# HANDWRITTEN CHARACTER RECOGNITION USING MODIFIED GRADIENT DESCENT TECHNIQUE OF NEURAL NETWORKS AND REPRESENTATION OF CONJUGATE DESCENT FOR TRAINING PATTERNS

***Manu Pratap Singh****

*Department of Computer Science, ICIS, Dr. B.R. Ambedkar University*
*P.O. Box Agra-282002, Uttar Pradesh, India*
*manu_p_singh@hotmail.com*

***V.S. Dhaka***

*Department of Computer Engineering, Maharaja Surajmal Institute of Technology*
*Janakpuri, New Delhi, India*
*vijay_dhaka89@yahoo.com*

*Corresponding Author

**(Received: February 8, 2007 – Accepted in Revised Form: December 11, 2008)**

**Abstract** The purpose of this study is to analyze the performance of Back propagation algorithm with changing training patterns and the second momentum term in feed forward neural networks. This analysis is conducted on 250 different words of three small letters from the English alphabet. These words are presented to two vertical segmentation programs which are designed in MATLAB and based on portions (1/2 and 2/3) of average height of words, for segmentation into characters. These characters are clubbed together after binarization to form training patterns for neural network. Network was trained by adjusting the connection strengths on each iteration by introducing the second momentum term. This term alters the process of connection strength fast and efficiently. The conjugate gradient descent of each presented training pattern was found to identify the error minima for each training pattern. The network was trained to learn its behavior by presenting each one of the 5 samples (final input samples having $26 \times 5 = 130$ letters) 100 times to it, thus achieved 500 trials indicate the significant difference between the two momentum variables in the data sets presented to the neural network. The results indicate that the segmentation based on 2/3 portion of height yields better segmentation and the performance of the neural network was more convergent and accurate for the learning with newly introduced momentum term.

**Keywords** Character Recognition, Feed Forward Neural Network, Segmentation, Back Propagation, Conjugate Gradient Descent

**چکیده** هدف از این مطالعه، تحلیل عملکرد الگوریتم «پس انتشار» دارای الگوهای آموزشی متغیر و عبارت گشتاور دوم در شبکهٔ عصبی تغذیهٔ پیشرو است. این تحلیل بر روی ۲۵۰ کلمه متفاوت انجام می‌شود که از سه حرف کوچک الفبای انگلیسی تشکیل شده‌اند. این کلمات به دو برنامهٔ تقطیع عمودی در قالب MATLAB که مبتنی برارتفاع اجزای متوسط کلمات (به‌میزان $\frac{1}{2}$ و $\frac{2}{3}$) طراحی شد تا به‌صورت حروف تقطیع شوند. این کاراکترها بعد از تفکیک‌سازی دوگانه با یکدیگر تلفیق می‌شوند تا الگوهای آموزشی لازم را برای شبکه عصبی فراهم‌کنند. شبکه مزبور با تنظیم توان ارتباطی در هر نوبت تکرار و از طریق وارد کردن عبارت گشتاور دوم سازماندهی می شود. این عبارت پردازش توان ارتباطی را به طور مؤثر و سریعی تغییر می‌دهد. افت گرادیان مزدوج در مورد هر یک از هر الگوی آموزشی جهت شناسائی حداقل خطا مشاهده شد. به شبکه آموزش داده شد که رفتار خود را با ارائه هر ۵ نمونه (هر نمونه ورودی نهایی دارای $۲۶ \times ۵ = ۱۳۰$ حرف) به‌میزان ۱۰۰ بار بیاموزد. بنابراین با توجه به ۵۰۰ بار آزمایش، تفاوت بسیاری بین دو متغیر گشتاور را در مجموعهٔ داده ها به شبکهٔ عصبی نشان می دهد. نتایج نشان می دهند که فرایند تقطیع بر مبنای $\frac{2}{3}$ بخشی از ارتفاع کلمات، تقطیع بهتری را به دست می‌دهد و عملکرد شبکهٔ عصبی برای آموزش با وارد کردن عبارت گشتاور جدید همگراتر و دقیق‌تر است.

## 1. INTRODUCTION

Handwritten character recognition is the operation that seeks to read gray scale/binary images and interpret the image into machine/human understandable words for further processing. When

these types of inputs are read correctly by machine, this optimizes the processing speed and achieves high benefits in monetary terms [1].

Handwritten text recognition is an important aspect of the pattern recognition task. It plays the major role in many real world problem domains [2]. Its applications include recognition of postal envelopes [3,4], bank checks [5], examination forms [6], Visa application forms [7] and medical prescriptions [8] etc. The automated processing of handwritten material saves time and shows higher accuracy in terms of more data being processed in less time as compared to human processing [9].

The domain of handwritten text recognition has two completely different problems: Online text recognition [10] and offline text recognition [3,11]. Online recognition is the interpretation of human handwriting in "Real-Time", when it is created using media like-light pen, digital writers, touch panel etc. The applications of online handwriting recognition are very much evident in our life as PDA, digital writers and tablet-PCs are very commonly used. On the contrary offline recognition is the reading of handwritten text in the form of binary/gray scale image [2,4,6,11,12]. The documents which are written sometime in the past are considered as offline data for handwriting recognition. In this big domain of handwriting recognition both online case (which pertains to the availability of trajectory data during writing) and the offline case (which pertains to scanned images) are considered.

Many highly accurate systems have been developed to recognize offline handwritten numerals and characters [13-17] but their success has not been carried out on the word recognition domain, due to the difficult nature of unconstrained writing styles and diversity of English characters [18]. A wide variety of research has been done in offline handwritten text segmentation and recognition using neural networks. Segmentation is the process of decomposing a word image into sub-images, so that each image corresponds to a character. The importance of correct segmentation of the word into the characters is highly acknowledged [19]. To correctly segment the currency amount on bank checks into digits, drop fall method has shown good success using NIST database [5,20].

Different architectures and algorithms using artificial neural network and handwriting features have been described by many re-searchers [2,15,18,21,30]. Timar presented a novel trigger wave-based word segmentation algorithm which operates on the skeleton of the words after skew correction [2]. To classify totally unconstrained handwritten numerals three kinds of sophisticated neural network classifiers are presented: Multiple multilayer perceptron (MLP) classifier, Hidden Markov model (HMM)/MLP hybrid classifier and structure adaptive self-organizing map (SOM) classifiers [15]. A Novel method based on data-dependent densities in a Choquet fuzzy integral is shown to outperform neural networks [16]. A heuristic algorithm for segmentation of words into characters based on histogram of word image has yielded 75 % result [18].

Learning-based models have received wide consideration for pattern recognition problems. Neural network models have been reported to achieve enhanced performance compared to other existing models in many recognition tasks [29,32]. Support vector machines (herein after SVMs) have also been observed to achieve reasonable generalization accuracy, especially in implementations of handwritten digit recognition. For the pattern recognition task, SVMs have been used for isolated handwritten digit recognition [22,23], speech identification [24], face detection in images [25], and character classification [26]. The basic SVMs contain no prior knowledge of the pattern problem (for example, a large class of SVMs for the image recognition problem would give the same results if the pixels were first permuted randomly (with each image suffering the same permutation), an act of destruction that would leave the best performing neural networks rigorously handicapped) and much work has been done on incorporating prior information into SVMs [27]. Although SVMs have been proven for good generalization performance, they can be terribly slow in the test stage [28,31].

Here in this paper we propose a system that works on segmentation of offline words into characters using a new segmentation technique which is derived from the general cursive handwriting method for English letters and subsequently a gradient descent based approach for neural network's training and character recognition. This approach is a part of segment-then-classify approach of pattern classification.

The paper is organized as follows, in Section 2 the architecture and functioning of proposed system is presented in detail. Results and discussions are described in Section 3 and 4 respectively. Section 4 also gives the future path for continual work in this field. And in the last section the references used in this paper are stated.

## 2. SIMULATION DESIGN AND IMPLEMENTATION DETAILS

This section addresses the steps required in the process of character recognition technique used and shows a detailed description about the architecture of the complete system i.e. right from scanning the document to character recognition. In structured sections, the experiments and their outcomes at each stage are described.

### 2.1. Overall System Design and Experiments

The complete functioning of the system can be best explained by following block diagram (Figure 1):

The system is simulated using a feed forward neural network system that consists of 150 neurons in input layer, 10 neurons in hidden layer and 26 output neurons. The network has 150 input neurons that are equivalent to the input character's size as we have resized every character into a binary matrix of size 15 × 10. The number of hidden neurons is directly proportional to the system resources. The bigger the number, more resources are required. The number of neurons in hidden layers is kept at 10 for optimal results. The 250 word samples are gathered form 50 subjects of different ages including male and female for the input patterns. After the preprocessing module, 5 input samples were considered for training such that each sample consisted of a-z (26 lower case) letters. Each sample was presented to the network 100 times thus 500 experiments have been conducted by applying different kinds of constraints on the same segmentation algorithm. The constraint is based on the height of the word. The same neural network was used for all learning and testing experiments. The segmented characters are resized into 15 × 10 binary matrixes and are exposed to 150 input neurons. The 26 output neurons correspond to 26 lower case letters of the English alphabet.

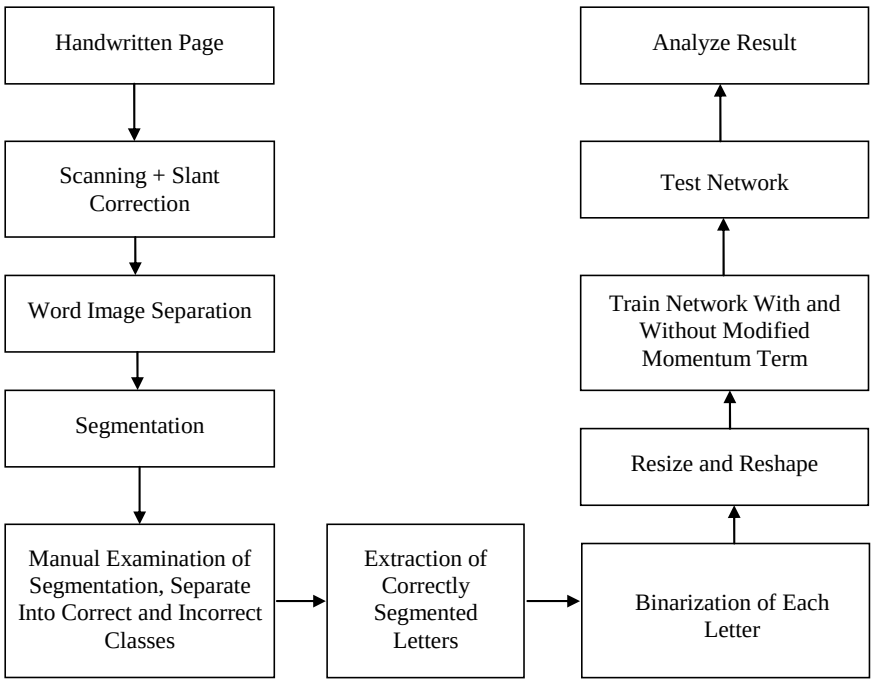



**Figure 1**. Overall steps involved in the character recognition system.

**2.2. Preprocessing** This step is considered as mandatory before segmentation and recognition. All hand printed documents are scanned into gray scale images. Many cursive words which are slanted at various angles in the image have a necessity to employ a slant detection and correction technique [1]. Each word is fitted into a rectangular box in order to be extracted from the document and this way they can contribute to the calculation of height and width.

**2.2.1. Average word height and width estimation** The average height and width of a word are calculated by simply computing an average function on height and weight measurements of every word. Each word is fitted into a rectangular box touching the word on its extreme points for example (Figure 2):

It is well understood that average height of the word is less than the width as each word comprises of 3 letters. The dimensions of each word sample are measured and average height and width are calculated as:

$$H_{avg} = \frac{1}{n} \sum_{i=1}^{n} h_i \quad (1)$$

Where $h_i$ represents height of individual $i^{th}$ word and the width average:

$$W_{avg} = \frac{1}{n} \sum_{i=1}^{n} w_i \quad (2)$$

Here, $w_i$ represents the width of individual $i^{th}$ word and,

n = 600 (total number of words) for Equations 1 and 2

**2.2.2. The segmentation process** The above calculated average height and width ($H_{avg}$ and $W_{avg}$) make the basis for implementation of segmentation process. In most cases it can be observed that in cursive handwritten text , the character are connected (if) to each other to form a word at a height less than half of the $H_{avg}$. The following samples depict the same phenomenon (Figure 3).

The proposed method is based on this approach,

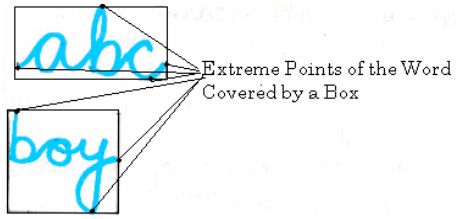



**Figure 2**. Examples of finding height and width of the words.

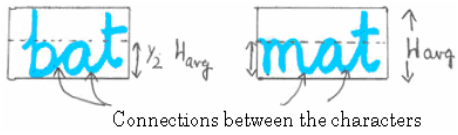



**Figure 3**. Examples of connections between the characters of the cursive word.

where $\frac{1}{2} * H_{avg}$ and $\frac{2}{3} * H_{avg}$ are taken into consideration while deciding segmentation points. Each word is traced vertically after converting the gray scale image into a binary matrix. This binarization is done using logical operation on gray intensity level as follows:

$$\text{If } (I >= \text{level}) \quad (3)$$

$$I = 0$$

Else

$$I = 1$$

Where

$$0 < \text{level} <= 1$$

Where level, the threshold parameter is the threshold parameter based on the gray-scale intensity of the text in document. More intensity leads to higher threshold value. This parameter is decided based on the following table:

The first column of Table 1 represents the intensity of text present in the document. This intensity is a gray-scale value when the document is saved in gray scale image format after scanning and the second column of this table represents the corresponding value of threshold level for binarization process.

The average height and width ($H_{avg}$ and $W_{avg}$) calculated are near to 20 and 38 respectively, so every word image is resized to fit into size 20 × 38. Based out of these dimensions, an assumption is made from binarization process i.e. only those black pixels should be considered which are at least 5 pixels apart from each other.

The threshold parameter along with the grayscale image is used as an input to the binarization program designed in MATLAB. The output is a binary matrix which represents the image. In the segmentation process this matrix is traced vertically down to find the first hit of a black pixel and segmentation points are decided depending on the row and column number of these black pixels.

The judgment of segmentation point is best on following algorithm (Figure 4):

Required: Intensity I of the image in gray-scale format and level of threshold value based on Table 1.

**TABLE 1. Intensity/Threshold Comparison Table.**

| Gray-Scale Intensity of the Text | Value of Threshold Level |
|---|---|
| 0 – 0.25 | 0.20 |
| 0.26 – 0.50 | 0.45 |
| 0.51 – 0.75 | 0.65 |
| 0.76 – 1.0 | 0.85 |

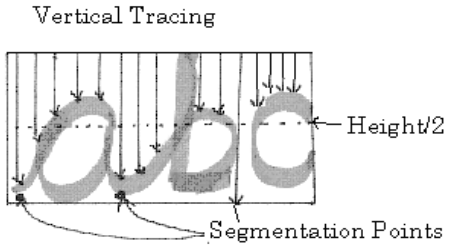



**Figure 4**. Vertical segmentation technique.

Algorithm 1:

**Vertical-Segment (I)**

1. **Repeat** for each column (**i**) in image matrix I starting from position I (0,0).
2. **Repeat** for each row (**j**) element in column.
3. Check **if I (i,j)** is 0 (black pixel) and row number (**j**) > Height/2 then
   3.1. Check **if** (column number (**i**)<5) **or** (i-last found segmentation column<5) then
      3.1.1. Process next element.
      **Else**
      3.1.2. Store this segmentation point (column number i)
4. **If** no black pixel found in entire column then
      It is a segmentation point and save it for further processing.
5. **Repeat** step 1 to step 4 until all elements are read.
6. Cut the image in correspondence to segmentation points identified.

**2.2.3. Reshape and resizing of characters for sample creation** Every segmented character is first converted into a binary matrix C then resized to 15 × 10 matrix using nearest neighborhood interpolation [16] and reshaped to a 150 × 1 logical vector so that it can be presented to the network for learning. The matrix is reshaped using following algorithm:

Algorithm 2:

**Reshape-Characters** (C, 150, 1)

1. Find the transpose of character's binary matrix C
2. Read every element vertically i.e. column wise.
3. Save it in the new matrix according to its dimensions in row first manner.
4. Repeat step 2-3 until every element is read.

The output of this algorithm is a binary matrix of size 150 × 1 which is used as an input to the neural network for learning and testing. Binary matrix representation of character 'a' can be defined as in Figure 5a. The resized characters are clubbed together in a matrix of size (150,26) to form a SAMPLE.

So the final samples for network training/testing were made as follows (Figure 5 and 6):

**2.3. Neural Network's Architecture and Functional Details** The architecture selected for the neural network was a multilayer preceptor (MLP). It consists of three layers, i.e. input layer, hidden layer and output layer. The processing elements of input layer use the liner output function and the units of hidden and output layers use non-liner differentiable output function as shown in Figure 5.

On this network the 150 processing elements are used in input layer and 26 units are used in output layer.

The output of the network can be determined as follows:

$$y_k = f(\sum_{i=1}^{n} Z_i W_{ik}) \quad (4)$$

Where $f$ is the output function,

$Z_i$ is the output of hidden layer and

$W_{ik}$ is the weight between hidden and output layers. Further, the hidden layer's processing unit output;

$$Z_i = f(\sum_{j=1}^{n} V_{ij} X_j) \quad (5)$$

Where $X_j$ is the output of input layer and $V_{ij}$ is the weight between input and hidden layer.

The LMS error between the desired and actual output of the network is:

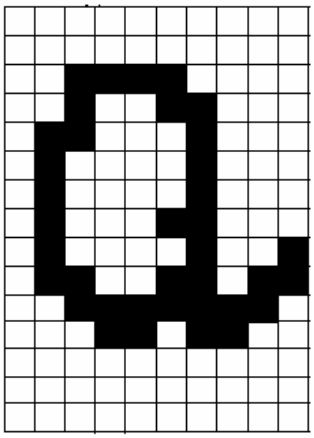
$$E = 0.5\sum_{k}[t_k - y_k]^2 \quad (6)$$

Where $t_k$ is desired output.

The error minimization can be shown as;

$$\partial E / \partial W_{ik} = [t_k - y_k] f'(y_k) z_i \quad (7)$$

Weights modifications on the hidden layer can be defined as;



(a)

1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1
1 1 0 0 0 0 1 1 1 1
1 1 0 1 1 0 0 1 1 1
1 0 0 1 1 1 0 1 1 1
1 0 1 1 1 1 0 1 1 1
1 0 1 1 1 1 0 1 1 1
1 0 1 1 1 0 0 1 1 1
1 0 1 1 1 1 0 1 1 0
1 0 0 1 1 0 0 1 0 0
1 1 0 0 0 0 0 0 0 1
1 1 1 0 0 1 0 0 1 1
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1

(b)

1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
….
150*1 Matrix
1
1

(c)

**Figure 5**. (a) Binary matrix representation of character 'a', (b) binary representation and (c) reshaped sample

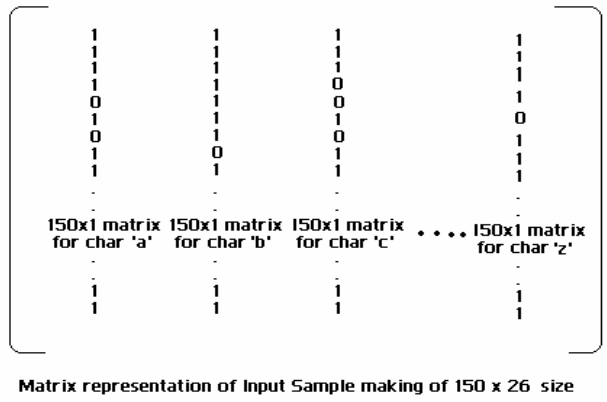



**Figure 6**. Input training/testing sample preparation.

$$\Delta V_{ij} = \partial E / \partial V_{ij} = \sum_k \partial k * (\partial y_k / \partial v_{ij})[t_k - y_k] f'(y_k) z_i \quad (8)$$

[Let, $\partial k = [t_k - y_k]\, f'(y_k)$ ]

and we have

$$\Delta V_{ij} = \sum_k \partial k * W_{ik} f'(z_i) \quad (9)$$

Thus the weight updates for output unit can be represented as;

$$W_{ik}(t+1) = W_{ik}(t) + \eta\, \Delta W_{ik}(t) + \alpha\, \Delta W_{ik}(t-1) \quad (10)$$

Where

| | |
|---|---|
| $W_{ik}(t)$ | Is the state of weight matrix at iteration t, |
| $W_{ik}(t+1)$ | Is the state of weight matrix at next iteration, |
| $W_{ik}(t-1)$ | Is the state of weight matrix at previous iteration, |
| $\Delta W_{ik}(t)$ | Is current change/modification in weight matrix and |
| $\alpha$ | Is standard momentum variable to accelerate learning process. |

This variable depends on the learning rate of the network. As the network yields the set learning rate, the momentum variable tends to accelerate the process.

The network was made to learn the behavior with this Gradient Descent. For next trial the gradient momentum term is modified by adding one more term i.e. the second momentum term.

$$W_{ik}(t+1) = W_{ik}(t) + \eta\, \Delta W_{ik}(t) + \alpha\, \Delta W_{ik}(t-1) + \beta\, \Delta W_{ik}(t-2) \quad (11)$$

When the weight is updated with this equation and computed sequentially, it leads to the following benefits:

1. Formed as a sum of the current (descent) gradient direction and a scaled version of the previous correction.
2. Faster learning process which is evident from learning graphs (Figure 7 and 8).

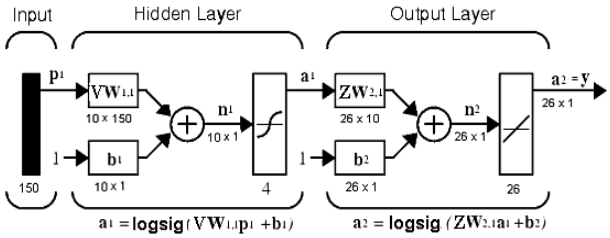



**Figure 7**. Architecture of the neural network used in the recognition system.

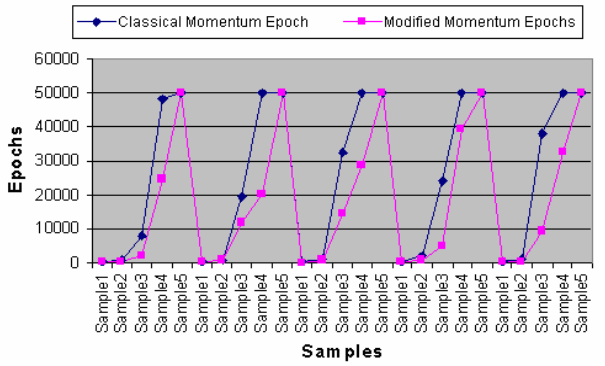



**Figure 8**. The comparison chart for handwritten english characters classification epochs of learning based on two momentum types.

3. Weight modification is based on the behavior learned from the latest 3 iterations instead of 2.

Second derivative (minima) of this gradient descent is computed using MATLAB's built-in function Diff () to find the status of convergence for the 500 trials. For a vector X, the value of Diff(X) is calculated by computing [X(2)-X(1), X(3)-X(2), ... X(n)-X(n-1)]. For a matrix, Diff(X, N) is the N-th order difference along the first non-singleton Dimension (denoted by DIM). If N > = size (X,DIM), Diff takes successive differences along the next non-singleton dimension.

This second derivative represents the points of local minima on the unknown error surface of the training input patterns. The convergence of learning process is judged based on the variation among these points of local minima.

The neural network was exposed to 5 different samples as achieved in section 2.2.3, each sample being presented 100 times. The network was trained 50 times with classical momentum term and 50 times with modified momentum term. Actual output of the network was obtained by "COMPET" function. This is a competitive transfer function which works as follows:

Function 1 Compet (net N)
Start:
[S,Q] = Size (N)
(N being network simulated vector)
[Maxn,Indn] = max (N, [ ], 1)
Result Matrix A = Sparse (indn,1: Q,Ones (1,Q),S,Q)
End Function.

This "COMPET" function puts 1 at the output neuron in which the maximum trust is shown and sets the remaining neuron's result into '0' status. The output matrix is a binary matrix of size (26,26). The output is of the size 26 × 26 because each character has 26 × 1 output vector. The first 26 × 1 column stores the first character's recognition output, the following column will be for next character and so on for 26 characters. For each character the 26 × 1 vector will contain value '1' at only one place. For example character 'a' if correctly recognized, will result in [1,0,0,0…all…0].

The difference between the desired and actual output is calculated for each cycle and the weights are adjusted with Equations 10 and 11. This process continues till the network converges to the allowable or acceptable error.

## 3. RESULTS

The following parameters are used in all the experiments:

The segmentation technique yields the following results with two applied constraints:

The first column of table represents the constraints applied on the segmentation process based on the height of the word. The second and third columns represent the number of correctly and incorrectly segmented words respectively with the two applied constraints. The last column shows the percentage of correctly segmented words. It is clearly evident that the $\frac{2}{3} * H_{avg}$ constraint leads to better results for proposed vertical segmentation technique.

The learning process results of the network in terms of the number of training iterations depicted as Epoch are represented in the following table:

In the table above specified; Epoch 1 represents the number of network iterations for a particular sample when presented to the neural network 5 times for learning with classical momentum term. Epoch 2 represents the number of iterations with modified momentum term for each sample.

The descent gradient is the characteristic of error surface. If the surface changes rapidly, the gradient calculated will be a large number; this will give a poor indication of the true "right error correction path". Hence the smaller gradient is always the desirable one. On the other hand, if the surface is relatively smooth, a larger learning rate will speed convergence. This rationale is based on the knowledge of the shape of the error surface.

The samples were trained in the following fashion:

Sample1-> Sample2-> Sample3-> Sample4-> Sample5-> Sample1-> Sample2-> Sample3-> Sample4-> Sample5-> Sample1-> Sample2-> Sample3-> Sample4->… So on.

Each epoch count is an average of 50 iterations.

The values of gradient descent are computed for each trial of learning. An average of 50 trials for one sample with same type of momentum term is taken into consideration for the subsequent

comparison table.

The network performances are achieved as shown in the following table. Column 2 of the table represents the present error in the network trained with classical weight update method. It is evident that the error is reduced when the modified momentum term is applied on the network.

The computed points of local minima on unknown error surface are described in the following table. Column 1 represents the value with classical method of weight update; whereas column 2 represents the minima points with modified weight update method.

The network was tested with 2 samples. These samples were un-known to the networks as the networks (both classical and modified) were not trained with these samples. The recognition rates for these samples are shown in Table 2 and 3.

**TABLE 2. Parameters v/s their Values used in All Learning Processes.**

| Sr. No. | Parameter | Value |
|---|---|---|
| 1 | Learning/Training Goal for Entire Network | 0.01 |
| 2 | Error | 0.001 |
| 3 | Classical Momentum Term ( $\alpha$ ) | 0.90 |
| 4 | Modified Momentum Term ( $\beta$ ) | 0.05 |
| 5 | Performance Function | Sum Square Error |
| 6 | Maximum Epochs | 50000 |
| 7 | Initial Weights and Biased Term Values | Randomly Generated Values Between 0 and 1 |

**TABLE 3. Results of the Vertical Segmentation Technique with Two Constraints.**

| Segmentation Constraint | Correctly Segmented Words (Out of 600) | Incorrect Segmented Words (Out of 600) | Success Percentage |
|---|---|---|---|
| Height/2 | 427 | 173 | 71.16 % |
| 2 * Height/3 | 498 | 102 | 83 % |

## 4. CONCLUSION

The proposed method for the handwritten words recognition using the segmentation and descent gradient approach, showed a remarkable enhancement in the performance. The results as shown in Table 4, for the different sample of three character words, represent the number of iterations as epochs that a sample has contributed to the learning of input patterns. The proposed method requires less epochs (training time). Two different approaches have been used for this training. The results from first column of Table 5 represent the training with classical method of descent gradient. The second column of Table 6 shows the results of improved version of descent gradient. In this case the values show that the error surface is smoother in the case of modified momentum weight update mechanism. The smaller gradient achieved in the case of modified momentum term are evident that the error correction is downwards and the accuracy has increased.

This paper has introduced an additional momentum term in the weight-age modification, this additional momentum term accelerates the process of convergence and the network shows better performance. The second derivative of the error gradient has been computed. This conjugate descent represents the local minima points on the error surface for the presented training patterns. The conjugate descent has been computed for both methods as shown in Table 7 and 8.

**TABLE 4. Comparison of Network Training Iterations (Epochs) Between the Two Learning Trails.**

| Sample | Epoch 1 (Classical Momentum Term) | | | | | Epoch 2 (Modified Momentum Term) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Sample 1 | 247 | 351 | 486 | 312 | 365 | 318 | 269 | 193 | 299 | 431 |
| Sample 2 | 654 | 837 | 1173 | 1980 | 1426 | 429 | 906 | 724 | 744 | 313 |
| Sample 3 | 7852 | 19286 | 32459 | 24290 | 38031 | 2143 | 11653 | 14310 | 5028 | 9341 |
| Sample 4 | 48024 | 50000 | 50000 | 50000 | 50000 | 14548 | 20182 | 28845 | 39207 | 32638 |
| Sample 5 | 50000 | 50000 | 50000 | 50000 | 50000 | 50000 | 50000 | 50000 | 50000 | 50000 |

**TABLE 5. Comparison of Gradient Values of the Network for the Momentum Terms.**

| Sample | Gradient 1 (Classical Method) | Gradient 2 (Modified Method) |
|---|---|---|
| Sample 1 | 1981400 | 2029280 |
| Sample 2 | 5792000 | 2021500 |
| Sample 3 | 7018400 | 1723310 |
| Sample 4 | 1173900 | 843.094 |
| Sample 5 | 62263.9 | 21897.8 |

**TABLE 6. Error Level Attained by the Neural Network Trained with Both Methods.**

| Sample | Error with Classical Weight Update Method | Error with Modified Weight Update Method |
|---|---|---|
| Sample 1 | 0 | 0 |
| Sample 2 | 0 | 0 |
| Sample 3 | 0 | 0 |
| Sample 4 | 0.00343787 | 0 |
| Sample 5 | 0.00787574 | 0.00491716 |

**TABLE 7. Comparison of minima (Second Derivative of Performance) for Two Methods of Weight Update for the Handwritten English Word Recognition of Three Letters.**

| Sample | Minima 1 (Multiplied by 10 ^ 10) | Minima 2 (Multiplied by 10 ^ 10) |
|---|---|---|
| Sample 1 | 527760 | 897490 |
| Sample 2 | 6635599 | 5387800 |
| Sample 3 | 6959000 | 7292400 |
| Sample 4 | 8403000 | 0.024474 |
| Sample 5 | 0.017583 | 0.023112 |

**TABLE 8. Comparison of Recognition Percentage with Test Samples Executed in Classical Feed Forward Neural Network and Modified Neural Network.**

| No of Characters in Test Sample | Classical Momentum Neural Network | | Classical Momentum Neural Network | | Recognition Percentage | |
|---|---|---|---|---|---|---|
| | Correct | Incorrect | Correct | Incorrect | Classical | Modified |
| 26 | 17 | 9 | 24 | 2 | 65.38 | 92.30 |
| 26 | 20 | 6 | 25 | 1 | 80 | 96.15 |

The performance can be observed for the modified technique as represented above in Figure 9. The success in the recognition also depends upon the segmentation criterion. It means that if we change the segmentation or the method for separating the characters, the success of performance will also be changed.

The proposed segmentation technique has been quite successful in segmenting all types of characters (cursive and non cursive). The segmentation technique yields better results when the 2/3* height constraint is applied. This constraint gives 83 % correct segmentation success whereas 71 % success rate is achieved in case of 1/2* height constraint.

The second momentum term used is useful for finding the rate of convergence. The conjugate descent has been computed for the representation of the points of local minima on the unknown error surface of the training input patterns. It helps in the investigation for the convergence of the network and also to identify present error in the training of a sample. Due to the back-propagation of error element in Multilayer Perceptron (MLP), it frequently suffers from the problem of Local-Minima, hence the samples may not converge so as the case with Sample 5 (Figure 10 and 11).

Nevertheless, more work needs to be done especially on the test for large complex handwritten characters. Some future work should also be explored. The proposed work can be carried out to recognize English words of different character lengths. The segmentation technique suggested in this paper is also useful to segment words of all lengths.

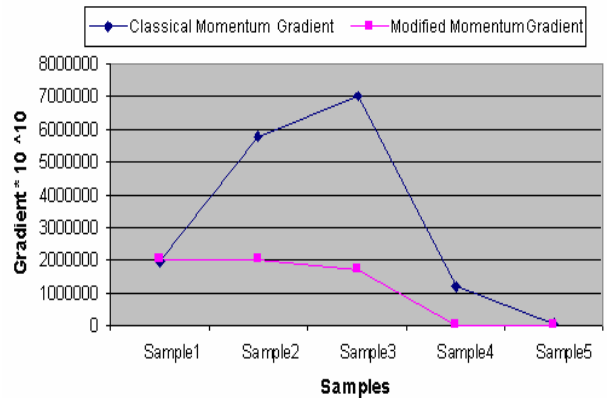



**Figure 9**. The comparison chart for gradient (conjugate descent) for handwritten english three letters word recognition for two different momentum based learning (gradient 1 with classical weight update and gradient 2 with modified method).

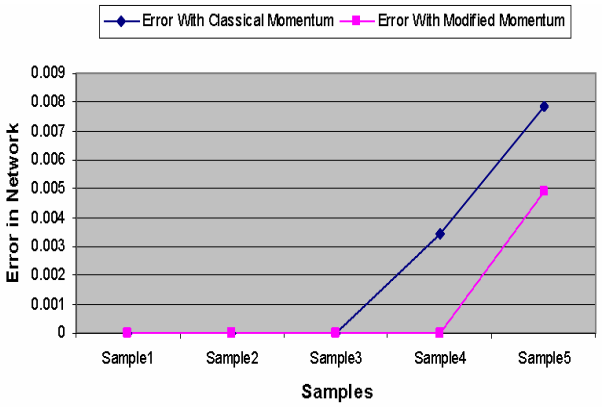



**Figure 10**. The comparison chart for minimized error for handwritten English three letter word recognition (error 1 with classical weight update and error 2 with modified method).

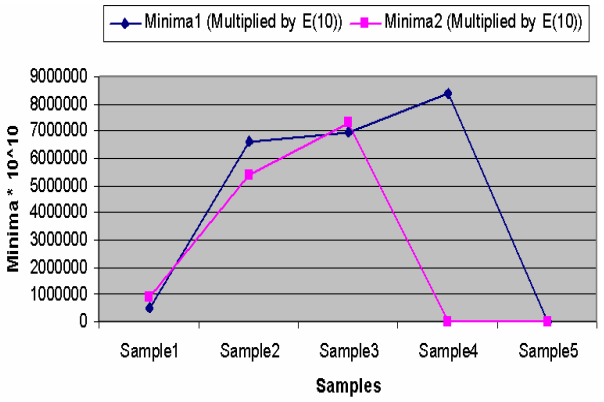



**Figure 11**. The comparison chart for minima (second derivative of conjugate descent) for handwritten English three-letter word recognition (minima1 with classical weight update and minima 2 with modified method).

## 5. REFERENCES


1. Andreyev, Y.V., Belsky, Y.L., Dmitriev, A.S. and Kuminov, D.A., "Information Processing using Dynamic Choas: Neural Network Implementation", ***IEEE Trans. Neural networks***, Vol. 7, (1996), 290-299.

2. Timar, G., Karacs, K. and Rekeczky, C., "Analogical Preprocessing and Segmentation Algorithm for offline Hand Writing Recognition", ***Journal of Circuits Systems and Computers***, Vol. 12, (2003), 783-804.
3. Chen, M.Y., Kundu, A. and Zhou, J., "Off-Line Handwritten Words Recognition using a Hidden Markov Model Type Stochastic Network", ***IEEE Trans. Pattern Analysis and Machine Intelligence***, Vol. 15, No. 5, (1994), 481-496.
4. Yacoubi, A.E.I., Gilloux, M., Sabourin, R. and Suen, C.Y., "Unconstrained Handwritten Word Recognition using Hidden Markov Models", ***IEEE Trans. Pattern Analysis and Machine Intelligence***, Vol. 21, No. 8, 1999), 752-760.
5. Palacios, R. and Gupta, A., "A System for Processing Handwritten Bank Checks Automatically", ***Image and Vision Computing***, Vol. 26, No. 10, (October 2008), 1297-1313.
6. Plamondon, R. and Srihari, S.N., "On-Line and off-Line Handwriting Recognition: A Comprehensive Survey", ***IEEE Trans. on Pattern Analysis and Machine Intelligence***, Vol. 22, No. 1, (January 2000) 63-84.
7. Chiang, H. and Gader, P., "Recognition of Handprinted Numerals in VISA Card Application Forms", ***Mach. Vision Applicat.***, Vol. 10, (1997), 144-149.
8. Byrne, S., Cunningham, P., Barry, A., Graham, I., Delaney, T. and Corrigan, O.I., "Using Neural Nets for Decision Support in Prescription and Outcome Prediction in Anticoagulation Drug Therapy", ***Proceedings of 5$^{th}$ International Workshop on Intelligent Data Analysis in Medicine and Pharmacology at 14$^{th}$ European Conference on Artificial Intelligence***, Berlin, Germany, (2000), 9-12.
9. Srihari, S., Govindraju, V. and Srihari, R., "Handwritten Text Recognition", ***Proc 4$^{th}$ International Workshop on Frontiers in Handwriting Recognition***, Taiwan, Vol. 35, (1994), 265-275.
10. Jaeger, S. Manke, S., Reichert, J. and Waibel, A., "Online Handwriting Recognition: The NPEN++ Recognizer", ***International Journal Document Analysis and Recognition***, Vol. 3, (2001), 169-180.
11. Procter, S., Illingworth, J. and Mokhtarian, F., "Cursive Handwriting Recognition using Hidden Markov Models and a Lexicon Driven Level Building Algorithm", ***IEEE Proc Vision, Image and Signal Processing***, Vol. 35, (2000), 332-367.
12. Srihari, S.N., "Recognition of Handwritten and Machine Printed Text for Postal Address Interpretation", ***Patterns Recognition Letters***, Vol. 14, (1993), 291-302.
13. Sven, C.Y., Legamlt, R., Nadal, C., Cherios, M. and Lam, L., "Building a New Generation of Handwriting Recognition System", ***Pattern Recognition Letters***, Vol. 14, (1993), 305-315.
14. Lee, S.W., "Multi-Layer Cluster Neural Network for Totally Unconstrained Handwritten Numeral Recognition", ***Neural Networks***, Vol. 8, (1995), 783-792.
15. Cho, S.B., "Neural-Network Classifiers for Recognizer for Totally Unconstrained Handwritten Numerals", ***IEEE Trains on Neural Network***, Vol. 8, (1997), 43-53.
16. Garder, P.D., "Fusion of Handwritten Word Classifiers", ***Pattern Recognition Letters***, Vol. 17, (1996), 577-584.
17. Srihari, S.N., "Recognition of Handwritten and Machine Printed Text for Postal Address Interpretation", ***Patterns Recognition Letters***, Vol. 14, (1993), 291-302.
18. Blumenstein, M. and Verma, B., "A New Segmentation Algorithm for Handwritten Word Recognition", IJCNN '99, Washington, U.S.A., (1999).
19. Casey, R.G. and Lecolient, E., "A Survey of Methods and Strategies in Characters Segmentation", ***IEEE Trans. Pattern Analysis and Machine Intelligence***, Vol. 18, (1996), 690-706.
20. Palacios, R., Gupta, A. and Wang, P.S.P., "Feedback-Based Architecture for Reading Courtesy Amounts on Checks", ***Journal of Electronic Imaging***, Vol. 12, No. 1, (January 2003), 194-202.
21. Kim, G. and Govindraju, V., "A Lexicon Driven Approach to Handwritten word Recognition for Real-Time Applications", ***IEEE Trans. PAMI***, Vol. 19, No. 4, (1997), 366-379.
22. Cortes, C. and Vapnik, V., "Support Vector Networks", ***Machine Learning***, Vol. 20, (1995), 273-297.
23. Burges, C. and Scholkopf, B., "Improving the Accuracy and Speed of Support Vector Machines", ***Proceedings of the Advances in Neural Information Processing Systems***, Vol. 9, (1997), 375-381.
24. Schmidt, M., "Identifying Speakers with Support Vector Networks", In Interface, Sydney, Australia, (1996).
25. Osuna, E., Freund, R. and Girosi, F., "Training Support Vector Machines: An Application to Face Detection", In Proceedings CVPR'97, 130-136.
26. Joachims, T., "Text Categorization with Support Vector Machines: Learning with Many Relevant Features", ***In Proceedings of ECML-98, 10$^{th}$ European Conference on Machine Learning, Number 1398 in Lecture Notes in Computer Science***, Springer Verlag, Heidelberg, DE, (1998), 137-142.
27. Scholkopf, B., Smola, A. and Muller, K.R., "Support Vector Methods in Learning and Feature Extraction", ***Australian Journal of Intelligent Information Processing Systems***, Vol. 1, (1998), 3-9.
28. Osuna, E. and Girosi, F., "Reducing the Run-Time Complexity in Support Vector Machines", Advances in Kernel Methods: Support Vector Learning, MIT Press, Cambridge, MA, (1999), 271-284.
29. Fereidunian, A.R., Lesani, H. and Lucas, C., "Distribution System Reconfiguration using Pattern Recognizer Neural Networks", ***International Journal of Engineering***, Trans. B: Applications, Vol. 15, No. 2, (2002), 135-144.
30. Mathur, S., Aggarwal, V., Joshi, H. and Ahlawat, A., "Offline Handwriting Recognition using Genetic Algorithm", ***6$^{th}$ International Conference on Information Research and Applications***", Varna, Bulgaria, (June-July 2008).
31. Perfetti, R. and Ricci, E.,"Reduced Complexity RBF Classifiers with Support Vector Centres and Dynamic Decay Adjustment", ***Neurocomputing***, Vol. 69, (October 2006), 2446-2450.

32. Liu, H. and Ding, X., "Handwritten Character Recognition using Gradient Feature and Quadratic Classifier with Multiple Discrimination Schemes", ***IEEE ICDAR'05***, (2005), 1520-5263.