Offline English Character Recognition

A Project Report

submitted in partial fulfillment of the requirements
for the award of the degree of

Bachelor of Technology

in

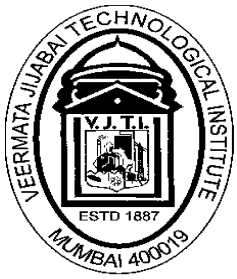
INFORMATION TECHNOLOGY

by

Aniket Ghag	091080028
Thyagarajan Radhakrishnan	091080011
Jivitesh Poojary	091080065
Sanket Malpure	091080042

Under the guidance of

Prof. V. K. Sambhe



DEPARTMENT OF COMPUTER ENGINEERING AND INFORMATION TECHNOLOGY

VEERMATA JIJABAI TECHNOLOGICAL INSTITUTE

H.R. Mahajani Marg, Matunga(E), Mumbai-400019

INDIA

May, 2013

Acknowledgements

First and foremost, we take this opportunity to thank our project guide, **Prof. Vijay K. Sambhe**, for his guidance and support. We will forever remain grateful for the constant support and guidance extended by our guide, in making this project. Through our many discussions, he helped us form and solidify ideas. The invaluable discussions we had with him and the constant motivation has all led to the development of this project.

We also wish to express our sincere thanks to the Head of department, **Dr. B. B. Meshram** and the Information Technology department for providing all the facilities to complete this project. We also wish to thank **Prof. Sandeep S. Udmale** and the departmental staff members for their support.

Finally, we would like to thank all our friends for the continual encouragement and the positive support.

Mr. Aniket Ghag (IT091080028)
Mr. Thyagarajan Radhakrishnan (IT091080011)
Mr. Jivitesh Poojary (IT091080065)
Mr. Sanket Malpure (IT091080042)

Abstract

Off-line English Character Recognition is a challenging and interesting research area in the field of pattern recognition and image processing. It deals with extracting and recognizing handwritten characters or machine text from scanned images which can then be used in a wide variety of applications. Here, we first analyze some of the recent implementations in this emerging field of research. In this paper, we have proposed two new techniques for recognizing offline English characters using Neural Networks: The first technique uses identity matrices, while the second technique uses ASCII codes. Both these techniques are implemented to train the Neural Network. We have used various neural networks like the feed-forward back-propagation network, Elman back-propagation network and the fitting network. The feature extraction technique that we have proposed is by dividing the character image into 10x10 square matrices and taking their individual sums of elements to finally create a vector that represents the corresponding character. We then analyze and compare the results of applying the two techniques using various neural networks on an input image for further study.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Off-line English character recognition involves scanning a form or document written some time in the past. This means the individual characters contained in the scanned image will need to be extracted. It has application in the fields of document processing, postal services, text to speech converters, security applications, etc. Pattern recognition methodologies for only Optical Character Recognition (OCR) have been successfully applied to some extent. However, application of these methods for English character recognition in both handwritten and machine text form is discussable because of the infinite variations in shapes of characters resulting from writing habits, style, education, region of origin, social environment, mood, health and other conditions of the writer. Moreover, factors such as the writing instrument, writing surface, scanning methods, etc. also affect the efficiency of standard character recognition algorithms.

Character recognition is one of the most fascinating research areas in field of image processing and pattern recognition. It enables automation of processes and helps improve the man-machine interface in a wide variety of applications. Several research works are trying to focus on new techniques that would reduce processing time in addition to providing higher accuracy of recognition. English character recognition is classified into two types:

(1) **Off-line recognition**: the data (image) are acquired by a scanner after the writing process is over. It includes both machine text as well as handwritten text.

(2) **On-line recognition**: the data are captured during the writing process by a special pen on an electronic surface.

1.1 Conventional English Character Recognition Method

The conventional method for English character recognition consists of following stages:

1. Image acquisition
2. Preprocessing the image
3. Segmentation (for character extraction)
4. Character recognition

The first important step in any handwritten character recognition system is "Preprocessing" followed by "Segmentation and feature extraction" followed by "Character recognition".

1. In Image acquisition, image is captured using a scanner or a camera that consists of handwritten text.
2. Pre-processing includes the steps that are required to shape the input image into a form suitable for segmentation.
3. Segmentation refers to a process of partitioning an image into groups of pixels which are homogeneous with respect to some criterion. The result of segmentation is the splitting up of the image into meaningful regions. Two types of segmentation are performed (in order):
    - Line segmentation: the image is segmented row-wise
    - Word segmentation: each rows is segmented column-wise
    - Finally, characters can be extracted using suitable feature extraction techniques. Each character is then resized before going into the character recognition stage
4. Character recognition is done by matching the characters extracted with the letters in the database using suitable algorithms.

1.2 Motivation

Our motivation regarding working in this research area is due to fact that the recognition of off-line character is more complex than the on-line case due to the presence of noise in image acquisition process and the loss of temporal information such as the writing sequence of characters. This data is usually very helpful during the recognition process. In addition, several applications including bank processing, document reading and postal address recognition require off-line English character recognition system. It is the most cost-effective and speedy method. Technology has made it possible to free acres of storage space by transforming all paper documents to be stored on a single machine. Hence, the off-line English character recognition continues to be an active area for research towards exploring newer techniques that would improve recognition accuracy.

1.3 Problem Definition

Off-line English Character Recognition system aims to recognize English text from a captured/scanned image. It enables a person to write something on a piece of paper, convert it into an image and then into text. It includes identifying various styles of handwritten text as well as various fonts of machine text. It contributes immensely to the advancement of an automation process and can improve the interface between man and machine in numerous applications. It would help to free large storage space which would otherwise be required for storing paper documents and files in offices and at homes. Searching for particular files or documents also becomes very easy when they are converted and stored in electronic form. Here, we focus only on offline English character recognition.

1.4 Overview

We started our discussion with an introduction to Offline English character recognition, the conventional method adopted, the motivation behind choosing this topic and the problem definition. In Chapter 2, we discuss the work done previously in the field of offline character recognition. In Chapter 3, we introduce the basis of our algorithm and then describe our proposed algorithm and techniques adopted for training the neural network and for character recognition. The results thus obtained are further discussed in Chapter 4. In Chapter 5 we conclude the topic and also mention the future scope of our project.

Chapter 2

Literature Survey

The very first effort in the direction of CR was made by Tyuring who attempted to develop an aid for the visually handicapped. The first character recognizer appeared in around 1940s. The early works were based either upon machine-printed text or upon a small set of well-separated handwritten text or symbols. Machine-printed CR generally used template matching and for handwritten text, low-level image processing techniques were used on the binary image to extract feature vectors, which were then fed to statistical classifiers. A good survey of the CR techniques used until 1980s can be found in Tappert et al. [1].

The period from 1980 - 1990 witnessed a growth in CR development due to rapid growth in information technology. Structural approaches were initiated in many systems in addition to the statistical methods. The syntactic and structural approaches require efficient extraction of primitives which include different types of line segments and curves. But there existed an upper limit in the recognition rate, because the CR research was focused basically on the shape recognition techniques without using any semantic information. Historical review of CR research and development during 1980-1990 can be found in Mori et al. [2] for off-line cases.

In Srihari et al. [4], we have seen that most of the off-line successes have come with handwriting recognition, especially for isolated characters and words in domains, such as postal addresses, bank checks. However, there was difficulty in analysis of documents with complex layouts, recognition of degraded printed text, and the recognition of running handwriting. Off-line processing faced major challenges in word and line separation, segmentation of words into characters and recognition of words when lexicons are of varying sizes. After 1990, image processing techniques and pattern recognition were combined

using artificial intelligence. Along with powerful computers and more accurate electronic equipment such as scanners, cameras and electronic tablets, there came in efficient, modern use of methodologies such as neural networks (NNs), hidden Markov models (HMMs), fuzzy set reasoning, and natural language processing.

The study of literatures in Dash et al. [19] described below conveys that multiple algorithms and techniques have been used to accomplish the task of English character recognition using Neural Network (NN) as a backend of character classification due to its faster and reliable computation: Basu et al. [11] used Multilayer Perceptron (MLP) for Bangla alphabet recognition. Accuracy achieved in this work: 86.46% and 75.05% on training and testing samples respectively. Neil et al. [12] proposed an optical correlator-neural network architecture for pattern recognition in which English alphabets were used as patterns for training and testing purposes. Pal et al. [13] proposed NN based English character recognition system. In this work, MLP with one hidden layer was used. About 500 testing were carried out to test the performance of the design. The best case accuracy obtained in this work was 94%. Perwej et al. [14] worked on English alphabet recognition using NN. In this work, binary pixels of the alphabets were used train the NN. The accuracy achieved was found to be 82.5%.

Pal et al. [15] proposed a modified quadratic classifier approach for handwritten numerals of six popular Indian scripts with high level of recognition accuracy. Dinesh et al. [16] used horizontal and vertical strokes and end points as feature for handwritten numerals. This method reported an accuracy rate of 90.5% in best case. However, this method used a thinning method resulting in loss of features. Yanhua et al. [17] recommended a novel Chinese character recognition algorithm which was based on minimum distance classifier. The algorithm attempted to work with two classes of feature extraction-structure and statistics. The statistic feature decided the primary class and the structure feature used to identify Chinese characters. Huiqin et al. [18] proposed a distribution based algorithm based on image segmentation and distribution of pixels. Deflection Correction method was adopted for flexibility as well as reduction of matching error. The method gave excellent result and was robust.

Som et al. [20] proposed an HCR system using Multiresolution Technique and Euclidean Distance Metric, whose results showed an average recognition accuracy of 90%. It compared its results with those of some other authors: The method used by Chen et al. [21] gets recognition rate of 92.20% for handwritten numerals. The method of Romero et al. [22] gets 90.20% recognition accuracy over the test data for handwritten numerals. The

method of Mowlaei et al. [23] gets recognition rates of 92.33% and 91.81% for 8 classes of handwritten Farsi characters and numerals respectively.

Pradeep et al. [6] proposed an English alphabet characters recognition system using a new type of feature extraction, namely, diagonal feature extraction. Two approaches using 54 features and 69 features were chosen to build the Neural Network. The neural network recognition system was trained using the horizontal and vertical feature extraction methods for comparison. Six different recognition networks were built. Results revealed that 69 features gave better recognition accuracy than 54 features for all the types of feature extraction: 97.8% for 54 features and 98.5% for 69 features. Chin et al. [24] used a Bayesian framework for deformable pattern recognition and achieved an accuracy of 94.7 percent on 11,791 test examples by using only 23 prototypes.

Morita et al. [25] presents a summary about the recent advances in terms of character, numeral, word and sentence recognition. The table below presents some recent contributions to the field of character recognition evaluated on NIST database:

Method	**Training Set**	**Validation Set**	**Test Set**	**Recognition Rate (%)**
(Oh, 1998)				
Uppercase (26 classes)	26000	-	11941	90.0
(Dong, 2001)				
Lowercase (26 classes)	23937	-	10688	92.3
(Koerich, 2002)				
Uppercase (26 classes)	37440	12092	11941	92.3
Lowercase (26 classes)	37440	11578	12000	84.6
Upper/Lower (52 classes)	74880	23670	23941	85.5
(Britto, 2004)				
Uppercase (26 classes)	37440	12092	11941	90.0
Lowercase (26 classes)	37440	11578	12000	84.0
Upper/Lower (52 classes)	74880	23670	23941	87.0

Table 2.1: Some recent contributions in character recognition

In recent years, many researchers have combined such techniques in order to improve the recognition results, but no system to date has achieved the goal of system acceptability. There has always been a scope of improvement.

Chapter 3

Implementation

Our proposed method includes the implementation of Neural networks based algorithms.

3.1 Neural Networks (NN)

A NN is defined as a computing architecture that consists of a massively parallel interconnection of adaptive "neural" processors. Because of its parallel nature, it can perform computations at a higher rate compared to the classical techniques. Because of its adaptive nature, it can adapt to changes in the data and learn the characteristics of input signal. An NN contains many nodes. The output from one node is fed to another one in the network and the final decision depends on the complex interaction of all nodes. Working of a NN is shown in Fig. 3.1.

Neural network recognizers learn from an initial image training set. The trained network then makes the character identifications. Each neural network uniquely learns the properties that differentiate training images. It then looks for similar properties in the target image to be identified. Neural networks are quick to set up.

First, we construct a suitable neural network and train it properly. Then, we extract characters one by one and map the target output for training purpose. After automatic processing of the image, the training dataset is used to train "classification engine" for recognition purpose.

Different stages are as given below:

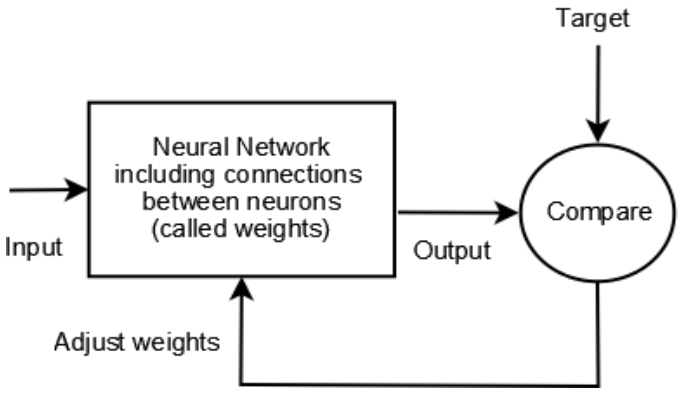


Figure 3.1: Working of a NN

1. **Input:** Samples are read to the system through a scanner.

2. **Preprocessing:** Preprocessing converts the image into a form suitable for subsequent processing and feature extraction.

3. **Segmentation:** The most basic step in CR is to segment the input image into individual characters. This step separates out sentences from text and subsequently words and letters from sentences.

4. **Feature extraction:** Extraction of features of a character forms a vital part of the recognition process. Feature extraction captures the vital details of a character.

5. **Classification:** During classification, a character is placed in the appropriate class to which it belongs.

6. **Training and Recognition:** Each neural network uniquely learns the properties that differentiate training images. It then looks for similar properties in the target image to be identified. CR systems use the methodologies of pattern recognition like NN which assigns an unknown sample into a predefined class.

Attributes are important and can have a crucial impact on end results. Some of the most important attributes include the width (in pixels) of the image, the height (in pixels) of the image and the total number of "on" pixels in the character image. Neural networks can be used, if we have a suitable dataset for training and learning purposes. Datasets are one of the most important things when constructing new neural network. Without

proper dataset, training will be useless. So, to get a proper dataset, first we have to scan the image. After the image is scanned, we define processing algorithm, which will extract important attributes from the image and map them into a dataset. Extracted attributes will have numerical values and will be usually stored in arrays. With these values, neural network can be trained and we can get a good end result.

Thus, the stages involved in Neural Network based Algorithms can be represented as shown below:

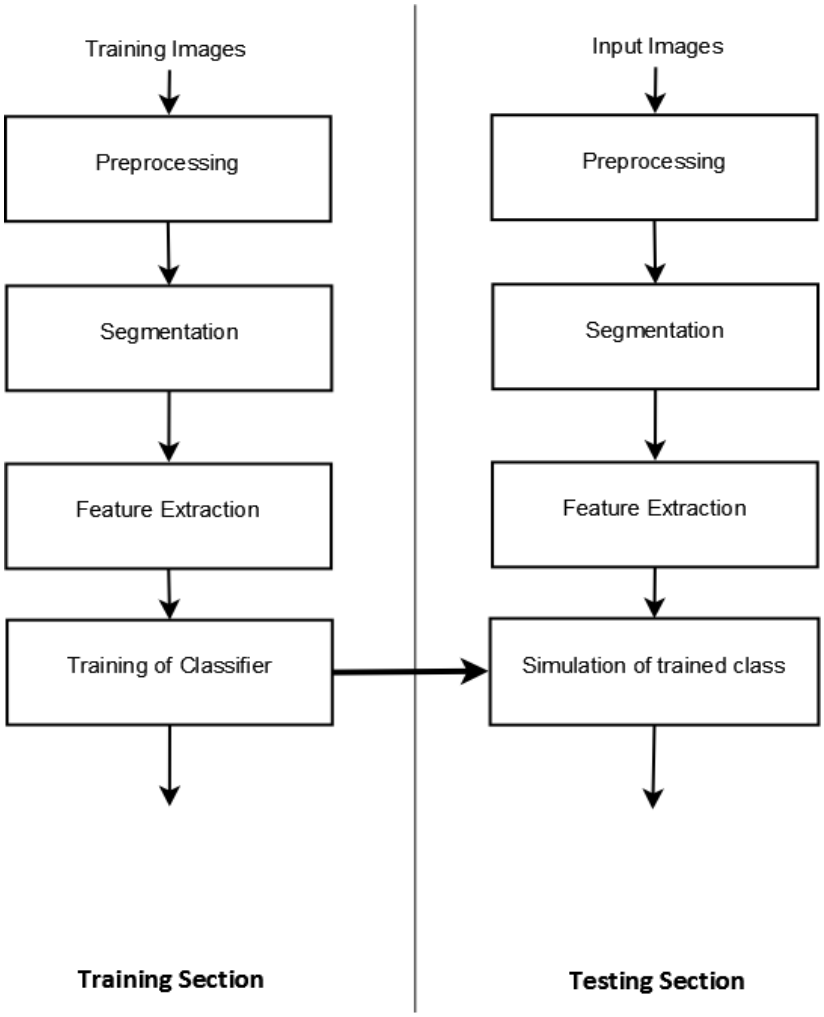


Figure 3.2: Stages in NN Algorithm

3.2 Proposed Algorithm For Training The Neural Network

Our proposed algorithm for training the NN is shown below:

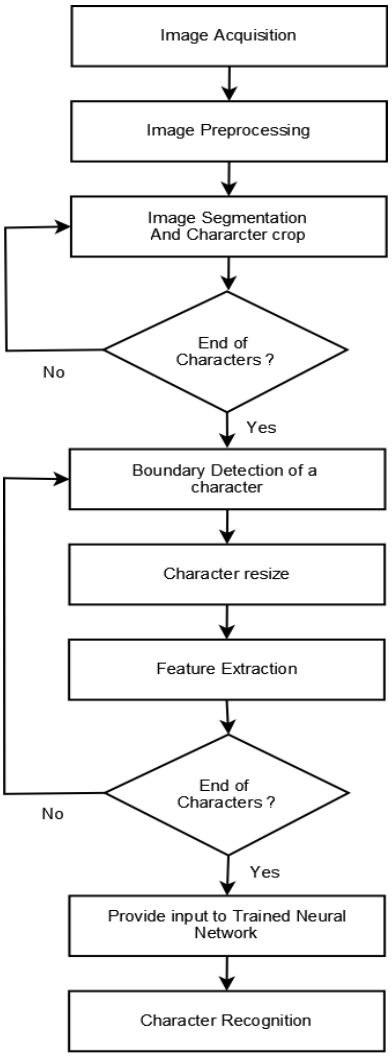


Figure 3.3: NN Training Algorithm

Next, we discuss the steps involved in our proposed algorithm for training the NN.

(1) Image Acquisition:

In this step, we acquire input images for training the neural network. The input image can contain handwritten or machine text in RGB or grayscale format as shown below:

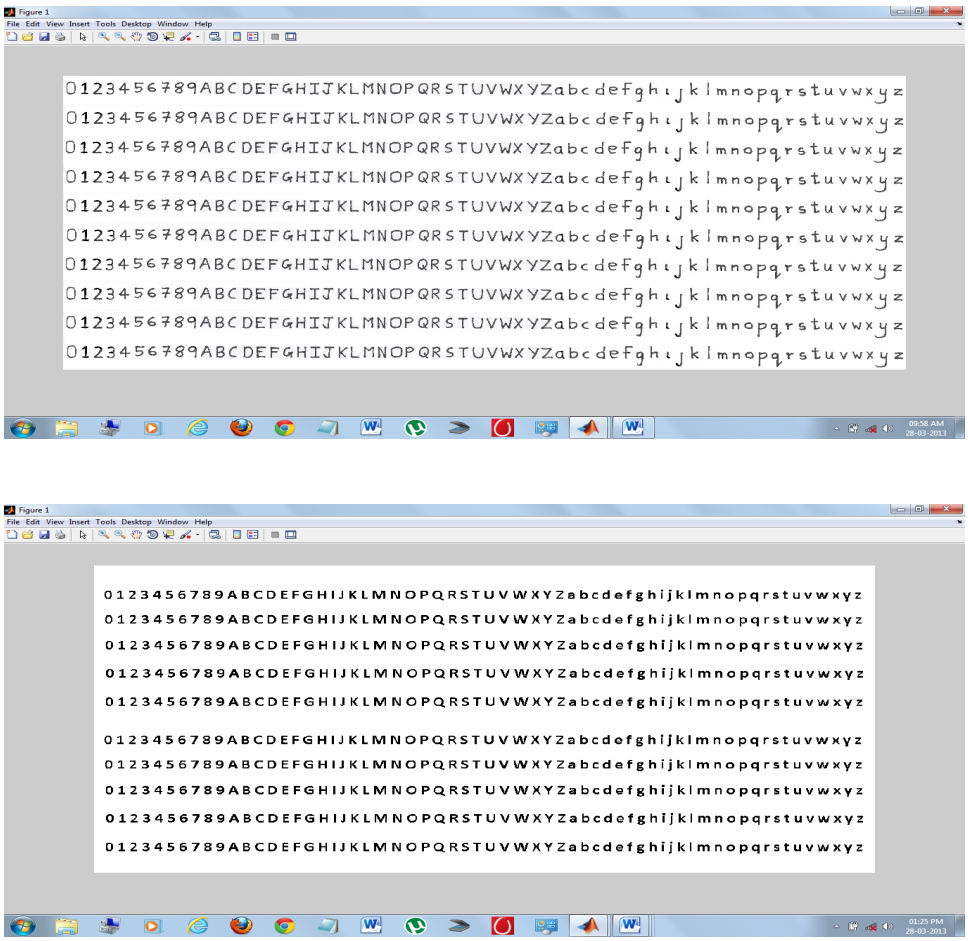


Figure 3.4: Training Images: Handwritten & Machine Text

(2) Image Preprocessing:

Image preprocessing contains various steps as shown below in the flowchart:

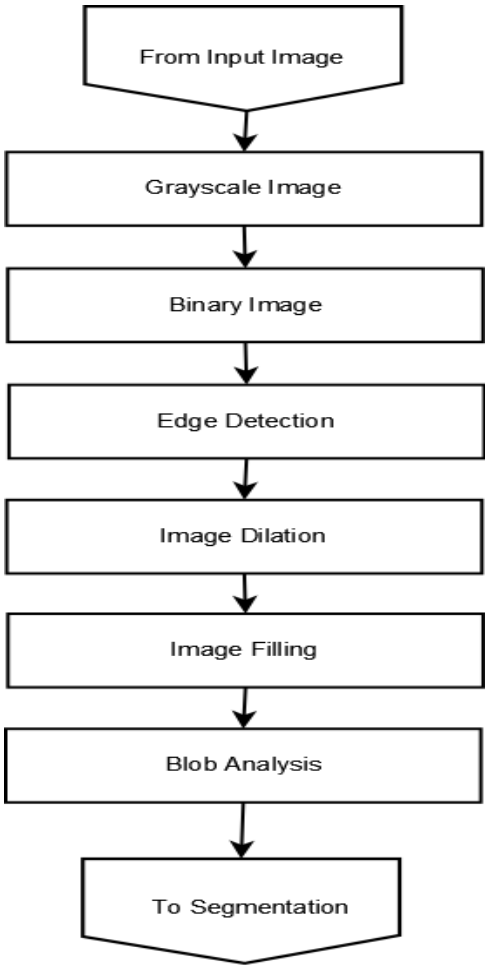


Figure 3.5: Image Preprocessing Steps

As an example, we consider the input image containing handwritten characters for performing preprocessing, as shown below:

Step 1: We convert the input image into grayscale format so that we can later convert it into binary image using suitable threshold.

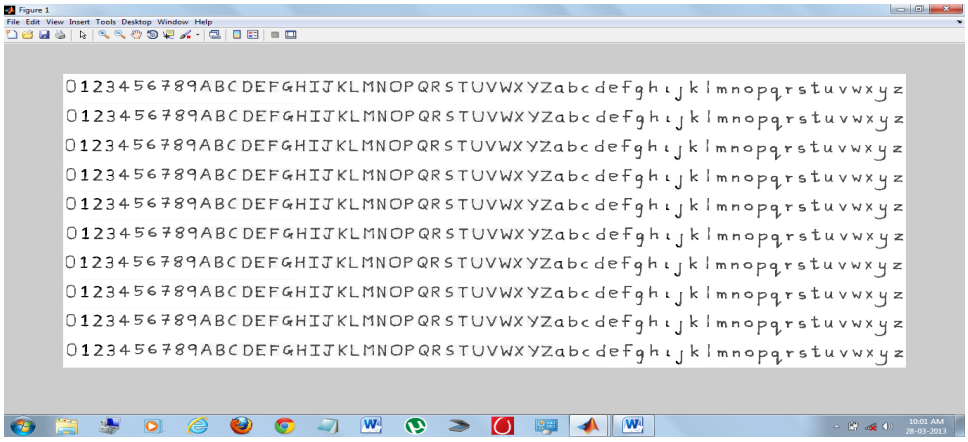


Figure 3.6: Grayscale Image

Step 2: We convert grayscale image into binary image which contains only 1s (for white) and 0s (for black) using suitable threshold

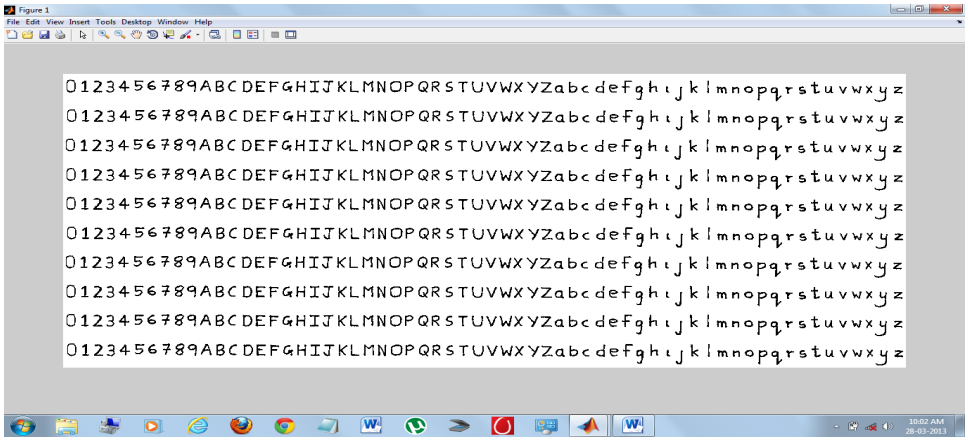


Figure 3.7: Binary Image

Step 3: We find the edges of all characters in the binary image. This is accomplished by converting it into another binary image of the same size, with 1's representing edges and 0's elsewhere.

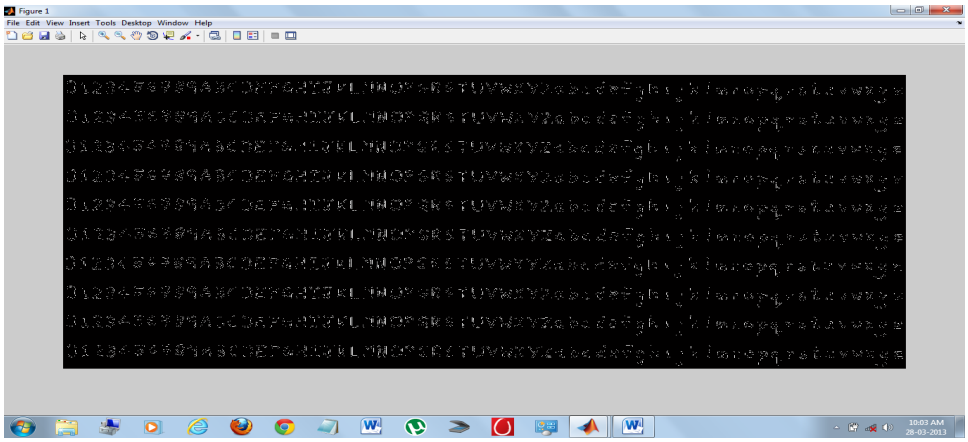


Figure 3.8: Edge Detection

Step 4: In the above image, edges are not properly connected. Hence we use suitable structuring element (square) to join the edges of each character. This process is known as Image Dilation.

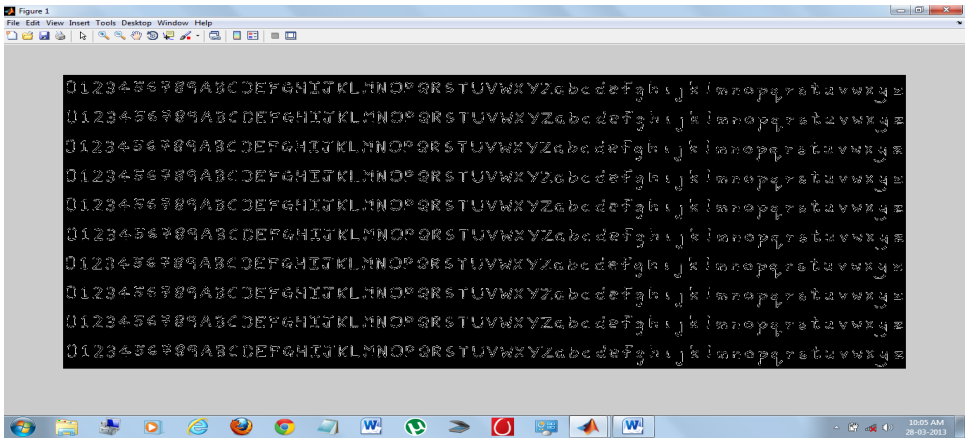


Figure 3.9: Image Dilation

Step 5: We fill holes present (if any) in the above binary image. A hole is a set of background pixels that cannot be reached by filling in the background from the edge of the image. This process is known as Image Filling.

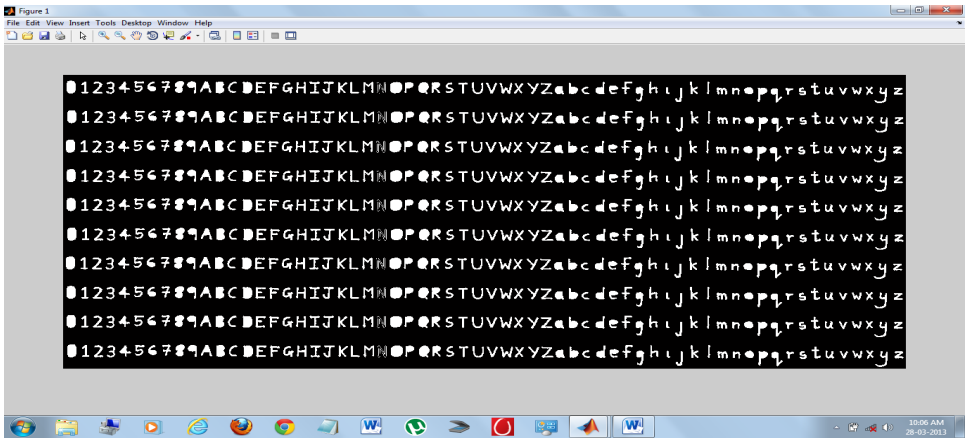


Figure 3.10: Image Filling

Step 6: We identify connected components i.e. each character from the above binary image, label them and store it in an array. The size of the resulting array provides us the number of characters. This is known as Blobs Analysis.

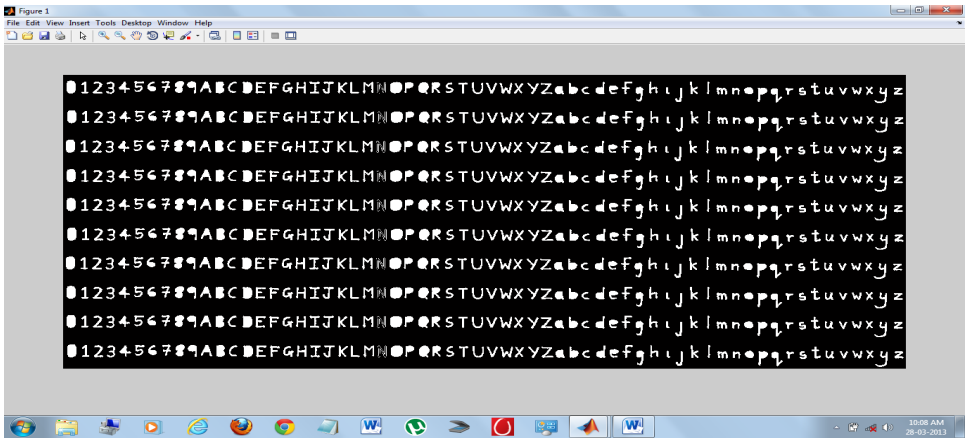


Figure 3.11: Blob Analysis

(3) Image Segmentation and Character crop:

The flowchart for Image Segmentation and Character crop is shown below:

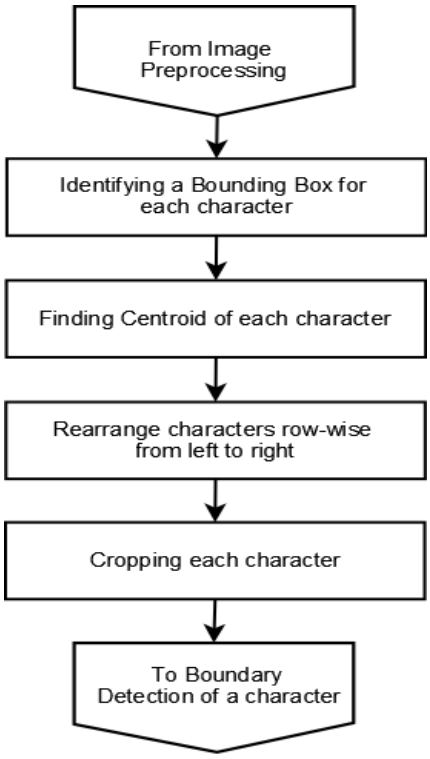


Figure 3.12: Image Segmentation and Character crop

In this step, we use the region that encloses each character obtained from the preprocessing stage to measure the set of properties for each connected component (character). Some of the properties include 'Area', 'BoundingBox', 'Centroid', 'FilledArea', 'FilledImage', 'Image' and 'PixelList'. The properties that we consider for our implementation are 'BoundingBox' and 'Centroid'.

The array obtained in Step 6 of the Image Processing stage contains characters in an order such that the entire image is scanned from left to right and the characters are labeled and stored in the array in the order in which they are encountered. This means that in most cases, the initial set of characters encountered are the characters present in every row and first column, then second column and so on. The labeling usually is also random

and not row-wise depending on the character present at that position. So, this random labeling affects the order of cropping and hence the output of recognition also becomes random and not in the order of characters present in the image.

Hence, we rearrange the characters in a row-wise manner from left to right and store them in that order in an array. This rearrangement helps us to crop and later recognize each character in an orderly fashion i.e. row-wise from left to right and it becomes easy to map the characters in the input image with those in the output and also helps in finding the recognition accuracy rate.

(4) Boundary Detection of a Character:

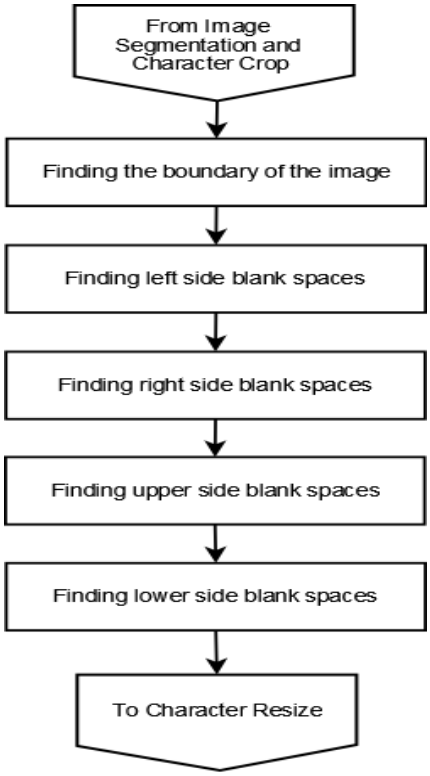


Figure 3.13: Boundary Detection of a Character

The aim of this step is to retain the character image by detecting the boundary of the character in such a way that the boundary of the image circumscribes the character leaving the extraneous regions along the boundary (if any).

In this stage, we use the image having some definite boundary obtained from the image segmentation and character crop stage. Using this boundary, we find the left side blank spaces, right side blank spaces, upper side blank spaces and the lower side blank spaces. These blank spaces indicate the extraneous regions in the boundaries of the character image.

One way to identify these blank spaces is described as follows:

We find the size of the binary image obtained from the Image Preprocessing stage. This gives us the width and the height (boundary dimensions). Since the image is in binary format, each pixel value represents a 1 (white) or a 0 (black). Note that white represents the background whereas black represents the character region. Now, to identify left side blank spaces, we find the sum of the values of pixels (0 or 1) of the first column. If this sum is equal to the height of the image i.e. all the values in that column are 1s, it means that it is an extraneous region and hence must be discarded. This procedure is repeated for subsequent columns until the sum of that column becomes less than the height. This implies the presence of a 0 in that column i.e. a part of the character is present at that position. Similarly, we repeat this procedure for right side blank spaces, starting from the right most position of the image and moving towards the left until we encounter a part of the character. In the same way, we find the upper and lower side blank spaces. The only difference here is that we find the sum of individual rows and compare it with the width of the binary image. In fact, using this procedure, we find the new width and height of the character image which exactly bounds the character without any extraneous regions present.

In short, we find out the size of a character and remove any extra regions surrounding the character. Thus we get the exact boundary of the character. Each character image is cropped to this size and used for further processing.

(5) Character Resize:

The cropped character images obtained from the previous step are of varying sizes as it depends on the size and shape of the characters. Hence, it becomes necessary to resize these images into some standard size for ease of processing. In this step, we resize each cropped character image into a 70x50 image. This resized image is then fed as input for feature extraction.

(6) Feature Extraction:

In this step, we divide the 70x50 character image obtained from the previous step into multiple 10x10 matrices starting from left. Thus, we obtain 5 such smaller matrices of size 10x10 from one row and we have 7 such rows. Thus, we obtain a total of 5*7=35 smaller matrices of size 10x10.

We convert each 5x7 character representation into a single 35x1 vector. For this, we take the sum of all the elements present in the 1st smaller matrix of size 10x10. This gives us 1st element of our 35x1 vector. Using the same procedure, we get all the 35 elements of the vector.

This gives us the character representation in vector form. The specialty of this vector is that this vector is unique for the character it represents. We repeat this procedure for all the characters in the training data. Thus we get an input matrix for training neural network.

(7) Train Neural Network:

We propose to perform Offline English character recognition using 3 different architectures of Neural Networks:

1. Feedforward Backpropagation Neural Network (FFBPNN)
2. Elman Backpropagation Neural Network (EBPNN)
3. Fitting Neural Network (FNN)

1. Feedforward Backpropagation Network (FFBPNN):

The first term, "feedforward" describes how this neural network processes and recalls patterns. In a feedforward neural network, neurons are only connected foreword. Each layer of the neural network contains connections to the next layer (for example, from the input to the hidden layer), but there are no connections back.

The term "backpropagation" describes how this type of neural network is trained. Backpropagation is a form of supervised training. When using a supervised training method, the network must be provided with both sample inputs and anticipated outputs. The anticipated outputs are compared against the actual outputs for given input.

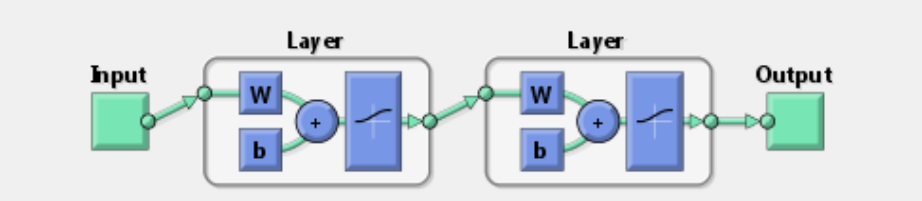


Figure 3.14: FFBPNN

Using the anticipated outputs, the backpropagation training algorithm then takes a calculated error and adjusts the weights of the various layers backwards from the output layer to the input layer.

Thus, the backpropagation and feedforward algorithms are used together- to create a neural network that uses the feedforward algorithm to determine its output and uses the backpropagation training algorithm.

2. Elman Backpropagation Network (EBPNN):

EBPNN is defined as follows:

"A three-layer network is used (arranged vertically as x, y, and z in the figure), with the addition of a set of "context units" (u in the figure). There are connections from the middle (hidden) layer to these context units fixed with a weight of one. At each time step, the input is propagated in a standard feed-forward fashion, and then a learning rule is applied. Fixed back connections result in the context units always maintaining a copy of the previous values of the hidden units (since they propagate over the connections before the learning rule is applied). Thus the network can maintain a sort of state, allowing it to perform such tasks as sequence-prediction that is beyond the power of a standard multilayer perceptron."

Most of the neural network architectures proposed by Jeffrey Elman were recurrent and designed to learn sequential or time-varying patterns. In this way, the algorithms could recognize and predict learned series of values or events.

Elman's definition of a context revolved around prior internal states, and thus he added a layer of "context units" to a standard feedforward net. In this way, the states of the hidden units could be fed back into the hidden units during the next stage of input.

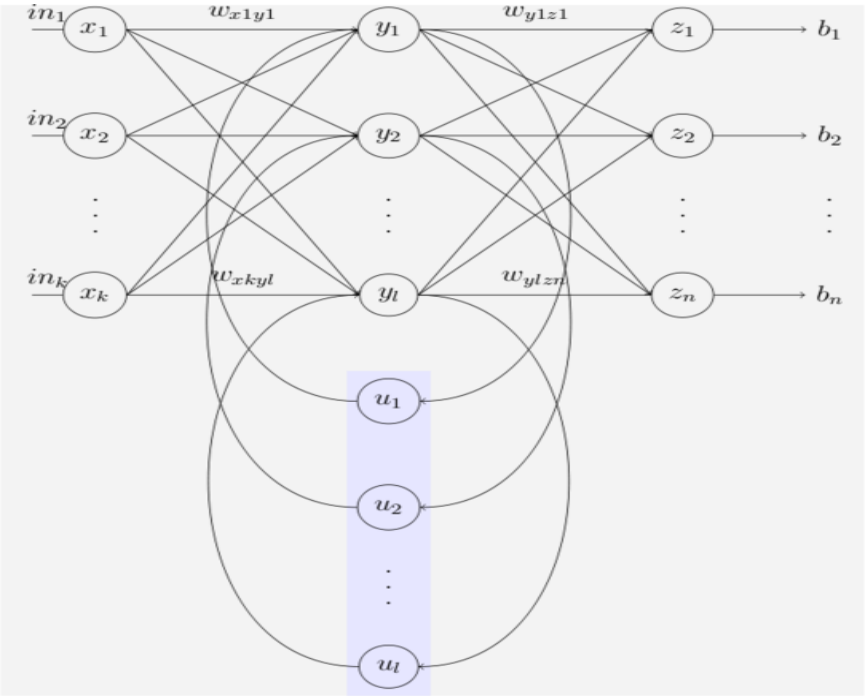


Figure 3.15: EBPNN

Jeffrey Elman (1990) best describes EBPNN as follows:

"Both the input units and context units activate the hidden units; and then the hidden units feed forward to activate the output units. The hidden units also feed back to activate the context units. This constitutes the forward activation. Depending on the task, there may or may not be a learning phase in this time cycle. If so, the output is compared with a teacher input and backpropagation of error is used to incrementally adjust connection strengths. Recurrent connections are fixed at 1.0 and are not subject to adjustment. At the next time step t+1 the above sequence is repeated. This time the context units contain values which are exactly the hidden unit values at time t. These context units thus provide the network with memory."

Thus, an Elman neural network is feed forward network with an input layer, a hidden layer, an output layer and a special layer called context layer. The output of each hidden neuron is copied into a specific neuron in the context layer. The value of the context neuron is used as an extra input signal for all the neurons in the hidden layer one time

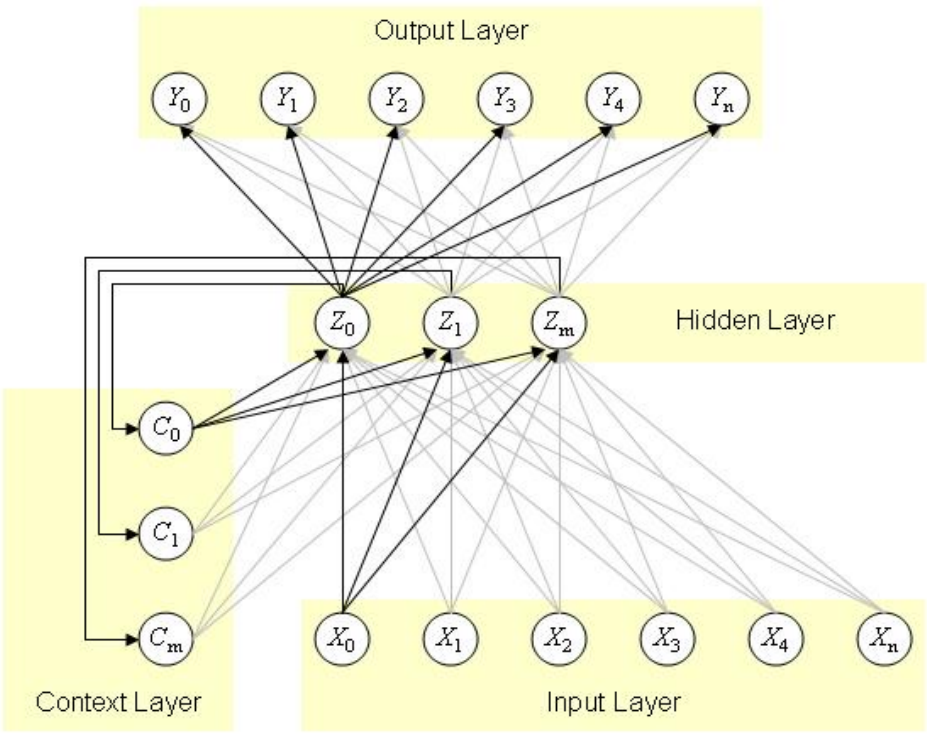


Figure 3.16: Common Structure of Elman Network

step later. In an Elman network, the weights from the hidden layer to the context layer are set to one and are fixed because the values of the context neurons have to be copied exactly. Furthermore, the initial output weights of the context neurons are equal to half the output range of the other neurons in the network. The Elman network can be trained with gradient descent back propagation and optimization methods.

3. Fitting Neural Network (FNN):

Fitting networks are feedforward neural networks used to fit an input-output relationship. In fitting problems, you want a neural network to map between a data set of numeric inputs and a set of numeric targets.

Examples of this type of problem include estimating house prices from such input variables as tax rate, pupil/teacher ratio in local schools and crime rate, estimating engine emission levels based on measurements of fuel consumption and speed or predicting a pa-

tient's body fat level based on body measurements.

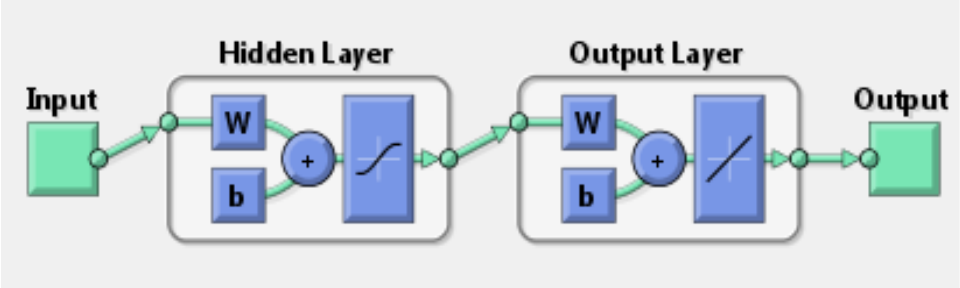


Figure 3.17: FNN

Using the above mentioned Neural Networks in MATLAB, we will train and test our English character recognition system. Now we will describe 2 techniques that we will be using to train the Neural Networks.

Techniques for Training the Neural Network:

To train the Neural Network, we have proposed 2 different techniques:

1. **Using an Identity matrix:**

   In this technique, we use an identity matrix as a target matrix for training the neural network. We consider 62 different characters in the training state:

```
0 - 9 = 10 characters
A - Z = 26 characters
a - z = 26 characters
Hence, total characters = 62.
```

   Hence, we use an identity matrix of order 62x62. Each diagonal element in the matrix is 1 representing separate characters. Thus, there are 62 such neurons in the output layer of the neural network that can be fired. Hence, the neural network can

be trained to fire only one specific neuron when a particular character is fed to the input layer of the neural network.

```
Pros: More accurate
Cons: Requires more space
```

2. **Using ASCII values:**

   In this technique, we use ASCII values of characters as a target matrix for training the neural network. ASCII values are converted into binary format and stored into the target matrix. Each ASCII value needs 7 bits for its binary representation and there are total of 62 different characters. Therefore, we have a target matrix of size 7x62.

```
Pros: Requires less space
Cons: Less Accurate
```

   Once we have the input matrix and the target matrix, we can train the neural network

(8) Simulate Neural Network:

Once the neural network is trained, we simulate it to check the efficiency of the output it produces. We use a small part of the data from the training dataset to simulate the trained neural network.

3.3 Proposed Algorithm For Character Recognition

The flowchart for the character recognition algorithm is shown below:

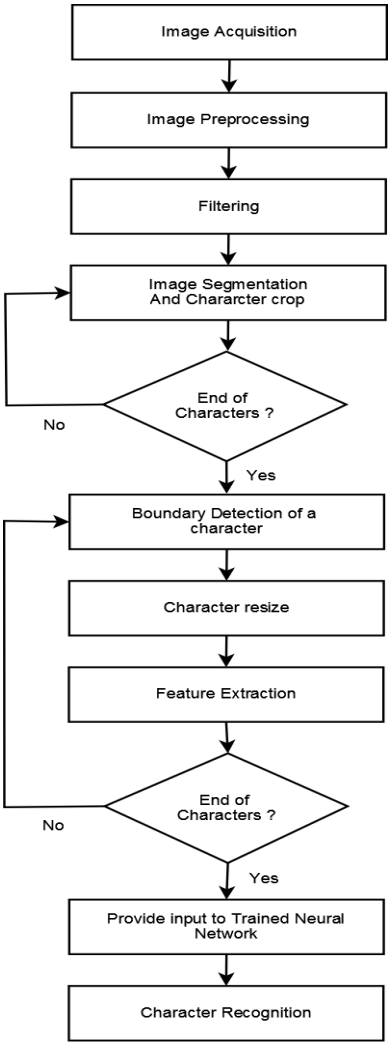


Figure 3.18: Character Recognition Algorithm

The steps involved in the character recognition algorithm are described below:

(1) Image Acquisition: Same as mentioned in 3.2

(2) Image Preprocessing: Same as mentioned in 3.2

(3) Filtering:

In this step, we separate the background from the original image and then subtract background image from the original image. Thus, we obtain a clear image of the characters to be recognized and the unwanted background noise is removed.

(4) Segmentation and Character Crop: Same as mentioned in 3.2

(5) Boundary Detection of a Character: Same as mentioned in 3.2

(6) Character Resize: Same as mentioned in 3.2

(7) Feature Extraction: Same as mentioned in 3.2

(8) Provide input to trained Neural Network:

In this step, we provide characters to be recognized to the trained neural network in the form of an image. The Neural network generates output based on its training.

(9) Character Recognition:

In this step, we apply reverse method for recognizing characters i.e. either convert binary ASCII value into equivalent character or generate character based on the number of the neuron which had fired. Then we put the recognized characters into a text file and display it.

Thus, we have described our proposed character recognition algorithm.

Chapter 4

Results and Discussion

In this chapter, we show the results obtained by implementing the proposed character recognition algorithm on certain input images. We have implemented the algorithm in MATLAB with the help of the Image Processing Toolbox as well as the Neural Network Toolbox. They played an important role in every phase of our algorithm:

1. Image Preprocessing
2. Training the Neural Network
3. Testing the Neural Network

Here, we show only the results obtained by applying the algorithm on images containing handwritten text only due to its complex nature, although we have proposed our algorithm for both handwritten as well as machine text images.

We first show the results obtained by using the ASCII value technique on Feedforward Backpropagation network and Fitting network. Then we show the results obtained by using the Identity matrix technique on Feedforward Backpropagation network, Elman Backpropagation network and Fitting network. Finally, we compare the results obtained so far in tabular and graphical form.

4.1 Target: ASCII Values

Here, we use binary representation of ASCII values of characters as a target matrix for training the neural network. Size of target matrix is 7x62.

4.1.1 Newff: Feed-forward backpropagation network

The trained Newff network is shown below:

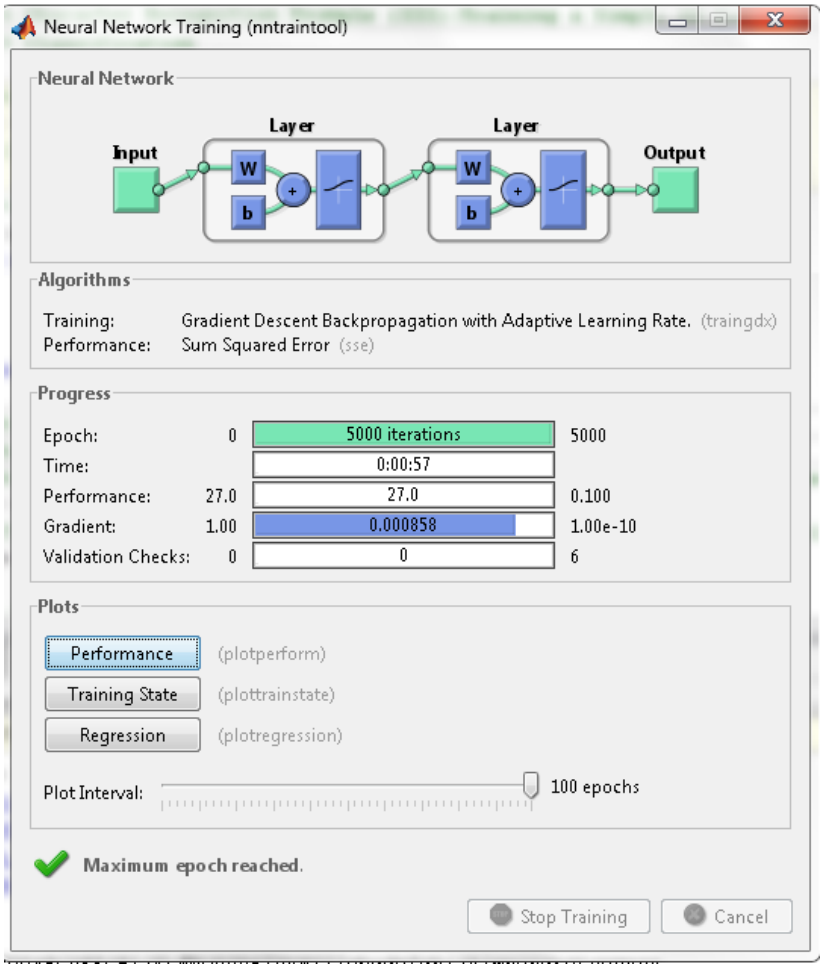


Figure 4.1: Training the Newff network (ASCII)

Newff Regression:

Linear Regression is the relation between variables when the regression equation is linear. A regression is a statistical analysis assessing the association between two variables. It is used to find the relationship between two variables.

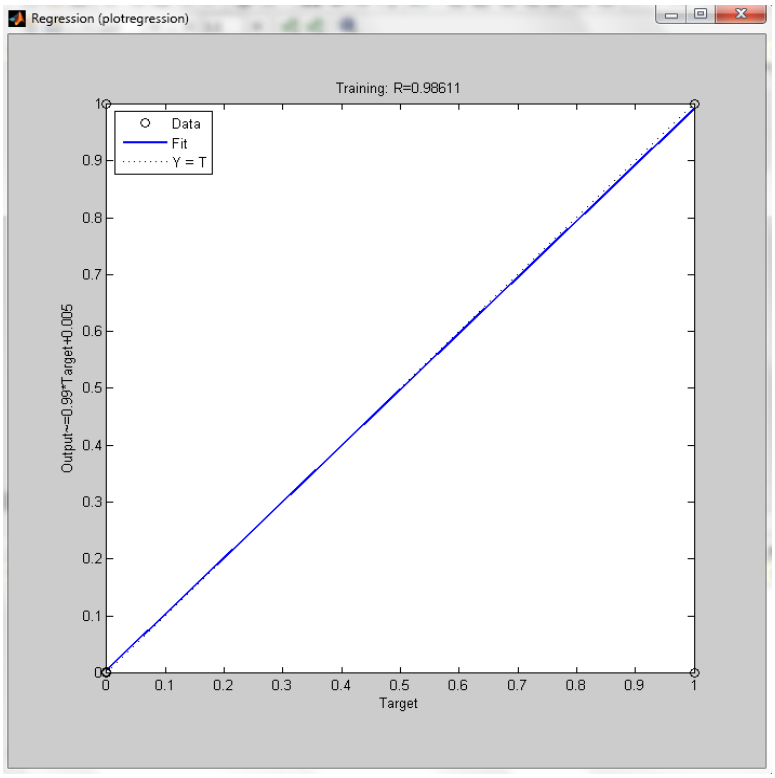


Figure 4.2: Newff Regression Graph (ASCII)

We use the formula, $y = a + bx$, where a and b are,

$$a = \frac{(\sum y)(\sum x^2) - (\sum x)(\sum xy)}{n(\sum x^2) - (\sum x)^2}$$
$$b = \frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum x^2) - (\sum x)^2}$$

and n is the sample size.

Here, we get the value of R as 0.98611 which is very close to 1. Hence the result is quite good.

Newff Output:

The result obtained is:

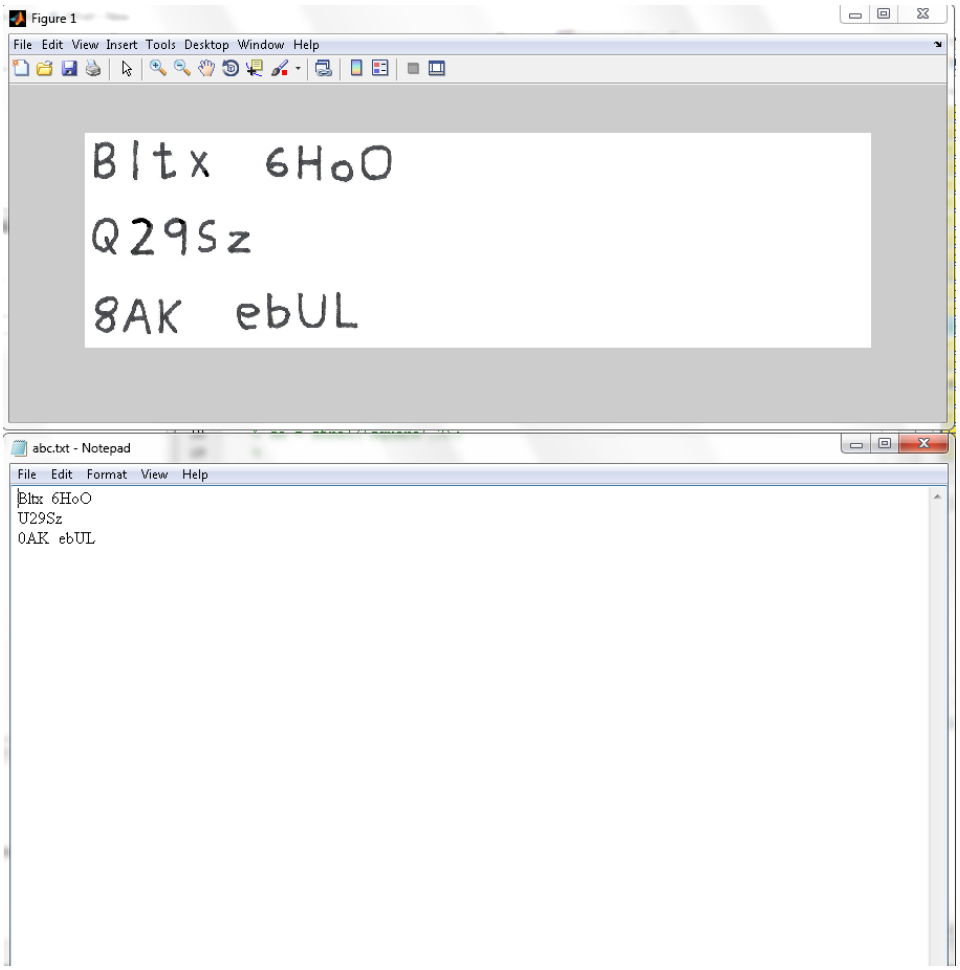


Figure 4.3: Newff Output (ASCII)

Out of 20 characters, newff function recognized 18 characters correctly.

4.1.2 Newfit: Fitting network

The trained Newfit network is shown below:

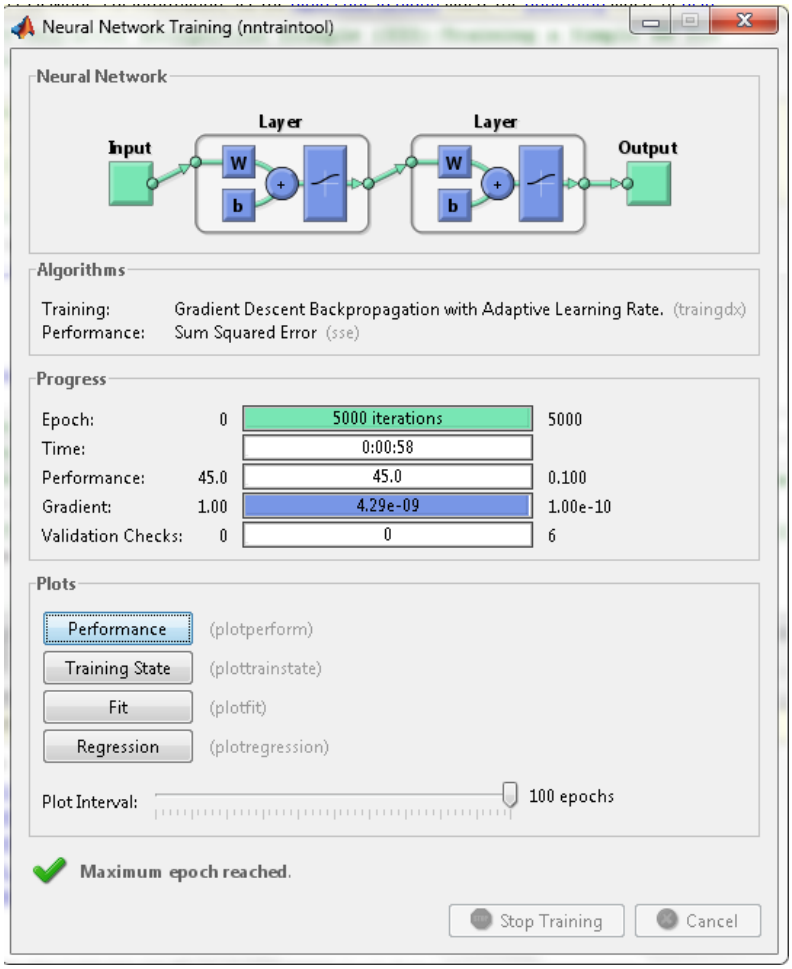


Figure 4.4: Training the Newfit network (ASCII)

Newfit Regression:

Linear Regression is the relation between variables when the regression equation is linear. A regression is a statistical analysis assessing the association between two variables. It is used to find the relationship between two variables.

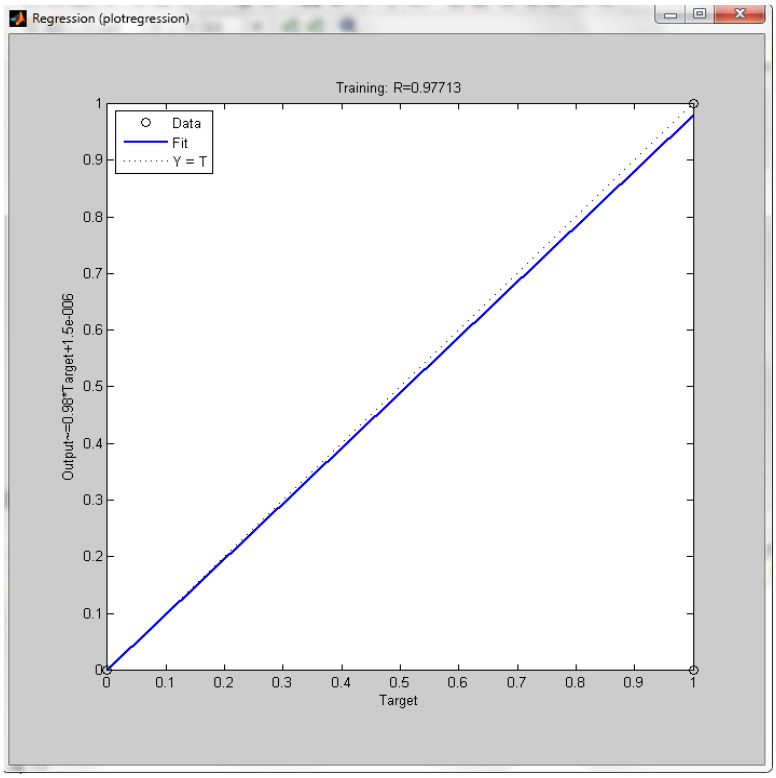


Figure 4.5: Newfit Regression Graph (ASCII)

We use the formula, $y = a + bx$, where a and b are,

$$a = \frac{(\sum y)(\sum x^2) - (\sum x)(\sum xy)}{n(\sum x^2) - (\sum x)^2}$$
$$b = \frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum x^2) - (\sum x)^2}$$

and n is the sample size.

Here, we get the value of R as 0.97713. Hence result is slightly less accurate as compared to newff function as we can see in the following output.

Newfit Output:

The result obtained is:

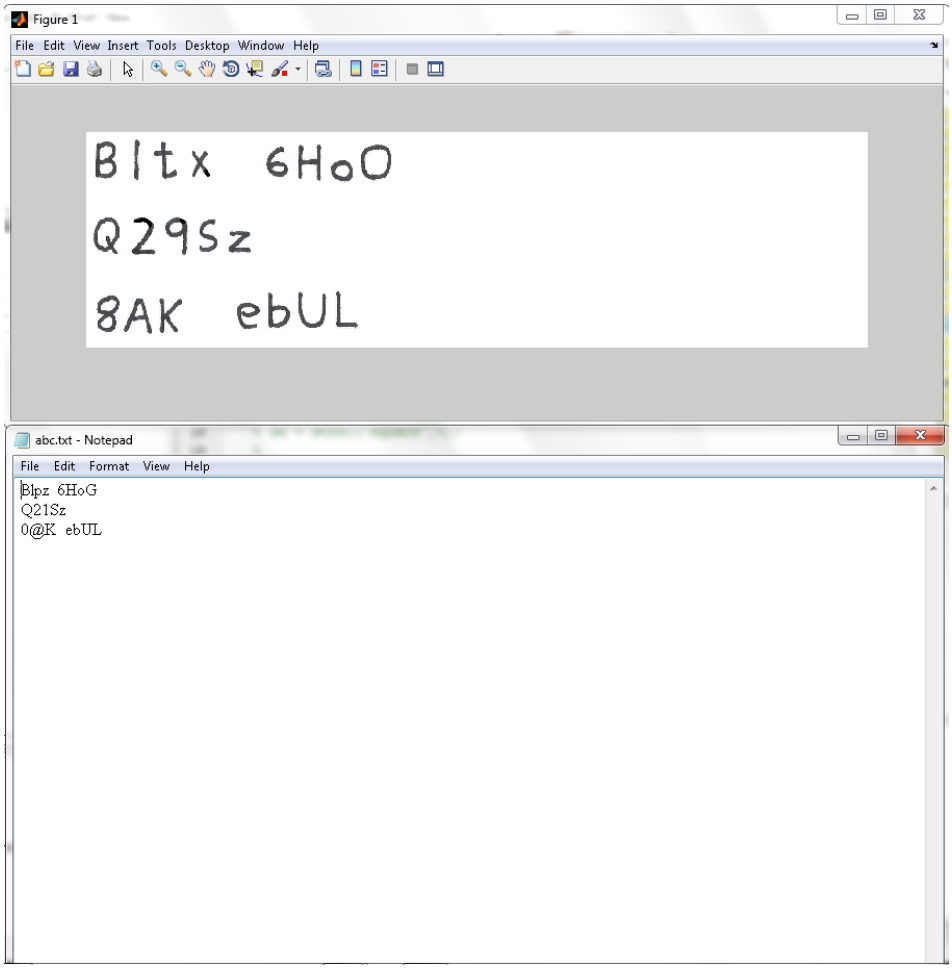


Figure 4.6: Newfit Output (ASCII)

Out of 20 characters, newfit function could recognize 14 characters correctly.

4.2 Target: Identity Matrix

Here, we use an identity matrix as a target matrix for training the neural network. Size of the identity matrix is 62.

4.2.1 Newff: Feed-forward backpropagation network

The trained Newff network is shown below:

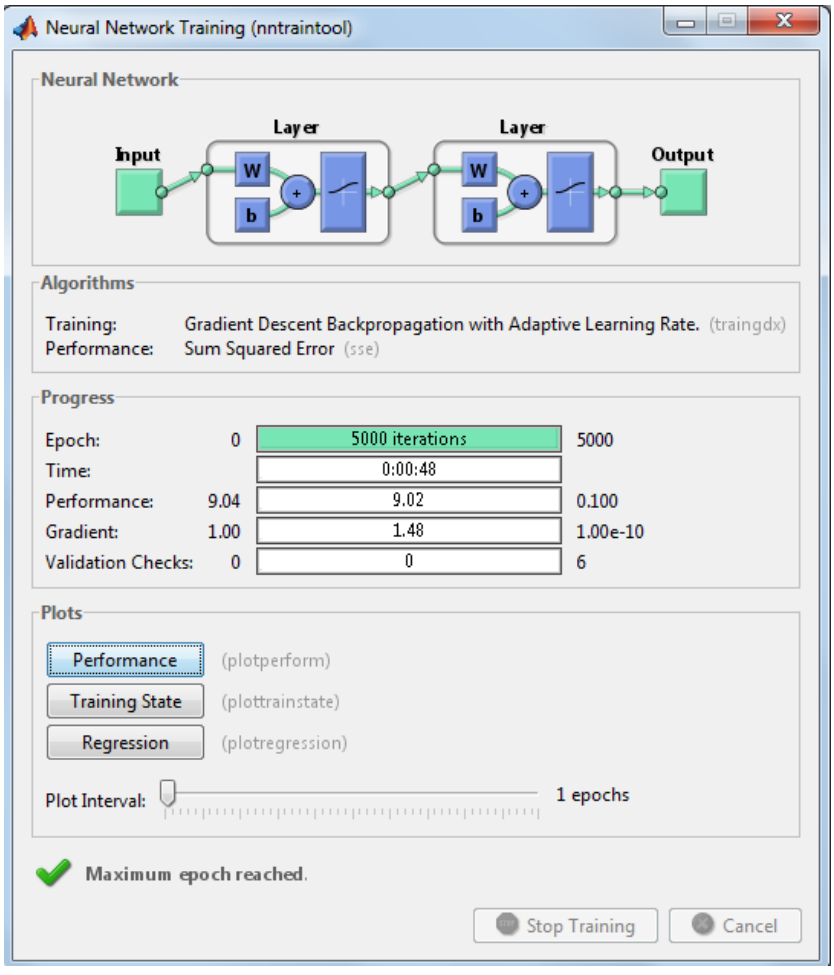


Figure 4.7: Training the Newff network (Identity)

Newff Regression:

Linear Regression is the relation between variables when the regression equation is linear. A regression is a statistical analysis assessing the association between two variables. It is used to find the relationship between two variables.

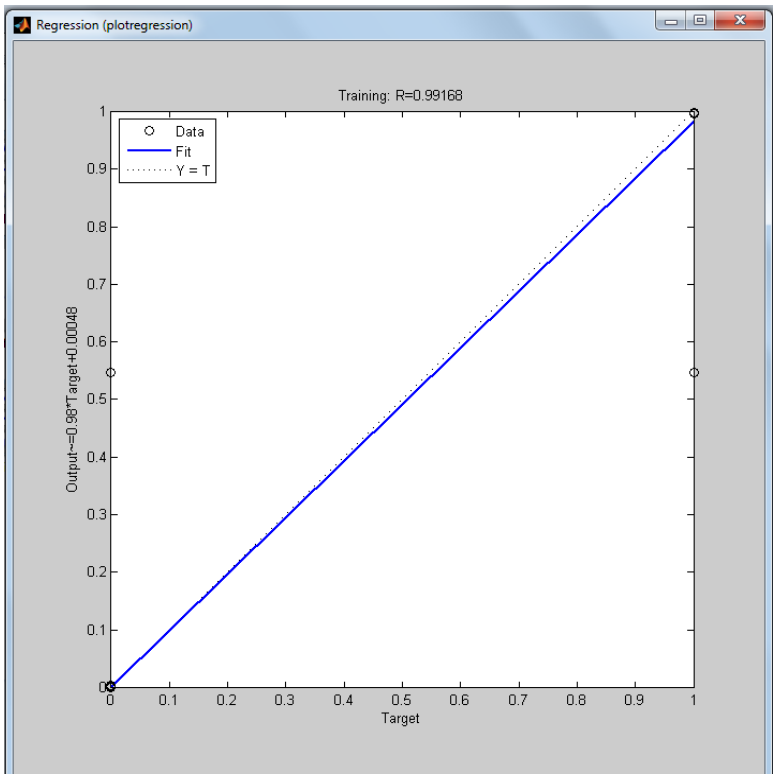


Figure 4.8: Newff Regression Graph (Identity)

We use the formula, $y = a + bx$, where a and b are,

$$a = \frac{(\sum y)(\sum x^2) - (\sum x)(\sum xy)}{n(\sum x^2) - (\sum x)^2}$$
$$b = \frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum x^2) - (\sum x)^2}$$

and n is the sample size.

Here, we get the value of R as 0.99168 which is quite good and can be seen in the output shown below.

Newff Output:

The result obtained is:

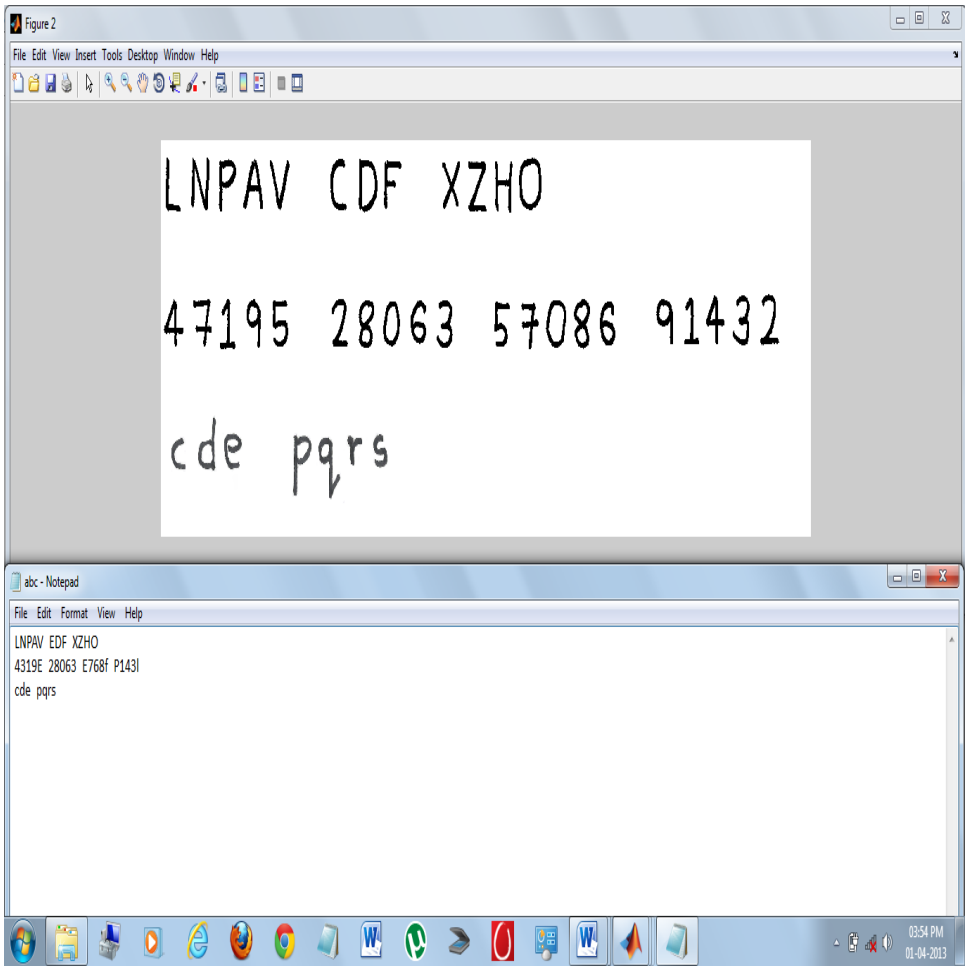


Figure 4.9: Newff Output (Identity)

Out of 39 characters, newff function could recognize 31 characters correctly.

4.2.2 Newfit: Fitting network

The trained Newfit network is shown below:

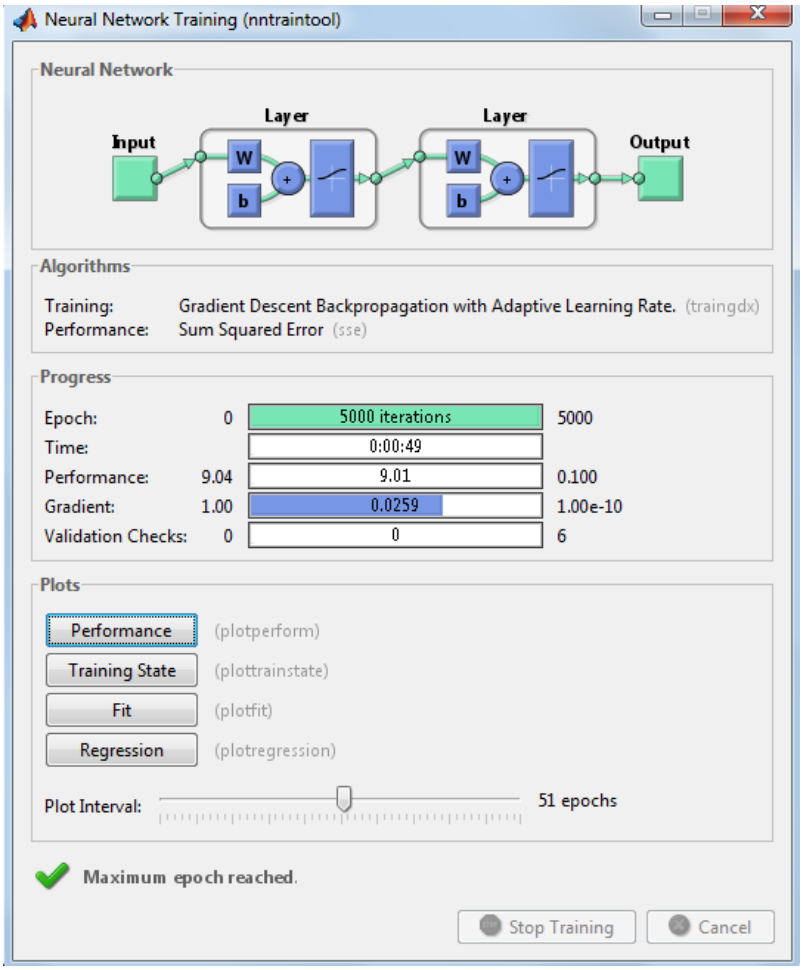


Figure 4.10: Training the Newfit network (Identity)

Newfit Regression:

Linear Regression is the relation between variables when the regression equation is linear. A regression is a statistical analysis assessing the association between two variables. It is used to find the relationship between two variables.

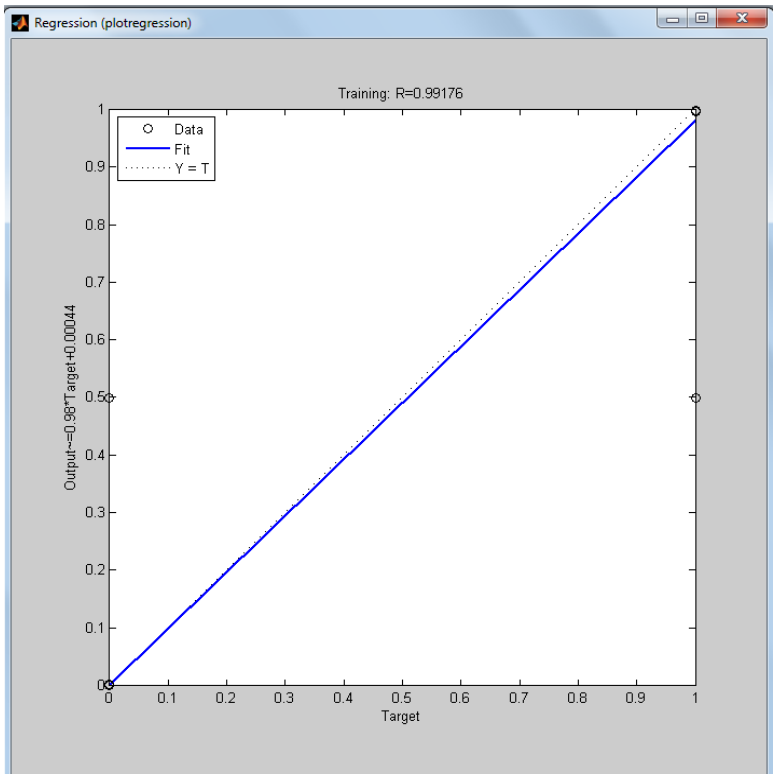


Figure 4.11: Newfit Regression Graph (Identity)

We use the formula, $y = a + bx$, where a and b are,

$$a = \frac{(\sum y)(\sum x^2) - (\sum x)(\sum xy)}{n(\sum x^2) - (\sum x)^2}$$
$$b = \frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum x^2) - (\sum x)^2}$$

and n is the sample size.

Here, we get the value of R as 0.99176 which is better than newff function but recognizes fewer characters correctly as shown below.

Newfit Output:

The result obtained is:

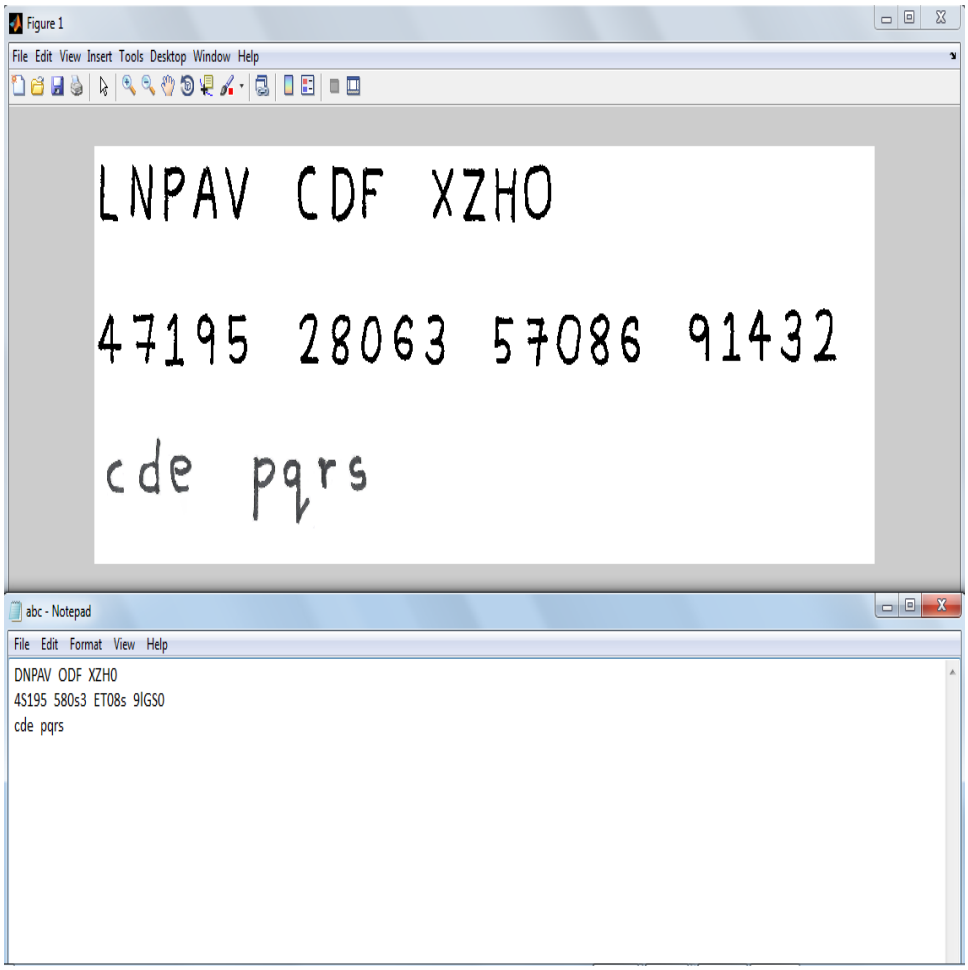


Figure 4.12: Newfit Output (Identity)

Out of 39 characters, newfit could recognize 27 characters correctly.

4.2.3 Newelm: Elman backpropagation network

The trained Newelm network is shown below:

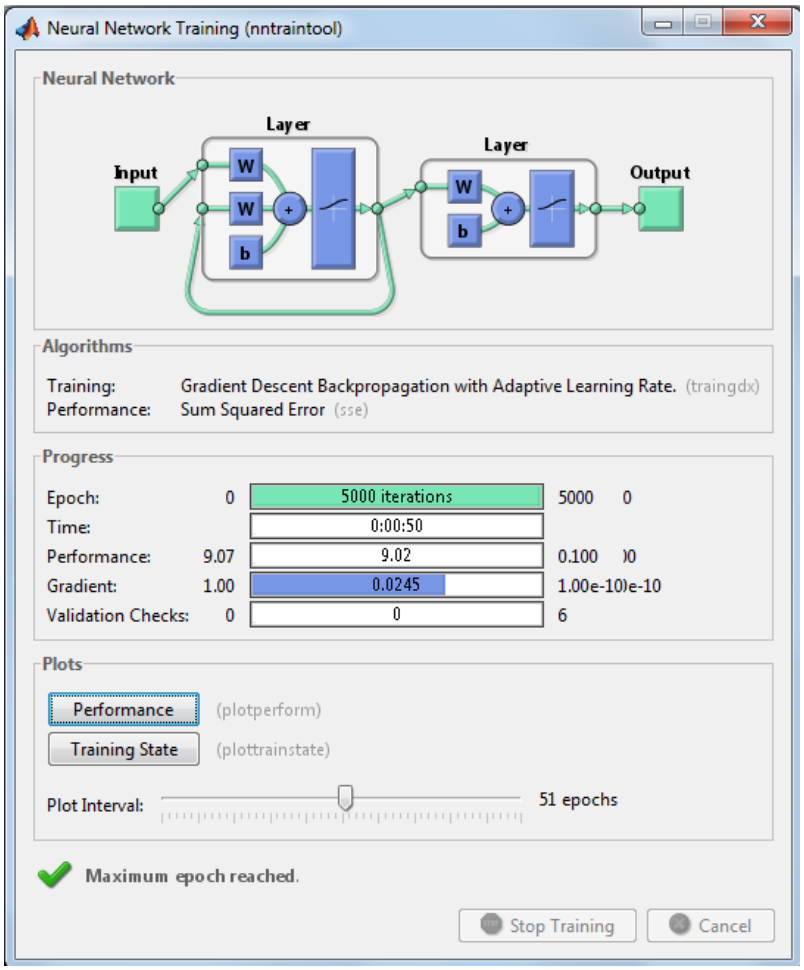


Figure 4.13: Training the Newelm network (Identity)

Newelm Output:

The result obtained is:

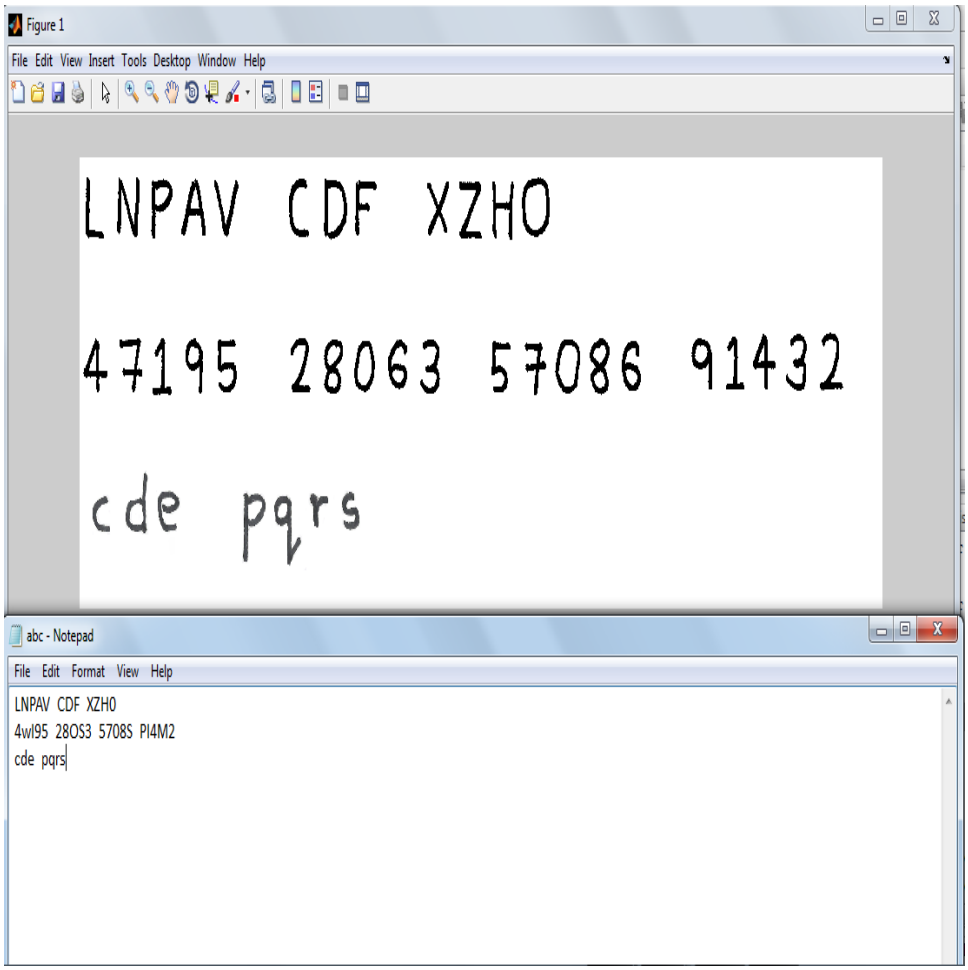


Figure 4.14: Newelm Output (Identity)

Out of 39 characters, newelm could recognize 32 characters correctly and produces the highest accuracy among the 3 networks.

4.3 Comparison of Results

Now, we compare and analyze the results obtained in 2 forms: Tabular and Graphical. This is done for both the techniques: ASCII values and Identity Matrix.

4.3.1 Comparison Table

Technique	**Neural Network**	**RR(%)**
ASCII	Newff	90.0
	Newfit	70.0
Identity Matrix	Newff	79.48
	Newfit	69.23
	Newelm	82.05

Table 4.1: Comparison of Recognition Rate (RR)

Recognition Rate (RR) is defined as the ratio of the number of characters correctly recognized and the total number of characters in the input image.

Recognition Rate is calculated by the formula:

$$RR = \frac{Number\ of\ characters\ correctly\ recognized}{Total\ number\ of\ characters\ in\ the\ image} \times 100\ \%$$

Using the above formula and using the results obtained in the previous 2 sections, we obtained the recognition rates as shown in the above table.

4.3.2 Comparison Graph for ASCII value technique

In the below graph, X-axis represents image index (Image Number) and Y-axis represents precision (RR). We took 5 samples of input images and calculated their RR for each NN i.e. feed-forward and fitting. Then we plotted corresponding co-ordinates and formed the graph as shown below.

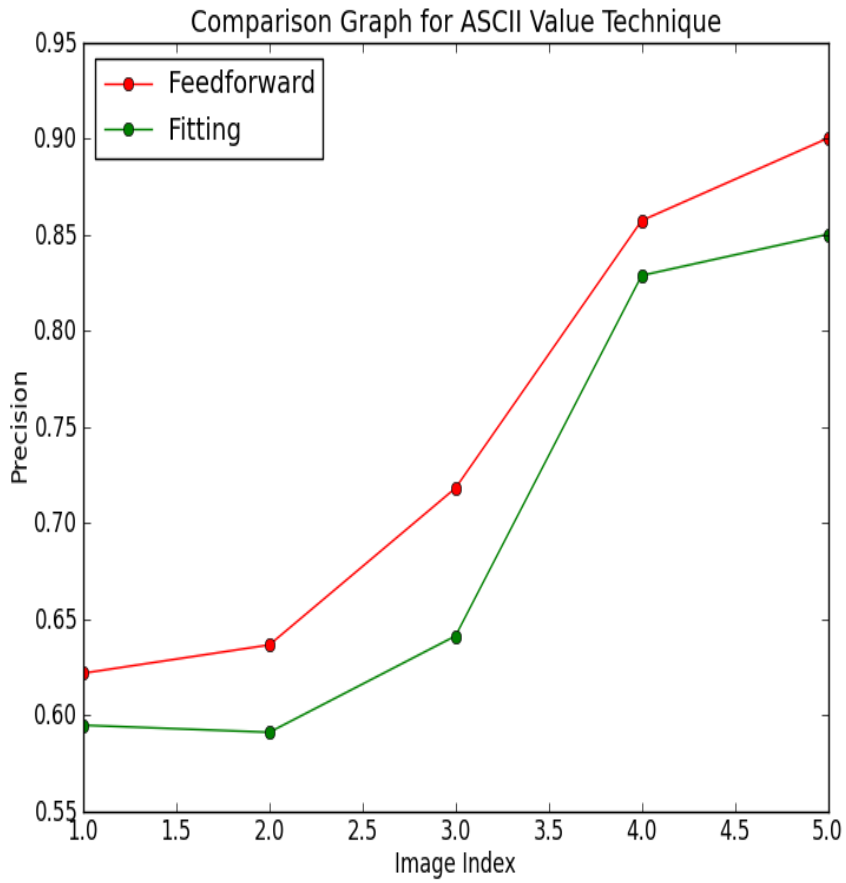


Figure 4.15: Comparison Graph for ASCII Value technique

Thus, we see that the Feedforward Backpropagation network performs best when used with 'ASCII value' technique.

4.3.3 Comparison Graph for Identity Matrix technique

In the below graph, X-axis represents image index (Image Number) and Y-axis represents precision (RR). We took 5 samples of input images and calculated their RR for each NN i.e. feed-forward, elman and fitting. Then we plotted corresponding co-ordinates and formed the graph as shown below.

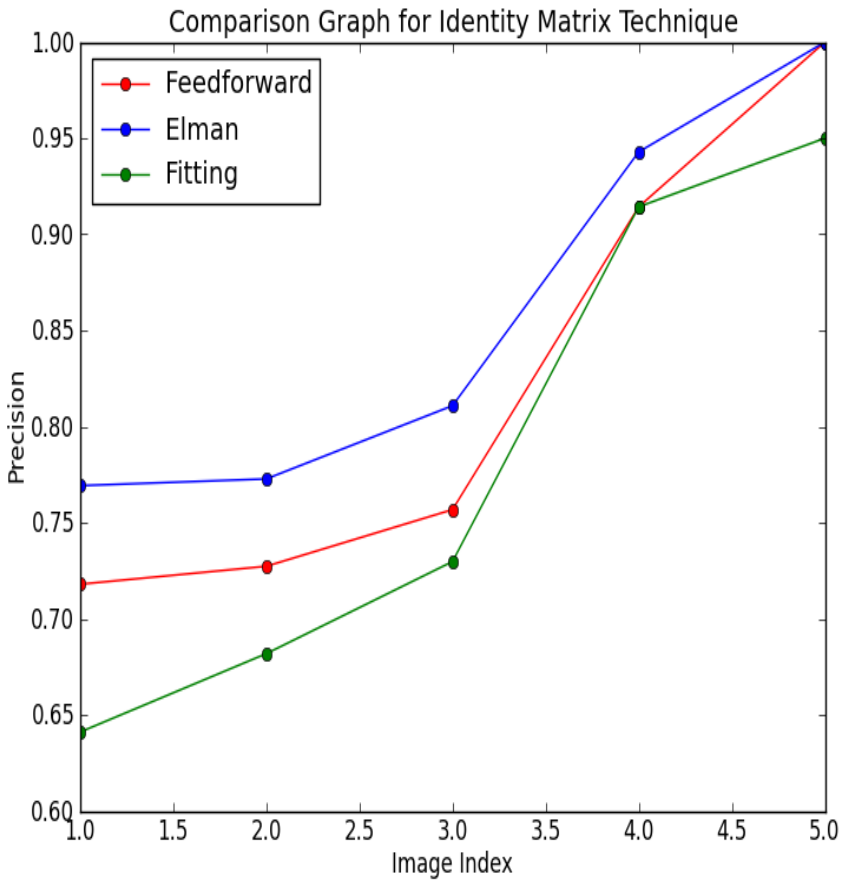


Figure 4.16: Comparison Graph for Identity Matrix technique

Thus, we see that the Elman backpropagation network performs best when used with the 'Identity Matrix' technique.

Chapter 5

Conclusion and Future Scope

Off-line English character recognition is a complex problem, especially handwritten text which are not easily solvable using standard OCR methodologies because of the infinite variations of character shapes in human writing. Neural Networks provide an efficient way to design an algorithm which instead of relying on a pre-specified rule for character recognition, 'learns' to identify characters by example.

Offline English character recognition using NN can be divided into three main parts:

1. Image pre-processing to get the training data
2. Training the neural network using suitable techniques
3. Testing the neural network for character recognition

We proposed an algorithm for training the neural network, for which we proposed 2 techniques: Identity Matrix and ASCII value. We also proposed an algorithm for character recognition and implemented it on 3 neural networks: Feedforward Backpropagation network, Elman Backpropagation network and Fitting network.

From the results obtained, we observed that 'Identity Matrix' technique performed better than 'ASCII' technique. Similarly, we also found out that the Elman backpropagation network performed best when used with the 'Identity Matrix' technique and Feedforward Backpropagation network performed best when used with 'ASCII value' technique.

5.1 Improving the Results

Some Ways of Improving the Results are:

- If the network is not sufficiently accurate, you can try initializing the network and the training again. Each time your initialize a feed-forward network, the network parameters are different and might produce different solutions.
- As a second approach, you can increase the number of hidden neurons. Larger numbers of neurons in the hidden layer give the network more flexibility because the network has more parameters it can optimize. Varying number of neurons in each layer provides distinct results and the best one can be chosen.
- A third option is to try a different training function.
- We also propose to use Eigenvalues for character recognition as character images are represented in matrix form and Eigenvalues are unique for each character.
- Implement ASCII value technique using Elman Backpropagation network.
- Finally, try using additional training data. Datasets are one of the most important things when constructing new neural network. Providing additional data for the network is more likely to produce a network that generalizes well to new data.

Bibliography

[1] C.Tappert, C.Suen, and T.Wakahara, "The state of the art in online handwriting recognition," Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 12, pp. 787-808, Aug. 1990.

[2] S. Mori, C. Suen, and K. Yamamoto, "Historical review of ocr research and development", Proceedings of the IEEE, vol. 80, pp. 1029-1058.

[3] "Optical Character Recognition Using Optimisation Algorithms"- Kirill Safronov, Dr.-Ing. Igor Tchouchenkov and Prof. Dr. Ing. Heinz Wörn.

[4] "On-Line and Off-Line Handwriting Recognition: A Comprehensive Study" - Rejean Plamondon and Sargur N. Srihari.

[5] "Handwritten Character Recognition using Neural Networks"-Sunith Bandaru.

[6] "Diagonal Based Feature Extraction For Handwritten Alphabets Recognition System Using Neural Network" - J. Pradeep, E. Srinivasan and S. Himavathi.

[7] "An Overview of Character Recognition Focused on Off-Line Handwriting"- Nafiz Arica and Fatos T. Yarman-Vural.

[8] "Hand Written English Character Recognition using Row-wise Segmentation Technique (RST)" - Rakesh Kumar Mandal, N R Manna.

[9] "Applying Neural Networks to Character Recognition" - Eric W. Brown.

[10] "Multiple Algorithms for Handwritten Character Recognition", Jonathan J. Hull, Alan Commike and Tin-Kam Ho.

[11] Basu, S. et al. (2005) "Handwritten 'Bangla' alphabet recognition using an MLP based classifier," Proceeding of 2nd National Conference on Computer Processing of Bangla, pp. 285-291.

[12] Manivannan, N. and Neil, M.A.A. (2010) "Optical correlator-neural network hybrid system for many patterns recognition," Proceeding of 8th International Symposium on Intelligent Systems and Informatics (SISY), pp.249-251.

[13] Pal, A. and Singh, D. (2010) "Handwritten English Character Recognition Using Neural Network," International Journal of Computer Science and Communication, Vol.1, No.2, pp. 141-144.

[14] Perwej, Y. and Chaturvedi, A. (2011) "Neural Networks for Handwritten English Alphabet Recognition," International Journal of Computer Applications, Vol. 20, No. 7, pp. 1-5.

[15] Pal, U. et al. (2007) "Handwritten numeral recognition of six popular scripts," 9th International conference on Document analysis Recognition, vol. 2, pp. 749-753.

[16] Dinesh, A. U. et al. (2007) "Isolated handwritten Kannada numeral recognition using structural feature and K-means cluster," IISN, pp. 125-129.

[17] Yanhua, M. and Chuanjun, L. (2009) "A Recognition Algorithm for Chinese Character Based on Minimum Distance Classifier," Proceeding of 2nd International Workshop on Computer Science and Engineering, vol.2, pp.246-249.

[18] Huiqin L. et al.(2011) "The Research of Algorithm for Handwritten Character Recognition in Correcting Assignment System," Proceeding of 6th International Conference on Image and Graphics (ICIG), pp.456-460.

[19] "Time Efficient Approach To Offline Hand Written Character Recognition Using Associative Memory Net", Tirtharaj Dash, International Journal of Computing and Business Research (IJCBR).

[20] "Handwritten Character Recognition Using Multiresolution Technique and Euclidean Distance Metric", Dileep Kumar Patel, Tanmoy Som, Sushil Kumar Yadav, Manoj Kumar Singh.

[21] G. Y. Chen, T. D. Bui and A. Krzyzak, "Contour-Based Handwritten Numeral Recognition Using Multiwavelets and Neural Networks," Pattern Recognition, Vol. 36, No. 7, 2003, pp. 1597-1604.

[22] D. J. Romero, L. M. Seijas and A. M. Ruedin, "Directional Continuous Wavelet Transform Applied to Handwritten Numerals Recognition Using Neural Networks," Java Caching System, Vol. 7, No. 1, 2007, pp. 66-71.

[23] A. Mowlaei, K. Faez and A. T. Haghighat, "Feature Extraction with Wavelet Transform for Recognition of Isolated Handwritten Farsi or Arabic Characters and Numerals," IEEE Digital Signal Processing, Vol. 2, No. 2, 2002, pp. 923-926.

[24] "A Bayesian Framework for Deformable Pattern Recognition With Application to Handwritten Character Recognition", Kwok-Wai Cheung, Student Member, IEEE, Dit-Yan Yeung, Member, IEEE, and Roland T. Chin, Member, IEEE.

[25] "Recent Advances in Handwriting Recognition", Flávio Bortolozzi, Alceu de Souza Britto Jr., Luiz S. Oliveira and Marisa Morita.

[26] "Fundamentals of Neural Networks, Architectures, Algorithms and Applications", Laurene Fausett, Pearson Education.

[27] "Digital Image Processing", Second Edition, by Rafael C. Gonzalez, Richard E. Woods, Pearson Education.

[28] "Principles of Soft Computing", S.N. Sivanandam and S.N. Deepa, Wiley India, 2007.