Analysis of Handwritten Hindi Character Recognition using Advanced Feature Extraction Technique and Back propagation Neural Network

Dayashankar Singh
M.M.M.University of Technology
Gorakhpur (UP), India

J.P. Saini
BIET, Jhansi (UP), India

D.S. Chauhan
GLA University, Mathura (UP), India

ABSTRACT
Feature extraction techniques play an important role in pattern recognition. Neural networks are being used for character/pattern recognition since last many years but most of the works are confined to English character recognition. Till date, a very little work has been reported for Handwritten Hindi Character recognition. The main challenge is to maintain high performance level with samples, which are distorted or written in more personal style. Handwritings of every person are different due to the great variations of individual writing styles, different size and orientation angle of the characters. In this paper, conventional feature extraction (Global pixel), Gradient feature extraction and 8-Directional Gradient Feature (8-DGF) Extraction, 16- Directional Gradient Feature Extraction (16-DGF) technique for handwritten Hindi character recognition have been implemented by using two layer fully connected Back propagation Feed-Forward neural network. The mouse has been initialized in graphics mode so that characters could be written directly on the screen. Once the character is written on screen, four input files are created consisting of the Global pixel input values, Gradient input values, 8- Directional Gradient input values and 16- Directional Gradient input values

Gradient input values, 8- Directional Gradient input values and 16- Directional Gradient input values respectively. Character has been taken in 32x32 pixels i.e.1024 neurons as an input. At hidden layer 12 hidden units or neurons have been taken. In this way, one hidden layer and one output layer have been taken. The network has been trained by giving the input data of respective network. One thousand samples were taken from 100 people of different age group, 10 samples from each person for training the network (train data) and testing the network (test data). Out of these samples, 500 samples were taken for training the network and 500 samples were taken for testing the network. A comparative analysis has been carried out, and experimental result shows that 8-DGF and 16-DGF extraction are better than conventional feature extraction and Gradient feature extraction in terms of recognition accuracy. The 16-DGF provides 96% recognition accuracy but it requires more training time as compared to 8-DGF extraction technique.

Keywords
Gradient Feature Extraction, Directional Gradient Feature Extraction, Recognition Accuracy

1. INTRODUCTION
Handwritten character recognition is a challenging problem in the field of pattern recognition. The difficulty is mainly because of large variations of individual writing style. Therefore, robust feature extraction technique is required to improve the performance of handwritten character recognition in terms of recognition accuracy. In this paper, neural network has been explored for handwritten Hindi character recognition and also several feature extraction techniques have been applied for handwritten character recognition on the baseline system to improve the recognition accuracy and to reduce the training time of the neural network. Neural networks have been widely used in the field of handwriting recognition. This paper describes a system for offline recognition of handwritten Hindi characters, a language widely spoken in India. It is also regarded as National Language of India. Pre-processing has been used primarily to reduce variations of handwritten Hindi characters and feature extraction for later processing to improve recognition accuracy. Feature extractor is essential for efficient data representation and extracting meaningful features for further processing. Feature extraction refers to the process of finding a mapping that reduces the dimensionality of patterns. As a major factor of influencing recognition performance, features play a very important role in handwriting recognition. This leads to the development of a number of features for handwriting recognition with better response. A systematic evaluation of features in a specific feature vector is very important for designing a new feature vector by combining different feature vectors. In this paper, in order to take online samples mouse has been initialized in graphics mode so that samples can be taken directly on the screen. Feed-forward neural networks with 12 hidden units and two output layers have been taken for the implementation. Since, sample collection has been done for only two Hindi characters; therefore only two neurons have been taken at output layer in order to recognize the two characters. Conventional feature extraction (Global pixels), Gradient feature extraction and two approach named as 8-division Directional Gradient Feature (8-DGF) extraction technique and 16-division Directional Gradient Feature (16-DGF) extraction technique have been implemented and an experimental analysis has been carried out for testing recognition accuracy and training time. Experimental result shows that 16-DGF extraction technique is providing better accuracy (96%) but it requires more training time as compared to 8-DGF feature extraction technique [1-21].

This paper has been organized as follows. Background and related available work is explained in detail in Chapter 2. Chapter 3 describes the back propagation feed-forward neural network, creating the network in mat lab, training algorithm etc. Chapter 4 describes conventional feature extraction (Global pixel), gradient feature extraction technique, 8-directional gradient feature extraction technique and an innovative16-directional gradient feature extraction technique in detail. Chapter 5 explains the implementation of gradient and 8-directional gradient feature extraction technique and 16-directional gradient feature extraction technique. Chapter 6 describes a detailed performance comparison of four techniques of handwritten Hindi character recognition, namely Global pixel technique, Gradient Feature extraction, 8-Directional Gradient Feature extraction and 16-directional gradient feature extraction technique. Finally, Chapter 7 provides conclusions and a brief description of future scope.

2. BACKGROUND AND RELATED WORK

2.1 Neural Network: An Overview

Neural Networks, which are simplified models of the biological neuron system, is a massively parallel distributed processing system made up of highly interconnected neural computing elements that have the ability to learn and thereby acquire knowledge and make it available for use.
Back propagation is a systematic method of training multilayer artificial neural networks. It is built on high mathematical foundation and has very good application potential. Even though it has its own limitations, it is applied to a wide range of practical problems and has successfully demonstrated its power. The Multilayer feed-forward network with Back propagation (BP) learning is sometimes called multilayer perceptron because of its similarity to perceptron networks with more than one layer. The generalized delta (back propagation) learning rule has been derived [1]. The back propagation algorithm (BPA), also called the generalized delta rule, provides a way to calculate the gradient of the error function efficiently using the chain rule of the differentiation. The error after initial computation in the forward pass is propagated backward from the output units, layer by layer, justifying the "backpropagation" [1].

2.2 Training of Neural Network

2.2.1 Supervised Training

Supervised training requires the pairing of each input vector with a target vector representing the desired output; together these are called a training pair. Usually a network is trained over a number of such training pairs. An input vector is applied, the output of the network is calculated and compared to the corresponding target vector, and the difference (error) is fed back through the network and weights are changed according to algorithms that tend to minimize the error. The vectors of the training sets are applied sequentially, and errors are calculated and weight adjusted for each vector, until the error for the training set is at an acceptably low level. [16, 17]

2.2.2 Unsupervised Training

It requires no target vector for the outputs, and hence, no comparisons to pre-determined ideal response. The training set consists of input vectors. The training algorithms modifies network weights to produce output vectors that are consistent; that is , both applications of one of the training vectors or application of vector that is sufficiently similar to it will produce the same pattern of outputs. The training process, therefore, extracts the statistical properties of the training set and groups similar vector into classes. Applying a vector from a given class to the input will produce a specific output vector, but there is no way to determine prior to training which specific output pattern will be produce by given input vector class. Hence the outputs of such a network must generally be transformed into a comprehensible form subsequent to the training process. [16,17]

2.3 Training Algorithm

2.3.1 Perceptron Training Algorithms

A Perceptron is trained by presenting a set of patterns to its input, one at a time, and adjusting the weights until the desired output occurs for each of them.

2.3.2 Delta Rule

An important generalization of the perceptron training algorithm, called the delta rule. The perceptron training algorithm may be generalized by introducing a term δ, which is the difference between the desired or target output T and the actual output A.

$$\delta = (T-A)$$

The case in which $\delta = 0$ corresponds to the output is correct and nothing is to be done.

In cases of $\delta > 0$ and $\delta < 0$, the perceptron training algorithm is satisfied if δ is multiplied by the value of each input xi and this product is added to the corresponding weight. To generalize this "learning rate" coefficient η multiplies the δ.xi product to allow control of the average size of weight changes. Symbolically, it can be represented as follows:

$$\Delta i = \eta\, \delta.xi$$
$$Wi(n+1) = Wi(n) + \Delta i$$

Where,

Δi = the correction associated with the ith input xi,
Wi(n+1) = the value of weight i after adjustment,
Wi(n) = the value of weight i before adjustment

The delta rule modifies weights appropriately for target and actual outputs for both continuous and binary inputs and outputs. [1, 8]

2.4 Available work related to Hindi Character Recognition

2.4.1 Introduction

Neural Network(s) are recently being applied on various kind of pattern recognition processing. Character recognition is an area of pattern recognition that has been the subject of considerable research during the last some decades. Many reports of character recognition of several languages, such as Chinese, Japanese, English, Arabic, and Farsi have been published but still recognition of handwritten Hindi characters using neural networks is an open problem. Hindi is a second official Indian language and it is widely used in many Indian states. Approximately four hundred million people use this language all over the world. In many Indian offices such as passport, bank, sales tax, railway, embassy, etc. the both English and Hindi languages are used. Therefore, it is a great importance to develop an automatic character recognition system for Hindi language. [1, 2]

2.4.2. Character Modelling

The Hindi language consists of 49 characters (13vowels, 36 consonants) and is written from left to right. A set of hand written Hindi characters is given below [1].

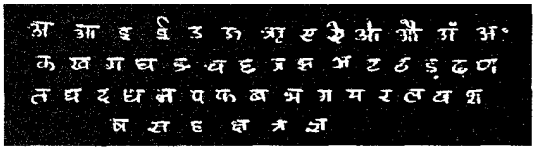

Figure: 1 A set of Hindi Character [1]

- Scanning and Skeletonization

Each character was scanned in 300 pixels per inch using Scanner HP-Scan Jet 11. The scanned character was converted into 1024 (32x32) binary pixels. The skeletonization process was used to binary pixel image and the extra pixels, which were not belonging to the backbone of the character, were deleted and the broad strokes were reduced to thin lines [1]. Skeletonization is illustrated in following Figure2.

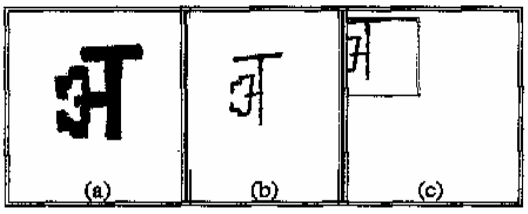

Figure: 2 Scanning and Skeletonization [1]

2.4.3. Multilayer Perceptron Network

The multilayer perceptron neural networks with the EBP algorithm have been applied to the wide variety of problems. We used a two-layer perceptron i.e., single hidden layer and an output layer. A structure of MLP network for Hindi character recognition is shown in Figure 3.

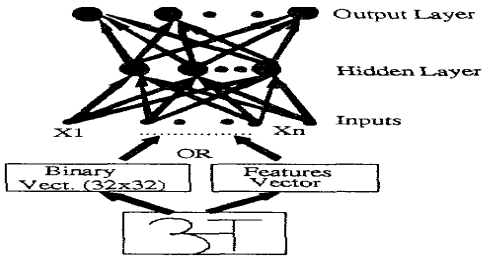

Figure: 3 Multilayer Perceptron Network [1]

2.5. Problem with available work

- The available work related to handwritten Hindi Character Recognition results poor accuracy and it also requires more training time to train the network. Whenever, neural network is used to recognize the characters from their pixel value combinations, we get the needed generalization. There will be a large number of units and accordingly, the weighted connections will increase. This will result a complex network. This will also require more storage space.
- Conventional feature extraction technique does not provide high recognition accuracy and it also requires higher memory storage.
- Backpropogation neural network with conventional feature extraction technique requires more training time.

2.6. Proposed Approach for the Solution

Gradient, 8-Directional Gradient Feature extraction technique and an innovative approach of 16-Directional Gradient Feature extraction techniques have been implemented in order to increase recognition accuracy and to reduce the training time to train the network [5].

Two layer backpropogation feed-forward neural network models has been implemented.

3. BACKPROPAGATION

3.1 Back propagation Training Algorithm

There are many variations of the back propagation algorithm, several of which we discussed in this chapter. The simplest implementation of back propagation learning updates the network weights and biases in the direction in which the performance function decreases most rapidly - the negative of the gradient. One iteration of this algorithm can be written

$$W_{k+1} = W_k - \alpha_k . g_k$$

Where Wk is a vector of current weights,

gk is the current gradient, and αk is the learning rate.

Gradient is calculated after every iteration and compared with threshold gradient value 10^{-10}. If gradient is greater than the threshold value then it performs next iteration. There are two different ways in which this gradient descent algorithm can be implemented:

- Incremental mode
- Batch mode

In the incremental mode, the gradient is computed and the weights are updated after each input is applied to the network. In the batch mode all of the inputs are applied to the network before the weights are updated [Matlab Toolbox].

- Batch Gradient Descent (traingd) The batch steepest descent training function is traingd. The weights and biases are updated in the direction of the negative gradient of the performance function. If you want to train a network using batch steepest descent, you should set the network trainFcn to traingd, and then call the function train. There is only one training function associated with a given network.

There are various training parameters associated with traingd: epochs, show, goal, time, min_grad, and lr. The learning rate lr is multiplied with the negative of the gradient to determine the changes to the weights and biases. The larger the learning rate, the bigger the step. If the learning rate is made too large, the algorithm becomes unstable. If the learning rate is set too small, the algorithm takes a long time to converge.

The training status is displayed for every iteration of the algorithm. (If show is set to NaN, then the training status never displayed.) The other parameters determine when the training stops. The training stops if the number of iterations exceeds epochs, if the performance function drops below goal, if the magnitude of the gradient is less than mingrad, etc. [Matlab Tool box]

4. FEATURE EXTRACTION

Feature extraction refers to the process of finding a mapping that reduces the dimensionality of patterns. In pattern recognition and in image processing, Feature extraction is a special form of reduction in dimensionality of patterns. There are various feature extraction techniques.

4.1 Conventional Feature Extraction (Global Pixel)

In conventional method a region is constituted which includes the character. If a line passes through a pixel, then corresponding pixel will be given value one (1), Otherwise it is taken as zero (0). Thus, it becomes necessary to store the pixel value combinations for every character. Now, when an input pattern is given, a suitable match with any of the stored patterns, it will be identified. In case of Hindi alphabets, there are different fonts for every character. The same character written by different person and when written a number of times will be different. Hence a generalized condition for recognizing a particular character cannot be specified, which is must for the algorithm approach. This is the major drawback of the conventional method. Another drawback is the large memory space is required to store the pixel values [1].

4.2 Gradient Feature Extraction

In gradient feature extraction technique, Sobel operator is used to calculate the gradient of each pixel. This method calculates the gradients of 900 pixels (30x30) which will be given as input to the neural network for training the various collected samples.

4.3 Directional Gradient Feature Extraction (8-DGF)

Directional gradient feature (8-DGF) extraction is advanced technique, which requires the division of gradient values into eight parts (1 to 8). The directional value has been taken 0 for the gradient value -1. Each part is having the span of 45-degree angle. If gradient falls between 0-45 degree then directional gradient feature will be taken as 1, if it falls between 46-90 degree then directional gradient feature will be taken 2, and so on. In this way, we reduce the domain of input values in only 8 discrete integer values.

4.6 Directional Gradient Feature Extraction (16-DGF): A new approach

Directional gradient feature (16-DGF) extraction technique has been developed and implemented in this paper, which requires the division of gradient values into sixteen parts (1 to16). The directional value has been taken 0 for the gradient value -1. Each part is having the span of 22.5 degree angle. If gradient falls between 0-22.5 degree then directional gradient feature will be taken as 1, if it falls between 22.5-45 degree then directional gradient feature will be taken 2, and so on. In this way, we reduce the domain of input values in sixteen discrete integer values.

5. IMPLEMENTATION OF VARIOUS FEATURE EXTRACTION TECHNIQUES

This Chapter describes the entire details of various existing feature extraction techniques namely conventional feature extraction, gradient feature extraction, directional gradient feature extraction (8-DGF) as well as newly developed 16-DGF extraction technique. The feed-forward neural network has been selected for implementation of each and every feature extraction technique, which has been discussed in chapter 4.

5.1 Conventional Feature Extraction

The procedure of implementing conventional feature extraction (Global pixel) has been illustrated as follows.

- Capture image of handwritten Hindi character in 30x30 pixels
- Perform binarization of 30x30 pixel images into 32x32 matrixes by imposing boundary of zeros.
- Now 32x32 matrix is converted into a 1024x1 column matrix which is standard input for feed forward neural network.
- This 1024x1 column matrix can be used as input for training as well as simulation purpose.
- 1024x1 column matrix has been given to feed forward neural network as input.
- The goal for training is set as a 2x1 matrix (e.g. अ character , goal =[1 0]' and for character , goal = [0 1]')
- The command net = train (net, I, g) will train the feed forward network named net in Matlab with input = I(1024x1), having goal = g ([1 0]' or [0 1]')
- The goal of network training (10^{-5}) is meeting when gradient of successive training is less than 10^{-10}.
- For simulation the command is out = sim (net, I) where the simulation result is stored in out.

5.2 Gradient Feature Extraction

Each image of character is normalized into 32×32 size. The gradient operator, named Sobel operator is used in this thesis to calculate the gradient. The Sobel operator uses two templates to compute the gradient components in horizontal and vertical directions, respectively [3,4,5]. The templates are shown in Figure 4.

-1	0	1
-2	0	2
-1	0	1

Vertical

1	2	1
0	0	0
-1	-2	-1

Horizontal

Figure 4: Sobel operator templates

The two gradient components at location (i, j) are calculated by:

$$Gx = gv(i, j) = f(i-1, j+1) + 2f(i, j+1) + f(i+1, j+1) - f(i-1, j-1) - 2f(i, j-1) - f(i+1, j-1) \quad \text{----------(1)}$$

$$Gy = gh(i, j) = f(i-1, j-1) + 2f(i-1, j) + f(i-1, j+1) - f(i+1, j-1) - 2f(i+1, j) - f(i+1, j+1) \quad \text{----------(2)}$$

The gradient strength and the direction are calculated as follows:

$$G(i,j)=\sqrt{gv^2(i,j)+gh^2(i,j)}\text{----------(3)}$$

$$\theta=\arctan(Gy/Gx)\text{----------(4)}$$

The procedure of implementing gradient feature extraction has been illustrated as follows.

- Capture image of handwritten Hindi character in 30x30 pixels
- Perform binarization of 30x30 pixel images into 32x32 matrixes by imposing boundary of zeros.
- Implementation of Sobel operator on 32x32 matrixes for gradient calculation.
- Gradient values are in a 30x30 matrix with values ranges between 0 to 2π. And gradient is set to -1 where a pixel is surrounded by 8 black pixels
- Gradient values are in a 30x30 matrix with values ranges between 0 to 2π. The gradient is set to -1 where a pixel is surrounded by 8 black pixels
- Now this 30x30 matrix having gradient values is converted into 900x1 column matrixes and given as input to the feed forward neural network.
- This matrix can be used for training as well as simulation.
- The goal for training is set as a 2x1 matrix.
- The command net = train(net, I, g) will train the feed forward network named net in Matlab with input = I(900x1), having goal = g ([1 0]' or [0 1]')
- The goal of network training (10^{-5}) is met when gradient of successive training is less than 10^{-10}.
- For simulation the command is out = sim (net, I) where the simulation result is stored in out.

The flow chart of gradient feature extraction technique has been given below in Figure 5.

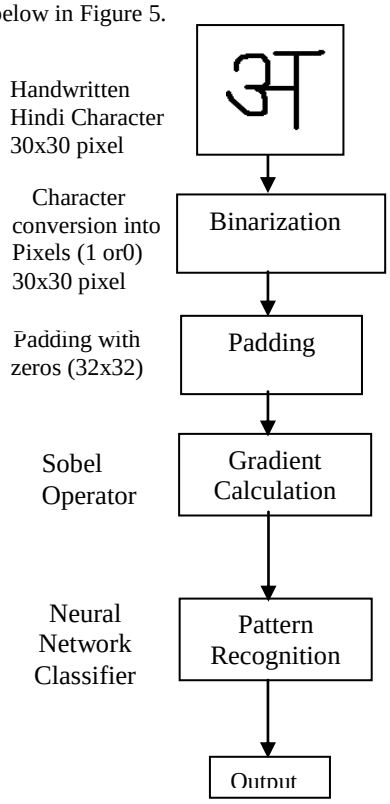

Figure 5: Flow Chart of Gradient Feature Extraction

5.3 Directional Gradient Feature Extraction (8-DGF)

Each image of character is normalized into 32×32 size. The gradient operator, named Sobel operator is used in this thesis to calculate the gradient. The Sobel operator uses two templates to compute the gradient components in horizontal and vertical directions, respectively [4]. The templates are shown in Figure 6..

-1	0	1
-2	0	2
-1	0	1

Vertical

1	2	1
0	0	0
-1	-2	-1

Horizontal Template

Figure: 6: Sobel operator templates

The two gradient components at location (i, j) are calculated by:

$$Gx = gv(i,j) = f(i-1,j+1) + 2f(i,j+1) + f(i+1,j+1) - f(i-1,j-1) - 2f(i,j-1) - f(i+1,j-1) \text{----------(1)}$$

$$Gy = gh(i,j) = f(i-1,j-1) + 2f(i-1,j) + f(i-1,j+1) - f(i+1,j-1) - 2f(i+1,j) - f(i+1,j+1) \text{----------(2)}$$

The gradient strength and the direction are calculated as fllows:

$$G(i,j)=\sqrt{gv^2(i,j)+gh^2(i,j)}\text{----------(3)}$$

$$\theta=\arctan(Gy/Gx)\text{ ---------- (4)}$$

5.3.1 Direction Calculation

Gradients calculated by sobel operator are divided into 8 equal parts depending upon their value falls between the range of angles as shown Table1. The gradients having the value -1 has been assigned direction 0.

Table 1: Direction Values

Gradient values (g)	Direction Values
g = -1	0
0 <= g < 0.79	1
0.78 <= g < 1.58	2
1.58 <= g < 2.36	3
2.36 <= g < 3.15	4
3.15 <= g < 3.93	5
3.93 <= g < 4.72	6
4.72 <= g < 5.5	7
5.5 <= g < 6.29	8

The gradient values and their respective direction values has been shown in following Figure 7.

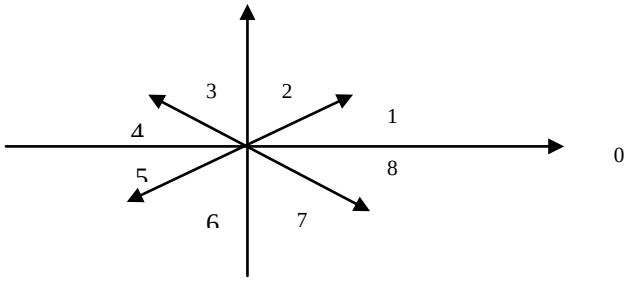

Figure 7: Direction values of gradients

5.3.2 Procedure

The procedure of implementing DGF extraction technique has been illustrated as follows.

- Capture the image of handwritten Hindi character in 30x30 pixels
- Perform binarization of 30x30 pixel images into 32x32 matrix by imposing boundary of zeros.
- Implementation of Sobel operator on 32x32 matrixes for gradient calculation.
- Gradient values are in a 30x30 matrix with values ranges between 0 to 2π. The gradient is set to -1 where a pixel is surrounded by 8 black pixels
- Now the gradient values will be decomposed into a set of directions as shown in table 5.1. Now we get only 9 discrete directional values.
- Now this 30x30 matrix having directional values is converted into 900x1 column matrixes and given as input to the feed forward neural network.
- This matrix can be used
- for training as well as simulation.
- The goal for training is set as a 2x1 matrix, goal =[1 0]' and goal = [0 1]')
- The command net = train(net, I, g) will train the feed forward network named net in Matlab with input = I(900x1), having goal = g ([1 0]' or [0 1]')
- The goal of network training (10^{-5}) is met when gradient of successive training is less than 10^{-10}.
- For simulation the command is out = sim (net, I) where the simulation result is stored in out.

5.4 Directional Gradient Feature Extraction (16-DGF): A New Approach

Each image of character is normalized into 32×32 size. The gradient operator, named Sobel operator is used in this thesis to calculate the gradient. The Sobel operator uses two templates to compute the gradient components in horizontal and vertical directions, respectively [4]. The templates are shown in Figure 8.

-1	0	1
-2	0	2
-1	0	1

Vertical

1	2	1
0	0	0
-1	-2	-1

Horizontal

Figure 8: Sobel operator templates

The two gradient components at location (i, j) are calculated by:

$$Gx = gv(i, j) = f(i-1, j+1) + 2f(i, j+1) + f(i+1, j+1) - f(i-1, j-1) - 2f(i, j-1) - f(i+1, j-1) \text{ ----------(1)}$$

$$Gy = gh(i, j) = f(i-1, j-1) + 2f(i-1, j) + f(i-1, j+1) - f(i+1, j-1) - 2f(i+1, j) - f(i+1, j+1) \text{ ----------(2)}$$

The gradient strength and the direction are calculated as fllows:

$$G(i,j)=\sqrt{gv^2(i, j) + gh^2(i, j)} \text{----------(3)}$$

$$\theta=\arctan(Gy / Gx) \text{ ---------- (4)}$$

5.5 Direction Calculation

Gradients calculated by Sobel operator are divided into 16 equal parts depending upon their value falls between the ranges of angles as shown Table 2. The gradients having the value -1 has been assigned direction 0.

Table 2: Direction Values

Gradient values (g)	Direction Values
g = -1	0
0<= g < 0.39	1
0.39< = g < 0.78	2
0.78 <= g < 1.17	3
1.17 < = g < 1.56	4
1.56 < = g < 1.95	5
1.95 < = g < 2.34	6
2.34 < = g < 2.73	7
2.73 < = g < 3.12	8
3.12 < = g < 3.51	19
3.51 < = g <3.90	10
3.90 < = g < 4.29	11
4.29 < = g < 4.68	12
4.68 < = g < 5.07	13
5.07 < = g < 5.46	14
5.46 < = g < 5.85	15
5.85 < = g < 6.28	16

The gradient values and their respective direction values have been shown in following Figure 5.4.

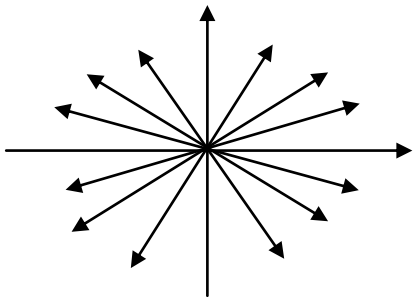

Figure: 9: Sixteen Direction values of gradients

5.3.2 Procedure

The procedure of implementing DGF extraction technique has been illustrated as follows.

- Capture the image of handwritten Hindi character in 30x30 pixels
- Perform binarization of 30x30 pixel image into 32x32 matrix by imposing boundary of zeros.
- Implementation of Sobel operator on 32x32 matrixes for gradient calculation.
- Gradient values are in a 30x30 matrix with values ranges between 0 to 2π. The gradient is set to -1 where a pixel is surrounded by 8 black pixels
- Now the gradient values will be decomposed into a set of directions as shown in table 5.1. Now we get only 9 discrete directional values.
- Now this 30x30 matrix having directional values is converted into 900x1 column matrix and given as input to the feed forward neural network.
- This matrix can be used
- for training as well as simulation.
- The goal for training is set as a 2x1 matrix (e.g. for character , goal =[1 0]' and for character , goal = [0 1]')
- The command = train(net, I, g) will train the feed forward network named net in Matlab with input = I(900x1), having goal = g ([1 0]' or [0 1]')
- The goal of network training (10^{-5}) is met when gradient of successive training is less than 10^{-10}.
- For simulation the command is out = sim (net, I) where the simulation result is stored in out.

6. COMPARISON & ANALYSIS

A comparison has been carried out on 500 samples for training the network and 500 samples for testing the network by considering 50 iterations for each feature extraction techniques. The result has been shown in following table 3.

The table indicates that for the same set of hidden units and number of iterations, the classification time in case of directional input is better than gradient input and this time is the best against pixel input. The performance on test set is also proved to be the best (accuracy 96%) in case of directional input (16-DGF) against directional gradient (8-DGF), gradient and pixel inputs. Experiment also shows that the 8-DGF Extraction Technique requires less training and classification time while 16-DGF extraction technique yields the high recognition accuracy.

Table 3 shows that conventional Feature Extraction Technique is giving poor accuracy and also requires more time to train the network. Experiment shows that this technique requires more number of iterations for improving the accuracy. Experiment also shows that this technique requires more classification time also.

Table 3 shows that Gradient Feature Extraction Technique is giving good accuracy but it requires more time to train the network because of floating point calculation is being performed which requires more time. There are large variations in gradient values due to which network recognition problem with better accuracy exists. Experiment shows that this technique also requires more number of iterations for improving the accuracy.

Table 3 shows that Directional Gradient Feature Extraction (8-DGF*)* technique is giving good accuracy around 94% and it requires less time to train the network because of discrete integer calculation is being performed and the information content are more for the network to learn and recognize handwritten Hindi characters easily and quickly.

Table 3
Comparison of Results for handwritten Hindi character recognition

Input to MLPN	No of hidden units	No of iterations	Training time (sec)	Classification time (ms)	Performance on training set (%)
32x32 Pixel input	12	50	984	785	100
30x30 Gradient input	12	50	785	435	100
30x30 8-Directional Gradient (8-DGF)	12	50	395	410	100
30x30 Directional Gradient (16-DGF)	12	50	410	425	100

There is a small variation in direction values due to which network recognizes the character with high accuracy (e.g. sometime 1 and then 2). Experiment shows that this technique requires less number of iterations for training the network due to which its training time is reduced. Experiment also shows that only 15 iterations are sufficient for training the network fully.

Table 3 shows that Directional gradient feature extraction (16-DGF) technique is giving very good accuracy around 96% but it requires slightly more time to train the network and classify the character because of more discrete integer calculation is being performed and the information content are more for the network to learn and recognize handwritten Hindi characters easily and quickly. It requires slightly more training time because of voluminous information content. This technique can

be used for getting high accuracy on the small cost of training and classification time.

7. CONCLUSIONS AND FUTURE SCOPE

This paper is an effort towards increasing the accuracy of handwritten Hindi characters recognition. This work can further be extended to the character recognition of other Indian languages. For doing so, sufficient training and also a sufficient number of samples are required so that network could be trained for them .Here, in this paper 16- DGF extraction technique has been introduced, this provides very high accuracy for handwritten Hindi character recognition. This paper uses the same parameters and compare training time, classification time and recognition accuracy. From the experimental result it has been shown that16- DGF extraction technique is best in terms of recognition accuracy. This paper can be used for character recognition in any language without making any modification in network as well as feature extraction technique.

Although, a lot of efforts have been made to complete a great deal of work but still it seems tremendous scope for improving/enhancing this work further. The efforts can be made to develop an automatic system for recognizing all handwritten Hindi characters. Future scope

of work is that scanned samples can be taken as input to the network and it is expected that there will be 100% accuracy in Directional Gradient Feature Extraction Technique (16-DGF). It can be extended for Hindi word recognition by including regular expressions. It can also be extended for Hindi to English translation by including regular expressions and grammar.

REFERENCES

[1] Verma B.K, "Handwritten Hindi Character Recognition Using Multilayer Perceptron and Radial Basis Function Neural Network", IEEE International Conference on Neural Network, vol.4, pp. 2111-2115, 1995.

[2] Sutha.J, Ramraj.N, "Neural Network Based Offline Tamil Handwritten Character Recognition System", IEEE International Conference on Computational Intelligence and Multimedia Application, 2007 Volume 2, 13-15, Dec.2007, Page(s): 446-450, 2007.

[3] Hailong Liu, Xiaoqing Ding, "Handwritten character recognition using gradient feature and quadratic classifier with multiple discrimination schemes" Eighth IEEE International Conference on Document Analysis and Recognition, Vol. 1, Page(s): 19-23, August 29-1 Sept., 2005.

[4] Weipeng Zhang; Yuan Yan Tang; Yun Xue. "Handwritten Character Recognition Using Combined Gradient and Wavelet Feature" IEEE International Conference on Computational Intelligence and Security, Volume 1, Page(s) 662-667, Nov. 2006.

[5] Cheng-Lin Liu, "Normalization-Cooperated Gradient Feature Extraction for Handwritten Character Recognition" IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 29, Issue 8, Aug. 2007 Page(s) 1465-1469.

[6] Starzyk,J.A.Ansari, " Feedforward neural network for handwritten character recognition", IEEE International Symposium on Circuits and Systems(ISCAS), Volume 6, Page(s) 2884-2887, 1992.

[7] A. Rajawelu, M.T. Husilvi, and M.V.Shirvakar, "A neural network approach to character recognition." IEEE Trans. on Neural Networks, vol. 2, pp. 307-393, 1989.

[8] W.K. Verma, "New training methods for multilayer perceptrons," Ph.D Dissertation, Warsaw Univ. of Technology, Warsaw, March 1995

[9] B. Parhami and M. Taragghi, "Automatic recognition of printed Farsi text,"IEEE Pattern Recognition, Vol. no. 8, pp. 787-1308, 1990.

[10] C.C. Tappert, C.J. Suen and T. Wakahara,"The state of the art in outline handwriting recognition," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. PAMI-12, no.8, pp.707-808, 1990.

[11] D.S. Yeung, "A neural network recognition system for handwritten Chinese character using structure approach," Proceeding of the World Congress on Computational Intelligence, vo1.7, pp. 4353-4358, Orlando, USA, June 1994.

[12] D.Y. Lee, "Handwritten digit recognition using K nearest-neighbor, radial basis function and backpropagation neural networks,"IEEE Neural computation, vol. 3, Page(s) 440-449.

[13] E. Cohen, 1.1. Hull and S.N. Shrikari, "Control structure for interpreting handwritten addresses," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 16, no. 10, pp. 1049-1055, Oct. 1994.

[14] H. Almualim and S. Yamaguchi, "A method for recognition of Arabic cursive handwriting," IEEE Trans. on Pattern and Machine Intelligence, vol. PAMI-9, no 5, pp.715-722, Sept. 1987.

[15] I.S.I. Abuhaiba and S.A. Mahmoud, "Recognition of handwritten cursive Arabic characters," IEEE Transaction on PA&MI vol.16, no 6, pp. 664-672, June 1994

[16] Neural Computing Theory and Practices by Philip D. Wasserman

[17] Neural Networks, Fuzzy Logic, and Genetic Algorithms by S. Rajasekaran and G.A. Vijaylakshmi Pai.

[18] J. Hertz, A. Krogh and R. Palmer, "Introduction to the theory of neural computation," Addison-Wesley Publishing Company, USA, 1991.

[19] K. Yamada and H. Kami, "Handwritten numeral recognition by multilayered neural network with improved learning algorithm," IJCNN Washington DC, vol. 2, pp. 259-266, 1989.

[20] P. Morasso, "Neural models of cursive script handwriting," IJCNN, WA, vol. 2, pp. 539-542, June 1989.

[21] S.J. Smith and M.O. Baurgoin, "Handwritten character classification using nearest neighbor in large database," IEEE Trans. on Pattern and Machine Intelligence, vol. 16, pp. 915-919, Oct. 1994.