A PROJECT REPORT ON

AUTOMATED HANDWRITTEN DEVANAGRI CHARACTER RECOGNITION USING MACHINE LERNING

MACHINE LEARNING (ECE-GY- 6143)

SUBMITTED BY

Name of the student	N-ID
JEEVTESH SINGH	N14422671
ASHISH JAMBHULAKR	N12864266
SUSHANTH SHABNAVEES	N14792855
DEVASHISH PAREEK	N16747244

Under the Guidance of

Prof. SUNDEEP RANGAN

NEW YORK UNIVERSITY, TANDON SCHOOL OF ENGINEERING.

MAY-2019

ACKNOWLEDGEMENT

Under the esteem guidance of our project guide Prof. Sundeep Rangan , detailed study and simulation AUTOMATED HAND-WRITTEN DEVANAGARI CHARACTER RECOGNITION USING MACHINE LEARNING has been done. We are very thankful for his whole-hearted co-operation without which this project could not have been completed.

I would like to place my sincere thanks to the head of the department for its continuous help and motivation.

A special thanks goes to my team mates, who helped complete the project on time.

ABSTRACT

System based optical character recognition is generally used to recognize many printed fonts. Even so it is important to note that commercial software is not always satisfactory, and problems exists with unusual character sets, fonts and document of poor quality.

Optical Character Recognition (OCR) could not be extended to handwritten recognition due to large degree to variability in people's handwriting styles. The character recognition system is generally a two-step recognition in which the pattern is represented by a set of features and classification where decision rules for separating pattern classes are defined. The objective of proposed work is to observe the effects of variation of handwriting styles and to analyze the performance of the same by processing many characters.

In the proposed system supervised approach is used as classifier to compare the percentage rate for isolated character.

Keywords: Optical Character Recognition (OCR), Artificial Neural Network (ANN), Devanagari script, Handwritten Character Recognition

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviated Word	Expansion
OCR	Optical Character Recognition
AR	Aspect Ratio
GUI	Graphical User Interface

Contents

CHAPTER 1
INTRODUCTION

Brief History:

Artificial Intelligence is the most challenging part of Electronic sciences providing the machines human like capability. Since the invention of computers and machines, their capability to perform various tasks went on growing exponentially.AI provides with one of the major tasks by giving machine capability to see, translate, think, learn and the capacity to read the text. A good text recognizer has many commercial and practical applications, e.g. from searching data in scanned book to automation of any organization, like post office, which involve manual task of interpreting text. Many dissimilar techniques have been attempted to solve the problem of text recognition.

Until early 2000‘s most of the work done in character recognition was in Roman, Chinese, Japanese and Arabic, although we have 15 major scripts in India. In India, there are 18 official languages. Among these Hindi and Bangla are most popular. We are using Devanagari script in our project as most of the Indian languages can be written using Devanagari script, for example Hindi, Marathi, Sanskrit etc.

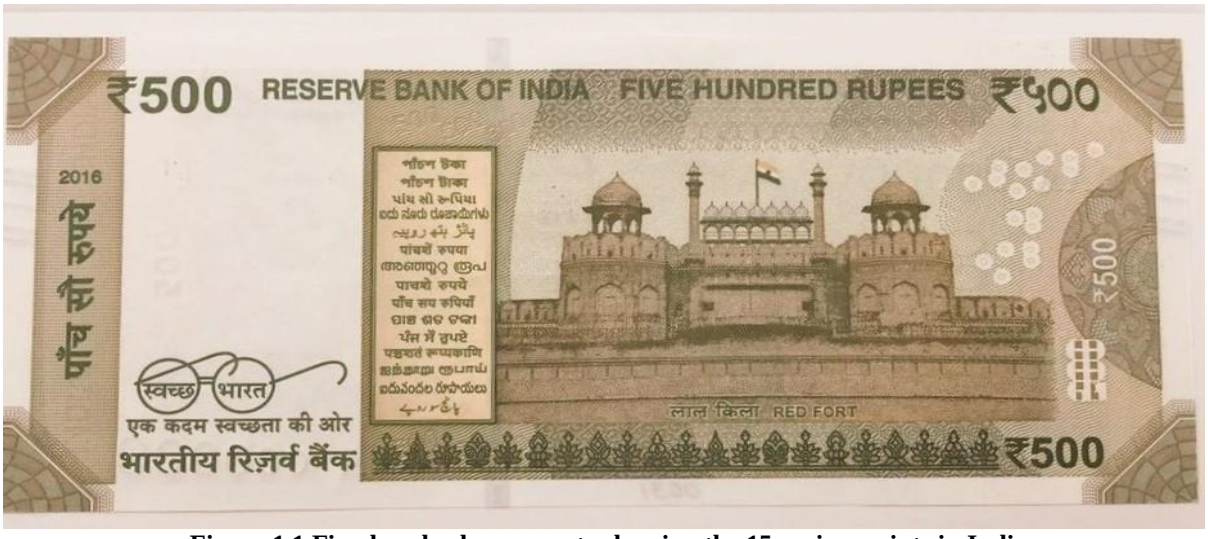


Figure 1.1 Five-hundred-rupee note showing the 15 major scripts in India

Template matching is one of the simplest approaches. In template matching each word is maintained for an input image, error or difference with each template is computed. Output is the symbol equivalent to minimum error. This technique does not have a good performance for hand written characters but works well for standard fonts. Feature extraction is another approach, in which orthogonal properties such as distance vectors are extracted by analyzing the statistical distribution of points. Box approach has been used for feature extraction, wherein, for each symbol a feature vector is calculated and stored in the database. Recognition is done by finding distance of feature vector of input image to that of stored in the database and outputting the symbol with minimum deviation. The obtained feature vector is given as input to the neural network. Neural network offers complete independence of recognition process and character set. The first step in neural network is training and so the neural network is trained by multiple samples of images of each character. After the training recognition process starts. In recognition process the input image is directly given to the neural network and then the recognized symbol, is outputted.

Neural Network is technical perception of a Human Brain. It is a processing paradigm that is inspired by the biological nervous system. It is a very large interconnection network between neurons. Neural network has the ability to derive the meaning from large complicated data and can be used to extract patterns and detect trends that are too complicated to be noticed.

The advantage of neural network is that the domain of the character set can be very easily extended; one just needs to train the network over a new set. Another advantage of neural network is that they are very robust to noise. The disadvantage of neural network is that it is very time consuming as a lot of training is required. Although the best of the recognition approaches are able to recognize all those texts that human can the major reason for this has been wide variation in writing practices and styles of different people and lot of context based information that human brain uses to interpret any text sample.

CHAPTER 2

LITERATURE SURVEY

The origin of character recognition was found in 1870 when Carey invented the retina scanner— an image transmission system using a mosaic of photocells [1]. The very first recognition devices interpreted the letter shape, which the OCRs read. These machines came into being in the early 1960s. The first widely commercialized OCR of this generation was the IBM 1418, which was designed to read a special IBM font, 407 [4].

After this, the devices were built to recognize machine as well as hand printed characters. This methodology was popularized by IBM 1287 OCR system, which was first unveiled at the 1965 New York world fair [4]. From around 1975-1985, a third generation of OCR systems was popularized which could recognize poorly printed character and hand-printed ones [10].

Ray and Chatterjee worked on Bangla character recognition and classified characters using features extracted from string connectivity criterion [7]. After this, Dutta provided a generalized solution for generation and analysis of Bangla and Devanagari script [8]. The system developed by Chaudhari and Pal was the first which did operations like skew correction, noise removal and segmentation of the image into lines, zones and characters. This approach involved a combination of feature and template matching for classification [9].

Even though, the first attempt for recognition of Devanagari script was done in 1977 but not much work was done on it [2]. The problem of handwritten recognition has been approached using methods such as discrete wavelet transforms (DWT) and Euclidean distance metric (EDM). This approach provided a classification rate of around 90%.

R. Jayadevan, Satish R. Kolhe, Pradeep M. Patil, and Umapada Pal, presents All feature-extraction techniques as well as training, classification and matching techniques useful for the recognition are discussed. An attempt is made to address the most important results reported so far and it is also tried to highlight the beneficial directions of the research till date. From the survey, it is noted that the errors in recognizing printed Devanagari characters are mainly due to incorrect character segmentation of touching or broken characters. Because of upper and lower modifiers of Devanagari text, many portions of two consecutive lines may also overlap and proper segmentation of such overlapped portions are needed to get higher accuracy

The recognition is carried out by modifying the exponential membership functions fitted to the fuzzy sets. These fuzzy sets are derived from features consisting of normalized distances obtained using the Box approach [5]. The membership function is modified by two structural parameters that are estimated by optimizing the entropy subject to the attainment of membership function to unity. The overall recognition rate is found to be 95% for Hindi numerals and 98.4% for English numerals. A new scheme for offline recognition of totally unconstrained handwritten characters using a simple multi-layer cluster neural network trained with the back-propagation algorithm is presented in [6]. Box approach method was used by M. Hanmandlu, to extract features of each character. The paper was well received in 2006, by the Department of Electrical Engineering, IIT Delhi.

Work done so far has not achieved 100% accuracy in this system due to variation in handwriting styles. Not much work has been done on handwritten Devanagri characters. The main problem faced in handwritten character recognition is the variation in handwriting styles. In villages, most of the people still prefer to write letters or postcards over digital writing.

CHAPTER 3

EXPERIMENT WORK

3.1 Process Overview

The character recognition system is usually validated by running it on independent test sets, on which the system has not been trained. For these tests to be conclusive, the validation sets should include a large numbers of samples to reflect the variety of writing styles that are found in real-life application.

3.1.1 Database:

For the purpose of validation, we need a standard database. For experimental analysis, we considered 150 samples for training and 15 samples for testing. These samples were taken from different professionals including Doctors, Engineers, Lawyers, Shopkeepers, Security Guards, etc. The images were scanned using HP LASERJET 4250. The training set captures as many variations and different styles of a character as possible. The features extracted from the training set are stored in the excel sheet and at the recognition time, used as reference features for comparing with those of an unrecognized character. So, if the training set contains a large number of samples with varied writing styles, the feature set computed from them will be able to reflect these variations, making the recognition system independent of the variations in writing styles.

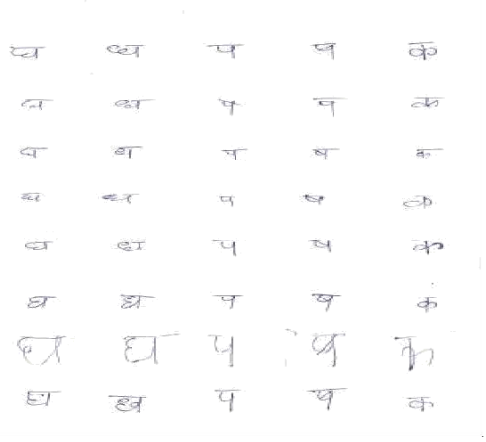


Figure 3.1 Database

The task of the recognition of handwritten numerals has been broken down into the following steps:

1. Binarization of sample image:
2. Correction of image slant
3. Normalization of the image to a standard size
4. Feature extraction
5. Recognition

To enable recognition, steps (1)-(3) are applied on a training set of all 5 characters as part of the pre-processing. While performing feature extraction, simultaneously the excel sheet of reference features is created.

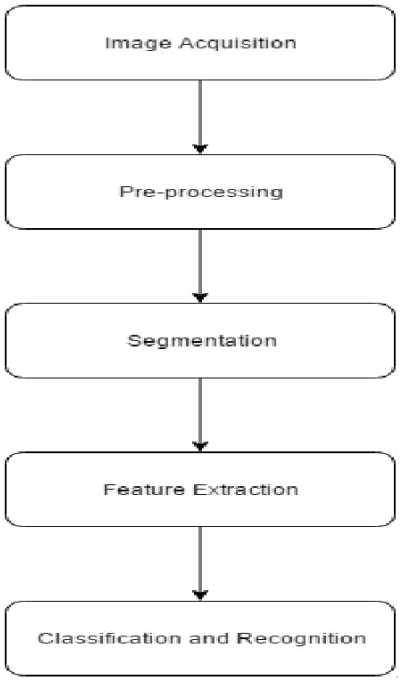


Figure 3.2 Process Flow Chart

3.2 Pre-Processing:

The steps of preprocessing are briefly discussed in the following

3.2.1 Image Acquisition:

Scanned images of hand written Devanagari script is given as an input. These images are segmented and compared to those in the database to recognize the character.

3.2.2 Binarization:

Frequently, binarization is carried out before the character recognition phase. Ideally an input character should have two tones, i.e., black and white pixels (commonly represented by 0 and 1, respectively). Image binarization converts an image of up to 256 gray levels into a two-tone image.

3.2.3 Segmentation:

Segmentation is one of the most important phases of OCR system. In order to have any chance character recognition, you will need to segment the characters properly. By applying good segmentation techniques, we can increase the performance of OCR. Segmentation sub divides an image into its constituent regions or objects. Basically, in segmentation, we try to extract the basic constituent of the script, which are certainly the characters. This is needed because our classifier recognizes these characters only. Even in good quality documents, some adjacent characters touch each other due to inappropriate scanning resolution. We used bounding box approach for the segmentation.

3.2.4 Slant Correction:

A practical character recognizer must be able to maintain high performance regardless of the size and slant of a given character or word. For handwritten characters, one of the major variations in writing styles is caused by slant, which is defined as the slope of the general writing trend with respect to the vertical line. It is important that the system be insensitive to slant. The slant correction algorithm is as follows:

1. The angle of each character is calculated and analyzed.
2. If the slope is directed clockwise, required angle is then added to the true angle of the character slope.
3. If the slope is directed anti-clockwise, the required angle is then subtracted from the true angle of the character slope.

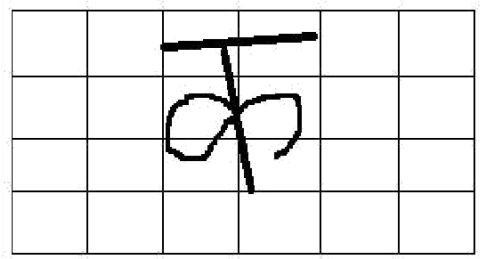


Figure 3.3 Slanted Character

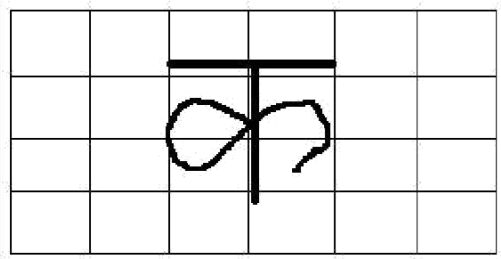


Figure 3.4 Slant Corrected Character

3.2.5 Size Normalization:

For Devanagari characters, depending on the AR of the numeral, categorization of character is done. AR is the ratio of height to width of the image. Irrespective of the size of a handwritten numeral, this AR is always found to be nearly constant for all its samples. Images having AR greater than 2 are fitted into a window size of 60x32 while others are fitted into a window of size 42x32.

Handwriting fluctuates greatly between people. Although humans can correct for this variability, it is a very complex task to have a computer understand off-line written text. To make the task easier, all the images are resized to the same size (42x32). This ratio is maintained for obtaining the information from characters which do not have any Matras ‘. The size is carefully selected for attaining the maximum information while compressing the size, which is nothing but the entropy of the image.

3.3 Feature Extraction

Box Approach Method [5]

The most important phase of character identification is Feature extraction. Each character is unique, thus distinguishing itself from the other character. Hence, it is very important to extract features in such a way that the recognition of different characters becomes easier on the basis of the individual features of each character. Following tasks have been performed to extract features in proposed method.

For extracting features, we used the Box approach. Slant corrected, normalized, binary character was used for extracting features. In this, the character of image size 42x32 was fitted into four horizontal and six vertical grid lines, thus dividing the image into 24 boxes of size 7x8. The vector distances of all white pixels within a box were calculated and averaged. This was performed for all 24 boxes and these were used as the features for each image.

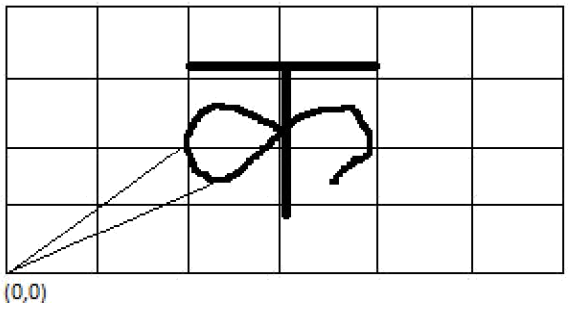


Figure 3.5 Distance Vector Calculation

By taking the bottom left corner as he absolute origin (0,0), the vector distance forks the k^{th} pixel in b^{th} box at location (i, j) is calculated as:

$$d_k^{\,b} = (i^2 + j^2)^{\frac{1}{2}}$$

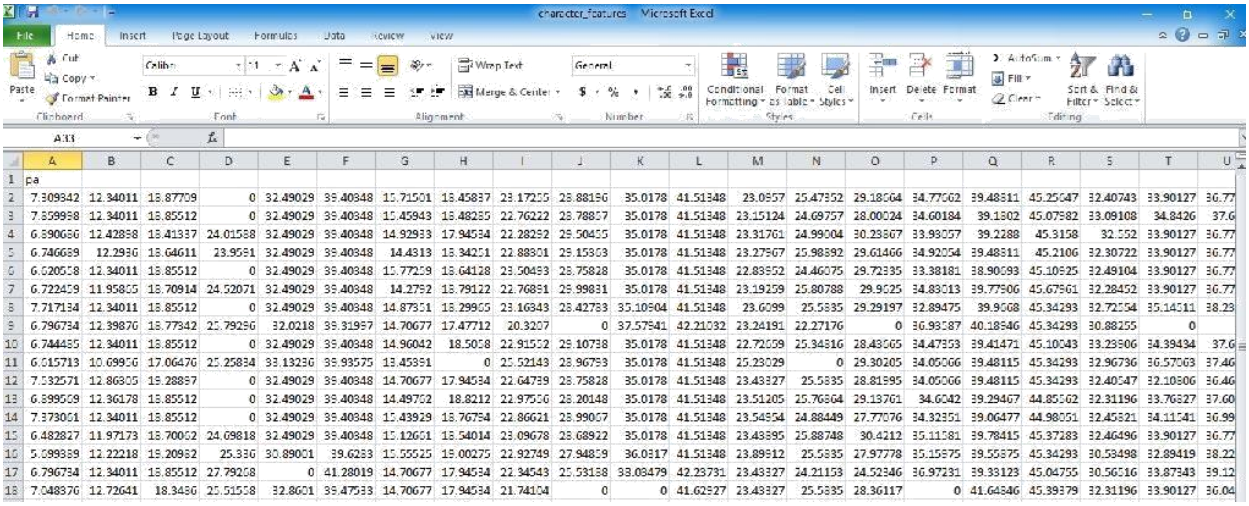
...eqn(1)

By dividing the sum of distances of all 1's pixels present in a box with their total number, a normalized vector distance (□) for each box is computed as:

$$\gamma_b = \frac{1}{n_b}\sum_{k=1}^{n_b} d_k{}^{b}$$

b= 1,2,3....24 ...eqn(2)

These 24 distance vectors constitute the complete feature set of a character, which are used for recognition. The points resulting from these 24 feature pairs have been back substituted in their respective boxes store generate the pattern of original character. All the features corresponding to each box are arranged in a excel sheet, as shown below for further analysis:



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	pa																				
2	7.309842	12.34011	18.87709	0	32.49029	39.40848	15.71501	18.45837	23.17255	25.88196	35.0178	41.51348	23.0957	25.47352	29.18064	34.77002	39.48811	45.25047	32.40743	39.90127	36.77
3	7.359958	12.34011	18.85512	0	32.49029	39.40848	15.45943	18.48285	22.76222	25.78857	35.0178	41.51348	23.15124	24.69757	28.00024	34.60184	39.1302	45.07582	33.09108	34.8426	37.6
4	6.890686	12.42838	18.41337	24.01588	32.49029	39.40848	14.92933	17.94584	22.28292	25.50455	35.0178	41.51348	23.31761	24.99004	30.23867	33.98057	39.2288	45.3158	32.552	39.90127	36.77
5	6.746689	12.2936	18.64611	23.9591	32.49029	39.40848	14.4313	18.34251	22.88301	25.15303	35.0178	41.51348	23.27567	25.98392	29.61466	34.92054	39.48811	45.2106	32.30722	39.90127	36.77
6	6.620558	12.34011	18.85512	0	32.49029	39.40848	15.77229	18.64128	23.50483	25.75828	35.0178	41.51348	22.83552	24.46075	29.72335	33.38181	38.90093	45.10525	32.49104	39.90127	36.77
7	6.722459	11.95805	18.70914	24.52071	32.49029	39.40848	14.2752	18.79122	22.76891	25.99831	35.0178	41.51348	23.19259	25.80788	29.9025	34.89013	39.77506	45.67961	32.28452	39.90127	36.77
8	7.717134	12.34011	18.85512	0	32.49029	39.40848	14.87301	18.29905	23.16343	25.42733	35.10504	41.51348	23.6099	25.5835	29.29197	32.89475	39.9068	45.34293	32.72254	35.14011	38.23
9	6.796734	12.39876	18.77342	25.79296	32.0218	39.31957	14.70677	17.47712	20.3207	0	37.57541	42.21032	23.24191	22.27176	0	36.98387	40.18546	45.34293	30.88255	0	
10	6.744485	12.34011	18.85512	0	32.49029	39.40848	14.96042	18.5058	22.91552	25.10738	35.0178	41.51348	22.72659	25.34816	28.48365	34.47353	39.41471	45.10043	33.23806	34.39434	37.6
11	6.015713	10.69956	17.06476	25.25834	33.13236	39.93575	13.45391	0	25.52143	28.96793	35.0178	41.51348	25.23029	0	29.30205	34.05066	39.48115	45.34293	32.96736	36.57063	37.46
12	7.532571	12.86805	19.28857	0	32.49029	39.40848	14.70677	17.94584	22.64739	25.75828	35.0178	41.51348	23.43227	25.5835	28.81995	34.05066	39.48115	45.34293	32.40547	32.10806	36.46
13	6.899509	12.36178	18.85512	0	32.49029	39.40848	14.49702	18.8212	22.97506	25.20148	35.0178	41.51348	23.51205	25.76864	29.13761	34.6042	39.29467	44.85562	32.31196	33.76827	37.60
14	7.373061	12.34011	18.85512	0	32.49029	39.40848	15.43929	18.76754	22.86621	25.99067	35.0178	41.51348	23.54854	24.88449	27.77076	34.32351	39.06477	44.98051	32.45821	34.11541	36.99
15	6.482827	11.97173	18.70052	24.09818	32.49029	39.40848	15.12661	18.54014	23.09678	25.68922	35.0178	41.51348	23.43895	25.88748	30.4212	35.11581	39.78415	45.37283	32.46496	33.90127	36.77
16	5.099389	12.22218	18.20982	25.336	30.89001	39.6283	13.55525	19.00275	22.92749	27.94829	36.0817	41.51348	23.89512	25.5835	27.97778	35.15875	39.55875	45.34293	30.58498	32.89419	38.22
17	6.796734	12.34011	18.85512	27.79258	0	41.28019	14.70677	17.94584	22.34543	25.58188	38.08479	42.23731	23.43227	24.21153	24.52346	36.97231	39.33123	45.04755	30.56016	33.87343	39.12
18	7.048376	12.72641	18.3486	25.51558	32.8601	39.47533	14.70677	17.94584	21.74104	0	0	41.62527	23.43227	25.5835	28.36117	0	41.64846	45.39379	32.31196	33.90127	36.04

Figure 3.6 Distance Vector (24 features)

Steps:

1. For extracting features from the slant corrected, size normalized, binary image Character is considered
2. The box approach based on spatial division of character image is proposed for feature extraction.
3. In this, the character image of the size of 42×32 is fitted into horizontal and vertical grid lines of 6× 4.
4. Thus, in this case 24 boxes of size 7×8 are devised and then these boxes are superimposed on the image, so that some of the boxes will have a portion of the image and others empty.
5. There are two ways in which box approach method can be implemented.
   i) By taking the bottom left corner as the absolute origin (0, 0).
   ii) By taking the central box as the absolute origin (0, 0).

   We are using (i).
6. We must calculate the vector distance of each pixel present in each box. Then for each 24 boxes we need to divide the 24 vector distances of each box by the no. of pixels present in each box.
7. Vector distance for the pixel in B^{th} box at location (i, j) is calculated as:

$$d_k^{\ b} = (i^2 + j^2)^{\frac{1}{2}} \qquad \text{...eqn(1)}$$

8. For each character we will get 24 features.
9. We have 150 sample characters of Devanagari script.
10. We will extract a matrix of 150x24, where 150 is no. of characters and 24 is no. of features of each character.
11. This matrix is extracted in excel sheet.

3.4 Training of Network

To train our network we are using 30 samples of each character. In our network we are giving our features that are extracted as an input to neuron. So, we will be having 24 inputs for each sample. Firstly, we tested the network for two hidden layers, but the results were not satisfying as the classification achieved was around 26%. So, we decided to train our network using three hidden layers, having 16, 10 and 5 neuron in each successive layer until the final output comes. Three neurons are treated as final output as we are expecting three-bit output. In three hidden layered networks, efficiency achieved is better and no. of characters classified are much more than two hidden layer network, i.e, 11 out of 15.

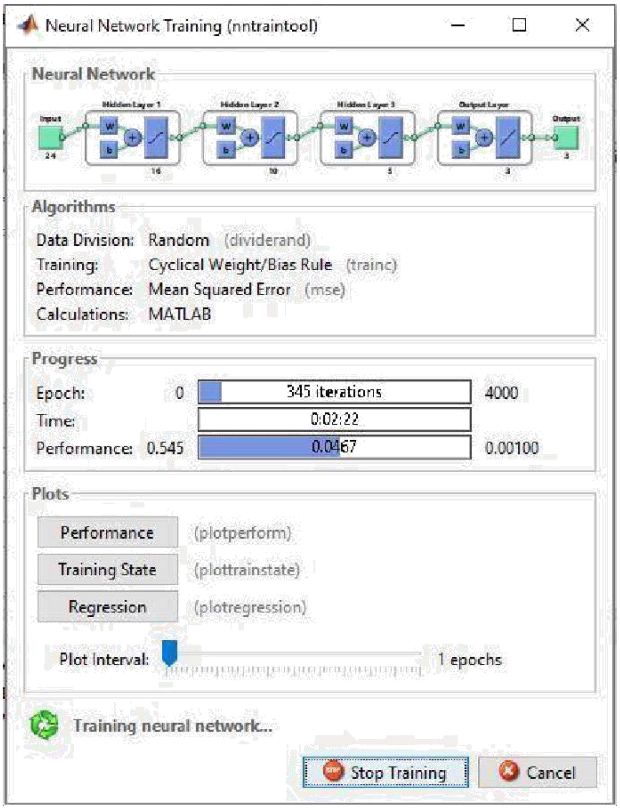


Figure 3.7 Training of Three Hidden Layer Network

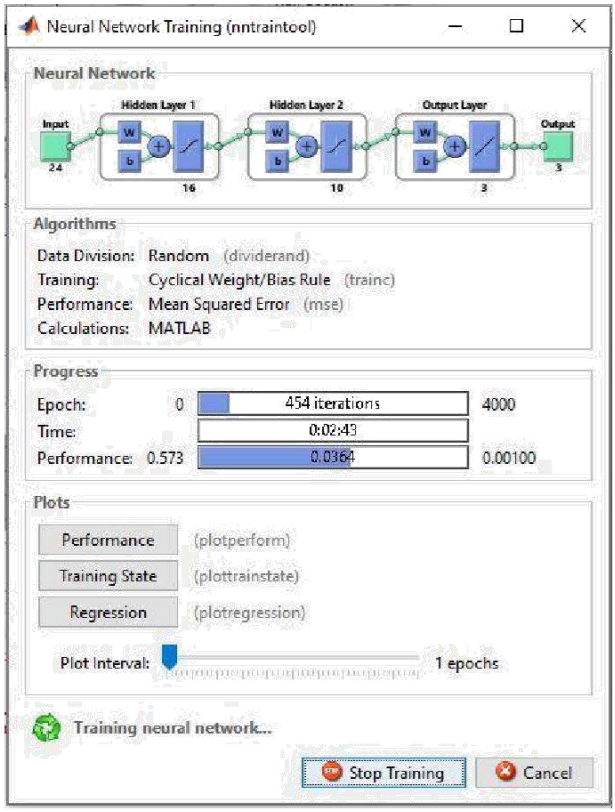


Figure 3.8 Training of Two Hidden Layer Network

3.4.1 Learning Rate (η)

Learning is a fundamental component to an intelligent system, although a precise definition of learning is hard to produce. In terms of an artificial neural network, learning typically happens during a specific training phase. Once the training has been done, it enters a production phase where it produces results independently. Training can take on many different forms, using a combination of learning paradigms, learning rules, and learning algorithms. We are performing the training keeping learning rate 0.5.

3.4.2 Number of Neurons in Hidden Layers

Hidden neurons are the neurons that are neither in the input layer nor the output layer. These neurons are essentially hidden from view, and their number and organization can typically be

treated as a black box to people who are interfacing with the system. Using additional layers of hidden neurons enables greater processing power and system flexibility. This additional flexibility comes at the cost of additional complexity in the training algorithm. Having too many hidden neurons is analogous to a system of equations with more equations than there are free variables: the system is over specified and is incapable of generalization. Having too few hidden neurons, conversely, can prevent the system from properly fitting the input data and reduces the robustness of the system.

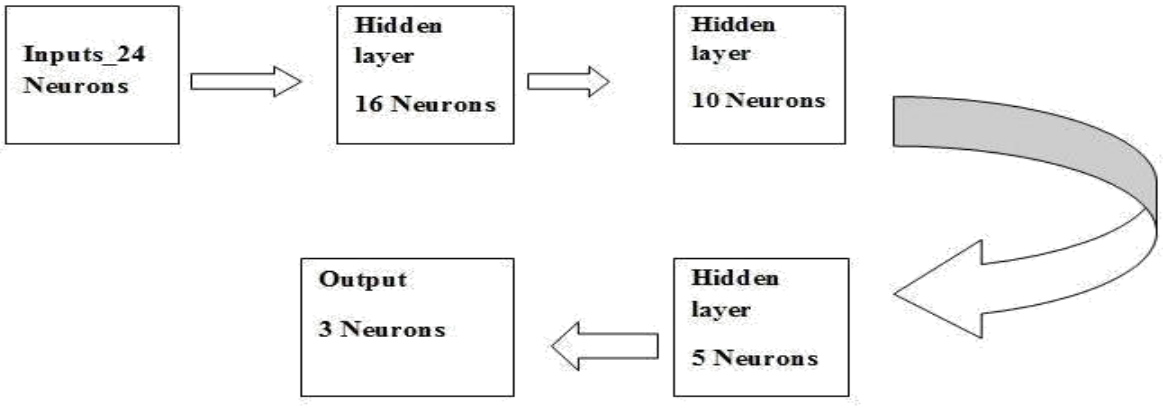


Figure 3.9 Neural Network Block Diagram

3.4.3 Epoch

Determines when training will stop once the number of iterations exceeds epochs. When training by minimum error, this represents the maximum number of iterations. The total epoch used in for training our neural network was equal to 4000 for achieving optimum results.

3.5 Backpropagation Algorithm

Backpropagation is a systematic method for training a multi-layer artificial neural network. An artificial neural network is a collection of interconnected processing elements or nodes. The nodes are termed simulated neurons as they attempt to imitate the functions of biological neurons. The nodes are connected via links. It is a multi-layer forward network using gradient descent or delta learning tool for back propagation error and thus minimizes the total squared error of the output computed by the network.

Initially, a weight is assigned at random to each link in order to determine the power of one node 's impact on the other. When the sum of input values reaches a threshold value, the node will produce the output 1 or 0 then. By adjusting the weights, the preferred output can be attained. This training process makes the network learn. The network, in other words, attains knowledge in much the same way human brains gain namely learning from experience. Backpropagation is one of the powerful artificial neural network techniques which is used acquire knowledge automatically.

Backpropagation method is also a supervised learning. The output is a real value which lies between 0 and 1 based on the sigmoid function.

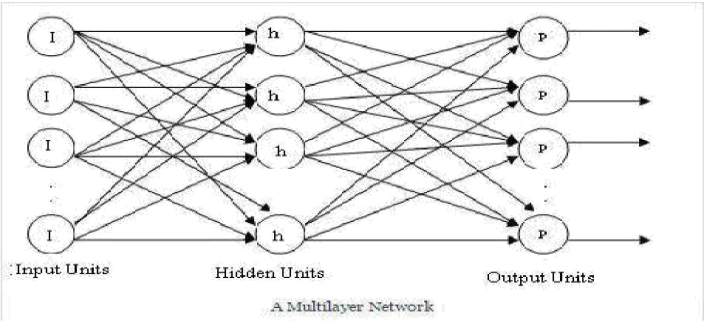


Figure 3.10 Neural Network

As the sum rises, the output approaches 1. As the sum falls, the output approaches 0.

A multilayer network is a kind of neural network which consists of one or more layers of nodes between the input and the output nodes. The input nodes pass values to the hidden layer, which in turn passes to the output layer. A network with a layer of input units, a layer of hidden units and a layer of output units is a two-layer network. A network with two layers of hidden units is a three-layer network, and so on. The multilayer network is the basis for backpropagation network.

As the name suggests, there is a backward pass of error to each internal node within the network, which is then used to calculate weight gradients for that node. The network learns by consecutively propagating forward the activations and propagating backward the errors as and when they occur.

Backpropagation network can deal with numerous types of data. It also has the capability to model a difficult decision system. If the tricky domain contains large amount of data, the network will require more input units or hidden units. Therefore, this will raise the complication of the model and increase its computational complexity. Additionally, it takes more time in resolving complex complications. In order to overcome this problem multi-backpropagation network is used.

In the commencement of the training method, weights are allotted to the connections arbitrarily. The training process is iterative. The training cases are given to the network iteratively. The actual outputs are related with preferred outputs and the errors are calculated. The errors are then propagated back to the network in order to modify the weights. The training process echoes till the desired outputs are obtained or the error reaches an acceptable level.

Steps:

Step 1: In the first step random values of weights are assigned.

Step 2: Feed Forward

An input signal is received by input unit and is transmitted to the hidden layer.

After this the hidden layer calculates the activation function and sends the signal to the output neuron. The output neuron calculates the activation function to give the response of the input pattern.

Step 3: Backpropagation of Errors

Error is calculated by comparing the associated activation Y_k output with desired output. Based on the error the factor δ_k (k=1,…m) is calculated and is used to distribute the error at output neuron Y_k back to all neurons in the previous layer.

Step 4: Updating the Weights and Biases

Each input unit receives the input signal x_i and transmits this signal to all the neurons of the hidden layer.

Step 5:

Each hidden neuron (Z_h,h=1,….p) sums it input signal and sends the signal to the neurons of output layer.

$$Z_{inj} = V_{oj} + \Sigma X_i \ X \ v_{ij} \quad \text{...eqn(3)}$$

Step 6:

Each output neuron (Y_k,k=1,……m) sums its weighted input signal

$$Y_{ink} = W_{ok} + \Sigma Z_j \ X \ W_{ij} \quad \text{...eqn(4)}$$

Step 7:

Error term is calculated as each output neuron receives a target pattern corresponding to a input pattern.

$\delta_k = (t_k - Y_k)\, f(Y_{ink})$ ...eqn(5)

Step 8:

Each hidden neuron sums its inputs from the neurons in the layer above

$\delta_{inj} = \Sigma\, \delta_j W_{jk}$ ...eqn(6)

The error is calculated $\delta_j = \delta_{inj}$.

Step 9

Each output neuron updates its bias and weights. The weight correction term is as follows

$$\Delta W_{jk} = \alpha \delta_k Z_j$$ ....eqn(7)

The bias correction term is as follows

$$\Delta W_{ok} = \alpha \delta_k$$ ...eqn(8)

Therfore,

$$W_{jk(new)} = W_{jk(old)} + \Delta W_{jk}$$
$$W_{ok(new)} = W_{ok(old)} + \Delta W_{ok}$$ ..eqn(9)

Step 9:

Each hidden neuron updates its bias and weights. The weight correction term is as follows

$$\Delta V_{jk} = \alpha \delta_j x_i$$ ...eqn(10)

The bias correction term is as follows

$$\Delta V_{oj} = \alpha \delta_j$$ ...eqn(11)

Therefore,

$$V_{ij(new)}=V_{ij(old)}+\Delta V_{ij},$$
$$V_{oj(new)}=V_{oj(old)}+\Delta V_{oj} \quad \text{...eqn(12)}$$

Step10: Stopping condition will be tested.

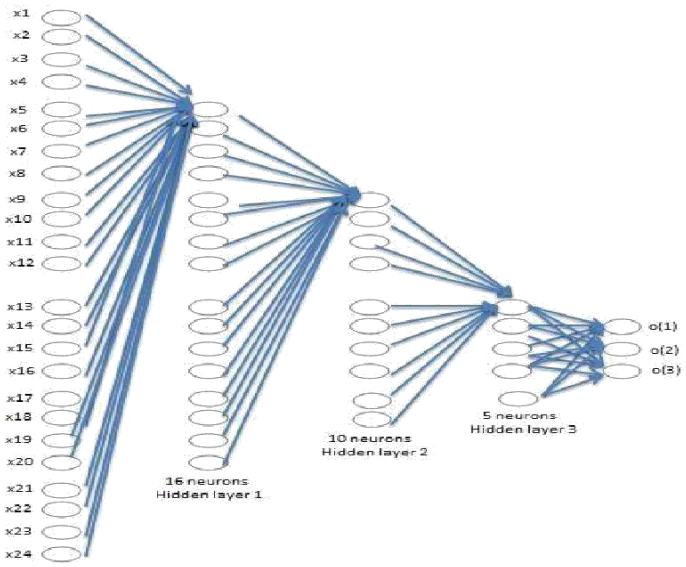


Figure 3.11 Backpropagation network with 24 input nodes, three hidden layers and an output layer

3.6 Testing of the network

The output obtained from backpropagation algorithm is compared against a threshold value. If the output is greater than threshold value, it is replaced with 1 else 0. The value of threshold was found by hit and trial method. Highest rate of classification was obtained for threshold value of 0.5. The modified output matrix is checked with the pre-defined target values of each character and the matching target and corresponding character are displayed as the recognized value.

CHAPTER 4

RESULTS

4.1 Comparison of 2 hidden layered and 3 hidden layered Network

Table 1.1 Comparison of networks developed

	3 Hidden Layers	2 Hidden Layers
Accuracy	73.34%	26%
Epochs	4000	4000
Performance Goals Set	0.001	0.001
Performance Goals Achieved	.00391	.0147
Learning rate	0.5	0.5
Computation Time	27 min	21 min

The network was first trained for one hidden layer for which the algorithm could achieve hardly any accuracy. Then the network was developed for various combinations of two hidden layers, the best out of which gave us 26% accuracy. Finally, we trained our network with three hidden layers i.e 16 neurons in hidden layer one, 10 neurons in hidden layer two and 5 neurons in hidden layer three. The success rate achieved after this was 73.3% as mentioned in Table 1.1. The three hidden layer network was trained at different learning rates i.e η=0.5, η=0.6, η=0.7 and η=0.85, η=0.9. The best results were obtained for η=0.5.

4.2 GUI Implementation

GUI was developed for the project that shows all the steps on window that pops up when the program is running.

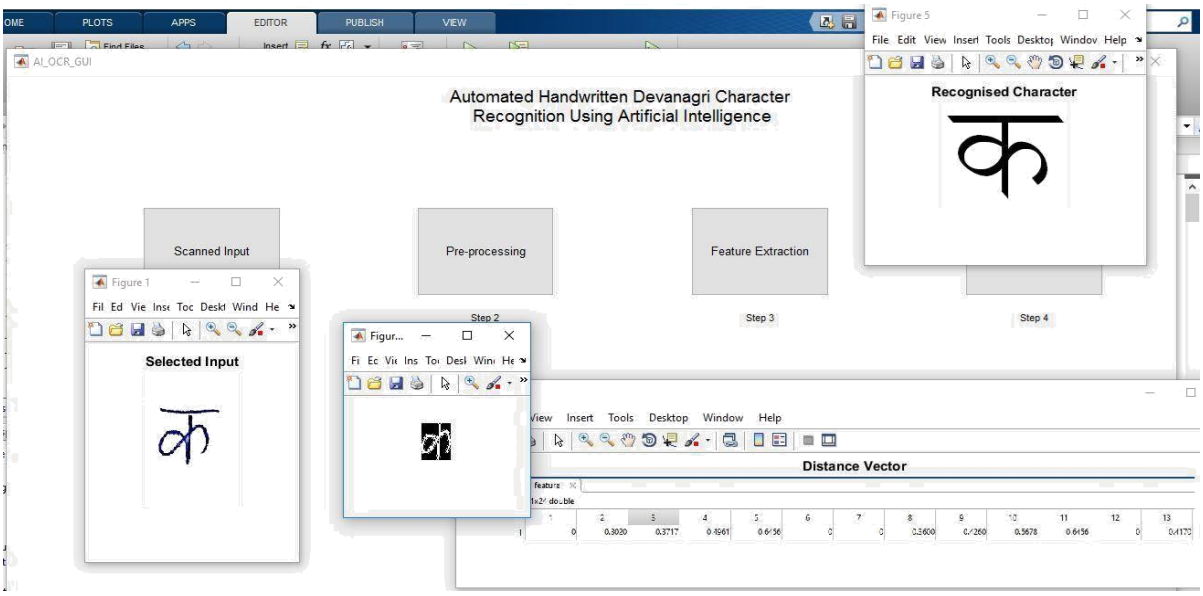


Figure 4.1 GUI showing steps of classified character

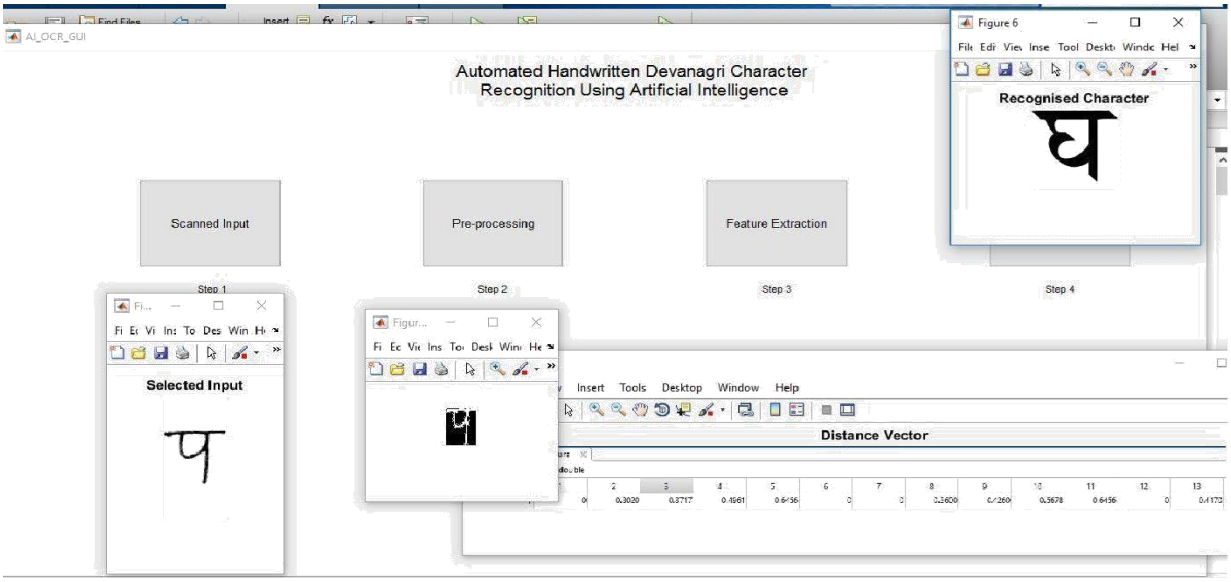


Figure 4.2 GUI showing steps of misclassified character

CHAPTER 5

CONCLUSION AND SCOPE OF IMPROVEMENT

5.1 CONCLUSION

In this paper, we have described a system for OCR of handwritten Devanagari script material. The recognition precision of the prototype implementation is promising, but more work needs to be done. We have used basic backpropagation algorithm for classification which gave us an accuracy of 73%. By using high end algorithms, we can achieve higher accuracy. The work has been done for only 5 characters of the Devanagari script and hence is character specific. We have also included similar looking characters to so that our algorithm can process the variations in the handwriting styles.

5.2 SCOPE OF IMPROVEMENT

• Segmentation of characters can be improved to include overlapping and joined characters. New algorithms can be developed to help the segmentation of joined hand written letters.

• The current algorithm can be extended for all the letters of the Devanagari script. This would involve preparation of a much larger data set. Also, different ratios will be needed for letters with different Matras '. It can be further improved to work on words with characters touching each other or even sentences.

• The algorithm used in this project though very efficient in classifying an image but lacks pinpoint accuracy in identifying a singular perfect match. In order to do so the algorithm must depend on the distributional properties of the image, this requires a rigorous analysis and study of particular image and still rigorous programming. This results at times the thresholds and limiting conditions being set by hit and trial approach, this can be avoided by more experimentation and further careful study of complex characters.

REFERENCES

1) J. Mantas, An overview of character recognition methodologies, Pattern Recognition 19 (1986) 425–430.

2) I.K. Sethi a n d B. Chatterjee, — Machine R e co gni t i on of constrained Hand printed Devnagari||, Pattern Recognition, Vol. 9, pp. 69-75, 1977.

3) R.M.K. Sinha and H.N. Mahabala, —Machine recognition of Devanāgarī script||, IEEE Transactions on Systems, Man and Cybernetics, 9(8), 1979, pp. 435-441.

4) S. Mori, C.Y. Suen, K. Yamamoto, Historical review of OCR research and development, Proc. IEEE 80 (1992) 1029–1058.

5) M. Hanmandlu, O.V.Ramana Murthy, —Fuzzy Model based recognition of handwritten numerals||, Pattern Recognition, 40(6), 2007, pp.1840-1854.

6) S.W. Lee, Offline recognition of totally unconstrained handwritten numerals using multi-layer cluster neural networks, IEEE Trans. Pattern Anal. Mach. Intell. 18 (6) (1996) 648– 652.

7) K. Ray, B. Chatterjee, Design of a nearest neighbor classifier system for Bengali character recognition, J. Inst. Electron. Telecom. Eng. 30 (1984) 226–229.

8) A.K. Dutta, A generalized formal approach for description and analysis for major Indian scripts, J. Inst. Electronic Telecom. Eng. 30 (1984) 155–161.

9) B.B. Chaudhuri, U. Pal, A complete printed Bangla OCR system, Pattern Recognition 31 (1998) 531–549.

10) S. Mori, K. Yamamoto, M. Yasuda, Research on machine recognition of hand-printed characters, IEEE Trans. Pattern Anal. Mach. Intell. 6 (1984) 386–405.

- R. Jayadevan, Satish R. Kolhe, Pradeep M. Patil, and Umapada Pal —Offline Recognition of Devanagari Script: A Survey||, IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS, VOL. 41, NO. 6, NOVEMBER 2011.