantly reduced and computational complexity of RP at each step is considerably smaller [77].

Hence, in this paper to accomplish the task of classification for *'Marathi'* characters, the feed forward multilayer neural network with back propagation learning rule and its two extended variants namely Adaptive Learning Rate Back Propagation (GDX) and Resilient Back propagation (RP) algorithm are used. The performance of these extended back propagation learning methods is evaluated and analyzed. Further, the simple structure similarity algorithm used on the same feed forward neural network architectures to analyze their performance for the training set of *"Marathi"* characters.

5. Structure Similarity Index Algorithm

Perceptual image quality metrics have explicitly accounted for human visual system (HVS) sensitivity to sub-band noise by estimating thresholds above which distortion is just-noticeable. Objective methods for assessing perceptual image quality traditionally attempted to quantify the visibility of errors (differences) between a distorted image and a reference image using a variety of known properties of the human visual system. A recently proposed class of quality metrics, known as structural similarity (SSIM), models perception implicitly by taking into account the fact that the HVS is adapted for extracting structural information (relative spatial covariance) from images and specific SSIM implemented both in the image space and the wavelet domain.

The motivation behind the structural similarity approach for measuring image quality is that the HVS has evolved to do visual pattern recognition in order to be able to extract the structure or connectedness of natural images. Based on this observation, it makes sense that a useful perceptual quality metric would emphasize the structure of scenes over the lighting effects. If PSNR (Peak Signal to Noise Ratio) value decreases quality of Image Increases and Vice versa with respect to the different image enhancement technique will give the different values for PSNR.

The structural similarity (SSIM) index is a method for measuring the similarity between two images [78]. The SSIM index is a full reference metric, in other words, the measuring of image quality based on an initial uncompressed or distortion-free image as reference. SSIM is designed to improve on traditional methods like peak signal-to-noise ratio (PSNR) and mean squared error (MSE), which have proven to be inconsistent with human eye perception. The difference with respect to other techniques mentioned previously such as MSE or PSNR is that these approaches estimate perceived errors; on the other hand, SSIM considers image degradation as perceived change in structural information. SSIM is commonly used for testing the quality of various lossy images. The SSIM index is a decimal value between 0 and 1. A value of 0 would mean zero correlation with original image, and 1 means the exact same image, so through this index, image comparison methods can be effectively compared. It is defined as:

$$SSIM(x,y)=[1(x,y)]^{\alpha}c[(x,y)]^{\beta}s[(x,y)]^{\gamma} \quad (24)$$

Here *(x, y)* are two images. *1 (x, y)* is the luminance comparison. *c (x, y)* is contrast comparison and s*(x,y)* is the structural comparison between two images and α>0, β>0 and γ>0 are used to adjust the importance of three parameters. In practice, one usually requires a single overall quality measure of the entire image, so mean SSIM (MSSIM) index is used to evaluate the overall image quality. The difference with respect to other techniques mentioned previously such as MSE or PSNR is that these approaches estimate perceived errors; on the other hand, SSIM considers image degradation as perceived change in structural information. Structural information is the idea that the pixels have strong inter-dependencies especially when they are spatially close. These dependencies carry important information about the structure of the objects in the visual scene. The SSIM metric is calculated on various windows of an image.

Structural information is the idea that the pixels have strong inter-dependencies especially when they are spatially close. These dependencies carry important information about the structure of the objects in the visual scene. The SSIM metric is calculated on various windows of an image. The measure between two windows x and y of common size *N*×*N* can represent as:

$$SSIM(x,y)=\frac{2\mu_x\mu_y+c_1)(2\sigma_{xy}+c_2)}{\mu_x^2+\mu_y^2+c_1)(\sigma_x^2+\sigma_y^2+c_{2)}} \quad (25)$$

With μ_x is the average of x, μ_y is the average of y; $\sigma_x^2$the variance of x; σ_y^2 the variance of y; $\sigma_{xy}$the covariance of x and y and $c_1=(k_1L)^2$, $c_2=(k_2L)^2$ two variables to stabilize the division with weak denominator.

In order to evaluate the image quality this formula is applied only on *LUMA*. *LUMA* represents the brightness in an image (the "black-and-white" or achromatic portion of the image). The resultant SSIM index is a decimal value between -1 and 1, and value 1 is only reachable in the case of two identical sets of data. Typically it is calculated on window sizes of 8×8. The window can be displaced pixel-by-pixel on the image but the authors propose to use only a subgroup of the possible windows to reduce the complexity of the calculation. Structural dissimilarity (DSSIM) is a distance metric derived from SSIM (though the triangle inequality is not necessarily satisfied).

$$DSSIM(x,y)=\frac{1-SSIM(x,y)}{2} \quad (25)$$

The peak signal-to-noise ratio, often abbreviated PSNR, is an engineering term for the ratio between the maximum possible power of a signal and the (or codec type) and same content. It is most easily defined via the mean squared error (MSE) which for two m × n monochrome images I and K where one of the images is considered a noisy approximation of the other is defined as MSSIM i.e. Measurement of structure similarity algorithm used for image structure analysis [79]. Thus the mean square error can define as [80]:

$$MSE=\frac{1}{mn}\sum_{i=0}^{m-1}[I(i,j)-K(i,j)^2 \quad (26)$$

Researcher has utilized this PSNR to judge the improved image quality with the help of MSE as follows:

$$PSNR=10\log_{10}(\frac{MAX_1^2}{MSE}) \quad (27)$$

$$=20\log_{10}(\frac{MAX_1}{\sqrt{MSE}}) \quad (28)$$

Here, $\boldsymbol{MAX_I}$ is the maximum possible pixel value of the image. When the pixels are represented using 8 bits per sample, this is 255. More generally, when samples are represented using linear PCM with B bits per sample, MAX_I is $2B-1$. For color images with three RGB values per pixel, the definition of PSNR is the same except the ***MSE*** is the sum over all squared value differences divided by image size and by three [81]. Typical values for the PSNR in noisy image and video compression are between 30 and 50 dB, where higher is better. When the two images are identical the MSE will be equal to zero, resulting in an infinite PSNR.

6. Simulation Design and Implementation detail

In this simulation framework we have consider two different multilayer feed forward neural network architectures, first multilayer neural network architecture (NN1) consist with 16 neurons in the input layer, 50 neuron in first hidden layer, 30 in second and 10 in third hidden layer and 6 neurons in output layer. Thus this neural network has the **16-50-30-10-6** architecture (one input layer, three hidden layers and one output layer). The second multilayer neural network architecture (NN2) consist with 16 neurons in the input layer, 50 neuron in first hidden layer, 25 in second and 15 in third hidden layer and 6 neurons in output layer. Thus this neural network has the **16-50-25-15-6** architecture (one input layer, three hidden layers and one output layer).

In this propose work, there are two experiments were conducted on two different neural network architectures with two different methods of learning i.e. Adaptive Learning Rate Back Propagation (GDX) and Resilient Back propagation (RP) algorithm on the given training set. There is another experiment is conducted with structural similarity (SSIM) index for the same training set. The training set consists with five different set of samples for handwritten *'Marathi'* characters from five different people. These *'Marathi'* characters are converted into their density function for input patterns by using MATLAB function. Each input pattern is presented with 16 features. Thus each input is a row vector of order 16*1. We consider five trails on three different values of learning rate parameters (η), the mean of five trails for each value of η is considered as the final result. The performance evaluation for GDX and RP learning algorithm are analyzed. Further the performance of these gradient descent methods are compared with structural similarity (SSIM) index method.

The learning methods those are used to train the neural network architectures are applied to neural networks for training in MATLAB environment. The MATLAB functions TRAINGDX and TRAINRP are used to train the neural networks. TRAINGDX is a network training function that updates weight and bias values according to gradient descent momentum and an adaptive learning rate. The TRAINGDX function considers the following inputs as shown in table 1.

NET	Neural network.
Pd	Delayed input vectors.
Tl	Layer target vectors.
Ai	Initial input delay conditions.
Q	Batch size.
TS	Time steps.
VV	Empty matrix [] or structure of validation vectors.
TV	Empty matrix [] or structure of test vectors.

Table 1: Input information for function TRAINGDX

This function returns the following output values as shown in table 2.

NET	Trained network.
TR	Training record of various values over each epoch:
TR.epoch	Epoch number.
TR.perf	Training performance.
TR.vperf	Validation performance.
TR.tperf	Test performance.
TR.lr	Adaptive learning rate.
Ac	Collective layer outputs for last epoch.
El	Layer errors for last epoch.

Table 2: Output values from the function TRAINGDX

Parameters with their values are used to train the neural network architecture with TRAINGDX learning rules are shown in table 3.

Epochs	1000	Maximum number of epochs to train
Goal	0	Performance goal
Lr	0.01	Learning rate
lr_inc	1.05	Ratio to increase learning rate
lr_dec	0.7	Ratio to decrease learning rate
max_fail	5	Maximum validation failures
max_perf_inc	1.04	Maximum performance increase

Mc	0.9	Momentum constant.
min_grad	1e-10	Minimum performance gradient
Show	25	Epochs between displays (NaN for no
Time	Inf	Maximum time to train in seconds

Table 3: Parameters used in TRAINGDX during training

TRAINGDX can train any network as long as its weights, net input and transfer function have derivative functions. In this learning rule back propagation is used to calculate derivative of performance with respect to weights and bias value, each variable is adjusted according to the gradient descent with momentum as described in equation (18) & (19). In this rule the learning rate is of adaptive nature. Hence in each epoch if performance decreases towards the goal then learning rate increases by factor lr_inc and if performance increases by more than factor max_perf_inc the learning rate is adjusted by the factor lr_dec and the change which increased the performance, is not made. The training in this learning function can be stopped due to anyone of following condition:

1) The maximum number of EPOCHS (repetitions) is reached.
2) The maximum amount of TIME has been exceeded.
3) Performance has been minimized to the GOAL.
4) The performance gradient falls below MINGRAD.
5) Validation performance has increase more than MAX_FAIL times since the last time it decreased (when using validation).

Now we consider our second training algorithm i.e. Resilient Back propagation (RP). In MATLAB this learning rule is implemented with the function name TRAINRP. TRAINRP is a network training function that updates weight according to the sequence of signs of the partial derivatives in each dimension of the weight space. The TRAINRP function considers the following inputs as shown in table 4.

NET	Neural network.
TR	Training record of various
Pd	Delayed input vectors.
Tl	Layer target vectors.
Ai	Initial input delay conditions.
Q	Batch size.
TS	Time steps.
VV	Either empty matrix [] or structure of validation vectors.
TV	Either empty matrix [] or structure of test vectors.

Table 4: Input information for function TRAINRP

This function returns the following output values as shown in table 5.

NET	Trained network.
TR	Training record of various values over each epoch:
TR.epoch	Epoch number.
TR.perf	Training performance.
TR.vperf	Validation performance.
TR.tperf	Test performance.
Ac	Collective layer outputs for last epoch
El	Layer errors for last epoch.

Table 5: Output values from the function TRAINRP

Following parameters with their values are used to train the neural network architecture with TRAINRP learning rules are shown in below in the table 6.

Epochs	1000	Maximum number of epochs to
Show	25	Epochs between displays (NaN for no
Goal	0	Performance goal
Time	Inf	Maximum time to train in seconds
min_grad	1e-6	Minimum performance
max_fail	5	Maximum validation failures
showCommand Line	0	Generate command-line
showWindow	1	Show training GUI
lr	0.01	Learning rate
delt_inc	1.2	Increment to weight change
delt_dec	0.5	Decrement to weight change
delta0	0.07	Intial pweight change
deltamax	50.0	Maximum weight change

Table 6: Parameter used in TRAINRP during training

TRAINRP can train any network as long as its weights, net input and transfer function have derivative functions. In this learning rule back propagation is used to calculate derivative of performance with respect to weights and bias variables, each variable is adjusted according to the learning equation as described in equations 20,21 and 22. The training in this learning function can be stopped due to any one of the following conditions occur:

1. The maximum number of EPOCHS (repetitions) is reached.
2. The maximum amount of TIME has been exceeded.
3. Performance has been minimized to the GOAL.
4. The performance gradient falls below MINGRAD.
5. Validation performance has increased more than MAX_FAIL times since the last time it decreased (when using validation)

7. Results and discussion

Results of both the experiment with two different approaches are discussed here. Result of first approach exhibit the performance evaluation analysis for the feed-forward neural network architectures with two extended back propagation learning algorithms i.e. Gradient descent with momentum & adaptive back propagation (TRAINGDX) and Resilient Back propagation (RP) algorithm on the given training set. Whereas results of the second approach exhibit the performance evaluation analysis for the same sample pattern sets with structural similarity (SSIM) index. The performances of all the tree approaches are analysised. The performance evaluation is considered on the basis of rate of correct classification i.e. recognition, first in between TRAINGDX and TRAINRP for the NN1 and NN2 neural network architectures then after between both the learning methods and structural similarity index for the sample patterns as well as for the test or sample patterns not used in the training set. The results for the first performance analysis are presented in table 7 as follows:

	Simulation Results for Classification of Handwritten Marathi Characters using GDX				**Simulation Results for Classification of Handwritten Marathi Characters using RP**			
Sample	Sample size	Recognition	Non Recognition	Percentage of Recognition	Sample size	Recognition	Non Recognition	Percentage of Recognition
	30	19	11	63.33	30	19	11	63.33
	30	19	11	63.33	30	17	13	56.66
	30	19	11	63.33	30	19	11	63.33
	30	19	11	63.33	30	19	11	63.33
	30	19	11	63.33	30	19	11	63.33
	30	19	11	63.33	30	18	12	60.00
	30	19	11	63.33	30	18	12	60.00
	30	19	11	63.33	30	18	12	60.00
	30	19	11	63.33	30	19	11	63.33
	30	19	11	63.33	30	18	12	60.00
	30	19	11	63.33	30	19	11	63.33
	30	19	11	63.33	30	18	12	60.00
	30	19	11	63.33	30	19	11	63.33
	30	19	11	63.33	30	20	10	66.66
	30	17	13	56.66	30	18	12	60.00
	30	18	12	60.00	30	19	11	63.33
	30	19	11	63.33	30	18	12	60.00
	30	19	11	63.33	30	17	13	56.00
	30	19	11	63.33	30	18	12	60.00
	30	18	12	60.00	30	16	14	53.33
	30	19	11	63.33	30	18	12	60.00
	30	18	12	60.00	30	18	12	60.00
	30	19	11	63.33	30	18	12	60.00
	30	17	13	56.00	30	15	15	50.00
	30	18	12	60.00	30	17	13	56.00
	30	19	11	63.33	30	19	11	63.33
	30	19	11	63.33	30	18	12	60.00
	30	18	12	60.00	30	17	13	56.00
	30	17	13	56.00	30	16	14	53.33

ळ	30	19	11	63.33	30	18	12	60.00
क्ष	30	19	11	63.33	30	17	13	56.00
ज्ञ	30	18	12	60.00	30	17	13	56.00

Table 7a: Simulation Results for Classification of Handwritten Marathi Characters using GDX and RP for NN1

Simulation Results as shown in the table 7 for Classification of Handwritten Marathi Characters using GDX and RP learning algorithms on NN1 neural network architecture are indicating that the character recognition rate of Gradient descent with momentum and adaptive learning rate back propagation (GDX) is more than Resilient back propagation (RP) algorithm. On an average recognition rate for GDX is 61.33 percentages and it is 60 percentages for RP. The simulation results as shown in table 8 for classification of handwritten Marathi characters using GDX and RP learning algorithm on NN2 neural network architecture are also indicating that the character recognition rate of Gradient descent with momentum and adaptive learning rate back propagation (GDX) is more than Resilient back propagation (RP) algorithm. On an average recognition rate for GDX is 60 percentages and it is 56 percentages for RP. It shows that GDX is more efficient than RP for the given training set. The NN1 architecture is exhibiting more efficient performance than NN2 architecture. The performance of both the neural network architecture can see from the table 7 and 8. The rate of recognition is slightly higher for NN1 architecture with respect to NN1. The reason is quite obvious because as the more number of neurons are used in hidden layer the performance for nonlinear classification will improve.

	Simulation Results for Classification of Handwritten Marathi Characters using GDX				**Simulation Results for Classification of Handwritten Marathi Characters using RP**			
Sample	Sample size	Recognition	Non Recognition	Percentage of Recognition	Sample size	Recognition	Non Recognition	Percentage of Recognition
अ	30	18	12	60.00	30	17	13	56.66
इ	30	18	12	60.00	30	17	13	56.66
ए	30	17	13	56.66	30	17	13	56.66
ओ	30	19	11	63.33	30	18	12	60.00
क	30	17	13	56.66	30	16	14	53.33
ख	30	18	12	60.00	30	18	12	60.00
ग	30	18	12	60.00	30	18	12	60.00
घ	30	19	11	63.33	30	18	12	60.00
च	30	18	12	60.00	30	17	13	56.66
ज	30	19	11	63.33	30	17	13	56.66
झ	30	19	11	63.33	30	18	12	60.00
त्र	30	19	11	63.33	30	18	12	60.00
ट	30	18	12	60.00	30	17	13	56.66
ड	30	19	11	63.33	30	19	11	63.33
ण	30	18	12	60.00	30	17	13	56.66
त	30	18	12	60.00	30	17	13	56.66
थ	30	17	13	56.66	30	16	14	53.33
द	30	18	12	60.00	30	17	13	56.00
न	30	18	12	60.00	30	17	13	56.00
प	30	18	12	60.00	30	17	13	56.00
फ	30	19	11	63.33	30	18	12	60.00
ब	30	17	13	56.00	30	16	14	53.33
म	30	19	11	63.33	30	18	12	60.00
र	30	17	13	56.00	30	15	15	50.00
ल	30	19	11	63.33	30	18	12	60.00
स	30	17	13	56.00	30	16	14	53.33
श	30	18	12	60.00	30	17	13	56.00
ष	30	18	12	60.00	30	17	13	56.00
ह	30	18	12	60.00	30	17	13	56.00
ळ	30	19	11	63.33	30	18	12	60.00
क्ष	30	18	12	60.00	30	17	13	56.00
ज्ञ	30	18	12	60.00	30	17	13	56.00

Table 7b: Simulation Results for Classification of Handwritten Marathi Characters using GDX and RP for NN2

Training Results for Classification of Handwritten *'Marathi'* Characters for NN1 using GDX and RP training algorithms for given training set are shown from figure 4 to 9 for resultant values of Epochs, MSE, Gradient, validation checks, and regression.

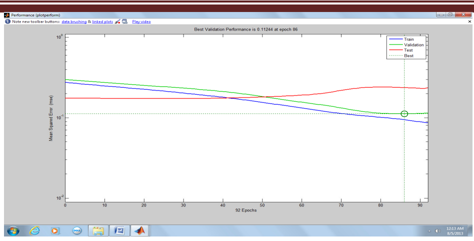

Figure 4: Performance evaluation for epochs and MSE for GDX

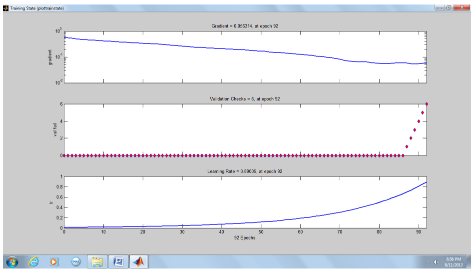

Figure 5: Performance evaluation for epochs, gradient and validation check for GDX

The performance evaluation of figure 1 is showing that for given training set the best performance validation is 0.11244 at epoch 86 and Mean Square Error value is 0.1124. MSE is greater than zero, it means there is bias in the sample data and it is less than 2% (i.e. 98% accuracy in the result). An MSE of zero, meaning that the estimator $\hat{\theta}$ predicts observations of the parameter θ with perfect accuracy, is the ideal, but is practically never possible. A value of MSE is used for comparative purposes. Two characters (standard and sample) are compared using their MSEs as a measure of how well they explain a given set of observations. Both linear regression techniques such as analysis of variance estimate the MSE as part of the analysis and use the estimated MSE to determine the statistical significance of the factors or predictors under study. The goal of experimental design is to construct experiments in such a way that when the observations are analyzed, the MSE is close to zero relative to the magnitude of at least one of the estimated treatment effects.

The performance evaluation of figure 2 is showing that for given training set gradient value is 0.05at epoch 92. Total 6 Validation checks are performed and learning rate is 0.89 i.e. 89%. The gradient of a point is a vector pointing in the direction of the steepest slope or grade at that point. The steepness of the slope at that point is given by the magnitude of the gradient vector. The gradient value shows how a scalar field changes in other directions, rather than just the direction of greatest change.

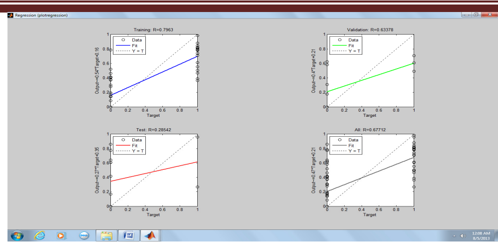

Figure 6: Performance evaluation of Regression for GDX

The performance evaluation of figure 3 is showing that for given training set Regression value for Training is 0.7963, validation is 0.6337, test is 0.2854 and for all parameters regression value is 0.6771.

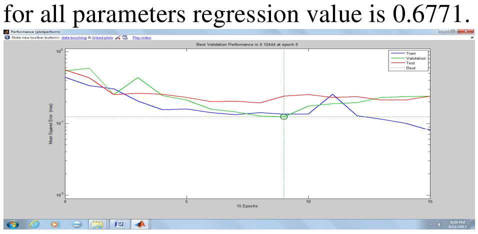

Figure 7: Performance evaluation for epochs and MSE for RP

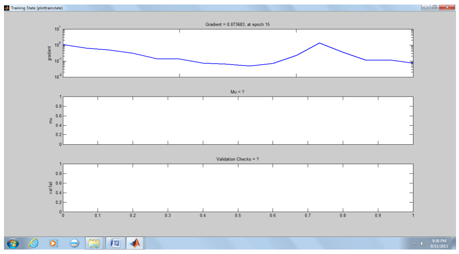

Figure 8: Performance evaluation for epochs, gradient and validation check for RP

The performance evaluation of figure 4 is showing that for given training set the best performance validation is 0.11244 at epoch 86 and Mean Square Error value is 0.1244. MSE is greater than zero, it means there is bias in the sample data and it is less than 2% (i.e. 98% accuracy in the result). The performance evaluation of figure 5 is showing that for the given training set gradient value is 0.073 at epoch 15.

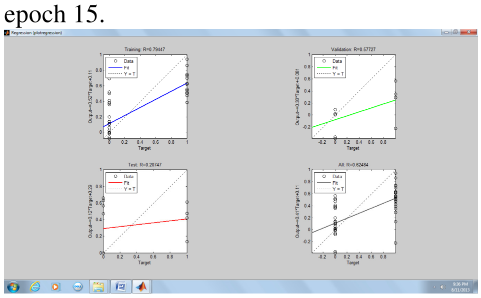

Figure 9: Performance evaluation of Regression for RP

The performance evaluation of figure 6 is showing that for given training set Regression value for Training is 0.7944, validation is 0.5772, test is 0.207 and for all parameters regression value is 0.624.

Characters	**Measured Structure Similarity Recognition Rate in %**				
	Sample 1	**Sample 2**	**Sample 3**	**Sample 4**	**Sample 5**
अ	15	37	88	20	20
इ	20	15	25	12	12
ए	29	45	22	34	27
ओ	29	33	49	53	55
क	32	15	31	22	17
ख	45	32	35	18	31
ग	11	27	26	24	17
ध	33	23	15	22	45
च	34	33	23	21	16
ज	63	54	88	44	64
झ	54	45	64	46	41
त	32	25	44	49	42
ट	11	23	21	15	20
ड	32	31	27	30	29
ण	50	47	51	48	49
त	29	45	22	34	27
थ	11	27	26	24	17
द	29	33	19	23	35
न	24	28	27	21	28
प	31	28	27	31	20
फ	26	37	29	21	27
ब	32	31	27	30	29
म	15	27	18	20	20
र	36	38	36	29	31
ल	45	47	48	41	39

Table 8: Handwritten *'Marathi'* Script Recognition Accuracy

The performance evaluation of both the training algorithm for the given training set with NN1 architecture are indicating from figure 1 to 6 that the performance of GDX is better than RP algorithm in terms of number of epochs, mean square error and gradient value. It can be observed that the GDX algorithm takes more number of epochs to train the system as compare to RP. MSE value for is less in GDX as compare to MSE value .Gradient value for GDX is more than gradient value for RP. Gradient has been increased concurrently in GDX Feed Forward Network. It means the character recognition rate has been increased in GDX. Hence GDX Feed Forward Neural Network is more efficient to recognize handwritten *'Marathi'* characters.

The experiment for structure similarity index is carried out for the standard images of *'Marathi'* scripts with handwritten *'Marathi'* characters of 240 samples for comparison. The performances of accuracy for SSIM are shown in the following table 8. Our goal is to provide the noise free image near to original image and it has been checked by MSSIM (Measurement of Structure Similarity Algorithm). As the MSSIM values increases the noise decrease so MSSIM and Noise values are running inversely proportional

Handwritten *'Marathi'* Script Recognition Accuracy is shown in the table 8 the structure similarity between Handwritten *'Marathi'* characters (sample) and standard character the recognition rate is 11% to maximum 88%. It can be achieved more if there is less noise in the sample character.

In this experimental work, an attempt is made to apply measured structure similarity approach to off-line recognition of handwritten *'Marathi'* characters. Maximum performance rate of SSIM was found to up to 88 percent which is most promising than other methods. But for most of the handwritten *'Marathi'* characters structure similarity is below 50 percent. These results suggest applying image enhancement techniques to remove the noise from the images of handwritten Marathi characters and reapply structure similarity algorithm to check percentage of structure similarity between standard *'Marathi'* characters and sample handwritten *'Marathi'* characters.

The performance of all the three approaches i.e. TRAINGDX & TRAINRP of neural networks and SSIM of structure similarity index are compared with each other on the basis of percentages of successful classification for the given samples of handwritten *'Marathi'* characters. The performance evaluation can show in the table 9.

	Classification of Handwritten Marathi Characters using GDX	Classification of Handwritten Marathi Characters using RP	Structure Similarity (SSIM) value is more than or equal to 20%
Sample	Percentage of	Percentage of	Percentage of

	Recognition	Recognition	Recognitio n
अ	63.33	63.33	100
इ	63.33	56.66	20
ए	63.33	63.33	100
ओ	63.33	63.33	80
क	63.33	63.33	60
ख	63.33	60.00	80
ग	63.33	60.00	60
घ	63.33	60.00	80
च	63.33	63.33	80
ज	63.33	60.00	80
झ	63.33	63.33	100
ञ	63.33	60.00	100
ट	63.33	63.33	100
ड	63.33	56.66	20
ण	63.33	63.33	100
त	63.33	63.33	80
थ	63.33	63.33	60
द	63.33	60.00	80
न	63.33	60.00	60
प	63.33	60.00	80
फ	63.33	63.33	80
ब	63.33	60.00	80
म	63.33	63.33	100
र	63.33	60.00	100
ल	63.33	63.33	100
स	63.33	56.66	20
श	63.33	63.33	100
ष	63.33	63.33	80
ह	63.33	63.33	60
ळ	63.33	60.00	80
क्ष	63.33	60.00	60
ज्ञ	63.33	60.00	80

Table 9: Performance evaluation for successful Classification of Handwritten *'Marathi'* Characters using GDX, RP and SSIM

Maximum performance rate of SSIM for successful classification is found up to 88 percent which is most promising with respect to methods of neural networks as it can be observed from the table 9. But for the recognition of handwritten Marathi characters noise should be removed from the images to enhance character recognition rate. In future work some image enhancement techniques should be used to improve the quality of images before testing structure similarity between standard Marathi character and sample handwritten Marathi character. These Handwritten Marathi Characters has been taken from various users (subject) and it provides accurate results of recognizing the data.

8. Conclusion

In this present work the Performance evaluation of multilayer neural networks trained with Gradient descent with momentum & adaptive back propagation (TRAINGDX) and Resilient Back propagation (RP) learning algorithms for classification of handwritten *'Marathi'* characters are considered and analysised. This analysis is performed in terms of Number of epochs, mean square error and gradient value. An attempt is made to apply measured structure similarity approach to off-line recognition of handwritten *'Marathi'* characters. Number of epochs, mean square error and gradient value for GDX is less than Number of epochs, mean square error and gradient value for RP. MSE has been reduced concurrently in RP Feed Forward Network. It means the character recognition rate has been increased in GDX. Hence GDX Feed Forward Neural Network is more efficient to recognize handwritten Marathi characters. The performance of trained multilayer neural networks is further evaluated with maximum performance rate of SSIM and analysis.

The results of the experiments are indicating and suggesting that a successful attempt is made to apply measured structure similarity approach to off-line recognition of handwritten Marathi characters. Maximum performance rate of SSIM was found up to 88 percent which is most promising than other methods. But for most of the handwritten Marathi characters structure similarity is below 20 percent. These results suggest applying image enhancement techniques to remove the noise from the images of handwritten Marathi characters and reapply structure similarity algorithm to check percentage of structure similarity between standard Marathi characters and sample handwritten Marathi characters. It has been observed that, Structure Similarity index value can be achieved up to 99.99 percent for the handwritten

Marathi characters and the standard Marathi character that looks same in structure.

Although the current network was trained on the full character data, it would be relatively easy to train a set on just the extracted feature data. The modification has been made, but after several thousand minutes of computer time on numerous different machines no such net has yet been made to settle. It is expected that such a combined approach would maintain the desirable feature of the feature extraction method of generally making human-like mistakes while keeping the neural net method's identification speed. Careful adjustments to both the network's topography and the feature extraction process could radically improve the accuracy. Overall the results of the experiment were promising but far from good in their present form. If one method had to be chosen over the other, the results here would indicate that the feature extraction method using standard AI techniques for classification would be the better choice. The possibility of a hybrid algorithm makes other options possible, and quite possibly the most potential lies down this path.

The system created in this work facilitates the various branches that are using handwriting as the base for the work in the field of criminology, investigating agencies. Now they need not only rely on the figure prints of the person but also handwriting recognition system can make their work easier. Various bank forgeries can also be avoided by using these systems in the bank and other related institutions.

9. References

[1] Christenson,P., Maurer, A., and Miner,G.," Handwriting recognition by neural network.",(2005).

[2] Source: http://www.acm.org/ubiquity/, Ubiquity, vol.5 (7), (2004).

[3] K. Fukushima and N. Wake, "*Handwritten alphanumeric character recognition by the neocognitron*," IEEE Trans. on Neural Networks, 2(3) 355-365 (1991).

[4] Manish Mangal and Manu Pratap Singh, "Analysis of Classification for the Multidimensional Parity-Bit-Checking Problem with Hybrid Evolutionary Feed-forward Neural Network." Neurocomputing, Elsevier Science, (70) 1511-1524 (2007).

[5] D. Aha and R. Bankert, "*Cloud classification using error-correcting output codes*", Artificial Intelligence Applications: Natural Resources, Agriculture, and Environmental Science, 11(1) 13-28 (1997).

[6] Friedman, M., and Kandel, A.," Introduction to Pattern Recognition: Statistical,Structural, Neural, and Fuzzy Logic Approaches.", World Scientific, Singapore, (1999).

[7] Duda, R.O., Hart P.E., and Stork, D.G.," Pattern classification (2nd edition)" Wiley, New York, ISBN 0-471-05669-3, (2001).

[8] Yahya, H. Z., Lakmal, D. and Seneviratne, K. A., "Stability analysis of a three-term back propagation algorithm", Neural Networks, vol. 18 (10), pp. 1341-1347, (2005)

[9] Sprinkhuizen, I. J. K. and Egbert, J. W. B., "The local minima of the error surface of the 2-2-1 XOR network", Annals of Mathematics and Artificial Intelligence, vol. 25 (1-2), pp. 107-136, (1999)

[10] M. Riedmiller, "*Rprop - Description and Implementation Details Technical Report*", University of Karlsruhe: W-76128 Karlsruhe (1994).

[11] Y. L. Murphey, Y. Luo, "*Feature extraction for a multiple pattern classification neural network system*", IEEE International Conference on Pattern Recognition, (2002).

[12] L. Bruzzone, D. F. Prieto and S. B. Serpico, "*A neural-statistical approach to multi-temporal and multisource remote-sensing image classification*", IEEE Trans. Geosci. Remote Sensing, (37) 1350-1359 (1999)

[13] J. N. Hwang, S. Y. Kung, M. Niranjan, and J. C. Principe, "*The past, present, and future of neural networks for signal processing*", IEEE Signal Processing Magazine, 14(6) 28-48 (1997).

[14] C. Lee and D. A. Landgrebe, "*Decision boundary feature extraction for neural networks*," IEEE Trans. Neural Networks, 8(1) 75–83 (1997).

[15] C. Apte, et al., "*Automated Learning of Decision Rules for Text Categorization*", ACM Transactions for Information Systems, 12 233-251 (1994).

[16] Manish Mangal and Manu Pratap Singh, "*Handwritten Marathi Characters Recognition using Hybrid evolutionary Feed-Forward Neural Network*", Malaysian Journal of Computer Science, 19(2) 169 –187 (2006).

[17] Y. Even-Zohar and D. Roth, "*A sequential model for multi class classification*", In EMNLP-2001, the SIGDAT. Conference on Empirical Methods in Natural Language Processing, 10-19 (2001)

[18] S. N. Srihari, "*Recognition Of Handwritten and Machine printed text for postal address Interpretation*", Patterns Recognition Letters, 14 (1993).

[19] M. Friedman, and A. Kandel," Introduction to Pattern Recognition: Statistical, Structural, Neural, and Fuzzy Logic Approaches." World Scientific, Singapore, (1999).

[20] R. O. Duda, P. E. Hart and D. G. Stork, "Pattern classification (2nd edition)" Wiley, New York, ISBN 0-471-05669-3, (2001).

[21] F. V. Jensen,"Bayesian Networks and Decision Graphs." Springer, (2001).

[22] T. Cover, and P. Hart,"Nearest Neighbor Pattern Classification." IEEE Trans. on Information Theory, 13(1) 21-27 (1967).

[23] E. D. Sontag, "Feed-forward nets for interpolation and classification", J. Computing System Sciences, 45 20-48, (1992)

[24] M. P. H. Wu, "Hand Written Character Recognition", The school on Information Technology and Electrical Engineering, University of Queensland,(2003).

[25] B. Muller and J. Reinhardt,"Neural Networks: An Introduction." Physics of Neural Network, New York: Springer-Verlag, (1991).

[26] A. G. Barto, R. S. Sutton, and C. Anderson, "Neuron-like adaptive elements that can solve difficult learning

control problems," IEEE Transaction on Systems, Man and Cybernetics, 13, 834-846 (1983)

[27] J. Dayhoff, "Neural-Network Architectures: An Introduction" New York: Van Nostrand Reinhold, (1990).

[28] S. Grossberg, "Some networks that can learn, remember and reproduce any number of complicated space – time patterns", J. Math. Mech., vol. 1(19), pp. 53 – 91, (1991)

[29] R. S. Sutton, A. G. Barto, and R. J. Williams, "Reinforcement learning is direct adaptive optimal control", IEEE Control Systems Magazine, vol. 12, pp. 19 – 22, (1999)

[30] Martin M. Anthony, Peter Bartlett, "*Learning in Neural Networks: Theoretical Foundations*", Cambridge University Press, New York, NY, (1999).

[31] D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning internal representations by error propagation", MIT Press, Cambridge, 1, 318–362 (1986)

[32] K. Fukushima and N. Wake, "*Handwritten alphanumeric character recognition by the neocognitron*," IEEE Trans. on Neural Networks, 2(3) 355-365 (1991).

[33] Y.L. Murphey, Y. Luo, "*Feature extraction for a multiple pattern classification neural network system*", IEEE International Conference on Pattern Recognition, (2002).

[34] S. Shrivastava, and Manu Pratap Singh, "Performance evaluation of feed-forward neural network with soft computing techniques for hand written English alphabets", Journal of Applied Soft Computing, 11 1156-1182 (2011)

[35] T. Gergely, K. Krislof and Csaba, "*Analogical preprocessing and segmentation algorithm for offline hand writing recognition*", Journal of Circuits Systems and Computers, 12 (6) 783-804 (2003).

[36] S. Procter, J. Illingworth, F. Mokhtarian. "*Cursive Handwriting Recognition using Hidden Markov Models and a lexicon driven level building algorithm*", IEEE Proc Vision, Image and Signal Processing, 35 332-367 (2000).

[37] P. Schneider. "*Computer assisted spelling normalization of 18th century Marathi*", Language and Computers, 36 (2001).

[38] F. Rossant and I. Bloc, "*Robust and adaptive OMR system including Fuzzy modeling, fusion of musical rules, and possible error detection*", EURASIP J. Appl. Signal Process, 1 160-160 (2007).

[39] Guang-yuan Zhang, Jing-jiao li, & Ai-xia wang, "A New Recognition method for the handwritten Manchu Character unit", 5th Int. conf. on machine learning and cybernetics, Dalian, 13-16, IEEE, Aug-2006.

[40] Attaullah Khawaja, Shen Tingzhi, Noor Mohd Memon, Allaf Rajpar, "Recognition of printed Chinese characters by using Neural Network", IEEE 2006.

[41] S. R. pande, Saurabh Srivastava, Rajesh Lavania and manu Pratap Singh,"Performance Analysis of feed forward neural networks for the Recognition of scaled and rotated Handwritten alphabets using soft computing techniques", International Journal of Computer Science and Applications, 4(2) 97-113 (2011)

[42] Md. Marof Manwar, Waquar Haque and Padma Polash Paul, "A new approach for rotation Invariant optical character recognition using Eigen digit", IEEE 2007.

[43] U. Pal and Anirban sarkar, "Recognition of printed Urdu script", ICDAR 2003.

[44] S.K. Shah, A. Sharma, "Design and Implementation of optical character recognition system to recognize Gujarati script using template matching", Journal-ET, March- 2006.

[45] S. Hewavitharona, H.C. Fernando and N.D. Kodikara, "Off-line Sinhala Handwriting recognition using Hidden Markove Model", International conference on Recent Advances in Computer vision & Satellite Image processing", Space application centre, Ahmadabad, Dec. 2002.

[46] V. Alessandro, P. Michael, "*Combining Online and Offline Handwriting Recognition*", *ICDAR*, Seventh International Conference on Document Analysis and Recognition (ICDAR'03) 2, 844-853 (2003).

[47] R. Palacios and A. Gupta, "*A System for Processing Handwritten Bank Checks Automatically*", MIT Sloan, (2002).

[48] N. Gorski, V. Anisimov, E. Augustin, O. Baret, D. Price, J. Simon, "*A2iA Check Reader: A family of bank check recognition systems*", ICDAR -99, Proceedings of the Fifth International Conference on Document Analysis and Recognition, 523 – 526 (1999).

[49] S. Lee, "*Off-Line Recognition of Totally Unconstrained Handwritten Numerals Using Multilayer Cluster Neural Network*", IEEE Trans. Pattern Anal. Mach. V. Alessandro, P. Michael, "*Combining Online and Offline Handwriting Recognition*", *ICDAR*, Seventh International Conference on Document Analysis and Recognition (ICDAR'03) 2, 844 (2003)

[50] J. Fierrez, J. Ortega-Garcia, D. Ramos, and J. Gonzalez-Rodriguez, "*HMM-based on-line signature verification: Feature extraction and signature modeling*", Pattern Recognition Letter, 28 (16) 2325-2334 (2007).

[51] D. Xi, S. W. Lee, "*Extraction of reference lines and items from form document images with complicated background*", Pattern Recognition 38 (2) 289-305 (2005).

[52] M. H. Shaheed, "*Feedforward neural network based non-linear dynamic modeling of a TRMS using RPROP algorithm*", Aircraft Engineering and Aerospace Technology, 77 (1)13-22 (2005).

[53] A. E. Hassan, J. Wu and R. C. Holt, "*Visualizing Historical Data Using Spectrographs*" In Proceedings of the 11th IEEE international Software Metrics Symposium METRICS, IEEE Computer Society, Washington, (2005).

[54] C. Y. Suen, and R. Legault, C. Nadal, M. Cheriet and L. Lam, "*Building a New Generation of Handwriting Recognition Systems*", Pattern Recognition Letters 14 305-315 (1993).

[55] S. B. Cho, "*Neural-Network Classifiers for Recognizing Totally Unconstrained Handwritten Numerals*", IEEE Trans. on Neural Networks, 8 43-53 (1997).

[56] H. Yamada and Y. Nakano, "*Cursive Handwritten Word Recognition Using Multiple Segmentation Determined by Contour Analysis*", IEICE Transactions on Information and Systems, 79 (E) 464-470 (1996).

[57] F. Kimura, N. Kayahara, Y. Miyake and M. Shridhar, "*Machine and Human Recognition of Segmented Characters from Handwritten Words*", 4th International

Conference on Document Analysis and Recognition (ICDAR '97), Ulm, Germany, 866-869 (1997).

[58] P. D. Gader, M. Mohamed and J. H. Chiang, "*Handwritten Word Recognition with Character and Inter-Character Neural Networks*", IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics, 27 158-164 (1997).

[59] R. Sarkar, "A Script Independent Technique for Extraction of Characters from Handwritten Word Images", International Journal of Computer Applications, 1 (23) (2010)

[60] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical Pattern Recognition: A Review", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22 (1), pp. 4-37, (2000)

[61] S. Arora, "Performance Comparison of SVM and ANN for Handwritten Devnagari Character Recognition", IJCSI International Journal of Computer Science Issues, vol. 7 (3), (2010)

[62] U. Bhattacharya, S. Vajda, A. Mallick, B. B. Chaudhuri, and A. Belaid, "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), (2004)

[63] M. E. Petersen, D. Ridder, and H. Handels, "Image Processing with Neural Networks: A Review", Pattern Recognition, 35 2279–2301 (2002)

[64] P. M. Patil, and T. R. Sontakke, " Rotation, scale and translation invariant handwritten Devanagari numeral character recognition using general fuzzy neural network", Pattern Recognition, Elsevier, (2007).

[65] K. Y. Rajput, and S. Mishra, "Recognition and Editing of Devnagari Handwriting Using Neural Network", SPIT-IEEE Colloquium and Intl. Conference, Mumbai, India (2006).

[66] T. K. Bhowmik, S. K. Parui and U. Roy, "Discriminative HMM Training with GA for Handwritten Word Recognition", IEEE, (2008)

[67] S. B. Cho, "Fusion of neural networks with fuzzy logic and genetic algorithm", IOS Press, pp. 363–372, (2002)

[68] F. J. Pineda, "Generalization of back-propagation to recurrent neural networks", Physical Rev Letter, 59 2229–2232 (1987)

[69] B. Widrow and M. E. Hoff, "Adaptive switching circuits" IRE Eastern Electronic Show & Convention (WESCON1960), Convention Record, 4, 96–104 (1960)

[70] P. J. Werbos, "Beyond regressions: New tools for prediction and analysis in the behavioral sciences" PhD Thesis, Harvard University, Cambridge, MA, (1974)

[71] W. Finnoff, "Diffusion approximations for the constant learning rate backpropagation algorithm and resistance to local minima", Neural Comp., 6 (2) 285–295 (1994)

[72] H. S. M. Beigi, "Neural network learning through optimally conditioned quardratically convergent methods requiring no line search", In: Proc IEEE 36th Midwest Symp Circuits Syst. Detroit, MI, 1, 109-112 (1993)

[73] M. L. Minsky, S. Papert, "Perceptrons", MIT Press, Cambridge, MA, (1969)

[74] R. Battiti, and F. Masulli, "BFGS optimization for faster automated supervised learning", In: Proc. Int. Neural Network Conf. France, 2, 757-760 (1990)

[75] B. Yegnanarayana, "Artificial neural networks", PHI, New Delhi, (1999)

[76] K. L. Du, M. N. S. Swamy, "Neural Networks in a Soft Computing framework", Springer, 2006.

[77] J. M. Hannan and J. M. Bishop, "A comparison of fast training algorithms over two real problems", International proceeding of IEEE conference on Artificial neural networks, Cambridge, UK, 1-6, 91997)

[78] Z.Wang, A.C. Bovik, H. R. Sheikh and E. P. Simoncelli, "Image Quality Assessment: From Error Visibility to Structure Similarity" IEEE Transactions on Image Processing, 13(4) 600-612 (2004)

[79] P. Petra and A. Chid, "*Empirical evaluation of feature subset selection based on a real-world data set*", Engineering Applications of Artificial Intelligence, 17(3) 285-288 (2004).

[80] M. Ofer, "*Decision Region Connectivity Analysis: A Method for Analyzing High-Dimensional Classifiers*", Machine Learning, 48 (2002).

[81] J. Moody and C.J. Darken, "*Fast learning in networks of locally-tuned processing units*", Neural Computation, 1(2) 281-294 (1989).