Handwritten Gurmukhi Akshara Recognition Using Convolutional Neural Network

Dissertation

Submitted in partial fulfilment of the requirements for the award of degree

of

Master of Technology

in

Computer Science and Application

Submitted By

Maneet Manoor

(Roll No. 601403016)

Under the supervision of:

Dr. R.K. Sharma

Professor, CSED

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

THAPAR UNIVERSITY

June 2016

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, **"Handwritten Gurmukhi Akshara Recognition Using Convolutional Neural Network"**, in partial fulfilment of the requirements for the award of degree of Master of Technology in *Computer Science and Application* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of Dr. R.K. Sharma and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

Maneet Manoor

Maneet Manoor

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

Dr. R.K. Sharma 30/6/16

Professor, CSED

Countersigned by

Dr. Maninder Singh

Head

Computer Science and Engineering Department

Thapar University

Patiala

Dr. S. S. Bhatia

Dean (Academic Affairs)

Thapar University

Patiala

ACKNOWLEDGEMENT

The successful completion of any task would be incomplete without acknowledging the people who made it possible and whose constant guidance and encouragement secured the success. First of all, I would like to express my gratitude to **Dr. R.K. Sharma, Professor,** Computer Science and Engineering, Thapar University, Patiala for introducing me to topic Online Handwritten Character Recognition and for all his guidance and support. This thesis work was enabled and sustained by his vision and ideas. I have been amazingly fortunate to have an advisor who gave me the freedom to explore on my own and at the same time the guidance to recover when my steps faltered. His patience and support helped me overcome many crisis situations and finish this dissertation. He has been a source of inspiration for me.

I am grateful to **Dr. Maninder Singh,** Head of Department, and **Dr. Sanmeet Kaur, Assistant Professor,** Department of Computer Science and Engineering, entire faculty and staff of Computer Science and Engineering and then friends who devoted their valuable time and helped me in all possible ways towards successful completion of this work. I thank all those who have contributed directly or indirectly to this work. I would like to express my heartfelt thanks to my friends who with their thought provoking views, veracity and whole hearted co-operation helped me in doing this dissertation.

Lastly, I would also like to thank my family for their years of unyielding love and encouragement. They have always wanted the best for me and I admire their determination and sacrifice.

(Maneet Manoor)

ABSTRACT

Handwritten Character Recognition Systems are the systems that are capable of identifying the symbols/characters drawn on graphical interface by hand using some input device. Online Handwriting Recognition System includes different stages, namely, Pre-processing, Feature Extraction, Classification and Recognizing. In Pre-processing phase, basic algorithms are used for segmentation of characters, centring/normalizing the coordinates of the character drawn, smoothing and slant correction etc. Feature Extraction is a process in which we try to extract only that relevant information from the pre-processed character that will be sufficient to classify the character drawn. It can be a low level feature or high level feature. Different classifiers are used to classify and recognize the symbol/character. This report presents a Convolutional Neural Network based Gurmukhi akshara recognition system which is trained using Stochastic Diagonal Lavenberg-Marquardt Backpropogation algorithm. Convolutional Neural Networks are analogous to neural networks having learnable weights and biases which evaluates raw digitized pixels to final winner class scores. A comprehensive recognition rate of 75.2% is achieved using 3-fold cross-validation strategy on a set of 10,500 images.

Contents

List of Tables

List of Figures

CHAPTER 1

INTRODUCTION

1.1 Handwritten Character Recognition Systems

Handwritten Character Recognition Systems are the systems that are capable of identifying the patterns in an image or drawn using some input device like mouse, stylus etc on a computer screen, that follows the syntax and semantics of a language. These systems play very important role in our daily lives like identifying the house numbers displayed on board outside houses, scanning the car numbers, optical character recognition, reading the sign board etc. Lots of research work has been done on Recognition Systems. Different classifiers and feature extraction techniques have evolved which makes the recognition task easy. There is a need to study all these different classifiers and feature extraction techniques. For different languages as we do not get same recognition results when tested using a given classifier and a given feature extraction technique. If a system trained for English characters have shown 99% efficiency using some xyz classifier, that classifier is not guaranteed to give same results for some other language. This difference is due to the variations in language syntax and semantics and also the variations in writing styles of different writers. Language like English have small and capital letters of each character, but this is not the case with languages like Hindi, Punjabi and Malayalam etc. This is the perfect example of variations in syntax and semantics of different languages. Every writer has one's own writing styles, these variations also make the recognition task hard. We cannot train the system for all the styles and patterns possible for each and every character.

Handwriting Recognition System includes different stages, namely, Pre-processing, Feature Extraction, Classification and Recognizing. In Pre-processing phase, basic algorithms are used for segmentation of characters, centring/normalizing the coordinates of the character drawn, smoothing and slant correction etc. Noises and distortions are also removed from the input data. Some of the basic pre-processing techniques are removing duplicate points, size normalisation, slant correction, interpolation, sampling, anti aliasing and slope corrections etc. [8]. Next to the pre-processing phase, a recognition system does the feature extraction. Features are defined as the information that a classifier needs to classify the dataset into different classes. Feature extraction is a process in which we try to extract only that relevant

information from the pre-processed character that will be sufficient to classify the character drawn. It can be a low level feature or a high level feature. Once the relevant features are extracted, different classifiers can be used to classify the symbol/character.

Handwriting Recognition system are broadly classified into two categories – Online Handwriting Recognition system and Offline Handwriting Recognition system.

Online Handwriting Recognition system takes the input from the mouse/pen/stylus movements on a digital device screen. The input data can be a stroke or a complete character. These systems have following phases as shown in Figure 1 [8].

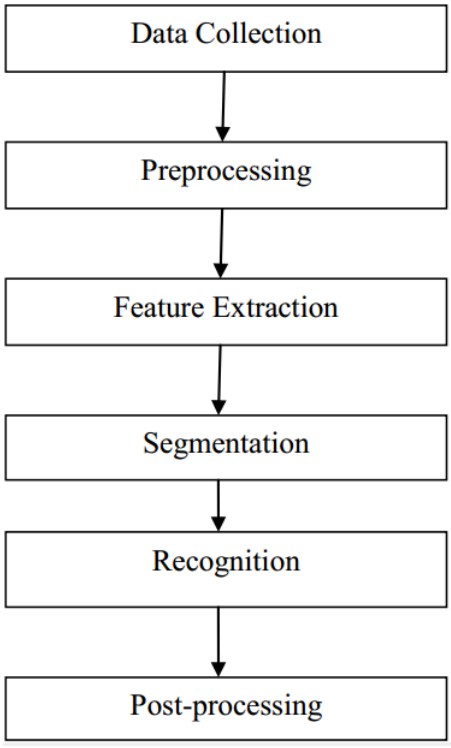


Figure 1. Phases in Online Handwritten Character Recognition

Offline Handwriting Recognition system takes the input from the image and then processes the data to a form that can be used to classify it. Often, these systems are called as Optical Character Recognition Systems (OCRS). These systems have two major phases – Character Extraction and Character Recognition.

1.2 Gurmukhi Script

The Punjabi language which is written using Gurmukhi script has its origin in Punjab, India and is among one of the 29 official languages spoken in India. Gurmukhi literally means 'from the mouth of the Guru' and was initially used to record the sayings by Sikh Gurus. This script consists of 35 characters and this is the reason why this script is also called as *Painti Akhri*. There is no case sensitivity and is written from left to right in horizontal lines.

The Gurmukhi character set is divided into various categories as shown in Figure 2 [6].

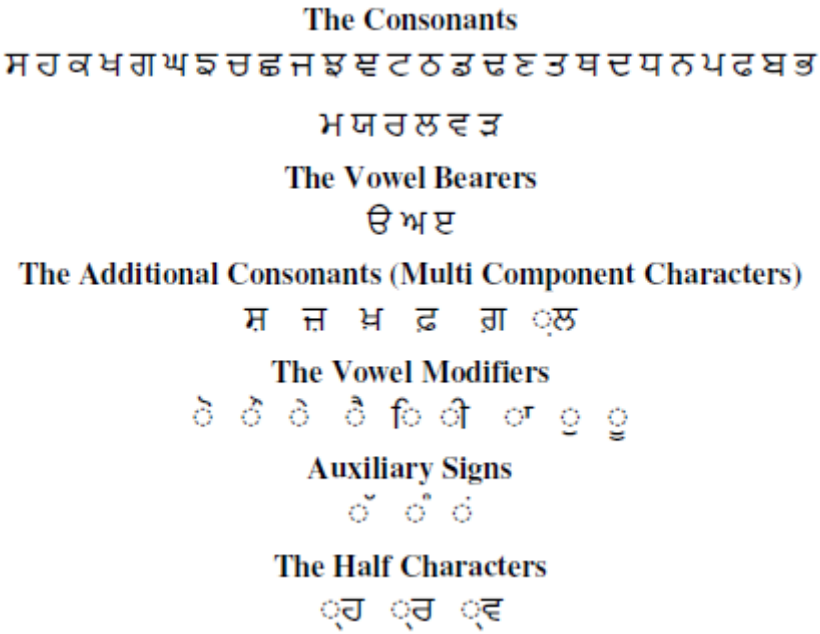


Figure 2. Complete Character Set of Gurmukhi

1.3 Convolutional Neural Network

Convolutional Neural Networks (ConvNets) are analogous to ordinary neural networks (NNs), comprised of neurons with learnable weights and biases. Each neuron computes dot product of the input and adds some bias to it.

ConvNets have been used for developing handwritten character recognition systems owing to the following facts:

- ConvNets extract significant features directly from the digitized images.

- A very little amount of pre-processing is required in ConvNets, as such this allows significant saving in computational cost.
- ConvNets fuse together three architectural ideas, namely, local receptive fields, shared weights and spatial / temporal sub sampling. These in turn help in achieving robustness against some degree of shift, scale and distortion [1].
- ConvNets doesnot require knowledge of syntax or semantic structures of the language [3].

The building blocks of ConvNets are 3 types of layers: Convolution Layer, Max Pooling Layer and Fully Connected Layer. Different topologies of all these layers are designed to build a ConvNet. Each layer processes the 3D input and transforms it to 3D output through a differentiable function.

1.3.1 Convolution Layer

Each Convolution layer consists of learnable filters that pass through full height and width of the 3D input. As this learnable filter extends through the 3D input, it computes the dot product between the values of a raw pixel image and a filter values as shown in Figure 3 [32]. It produces a 2D activation map of that filter. All the 2D outputs produced along the height and width of the input, and along with the depth are stacked together to produce full output of the layer.

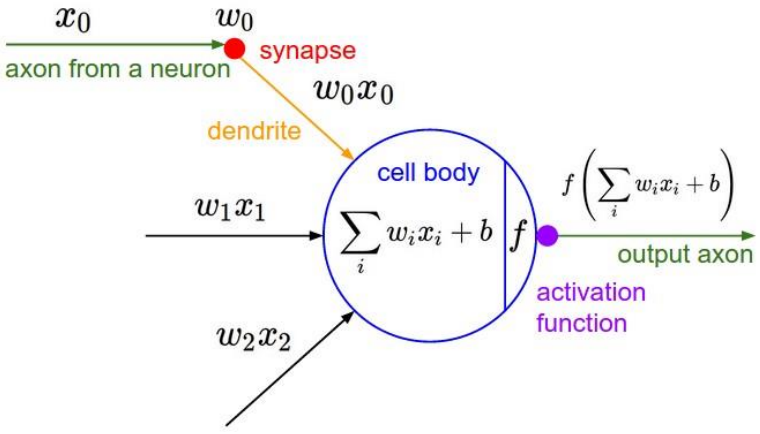


Figure 3. Working of Neuron

Local Connectivity: When working with the 3D input volume, it is impossible to connect all neurons to every previous layer's neuron. Instead each neuron is connected to a small unit. The extent of this unit is called as receptive field.

For example, if we have an input of size - 32×32×3 and receptive field is of size - 5×5, then every neuron will have 5×5×3 connections to the input as shown in Figure 4 [32].

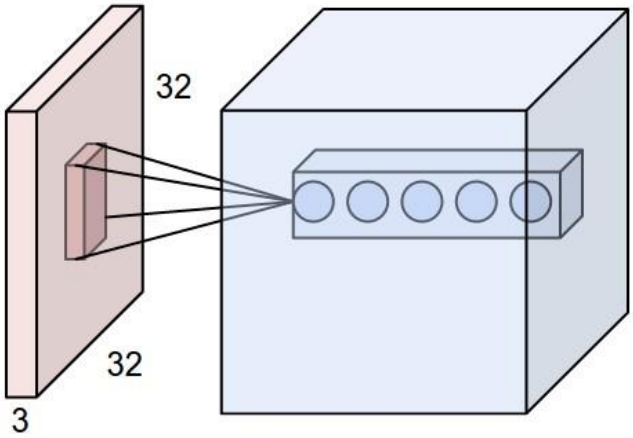


Figure 4. Local Connectivity

Spatial Arrangement: This arrangement explains how many neurons are there in output and how they are arranged. There are 3 things that control the size of output volume. These are: Depth, Stride and Zero-padding.

i. Depth: This controls the number of neurons in the output volume that will connect to same input. For example, if the convolution layer takes image as input, then different neurons will activate along the depth of the input depending on the different colour blobs of image. The set of neurons, working on the same unit of input, are referred as Depth Column.

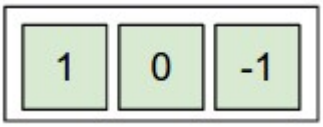
ii. Stride: It is defined as the step that is taken, when the filter is operating on the unit of the input along the height and width. For example, if we take stride as 1, the filter will jump along height/width with one step only. Higher stride value will result in less overlap and the output will be of small size. Conversely, lower stride results in heavy overlap and output generated is of bigger size.

iii. Zero Padding: This feature also controls the output volume size. Suppose w is size of the input, f is size of receptive field and s is stride value then, output volume size (O) is calculated as

$$O = [(w - f + 2 \times p) / s] + 1$$

Where p is size of the zero padding used.

For example, Output size (O) for input of size (w) = 5, stride (s) = 1 with zero padding (p) =1 is [(5 - 3 + 2) / 1]+1 = 5. Figure 5 [32] explains the output.



Neuron with receptive field (*f*) of 3.

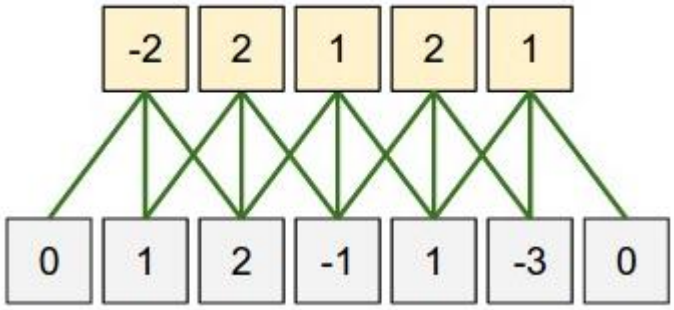


Figure 5. Dot Product of Input Vector and Receptive Field Vector with Stride = 1

For the same input but stride (*s*) = 2, output size (*O*) is [(5 - 3 + 2) / 2]+1 = 3.

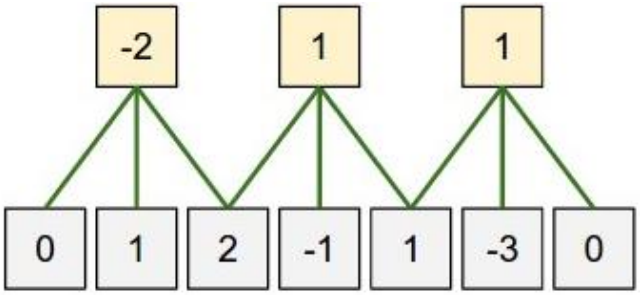


Figure 6. Dot Product of Input Vector and Receptive Field Vector with Stride = 2

1.3.2 Sub-Sampling Layer:

Task of this layer is to reduce the size of input which helps in controlling over fitting. It reduces the resolution of the image as shown in Figure 7 [32].

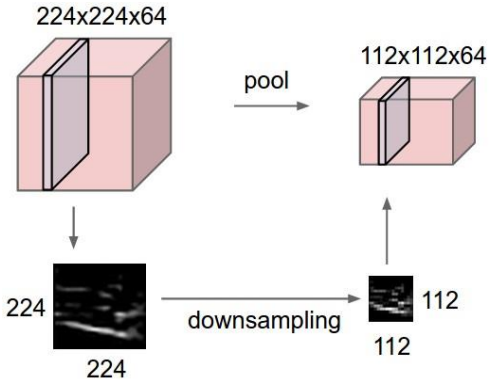


Figure 7. Working of Sub-Sampling layer

A Pooling layer with filter size 2×2, down samples the input by 2 along the height and width of the input as shown in Figure 8 [32].

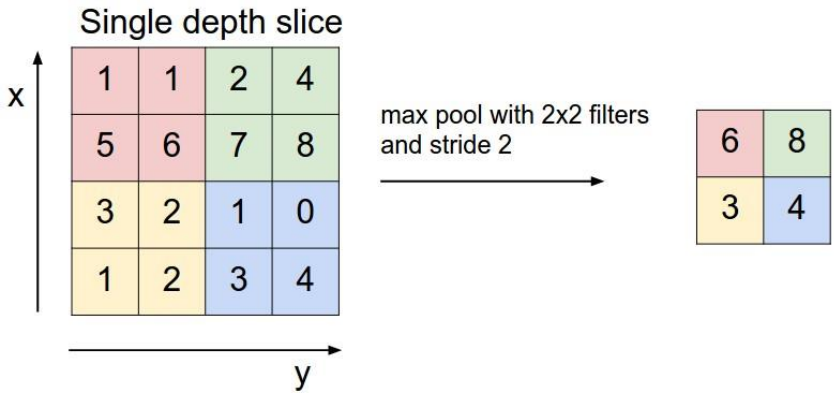


Figure 8. Downsampling of 4x4 Vector

1.3.3 Fully Connected layer:

This layer's neurons are connected to all the neurons of the previous layer. The output is computed as the matrix multiplication followed by addition of learnable bias to it.

1.3.4 Stochastic Lavenberg Marquardt Backpropogation Algorithm

Gradient based learning algorithm are of 2 types – Batch gradient and Stochastic gradient. In Batch gradient algorithms, parameters are updated once the exact gradients are calculated from the entire training dataset after training process. In stochastic mode, parameters are updated on the basis of a noisy gradient that is calculated using few samples of training dataset. These few samples are randomly selected from the entire dataset. Stochastic mode is

faster as compared to batch mode. The Stochastic Lavenberg Marquardt Backpropogation Algorithm is believed to converge faster as compared to the traditional version used without incurring additional cost. It also corrects the ill-functioning of the loss function. Owing to these reasons, we decided to train our system using stochastic learning mode.

1.4 Literature Review

Handwritten character recognition is an important issue in pattern recognition and a good amount of automation is required for this task. Bags of pre-processing, feature extraction and classification techniques have been evolved in past decades and calculable work has been done for character recognition of a number of languages. Following is the state of the art in the different phases of the recognition system.

1.4.1 State of the Art in Pre-processing

As discussed above, we have many steps to follow before we pass the data to classifier. Various algorithms have been proposed for pre-processing of the data which help to reduce the noise and increase the recognition results. According to Jeager *et al*. (2001), most commonly used pre-processing steps are size normalization, centering, interpolating missing points, smoothing, slant correction and resampling of points. While entering the data to system using pen/stylus, user may miss points. Unser *et al.* (1993) worked on inserting these missing points using Bezier insertion. Beigi *et al.* (1994) discussed the advantages of size normalization methods for online recognition systems. Kavallieratou *et al.* (2002) worked on smoothing of the character. Hosny *et al*. (2011) worked on recognizing Arabic characters using four pre-processing techniques, namely, Removal of duplicate points, Interpolating points, Smoothing and Resampling.

1.4.2 State of the Art in Feature Extraction

Next to pre-processing, feature extraction is done. Lots of feature extraction techniques have evolved in past few years. Every script has its own features, so feature extraction techniques are also different for all the scripts. There is no standard that a particular feature extraction technique will give good results for a given script. Rocha *et al.* (1994) proposed a method to recognize multifont printed characters emphasizing on structural description of character shapes. Lehal *et al.* (2000) worked on recognition of machine printed Gurmukhi script

extracting two types of features – primary and secondary. Verma *et al*. (2004) combined basic features like structural and directional characteristic with zoning data into a single feature vector. This technique works irrespective of character size.

1.4.3 State of the Art in Classification Techniques

Veltman *et al.* (1994) used Hidden Markov Model (HMM) to recognize lower case English alphabets. They achieved the average error rate of 6.9%.

Wakahara *et al.* (1997) used Stroke-based Affine Transformation (SAT) to recognize online cursive Kanji characters. Recognition rates of 98.4% and 96.0% are obtained on two types of test dataset, one written in cursive and other in square style.

Zho *et al.* (1999) worked on recognition of handwritten numerals using Quantum Neural Networks, which is a combination of Neural Networks and Fuzzy principals. A recognition rate of 99.1% has been reported by them.

Das *et al.* (2004) proposed a model for recognition of Bangla characters and numerals using Convex Hull based feature set. They obtained an accuracy of 76.9% on a dataset of 10,000 samples of Bangla Characters and 99. 5% on a dataset of 12000 samples of Bangla numerals.

Pal *et al.* (2007) experimented the modified quadratic classifier on Devanagari script. A comprehensive recognition rate of 94.2% is achieved using 5-fold cross-validation technique on a dataset of 36,172 handwritten characters.

Alijila *et al.* (2012) used Neural Networks for recognition of isolated Arabic handwritten characters. A recognition rate of 95.7% on untrained writers' dataset and 99.1% for trained writer's dataset is reported.

Jumanal *et al.* (2013) proposed a system to recognize online English characters using Genetic algorithm. The proposed algorithm is based on extracting spatial and temporal features from the character drawn. The system achieved an accuracy of 83.1% on a sample set of 5,200 handwritten characters.

Jignesh *et al.* (2015) worked on recognition of offline English character recognition using ANN. Features are directly extracted from pixels. Backpropogation Neural Network is used for classification purpose. They have reported accuracy of --- in their work.

1.4.4 State of Art in Gurmukhi Akshara Recognition Systems

Lehal *et al.* (1999) developed a complete OCR system for printed Gurmukhi characters, which classifies data using multistage recognition process using *K*-NN and Binary tree. The system uses two types of feature sets, namely, primary and secondary. Number of loops and junctions come under primary feature set. The number of endpoints and their location is considered under secondary feature set. They have reported an accuracy of 97.3% in this work. Lehal *et al.* (2001) extended the work by proposing a model for post-processing. This post-processing model makes decisions according to Punjabi grammar rules.

Sharma *et al*. (2008) implemented a recognition system for Gurmukhi aksharas using Elastic matching technique. The recognition is done in two stages. First stage evaluates the stroke. Second stage recognizes the character on the basis of stroke recognized in first stage. A recognition rate of 90.1% is achieved for 41 Gurmukhi characters. *Sharma et al.* (2009) extended this work proposing a new method for re-arrangement of recognized strokes for recognition of Punjabi words. Combinations of strokes and their re-arrangement were used to recognize words. The new system achieved the accuracy of 81.1% for a dataset of 2,576 Gurmukhi words.

Sharma *et al*. (2010) presented a HMM based Gurmukhi akshara recognition system. This system showed the accuracy of 91.9% for 2,460 samples of 41 Gurmukhi characters.

Kumar *et al.* (2012) tested different Feature-Classifier combination for isolated Gurmukhi offline character recognition. They used Linear Support Vector Machine (SVM), Polynomial SVM, RBF-SVM and *k*-NN as classifiers. Different feature extraction techniques like zoning feature, diagonal feature, directional feature, and parabola curve fitting based features to build a feature vector for a character. The proposed system achieved a recognition rate of 94.8%.

Sukhpreet *et al.* (2012) proposed a recognition system for isolated Gurmukhi characters using RBF-SVM as classifier and Gabor Filters as feature extraction technique. A cumulative recognition rate of 95.3% is achieved using 5-fold cross-validation technique on 7,000 samples of 35 Gurmukhi characters.

1.4.5 State of Art in Convolution Neural Network Based Recognition Systems

Convolutional Neural Networks have been experimented on MNIST database consisting of 70,000 samples of handwritten digits by LeCun *et al*. (1998). They reported an error rate of 0.95%.

Ahranjany *et al.* (2010) have tested ConvNets on extended IFH-CDB having 17,740 Farsi handwritten numerals and claimed an error rate of 0.83%.

ConvNets have also been analysed on Hoda database of 80,000 samples of Persian handwritten digits by Zamim *et al.* (2015) achieving an error rate of 0.07%.

1.5 Problem Statement

The Punjabi language which is written using Gurmukhi script has been investigated thoroughly using many classifiers like Hidden Markov Model (HMM), Support Vector Machine (SVM), k-Nearest Neighbor (*k*-NN) and Neural Network (NN) etc. However, ConvNets have not been used for classification purpose for this script. The present study is undertaken with a focus on developing a Convolution Neural Network based recognition system for Gurmukhi script.

CHAPTER 2

GURMUKHI AKSHARA RECOGNITION SYSTEM

Our Gurmukhi Akshara Recognition System is trained to recognize 35 characters (shown in figure 9) of Gurmukhi script. To build the system, we performed three major activities, namely, Data Collection, Pre-processing, Training the ConvNet.

Figure 9. Gurmukhi Characters

2.1 Data Collection

To train our recognition system, we collected 100 isolated sample images of each character from native writers and the database of 200 images of each isolated character has also been included as used in Kumar *et al.* (2012). Thus, we had 300 isolated sample images of each Gurmukhi character. Figure 10 shows the small sample set of 5 handwritten Gurmukhi aksharas written by 8 different writers in different styles.

Figure 10. First Five Gurmukhi Aksharas Written by Eight Different Writers in their Different Styles

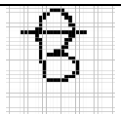
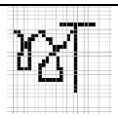
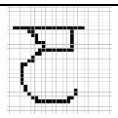
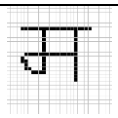
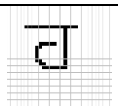
2.2 Pre-processing

Pre-processing is done to make the system robust against noises and different writing styles of different users. It also helps the easy learning with less parameter for the classifier thus, increasing the efficiency of any recognition system. The following are the pre-processing techniques applied:-

2.2.1 Minimum Bounded Box Extraction

In the data collection phase, different users entered characters of different sizes, shapes and styles. Before we use this database to train our system, we have to remove all the pixels belonging to marginal areas and extract only that area that minimally bounds the character. The elimination of these unwanted pixels will result in variable size character images that need to be normalised in size. Table 1 shows five sample characters canned in bounded box removing all the unwanted margins.

Table 1. First Five Gurmukhi Aksharas Canned in Minimum Bounded Box

Akshara written	Character canned in bounded box
	
	
	
	
	

2.2.2 Size Normalisation

After the extraction of character canned in minimum bounded box, this area is cropped out and resized to maximum 20×20 pixels image maintaining the original aspect ratio. Size normalisation makes the training and testing data uniform and helps a lot in recognition [7]. The aspect ratio (*R*) of image is calculated as:-

$$R = \begin{cases} \frac{w}{h}, & if\ w < h \\ \frac{h}{w}, & otherwise \end{cases}$$

Here *w* and *h* are width and height of canned character.

The resize factor (*f*), that will resize character to maximum of 20×20 pixels preserving the aspect ratio is calculated as:-

f = min (20/h, 20/w);

2.2.3 Zero Padding

When the number of rows (or columns) in the bounding box is less than 20, we need to pad such an image with zeros. For example, when cropped image is of size 18×20, it is then padded with zeros to make it a square image of size 20×20. This 20×20 image is then padded with zeros to make it 28×28 pixel image. This padding is done for the reason that all the relevant features like edges and corners of the character will come into the centre of convolution layer's receptive field area. Finally, the images are again padded with zeros to make it of 32×32 size. This is done to make input data compatible to LeNet-5 input layer. Table 2 shows five character images that are normalised in size and padded with zeros.

Table 2. First Five Gurmukhi Aksharas Canned in Minimum Bounded Box are Cropped and Centered to 28x28 Pixel Image.

Character canned in bounded box	Cropped and padded 28x28 image
	
	
	
	
	

2.3 Training the LeNet-5

LeNet-5 has 3 types of layers: Convolution layer, Subsampling layer, Fully connected layer arranged in following topology.

Input layer > Convolution layer (*C*1) > Subsampling layer (*S*2) > Convolution layer (*C*3) > Subsampling layer (*S*4) > Convolution layer (*C*5) > Fully connected layer (*F*6) > output layer.

LeNet-5 thus has total 6 layers when input layer and output layer are not counted. Figure 12 shows the complete topology of LeNet-5 [1]. Each layer contains trainable weights and biases. The input layer takes input from 32×32 pixel image having the largest character size of 20×20 capsulated by zero padding. The 20×20 pixel size character is centred in 28×28 window because of the reason that all the significant features like corners and edges come in the centre of receptive fields of Convolution layer. At last, the 28×28 image is padded with 0's to make it 32×32 pixel image. The pixel values are normalised to black for foreground pixels and white for background pixels.

The first convolution layer - *C*1 has 6 feature maps of 28×28 with stride 1. Each 5×5 unit of the input layer is connected to 5×5 unit of *C*1 layer.

The second sub sampling layer - *S*2 with 6 feature maps of 14×14 down samples the output of *C*1 layer. A unit of 2×2 of each feature map in *C*1 layer is connected to each feature map of *S*2 layer. This 2×2 receptive field is non-overlapping and the 4 inputs are added, multiplied by trainable coefficient and at last passed through the sigmoid function.

The third convolution layer *C*3 has 16 feature maps of 10×10 connected to 5×5 unit of *S*2 layer. Figure 11 shows the connection between *S2* and *C*3 layer [1].

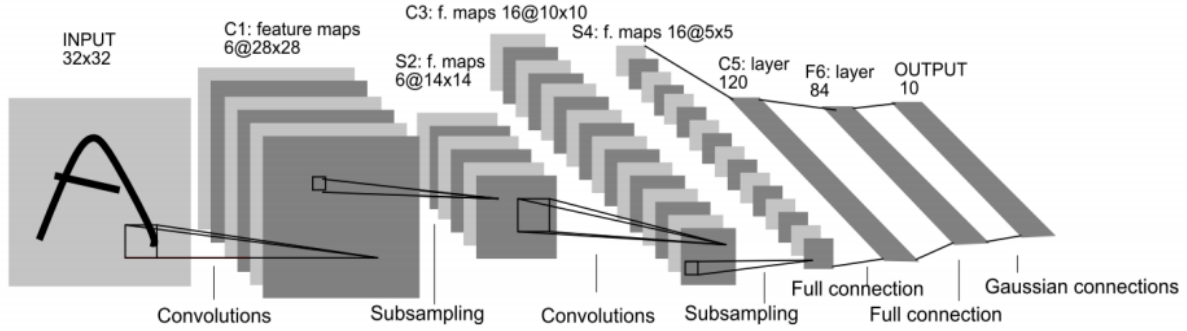


Figure 11. LeNet-5 Architecture.

The fourth sub sampling layer - *S*4 has 16 feature maps of 5×5. This layer again does the down sampling, which in turn reduces the resolution. Each unit of 2×2 of *C*3 layer is added, multiplied and passed through sigmoid function.

The fifth convolution layer - *C*5 has 120 feature maps. The unit of 5×5 of *S*4 layer feature maps is connected to each unit of *C*5 layer.

The sixth fully connected layer - *F*6 has 84 units and is fully connected to *C*5 layer. The input vector is multiplied to weight vector and bias is added. The output is then passed through a sigmoid squashing function.

The whole network is trained using stochastic backpropogation algorithm as it converges faster as compared to any gradient based technique without incurring any additional cost [1]. The Backpropogation technique used for training LeNet-5 is Stochastic Diagonal Lavenberg-Marquardt technique.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X				X	X	X			X	X	X	X		X	X
1	X	X				X	X	X			X	X	X	X		X
2	X	X	X				X	X	X			X		X	X	X
3		X	X	X			X	X	X	X			X		X	X
4			X	X	X			X	X	X	X		X	X		X
5				X	X	X			X	X	X	X		X	X	X

Figure 12. Each Row Represents the Connection of C3 Layer Feature Maps to S2 Layer Feature Maps.

After all the pre-processing is done, our database of 10,500 images is ready to train our recognition system. The network is trained using 7,000 images and tested on 3500 images using Stochastic Diagonal Lavenberg-Marquardt technique. The Hessian Diagonal re-estimation is done after each iteration using randomly selected 500 samples from the training dataset. We keep on increasing the iterations until we get the convergence in error rate.

CHAPTER 3

EXPERIMENTS AND RESULTS

We performed several experiments in this work. Experiments are divided into two phases. In the first phase, we tested the ConvNets on first five Gurmukhi aksharas – ੳ, ਅ, ੲ, ਸ and ਹ. In the second phase, we tested the ConvNets on all 35 Gurmukhi aksharas – ੳ, ਅ, ੲ, ਸ, ਹ, ਕ, ਖ, ਗ, ਘ, ਙ, ਚ, ਛ, ਜ, ਝ, ਞ, ਟ, ਠ, ਡ, ਢ, ਣ, ਤ, ਥ, ਦ, ਧ, ਨ, ਪ, ਫ, ਬ, ਭ, ਮ, ਯ, ਰ, ਲ, ਵ and ੜ. Table 3 shows statistics on these experiments.

Table 3. Experiment setup

Experiment #	# of training images	# of testing images
Phase – 1 Training for 5 Gurmukhi Aksharas		
1	25	25
2	40	10
3	1400	350
Phase – 2 Training for 35 Gurmukhi Aksharas		
4	1400	350
5	2100	350
6	2800	350
7	3500	700
8	7000	3500

3.1 Training the System for 5 Gurmukhi aksharas.

Experiment -1

In first experiment, we trained the system for only 5 Gurmukhi aksharas - ੳ, ਅ, ੲ, ਸ and ਹ using 25 images and tested it using 25 images. Training of the system is done till a convergence in error rate is obtained. In this experiment, system attained the convergence after 14 iterations and shows constant train/test error rate for next 20 iterations. For the next 5 iterations, test error drops from 44.0% to 36.0%. Table 4 shows error rates on train and test dataset. Figure 13 shows the train and test error rate for every iteration.

Table 4. Train and Test Error Rates for Experiment 1

Iterations	Eta	Train Error Rate %	Test Error Rate %
1	0.0005	24.0	36.0
2	0.0005	36.0	48.0
3	0.0002	32.0	72.0
4	0.0002	0.0	36.0
5	0.0002	0.0	28.0
6	0.0001	0.0	36.0
7	0.0001	0.0	36.0
8	0.0001	0.0	40.0
9	0.00005	0.0	44.0
10	0.00005	0.0	44.0
11	0.00005	0.0	40.0
12	0.00005	0.0	40.0

13	0.00001	0.0	40.0
14	0.00001	0.0	40.0
15	0.00001	0.0	44.0
16	0.00001	0.0	44.0
17	0.00001	0.0	44.0
18	0.00001	0.0	44.0
19	0.00001	0.0	44.0
20	0.00001	0.0	44.0
21	0.00001	0.0	44.0
22	0.00001	0.0	44.0
23	0.00001	0.0	44.0
24	0.00001	0.0	44.0
25	0.00001	0.0	44.0
26	0.00001	0.0	44.0
27	0.00001	0.0	44.0
28	0.00001	0.0	44.0
29	0.00001	0.0	44.0
30	0.00001	0.0	44.0
31	0.00001	0.0	44.0
32	0.00001	0.0	44.0

33	0.00001	0.0	44.0
34	0.00001	0.0	44.0
35	0.00001	0.0	44.0
36	0.00001	0.0	40.0
37	0.00001	0.0	40.0
38	0.00001	0.0	40.0
39	0.00001	0.0	40.0
40	0.00001	0.0	36.0

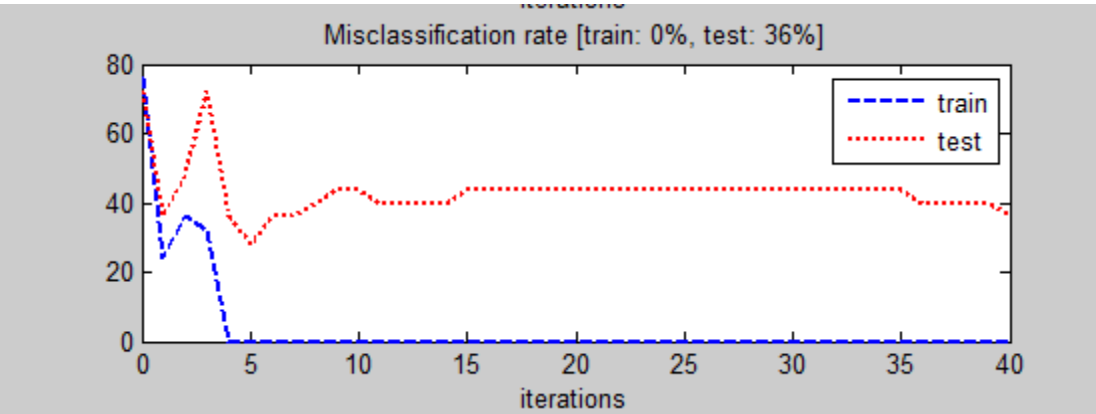


Figure 13. Misclassification Rate on Train and Test Dataset of Experiment – 1

The test error rate of 36% means 9 out of 25 images were misclassified. Table 5 shows misclassified characters with number of times they are misclassified.

Table 5. Misclassified Characters

Target Character	# of times misclassified
ੳ	4
ਅ	2
ਏ	1
ਸ	1

ਹ	1

Experiment - 2

To check if we can get good results by increasing the size of training dataset, we trained the system using 40 training images and tested using 10 testing images. The system attained the convergence after 14 iterations showing 0.0% train error rate and 10.0% test error rate. The 10% test error rate means 1 out of 10 testing images is misclassified. Only 'ਅ' is misclassified once. Table 5 shows error rates on train and test dataset. Figure 14 shows the train and test error rate for every iteration.

Table 6. Train and Test Error Rate for Experiment-2

Iterations	Eta	Train Error Rate %	Test Error Rate %
1	0.0005	20.0	40.0
2	0.0005	37.0	50.0
3	0.0002	42.0	60.0
4	0.0002	17.0	30.0
5	0.0002	15.0	10.0
6	0.0001	0.0	10.0
7	0.0001	5.0	20.0
8	0.0001	2.0	20.0
9	0.00005	7.0	40.0
10	0.00005	0.0	20.0
11	0.00005	0.0	10.0
12	0.00005	2.5	20.0

13	0.00001	0.0	20.0
14	0.00001	0.0	20.0
15	0.00001	0.0	10.0
16	0.00001	0.0	10.0
17	0.00001	0.0	10.0
18	0.00001	0.0	10.0
19	0.00001	0.0	10.0
20	0.00001	0.0	10.0
21	0.00001	0.0	10.0
22	0.00001	0.0	10.0
23	0.00001	0.0	10.0
24	0.00001	0.0	10.0
25	0.00001	0.0	10.0

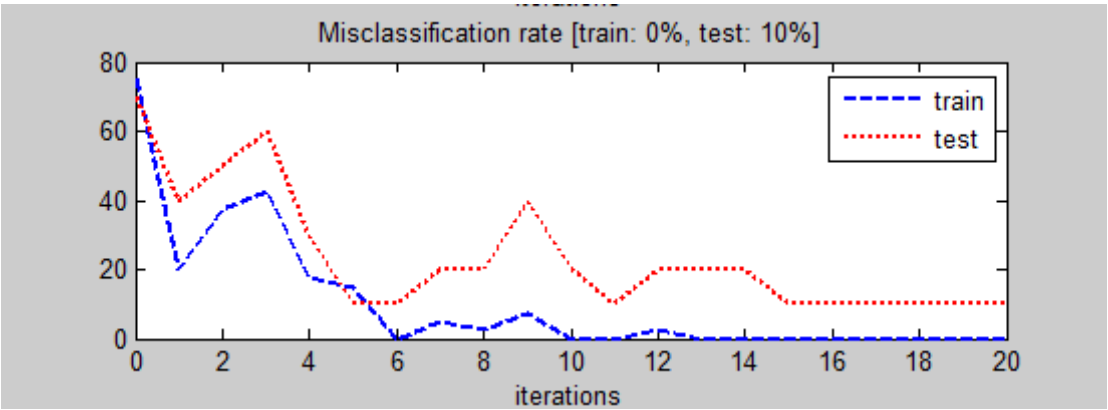


Figure 14. Misclassification Rate on Train and Test Dataset of Experiment – 2

Experiment – 3

In third experiment, we increased the training dataset size to 200 images and testing dataset to 50 images. The system attained the convergence after 13 iterations. By increasing the training dataset size to 200 images, the test error rate drops to 2.0%. Table 7 shows error rates on train and test dataset. Figure 15 shows the train and test error rate for every iteration.

Table 7.Train and Test Error Rate for Experiment-3

Iterations	Eta	Train Error Rate %	Test Error Rate %
1	0.0005	29.0	38.0
2	0.0005	2.0	0.0
3	0.0002	3.0	2.0
4	0.0002	1.0	2.0
5	0.0002	1.0	10.0
6	0.0001	3.0	0.0
7	0.0001	3.0	6.0
8	0.0001	12.0	12.0
9	0.00005	2.0	2.0
10	0.00005	2.0	2.0
11	0.00005	0.0	6.0
12	0.00005	0.0	2.0
13	0.00001	0.0	0.0
14	0.00001	0.0	2.0
15	0.00001	0.0	2.0
16	0.00001	0.0	2.0

17	0.00001	0.0	2.0
18	0.00001	0.0	2.0
19	0.00001	0.0	2.0
20	0.00001	0.0	2.0
21	0.00001	0.0	2.0
22	0.00001	0.0	2.0
23	0.00001	0.0	2.0
24	0.00001	0.0	2.0
25	0.00001	0.0	2.0

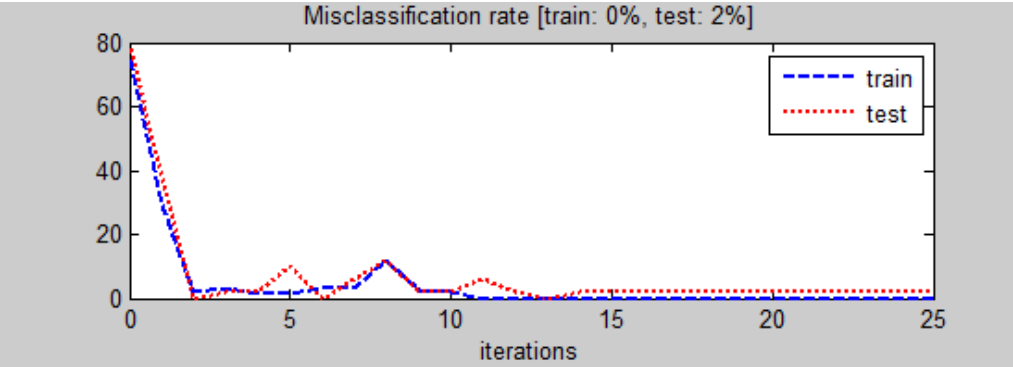


Figure 15. Misclassification Rate on Train and Test Dataset of Experiment – 3

The 2.0% test error rate means 1 out of 50 images were misclassified. Only 'ਅ' is misclassified once.

3.2 Training the System for 35 Gurmukhi aksharas

In the second phase, the system is trained for all 35 Gurmukhi aksharas. The training and testing dataset is increased in every experiment to check how the efficiency is affected.

Experiment – 4

Training of the system is done with 1400 training and 350 testing images. Misclassification rate of 19.0% on train data and 33.7% on test data is achieved. Training of the system is done till the convergence in error rate is obtained. After the system is trained for 20 iterations, the test error rate increases due to over training. Table 8 shows error rates on train and test dataset. Figure 16 shows the train and test error rate for every iteration.

Table 8. Train and Test Error Rates for Experiment - 4

Iterations	Eta	Train Error Rate %	Test Error Rate %
1	0.0005	89.9	91.7
2	0.0005	77.6	82.8
3	0.0002	82.7	82.0
4	0.0002	50.7	54.8
5	0.0002	63.2	69.1
6	0.0001	52.2	59.4
7	0.0001	62.8	73.1
8	0.0001	49.9	58.8
9	0.00005	47.0	54.8
10	0.00005	44.0	55.4
11	0.00005	54.7	66.7
12	0.00005	41.5	50.0
13	0.00001	45.1	57.4

14	0.00001	33.6	48.2
15	0.00001	37.5	48.5
16	0.00001	21.1	37.1
17	0.00001	20.1	33.4
18	0.00001	20.2	33.4
19	0.00001	18.2	32.8
20	0.00001	19.0	33.7

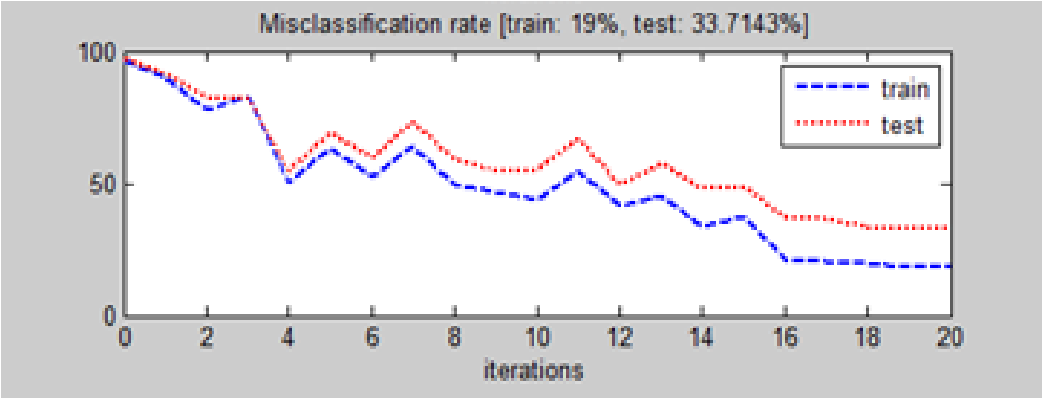


Figure 16. Misclassification Rate on Train and Test Dataset of Experiment – 4

Characters that are shown bold in table 9 are mostly misclassified by ConvNets. Table 10 shows these misclassified characters with the characters, they are confused with.

Table 9. Number of Times the Gurmukhi Characters are Misclassified in Experiment - 4

Target	# of times misclassified	Target	# of times misclassified	Target	# of times misclassified
ੳ	0	ਜ	4	ਨ	1
ਅ	6	ਝ	4	ਪ	4

ਏ	4	**ਢ**	9	ਫ	4
ਸ	2	ਟ	1	ਬ	3
ਹ	3	**ਠ**	5	**ਭ**	7
ਕ	0	ਡ	4	ਮ	2
ਖ	5	ਫ	3	**ਯ**	5
ਗ	2	ਣ	1	**ਰ**	9
ਘ	3	ਤ	1	ਲ	4
ਙ	2	**ਥ**	6	ਵ	4
ਚ	1	ਦ	2	ੜ	2
ਛ	0	ਧ	2	-	-

**Target characters in bold are most misclassified characters*

Table 10. Target Characters with Characters , They are Confused with.

Target Character	Misclassified with
ਅ	ਘ , ਯ
ਖ	ਯ
ਢ	ਵ

ਠ	ਣ, ੜ
ਥ	ਕ ,ਧ
ਭ	ਛ, ਣ
ਯ	ਘ , ਪ
ਰ	ੜ

Experiment – 5

Training of the system is done with 2100 training and tested with 350 images. Misclassification rate of 16.9% on train data and 23.9% on test data is achieved. Table 11 shows error rates on train and test dataset. Figure 17 shows the train and test error rate for every iteration.

Table 11. Train and Test Error Rate for Experiment - 5

Iterations	Eta	Train Error Rate %	Test Error Rate %
1	0.0005	90.5	92.5
2	0.0005	66.7	66.0
3	0.0002	53.0	57.3
4	0.0002	57.5	56.7
5	0.0002	68.1	68.5
6	0.0001	69.7	74.2
7	0.0001	54.7	64.0
8	0.0001	49.1	54.5

9	0.00005	41.4	47.3
10	0.00005	48.9	56.5
11	0.00005	47.9	56.5
12	0.00005	47.9	54.2
13	0.00001	29.9	41.7
14	0.00001	31.5	38.2
15	0.00001	41.5	49.1
16	0.00001	20.4	26.0
17	0.00001	18.4	26.8
18	0.00001	17.2	24.2
19	0.00001	16.1	26.0
20	0.00001	15.2	24.8
21	0.00001	15.0	25.1
22	0.00001	14.1	24.5
23	0.00001	14.0	23.7
24	0.00001	22.4	30.8
25	0.00001	33.3	38.2
26	0.00001	29.5	36.0
27	0.00001	26.9	34.0
28	0.00001	26.9	34.0
29	0.00001	22.9	30.2

30	0.00001	21.3	29.4
31	0.00001	21.2	28.8
32	0.00001	20.3	28.0
33	0.00001	19.1	28.8
34	0.00001	18.9	27.4
35	0.00001	18.8	26.2
36	0.00001	19.4	36.2
37	0.00001	18.1	25.8
38	0.00001	17.5	27.4
39	0.00001	17.2	27.1
40	0.00001	17.3	27.1
41	0.00001	16.3	28.2
42	0.00001	15.9	27.8
43	0.00001	15.6	26.0
44	0.00001	23.7	31.1
45	0.00001	20.1	30.2
46	0.00001	18.7	29.4
47	0.00001	18.2	26.0
48	0.00001	17.2	25.1
49	0.00001	16.9	25.8
50	0.00001	16.4	28.5

51	0.00001	17.7	26.8
52	0.00001	16.7	26.0
53	0.00001	16.4	27.1
54	0.00001	15.6	27.4
55	0.00001	15.4	26.8
56	0.00001	16.3	28.8
57	0.00001	15.8	26.4
58	0.00001	14.2	28.5
59	0.00001	13.2	25.4
60	0.00001	13.5	26.2
61	0.00001	13.8	25.2
62	0.00001	13.8	24.5
63	0.00001	13.5	25.4
64	0.00001	13.3	23.2
65	0.00001	16.9	24.9

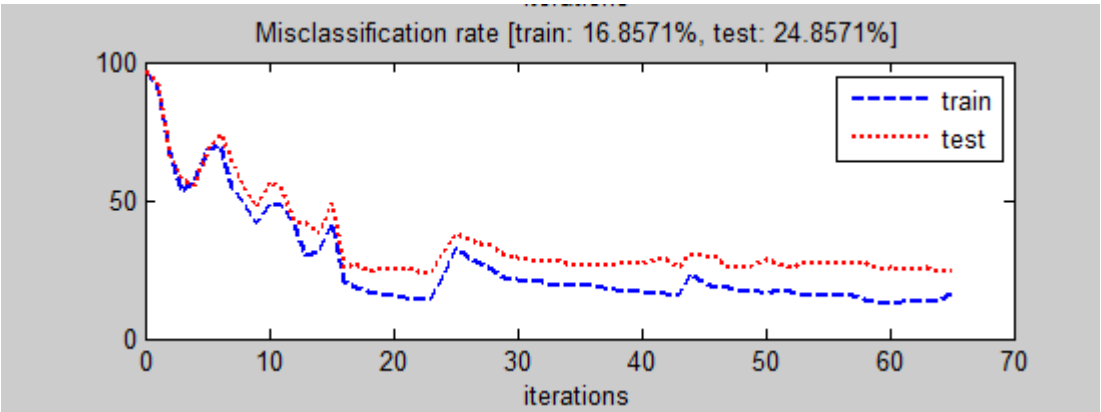


Figure 17. Misclassification Rate on Test and Train Dataset of Experiment - 5

The test error rate of 24% means 84 out of 350 images are misclassified. Characters that are shown bold in table 12 are mostly misclassified by ConvNets. Table 13 shows these misclassified characters with the characters, they are confused with.

Table 12. Number of Times the Gurmukhi Characters are Misclassified in Experiment - 5

Target	# of times misclassified	Target	# of times misclassified	Target	# of times misclassified
ੳ	0	ਜ	3	ਨ	2
ਅ	2	ਝ	3	ਪ	2
ਏ	3	**ਢ**	5	ਫ	2
ਸ	6	ਟ	1	ਬ	2
ਹ	2	ਠ	1	ਭ	1
ਕ	1	ਡ	3	ਮ	2

ਖ	5	**ਢ**	1	ਯ	1
ਗ	0	ਣ	1	**ਰ**	6
ਘ	3	ਤ	0	ਲ	4
ਝ	5	**ਥ**	8	**ਵ**	6
ਚ	2	ਦ	0	ੜ	2
ਛ	2	ਧ	2	-	-

**Target characters in bold are most misclassified characters*

Table 13. Target Characters with Characters, They are Confused with.

Target Character	Misclassified with
ਸ	ਗ
ਖ	ਥ
ਝ	ਝ
ਢ	ਵ
ਥ	ਕ ,ਧ
ਰ	ੜ, ਚ
ਵ	ਢ

Experiment – 6

Training of the system is done with 2800 training and tested with 700 images. Misclassification rate of 13.2% on train data and 23.5% on test data is achieved. System is trained for 90 iterations. Table 14 shows error rates on train and test dataset. Figure 18 shows the train and test error rate for every iteration.

Table 14. Train and Test Error Rate for Experiment - 6

Iterations	Eta	Train Error Rate %	Test Error Rate %
1	0.0005	26.5	34.7
2	0.0005	21.2	28.7
3	0.0002	13.7	25.0
4	0.0002	12.8	24.2
5	0.0002	24.0	28.2
6	0.0001	15.1	27.4
7	0.0001	13.6	23.7
8	0.0001	9.6	22.5
9	0.00005	25.6	33.9
10	0.00005	16.8	27.1
11	0.00005	14.9	26.3
12	0.00005	15.2	23.8
13	0.00001	12.0	25.3
14	0.00001	22.7	30.5
15	0.00001	16.3	25.6
16	0.00001	10.5	22.3
17	0.00001	10.2	21.3
18	0.00001	10.6	21.6

19	0.00001	11.2	21.4
20	0.00001	11.2	21.5
21	0.00001	10.5	21.5
22	0.00001	14.1	24.8
23	0.00001	12.5	23.1
24	0.00001	12.5	22.7
25	0.00001	13.2	23.5

.

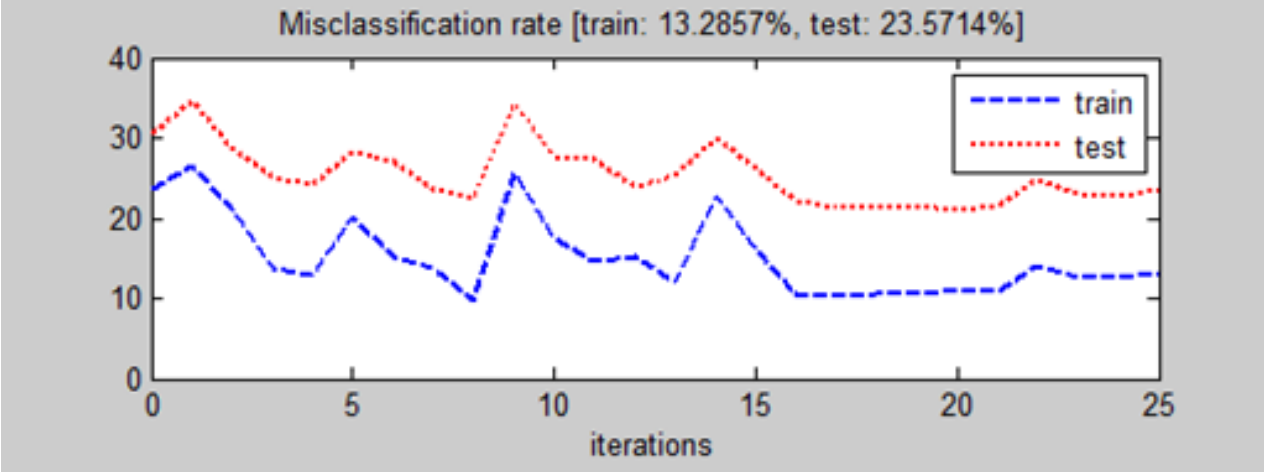


Figure 18. Misclassification Rate on Train and Test Dataset of Experiment – 6

The test error rate of 24% means 168 out of 700 images are misclassified. Characters that are shown bold in table15 are mostly misclassified by ConvNets. Table 16 shows these misclassified characters with the characters, they are confused with.

Table 15. Number of Times the Gurmukhi Characters are Misclassified in Experiment - 6.

Target	# of times misclassified	Target	# of times misclassified	Target	# of times misclassified
ੳ	0	ਜ	3	ਨ	3

ਅ	2	**ਝ**	3	**ਪ**	5
ਏ	3	**ਢ**	7	**ਫ**	8
ਸ	11	ਟ	4	**ਬ**	5
ਹ	5	ਠ	4	**ਭ**	5
ਕ	0	**ਡ**	7	ਮ	3
ਖ	6	**ਢ**	5	**ਯ**	5
ਗ	3	**ਣ**	5	**ਰ**	10
ਘ	6	**ਤ**	8	**ਲ**	5
ਙ	7	ਥ	4	**ਵ**	7
ਚ	1	**ਦ**	6	ੜ	2
ਛ	0	**ਧ**	8	-	-

**Target characters in bold are most misclassified characters*

Table 16 . Target Characters with Characters, They are Confused with.

Target Character	Misclassified with
ਸ	ਗ, ਮ
ਖ	ਥ
ਘ	ਮ
ਢ	ਵ
ਡ	ੜ, ਤ

ਢ	ਦ, ਚ
ਤ	ੜ
ਦ	ਢ
ਧ	ਥ, ਪ
ਫ	ਟ, ਰ
ਬ	ਛ, ਜ
ਭ	ਤ
ਯ	ਘ
ਰ	ਹ
ਲ	ਠ
ਵ	ਵ

Experiment – 7

Training of the system is done with 3500 training and tested with 700 images. Efficiency of the system is calculated using 6-fold cross-validation strategy. A comprehensive recognition rate of 74.5% is achieved. System is trained for 100 iterations for all 6 validations. Table 17-22 shows error rates on train and test dataset of all 6 validation tests. Figure 19-24 shows the train and test error rate for every iteration of all 6 validation tests.

Table 17. Train and Test Error Rate for Experiment – 7 (validation - 1)

Iterations	Eta	Train Error Rate %	Test Error Rate %
1	0.0005	19.6	36.5
2	0.0005	23.1	36.0
3	0.0002	23.0	30.0
4	0.0002	17.6	34.8
5	0.0002	16.8	34.2
6	0.0001	14.3	31.7
7	0.0001	17.0	31.3

8	0.0001	16.9	31.2
9	0.00005	12.8	30.6
10	0.00005	12.1	29.7
11	0.00005	11.1	28.1
12	0.00005	10.6	28.5
13	0.00001	10.3	27.5
14	0.00001	10.1	26.4
15	0.00001	10.6	28.5
16	0.00001	11.0	28.5
17	0.00001	10.9	29.0
18	0.00001	10.4	29.4
19	0.00001	10.7	29.5
20	0.00001	10.3	29.1

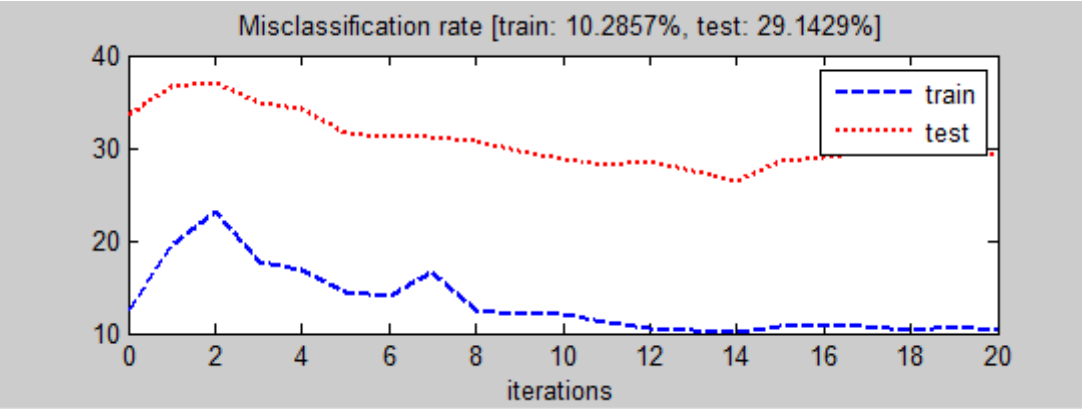


Figure 19. Misclassification Rate on Train and Test Dataset of Experiment – 7 (Validation - 1)

Table 18. Train and Test Error Rate for Experiment – 7 (validation - 2)

Iterations	Eta	Train Error Rate %	Test Error Rate %
1	0.0005	17.2	33.2
2	0.0005	17.3	34.1
3	0.0002	12.3	29.1
4	0.0002	11.6	28.4
5	0.0002	10.1	27.8
6	0.0001	8.6	27.7
7	0.0001	7.8	27.7
8	0.0001	8.1	27.0
9	0.00005	8.1	27.0
10	0.00005	7.8	27.3
11	0.00005	8.0	27.0
12	0.00005	11.8	29.3
13	0.00001	8.9	27.4
14	0.00001	8.6	27.2
15	0.00001	8.4	27.4
16	0.00001	8.6	27.3
17	0.00001	9.3	26.8
18	0.00001	9.2	27.1

19	0.00001	9.0	27.5
20	0.00001	8.9	27.3

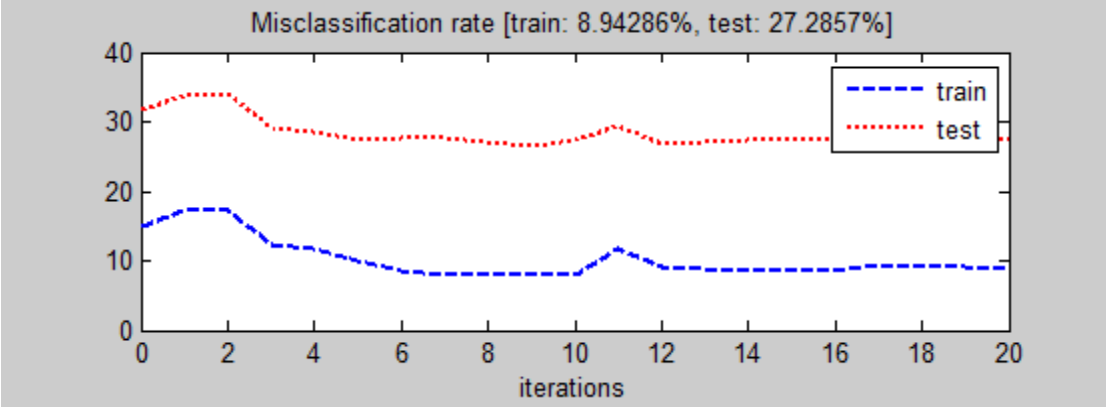


Figure 20. Misclassification Rate on Train and Test Dataset of Experiment – 7 (validation - 2)

Table 19. Train and Test Error Rate for Experiment – 7 (validation - 3)

Iterations	Eta	Train Error Rate %	Test Error Rate %
1	0.0005	25.3	31.3
2	0.0005	24.8	32.6
3	0.0002	17.4	25.6
4	0.0002	15.1	25.0
5	0.0002	15.8	25.3
6	0.0001	15.4	24.5
7	0.0001	13.9	24.8
8	0.0001	13.1	24.1
9	0.00005	12.2	24.0

10	0.00005	12.3	23.7
11	0.00005	13.7	24.3
12	0.00005	12.7	23.3
13	0.00001	12.4	23.4
14	0.00001	12.2	23.6
15	0.00001	12.4	23.2
16	0.00001	12.0	23.3
17	0.00001	12.3	23.4
18	0.00001	12.1	23.1
19	0.00001	11.6	23.6
20	0.00001	11.7	23.5

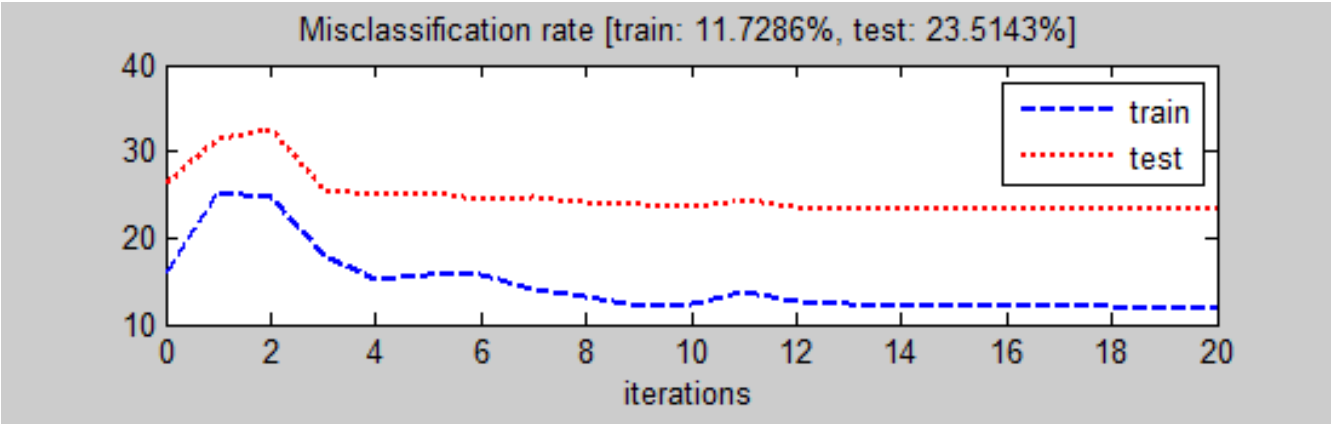


Figure 21. Misclassification Rate on Train and Test Dataset of Experiment – 7 (validation - 3)

‘

Table 20. Train and Test Error Rate for Experiment – 7 (validation - 4).

Iterations	Eta	Train Error Rate %	Test Error Rate %
1	0.0005	21.3	29.5
2	0.0005	23.4	31.2
3	0.0002	16.2	26.1
4	0.0002	14.5	25.4
5	0.0002	15.4	25.3
6	0.0001	12.4	23.3
7	0.0001	11.2	22.3
8	0.0001	10.5	23.2
9	0.00005	9.6	22.1
10	0.00005	9.3	21.7
11	0.00005	9.0	22.0
12	0.00005	9.3	22.1
13	0.00001	8.9	22.0
14	0.00001	8.7	22.6
15	0.00001	8.7	22.4
16	0.00001	8.6	22.5
17	0.00001	8.6	22.2
18	0.00001	8.7	22.5
19	0.00001	8.6	22.4

20	0.00001	8.7	22.4

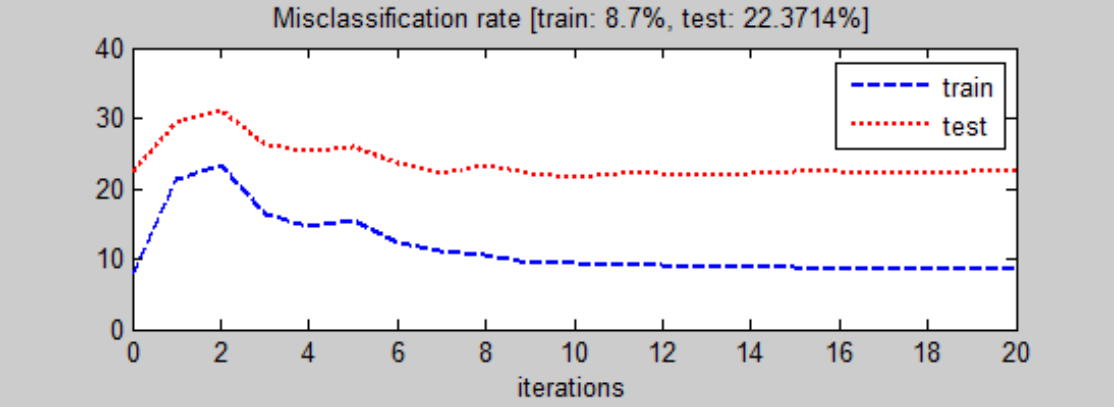


Figure 22. Misclassification Rate on Train and Test Dataset of Experiment – 7 (Validation - 4)

Table 21. Train and Test Error Rate for Experiment – 7 (validation - 5)

Iterations	Eta	Train Error Rate %	Test Error Rate %
1	0.0005	21.3	28.3
2	0.0005	18.5	28.4
3	0.0002	14.2	23.8
4	0.0002	14.2	23.7
5	0.0002	14.1	23.7
6	0.0001	11.7	23.2
7	0.0001	11.0	22.0
8	0.0001	11.4	21.6
9	0.00005	11.1	22.3
10	0.00005	10.3	21.7

11	0.00005	10.0	21.5
12	0.00005	10.4	22.1
13	0.00001	10.0	22.5
14	0.00001	10.1	21.7
15	0.00001	9.8	21.5
16	0.00001	9.6	21.4
17	0.00001	9.0	21.3
18	0.00001	9.5	20.9
19	0.00001	9.3	21.4
20	0.00001	9.4	20.7

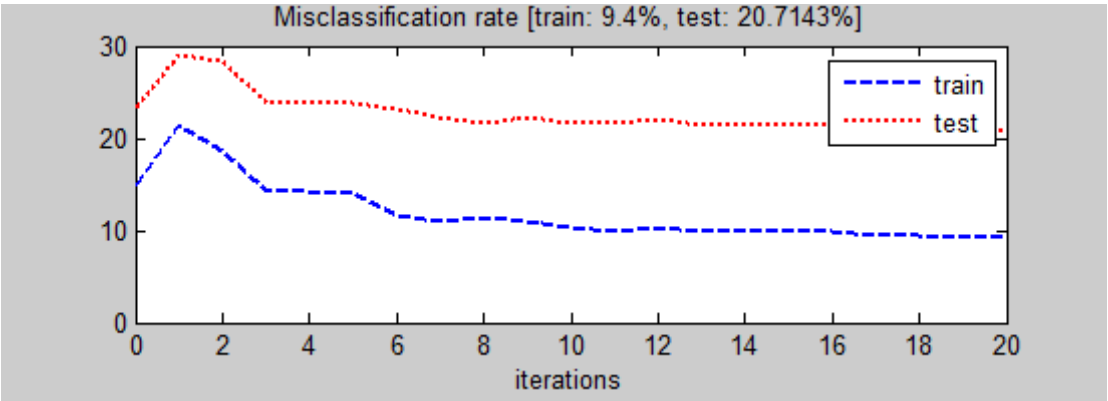


Figure 23. Misclassification Rate on Train and Test Dataset of Experiment – 7 (Validation - 5)

Table 22. Train and Test Error Rate for Experiment – 7 (validation - 6).

Iterations	Eta	Train Error Rate %	Test Error Rate %
1	0.0005	20.6	35.2

2	0.0005	17.6	33.7
3	0.0002	11.8	31.0
4	0.0002	13.7	30.6
5	0.0002	12.6	29.8
6	0.0001	10.5	29.3
7	0.0001	9.8	28.8
8	0.0001	12.5	31.5
9	0.00005	10.5	30.8
10	0.00005	10.7	30.2
11	0.00005	10.0	28.8
12	0.00005	9.5	30.1
13	0.00001	9.1	31.1
14	0.00001	8.9	30.1
15	0.00001	8.9	30.6
16	0.00001	8.7	29.4
17	0.00001	8.7	29.5
18	0.00001	8.5	29.7
19	0.00001	9.1	30.6
20	0.00001	8.7	30.1

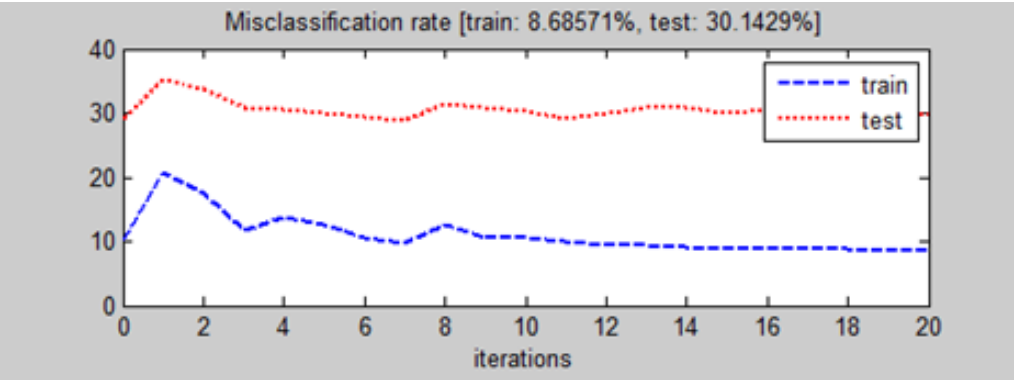


Figure 24. Misclassification Rate on Train and Test Dataset of Experiment – 7 (Validation - 6)

Table 23 Misclassified Characters

Target Character	Misclassified with
ਅ	ਘ , ਯ
ਖ	ਯ,ਥ
ਢ	ਵ
ਠ	ਣ, ੜ
ਥ	ਕ ,ਧ
ਭ	ਛ, ਣ
ਯ	ਘ , ਪ
ਰ	ੜ
ਸ	ਗ

Experiment – 8

Training of the system is done with 7000 training and tested with 3500 images. Efficiency of the system is calculated using 3-fold cross-validation strategy. A comprehensive recognition rate of 75.2% is achieved. System is trained for 150 iterations for all 3 validations. Table 24-26 shows error rates on train and test dataset of all 3 validation tests. Figure 25-27 shows the train and test error rate for every iteration of all 3 validation tests.

Table 24. Train and Test Error Rate for Experiment – 8 (Validation - 1).

Iterations	Eta	Train Error Rate %	Test Error Rate %

1	0.0005	18.5	33.6
2	0.0005	19.1	32.2
3	0.0002	11.2	27.8
4	0.0002	10.6	27.3
5	0.0002	8.7	27.5
6	0.0001	7.7	27.1
7	0.0001	7.7	27.1
8	0.0001	7.0	27.2
9	0.00005	6.0	26.0
10	0.00005	6.1	26.9
11	0.00005	5.8	26.5
12	0.00005	7.8	25.5
13	0.00001	7.6	25.5
14	0.00001	7.5	25.3
15	0.00001	7.3	25.3
16	0.00001	8.6	27.0
17	0.00001	8.3	27.0
18	0.00001	8.2	27.2
19	0.00001	8.0	26.3
20	0.00001	7.9	26.4

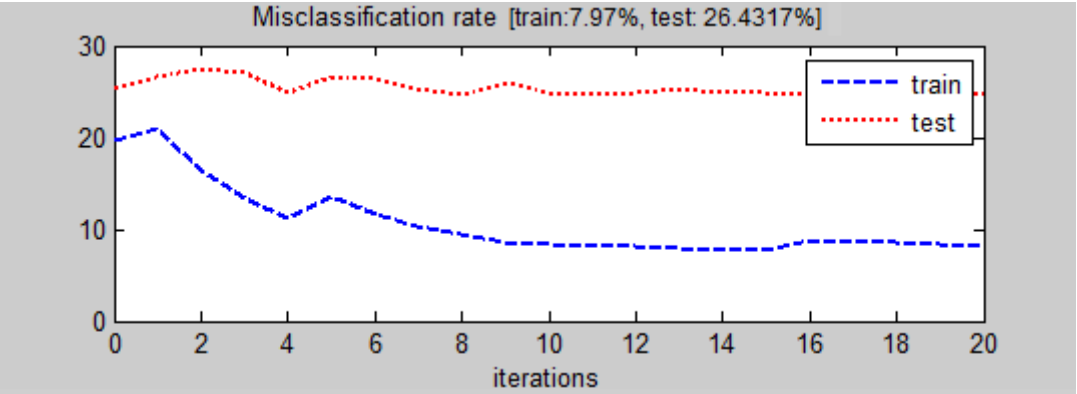


Figure 25. Misclassification Rate on Train and Test Dataset of Experiment – 8 (Validation - 1)

Table 25. Train and Test Error Rate for Experiment – 8 (validation - 2)

Iterations	Eta	Train Error Rate %	Test Error Rate %
1	0.0005	17.9	27.4
2	0.0005	18.9	29.1
3	0.0002	12.6	23.6
4	0.0002	17.3	27.1
5	0.0002	14.8	26.8
6	0.0001	11.7	24.2
7	0.0001	10.9	23.6
8	0.0001	10.4	23.7
9	0.00005	9.6	20.3
10	0.00005	9.4	22.8
11	0.00005	88.8	22.5
12	0.00005	8.9	22.7

13	0.00001	8.4	22.4
14	0.00001	8.3	22.2
15	0.00001	8.4	22.4
16	0.00001	8.4	22.1
17	0.00001	8.3	22.3
18	0.00001	8.2	22.3
19	0.00001	8.2	22.1
20	0.00001	8.1	22.5

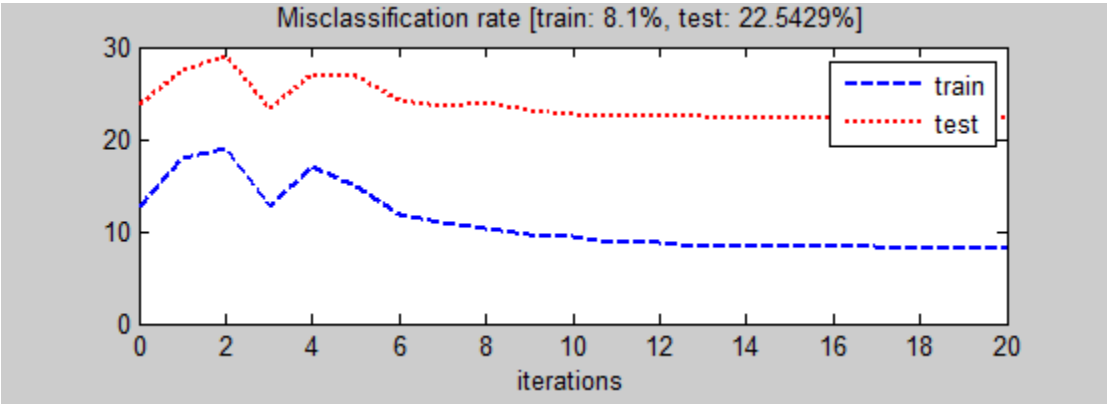


Figure 26. Misclassification Rate on Train and Test Dataset of Experiment – 8 (Validation - 2)

Table 26. Train and Test Error Rate for Experiment – 8 (validation - 3)

Iterations	Eta	Train Error Rate %	Test Error Rate %
1	0.0005	20.9	26.7
2	0.0005	16.3	27.4
3	0.0002	13.3	27.1

4	0.0002	11.1	25.0
5	0.0002	13.4	26.9
6	0.0001	11.5	26.4
7	0.0001	10.3	25.4
8	0.0001	9.6	24.7
9	0.00005	8.6	26.0
10	0.00005	8.3	25.0
11	0.00005	8.2	24.4
12	0.00005	8.1	24.1
13	0.00001	7.9	25.4
14	0.00001	7.7	25.0
15	0.00001	7.7	25.0
16	0.00001	8.6	24.5
17	0.00001	9.0	25.0
18	0.00001	8.6	25.1
19	0.00001	8.3	24.7
20	0.00001	8.2	24.7

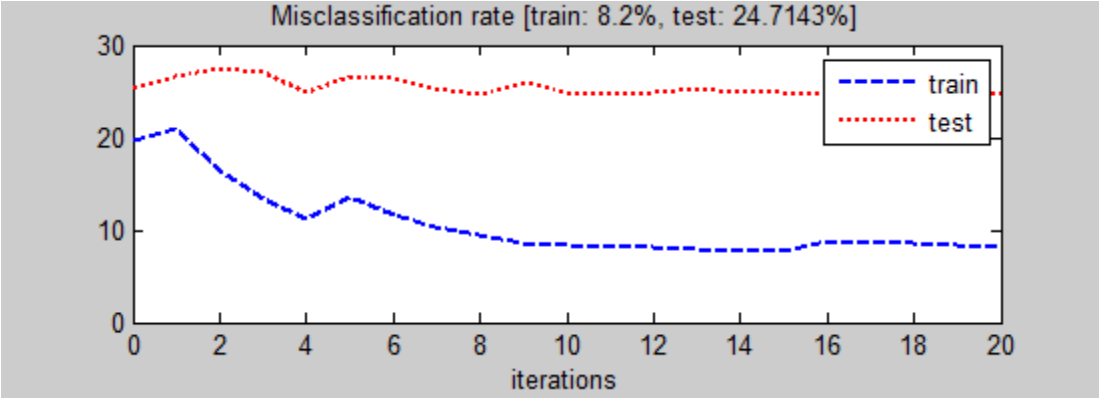


Figure 27. Misclassification Rate on Train and Test Dataset of Experiment – 8 (Validation -3)

Table 27 shows the misclassified characters with the characters they are confused with the most.

Table 27. Misclassified Characters

Target	Misclassified with
ਏ	ਟ
ਸ	ਮ, ਗ, ਥ
ਹ	ਰ
ਕ	ਰ, ਭ
ਖ	ਥ, ਮ
ਗ	ਸ, ਮ
ਝ	ਝ
ਜ	ਚ
ਵ	ਵ
ਟ	ਠ, ਏ, ਨ
ਠ	ਟ
ਡ	ਤ
ਦ	ਢ
ਤ	ਭ, ਡ
ਥ	ਖ

On analyzing Table 27, Gurmukhi character similar in shape is mostly misclassified.

CHAPTER 4

CONCLUSION AND FUTURE SCOPE

In my dissertation work, ConvNets based recognition system for Gurmukhi script is presented. This system has achieved reasonably good recognition results without using large amount of preprocessing and complicated feature extraction techniques. In all the experiments conducted, this has been concluded that, by increasing the size of training data the efficiency of the system increases. As future work, one can explore increasing the recognition accuracy. This can probably be achieved by following ways:

i. Growing the size of the training database.
ii. By using other backpropogation learning algorithm for the ConvNets.
iii. Fusion of two or three ConvNets trained could be used and then decision is made on basis of average scores calculated by these ConvNets.

As our system is trained for only 35 Gurmukhi characters, One can also explore the possibility of recognizing other Gurmukhi symbols including matras and numerals. Currently, there is no post-processor model is implemented for our system. A post processing model that combines Gurmukhi aksharas to words can also be explored in future. Presently, Character size in the training and testing images are of 20×20 centered in 32×32. He *et al*. (2005) experimented with 26×26 character size centered in 32×32 and was successful in increasing the recognition rate. One can try to do resizing of all training and testing images to 26×26 centered in 32×32 and then training the network.

REFERENCES

[1]Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, 1998.

[2]S. Ahranjany, F. Razzazi, and M. Ghassemian, "A very high accuracy handwritten character recognition system for Farsi/Arabic digits using Convolutional Neural Networks," in *2010 IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA*), 2010.

[3] X. Zhang and Y. LeCun, "Text Understanding from Scratch," *Arxiv.org*, 2015. [Online]. Available: http://arxiv.org/abs/1502.01710.

[4]Y. Zamani, Y. Souri, H. Rashidi, and S. Kasaei, "Persian handwritten digit recognition by random forest and convolutional neural networks," in *2015 9th Iranian Conference on Machine Vision and Image Processing (MVIP)*, 2015.

[5]D. Bouchain, "Character Recognition Using Convolutional Neural Networks," presented at *the Statistical Learning Theory seminar, University of Ulm, Germany*, 2006.

[6]M. Kumar, R. Sharma, and M. Jindal, "Offline handwritten Gurmukhi character recognition," in *Proceeding of the workshop on Document Analysis and Recognition - DAR '12*, pp. 94-99, 2012.

[7] C.L. He, P. Zhang, J. Dong, C.Y. Suen, and T.D. Bui,"The Role of Size Normalization on the Recognition Rate of Handwritten Numerals," in *The 1st IAPR TC3 NNLPAR workshop*, pp. 8-12. 2005.

[8] A. Sharma. "Online handwritten Gurmukhi character recognition," *Ph.D. thesis*, School of Mathematics and Computer Applications, Thapar University, Patiala, 2009.

[9] M. Unser, A. Aldroubi, and M. Eden, "B-Spline signal processing: part II - efficient design and applications," in *IEEE Transactions on Signal Processing*, vol. 41, no. 2, pp. 834-848, 1993.

[10] S. R. Veltman and R. Prasad, "Hidden markov models applied to online handwritten isolated character recognition", *IEEE Transactions on Image Processing*, vol. 3, no. 3, pp. 314-318, 1994.

[11] T. Wakahara and K. Odaka, "Online cursive kanji character recognition using stroke-based affine transformation," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 12, pp. 1381-1385, 1997.

[12] J. Zhou, Q. Gan, K. Krzyzak, and C. Y. Suen, "Recognition of handwritten numerals by quantum neural network with fuzzy features," in *International Journal of Document Analysis and Recognition*, vol. 2, pp. 30-36, 1999.

[13] A. Sharma, R. Sharma and R. K. Sharma, "Rearrangement of Recognized Strokes in Online Handwritten Gurmukhi Words Recognition," in 10th *International Conference on Document Analysis and Recognition*, pp. 1241-1245, 2009.

[14] H. Beigi, "Pre-processing and dynamics of online handwriting data, feature extraction and recognition," in *Proceedings of fifth IWFHR*, pp. 255-258, 1996.

[15] A. Sharma, "Online handwritten Gurmukhi character recognition," *Ph.D. thesis*, School of Mathematics and Computer Applications, Thapar University, Patiala, 2009.

[16] A. Sharma, R. Sharma, and R. K. Sharma, "Online Handwritten Gurmukhi Character Recognition Using Elastic Matching," in *Congress on Image and Signal Processing*, pp. 391-396,2008.

[17] S. Jaeger, S. Manke, J. Reichert, and A. Waibel, "Online handwriting recognition: the Npen++ Recognizer," in *International Journal of Document Analysis and Recognition*, vol. 3, no. 3, pp. 169-180, 2001.

[18] E. Kavallieratou, N. Fakatakis, and G. Kolkkinakis, "An unconstrained handwriting recognition system," in *International Journal of Document Analysis and Recognition*, vol. 4, no. 4, pp. 226-242, 2002.

[19] S. Singh, A. Aggarwal, and R. Dhir, "Use of Gabor Filters for Recognition of Handwritten Gurmukhi Character," in *International Journal of Advance Research in Computer Science and Software Engineering*, Vol. 2, Issue 5, 2012.

[26] G.S. Lehal and C. Singh, "A Gurmukhi script recognition system," in *Proceedings of the 15th International Conference on Pattern Recognition*, Vol. 2, pp. 557–560, 2000.

[25] G.S. Lehal, C. Singh, and R. Lehal, "Shape based post processor for Gurmukhi OCR," in *Proceedings of the Sixth International Conference on Document Analysis and Recognition*, Seattle, pp. 1105–1109,2001.

[26] U. Pal, N. Sharma, T. Wakabayashi, and F. Kimura, "Off-Line Handwritten Character Recognition Of Devnagari Script," in *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), Parana*, pp. 496 - 500, 2016.

[27] Nibaran Das *et al*., "Recognition of Handwritten Bangla Basic Characters and Digits using Convex Hull based Feature Set," in *International Conference on Artificial Intelligence and Pattern Recognition, Florida*, pp. 380-386, 2014.

[28] Basem Alijla and Kathrein Kwaik, " Online Isolated Arabic Handwritten Character Recognition Using Neural Network," in *T*he International Arab Journal of Information Technology, Vol.9, No. 4, 2012.

[29] Anuj Sharma, Rajesh Kumar, and R. K. Sharma, "HMM-Based Online Handwritten Gurmukhi Character Recognition" in Machine Graphics & Vision International Journal, pp.

439-449, 2010.

[30] Shilpa Jumanal and Ganga Holi, "On-Line Handwritten English Character Recognition Using Genetic Algorithm," in International Journal of Computer Trends and Technology, Vol. 4, 2013.

[31] Jignesh Rathod, Sonia Dewangan, Amarish Dubey, and Jitendra Kumar Sao, "A Novel Neural Net based Off-line English Character Recognition System," in International Journal of Research, Vol. 2, pp. 576-583 2015.

[32] Karpathy. CS231n Convolutional Neural Network for Visual Recognition [Online]. Available:http://cs231n.github.io/convolutional-networks/

APPENDIX-A
PAPERS COMMUNICATED

1. Maneet Manoor, R.K. Sharma, Isolated Handwritten Gurmukhi Akshara Recognition using Convolutional Neural Networks. In International Conference on Inventive Computation Technologies (ICICT), 2016.

   (Communicated)

APPENDIX –B

VIDEO PRESENTATION LINK

APPENDIX-C

PLAGIARISM REPORT

ORIGINALITY REPORT

4%	2%	4%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	dspace.thapar.edu:8080 Internet Source	<1%
2	www.sikhiwiki.org Internet Source	<1%
3	Kumar, Munish, R. K. Sharma, and M. K. Jindal. "Offline handwritten Gurmukhi character recognition : study of different feature-classifier combinations", Proceeding of the workshop on Document Analysis and Recognition - DAR 12 DAR 12, 2012. Publication	<1%
4	Ramappa, Mamatha Hosalli and Krishnamurthy, Srikantamurthy. "A Comparative Study of Different Feature Extraction and Classification Methods for Recognition of Handwritten Kannada Numerals", International Journal of Database Theory & Application, 2013. Publication	<1%
5	Advances in Intelligent Systems and Computing, 2015. Publication	<1%

6	Communications in Computer and Information Science, 2011. Publication	<1%
7	Pal, U.. "Indian script character recognition: a survey", Pattern Recognition, 200409 Publication	<1%
8	Submitted to University of Surrey Student Paper	<1%
9	Rajesh Kumar Bawa. "A Preprocessing Technique for Recognition of Online Handwritten Gurmukhi Numerals", Communications in Computer and Information Science, 2011 Publication	<1%
10	Submitted to VIT University Student Paper	<1%
11	Submitted to Universiti Tunku Abdul Rahman Student Paper	<1%
12	Zhihong Zhao. "Chinese License Plate Recognition Using a Convolutional Neural Network", 2008 IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application, 12/2008 Publication	<1%
13	Wei, Xue, Son Lam Phung, and Abdesselam Bouzerdoum. "Visual descriptors for scene categorization: experimental evaluation", Artificial Intelligence Review, 2015.	<1%

	Publication	
14	aljahdali.net Internet Source	<1%
15	Bhowmik, Showmik, Md. Galib Roushan, Ram Sarkar, Mita Nasipuri, Sanjib Polley, and Samir Malakar. "Handwritten Bangla Word Recognition Using HOG Descriptor", 2014 Fourth International Conference of Emerging Applications of Information Technology, 2014. Publication	<1%
16	Submitted to University of Florida Student Paper	<1%
17	Submitted to Guru Nanak Dev Engineering College Student Paper	<1%
18	www.academypublisher.com Internet Source	<1%
19	Perez-Carrasco, Jose-Antonio, Carmen Serrano, Begona Acha, Teresa Serrano-Gotarredona, and Bernabe Linares-Barranco. "Spike-Based Convolutional Network for Real-Time Processing", 2010 20th International Conference on Pattern Recognition, 2010. Publication	<1%
20	Ahranjany, Sajjad S., Farbod Razzazi, and Mohammad H. Ghassemian. "A very high	<1%

accuracy handwritten character recognition system for Farsi/Arabic digits using Convolutional Neural Networks", 2010 IEEE Fifth International Conference on Bio-Inspired Computing Theories and Applications (BIC-TA), 2010.
Publication

21	Advances in Intelligent Systems and Computing, 2013. Publication	<1%
22	Yann LeCun. "Object Recognition with Gradient-Based Learning", Lecture Notes in Computer Science, 1999 Publication	<1%
23	Lecture Notes in Computer Science, 2015. Publication	<1%
24	Choon Hui Teo. "Invariant Object Recognition Using Circular Pairwise Convolutional Networks", Lecture Notes in Computer Science, 2006 Publication	<1%

EXCLUDE QUOTES	ON	EXCLUDE MATCHES	< 8 WORDS
EXCLUDE BIBLIOGRAPHY	ON		