**Optical Character recognition for Devanagari Font**



**Submitted by –**

**Ashay Churi**

**Mayur Navarkar**

**Durgesh Waghmare**



**ST. JOHN COLLEGE OF ENGINEERING AND TECHNOLOGY**

**(ENGINEERING COLLEGE)**

**Vevoor, Manor Road, Palghar (E), District Thane - 401404**

**2013-2014**

**OPTICAL CHARACTER RECOGNITION FOR DEVANAGARI FONT**

**B. E. COMPUTER ENGINEERING**

**By**

**Ashay Churi**

**Mayur Navarkar**

**DUrgesh Waghmare**

**Under the guidance of**

**Mr. Pankaj Sonawane**

**Department of Computer Engineering**

Department of Computer Engineering

Year 2013-2014

**ST. JOHN COLLEGE OF ENGINEERING AND TECHNOLOGY**

**(ENGINEERING COLLEGE)**

****

**CERTIFICATE OF APPROVAL**

**For**

**Project Synopsis**

**This to certify that**

**Ashay Churi(PID-110)**

**Mayur Navarkar (PID-110)**

**Durgesh Waghmare (PID-1102097)**

**have satisfactorily carried out the Project work entitled “OPTICAL CHARACTER RECOGNITION FOR DEVANAGARI FONT “ in partial fulfillment of Bachelor of Engineering in Computer Engineering as laid down by University of Mumbai during the academic year 2013-2014.**

**.**

**Internal Guide: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**(MR.Pankaj Sonawane)**

**Internal Examiner External Examiner**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**( ) ( )**

**Principal**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**(Prof. (Dr) S. N. Takalikar)**

**Acknowledgement**

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement helped us throughout.

We are grateful to our project Guide **Mr. Pankaj Sonawane** for the guidance, inspiration and constructive suggestions that helped us in the preparation of this project.

We would also like to take the opportunity to thank our professors for their creative inputs and support.

All of the aforementioned people have a big part in clearing our doubts about the project and giving us confidence to go about for the project.

Ashay Churi

Mayur Navarkar

Durgesh Waghmare

**Abstract**

The aim of the project **‘OCR for Devanagari Font**’ is to develop OCR software for o recognizing the Devanagari Script Characters from the scanned digital image. OCR is an Optical character recognition and is the mechanical or electronic translation of images of handwritten or typewritten text (usually captured by a scanner) into machine-editable text. OCR is a field of research in pattern recognition, artificial intelligence and machine vision.

Handwritten and Typewritten recognition is used most often to describe the ability of a computer to translate human writing into text. This may take in one of the two ways, either by scanning of written text or by writing directly on peripheral input devices.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Sr. No** | **Contents** | **Page No** |
| 1.0 | Introduction  1.1 Aim and Objectives | 8  9 |
| 2.0 | Literature Survey | 10 |
| 3.0 | Existing System | 12 |
| 4.0 | Problem Statement | 13 |
| 5.0 | Scope of the Project | 13 |
| 6.0 | Constraint of the Project | 15 |
| 7.0 | Proposed System | 20 |
| 8.0 | Methodology | 25 |
| 9.0 | Requirement Gathering And Planning  9.1 Requirement Elicitation  9.1.1 Use Case Diagram  9.2 Feasibility Study  9.4 Time Line Chart  9.5 W.B.S Chart | 30  30  30  32  33  35 |
| 10.0 | Analysis  10.1 Class Diagram  10.2 Activity Diagram  10.3 Data Flow Diagram | 36  36  37  39 |
| 11.0 | UI Design | 40 |
| 12.0 | Appendix | 42 |
| 13.0 | Bibliography and References | 44 |

**LIST OF FIGURES**

Figure no. Description Page no.

**6.1** Addition of elements in OCR-B to differentiate **14**

two characters

**6.2** Serifs in a typeface (grey serifs) **15**

**6.3** Shadow Characters **15**

**6.4** Counters is the circular negative space (grey) **16**

**6.5** Example of few characters after the removal **17**

of Shiro Rekha

**6.6** Similarities between different characters once **17**

the Shiro Rekha is removed

**6.7** Similarities between different characters once **18**

the bottom strip is removed

**6.8** Similarities between descender and Halanta **18**

**6.9** Example of open counter (grey) **19**

**7.0** Proposed OCR System **20**

**7.1**  Procedure of Devanagari Recognition **21**

**7.2** Image before binarization (left); Image after  **22**

binarization (right)

**7.3** Horizontal Projection Profiles of a document **22**

for line segmentation

**7.4** Vertical Projection Profiles of a document for **23**

word segmentation

**7.5** Three Part Of Devanagari Word **23**

**7.6** The procedure of Hindi character segmentation **24**

**8.1** Backpropagation Neural Network **29**

**1.0 Introduction :**

Machine replicating human functions, like reading, is an old dream. However, over the last five decades, machine reading has grown from a dream to reality. Machine reading uses the principles of Optical Character Recognition (OCR). OCR has also become one of the most successful applications of technology in the field of pattern recognition and artificial intelligence. Since the mid 1950s, OCR has been a very active field of research and development. While the OCR technology for some scripts like Latin is fairly mature and commercial OCR systems like Nuance OmniPage Pro or ABBYY FineReader are available which can perform with high accuracy, it is still under development for other scripts like Chinese and Devangari.

Although a great deal of research has been done for OCR applications for Latin script, even theses   
OCR based machines are still not able to compete with human reading capabilities. This prob-  
lem is more prominent for other scripts for which OCR technology is relatively newer. Typefaces   
are very important in determining the performance of the OCR technology. Hence in order to   
improve the accuracy of the OCR system, typefaces which are specially designed for OCR are   
required. For Latin script, quite a few typefaces have been designed which are optimized for OCR.   
These specially designed typefaces have a unique and well defined character set which allows for   
greater accuracy in recognition. This in turn helps in building low cost systems which can recog-  
nize characters using simple algorithms. However, no Devanagari script font is available which is   
designed specifically for machine reading and we address this problem in this report

In general, documents contain text, graphics, and images. The procedure of reading the text component in such a document can be divided into three steps:

1. Document layout analysis in which the text component of the document is extracted.

2. Segmentation, i.e. extraction of characters from the text component of the document.

3. Recognition of the segmented characters.

Typically, the OCR character segmentation stage needs to be redesigned for each new script, while   
the other stages are easier to port from one script to another and can be generalized over large   
classes of languages. There is a great need for OCR related research in Indian languages as there   
are many technical challenges which are specific to Devanagari script. With the spread of comput-  
ers in organizations and offices, automatic processing and machine reading of paper documents   
is gaining importance in India. Although a lot of research is going on Devanagari script recogni-  
tion, there is no commercial OCR systems focusing on Devanagari based languages.

**1.1 Aim and objectives :**

**Aim:** To develop an OCR for Devanagari Script Characters recognition.

**Description:**

We are going to implement the software which will recognize the characters from offline document (in image format) and use it as individual user profile.

Here we are developing OCR which will recognize handwritten or type written Devanagari Script Chatacters characters. OCR is an Optical character recognition and is the mechanical or electronic translation of images of handwritten or typewritten text (usually captured by a scanner) into machine-editable text. OCR is a field of research in pattern recognition, artificial intelligence and machine vision.

## Project Objectives:

This software is for recognizing handwritten or typewritten characters and creating profile for each particular user. This software supports various languages .The software can be used for security purposes and for creating font of user’s handwriting.

## 2.0 Literature Survey :

Now a days, there are software’s for recognizing only the English characters. It recognizes and stores the characters in ASCII format.

Optical character fields. Because very few applications survive that use true optical techniques, the OCR term has now been broadened to include digital image processing as well.

Early systems required training (the provision of known samples of each character) to read a specific font. "Intelligent" systems with a high degree of recognition accuracy for most fonts are now common. Some systems are even capable of reproducing formatted output that closely approximates the original scanned page including images, columns and other non-textual components.

**In about 1965**, Reader's Digest and RCA collaborated to build an OCR Document reader designed to digitize the serial numbers on Reader's Digest coupons returned from advertisements. The fonts used on the documents were printed by an RCA Drum printer using the OCR-A font. The reader was connected directly to an RCA 301 computer (one of the first solid state computers). This reader was followed by a specialised document reader installed at TWA where the reader processed Airline Ticket stock. The readers processed documents at a rate of 1,500 documents per minute, and checked each document, rejecting those it was not able to process correctly. The product became part of the RCA product line as a reader designed to process "Turn around Documents" such as those utility and insurance bills returned with payments.

The United States Postal Service has been using OCR machines to sort mail since 1965 based on technology devised primarily by the prolific inventor Jacob Rabinow. The first use of OCR in Europe was by the British General Post Office (GPO). In 1965 it began planning an entire banking system, the National Giro, using OCR technology, a process that revolutionized bill payment systems in the UK. Canada Post has been using OCR systems since 1971.

**In 1974** Ray Kurzweil started the company Kurzweil Computer Products, Inc. and led development of the first omni-font optical character recognition system — a computer program capable of recognizing text printed in any normal font. He decided that the best application of this technology would be to create a reading machine for the blind, which would allow blind people to have a computer read text to them out loud. This device required the invention of two enabling technologies — the CCD flatbed scanner and the text-to-speech synthesizer.

**In 1978** Kurzweil Computer Products began selling a commercial version of the optical character recognition computer program. LexisNexis was one of the first customers, and bought the program to upload paper legal and news documents onto its nascent online databases.

**1992-1996** Commissioned by the U.S. Department of Energy (DOE), Information Science Research Institute (ISRI) conducted the most authoritative of the **Annual Test of OCR Accuracy** for 5 consecutive years in the mid-90s. Information Science Research Institute (ISRI) is a research and development unit of University of Nevada, Las Vegas. ISRI was established in 1990 with funding from the U.S. Department of Energy. Its mission is to foster the improvement of automated technologies for understanding machine printed documents.

One study based on recognition of 19th and early 20th century newspaper pages concluded that character-by-character OCR accuracy for commercial OCR software varied from 71% to 98%; total accuracy can only be achieved by human review. Other areas—including recognition of hand printing, cursive handwriting, and printed text in other scripts (especially those East Asian language characters which have many strokes for a single character)—are still the subject of active research.

recognition, usually abbreviated to OCR, is the mechanical or electronic translation of images of handwritten, typewritten or printed text (usually captured by a scanner) into machine-editable text.

OCR is a field of research in pattern recognition, artificial intelligence and machine vision. Though academic research in the field continues, the focus on OCR has shifted to implementation of proven techniques. Optical character recognition (using optical techniques such as mirrors and lenses) and digital character recognition (using scanners and computer algorithms) were originally considered separate

**3.0 Existing Systems :**

Currently there are many systems available which are capable of performing the optical character recognition. Some of them are as follows

**OmniPage :** It  is an [optical character recognition](http://en.wikipedia.org/wiki/Optical_character_recognition) application available from [Nuance Communications](http://en.wikipedia.org/wiki/Nuance_Communications). OmniPage was one of the first OCR programs to run on personal computers. It was developed in the late 1980s and sold by Caere Corporation, a company headed by [Robert Noyce](http://en.wikipedia.org/wiki/Robert_Noyce). The original developers were Philip Bernzott, John Dilworth, David George, Bryan Higgins, and Jeremy Knight. Caere was acquired by[ScanSoft](http://en.wikipedia.org/wiki/ScanSoft) in 2000. ScanSoft acquired Nuance Communications in 2005, and took over its name.OmniPage supports more than 120 different languages.

**ABBYY FineReader**[**:**](http://en.wikipedia.org/wiki/Help:IPA_for_English) It is an international software company that provides [optical character recognition](http://en.wikipedia.org/wiki/Optical_character_recognition), [documentcapture](http://en.wikipedia.org/wiki/Document_capture) and [language software](http://en.wikipedia.org/wiki/Language_software) for both PC and mobile devices.

The majority of ABBYY products, such as [ABBYY FineReader](http://en.wikipedia.org/wiki/ABBYY_FineReader), are intended to simplify converting paper documents to digital data. ABBYY also provides language products and services.

**Optical Character Recognition in Google Drive :** In Google Drive, we take your uploaded images or PDF files, scan the file, and use computer algorithms to convert the file into a Google document.

For best results, the image or PDF files need to meet certain requirements:

* **Resolution**: High-resolution files work best. As a rule of thumb, we recommend each line of text in the documents to be of at least 10 pixels height.
* **Orientation**: Only documents with horizontal left-to-right text are recognized. If you've accidentally scanned or captured a document in a different orientation, please use a program to retouch and edit images to rotate them before uploading to Google Drive.
* **Languages, fonts and character sets**: Our OCR engine supports a number of character sets, but support for non-Latin character sets is still experimental. You can select the language of your document from a drop-down menu. You'll get better results if your file includes common fonts such as Arial and Times New Roman.
* **Image quality**: Sharp images with even lighting and clear contrasts will work best. Motion blur or bad camera focus will decrease the quality of the detected text.

**4.0 Problem Statement :**

Currently there are many systems are available for performing the optical character Recognition .

All these systems are build for standard foreign languages like English , French , Spanish and German . They does not support the Recognition of regional languages like Marathi , Sanskrit , Hindi etc . We are going to build a Software which is capable of recognizing Devanagari Script Characters .

Our main aim is to recognize these characters and to store or save them into a standard text file so that the user can easily go through them and can perform various text related operations like Copy , Paste , Find , Replace etc .

**5.0 Scope of The Project :**

This system can be used by multiple users. We can do this by improving our software for recognizing the handwriting of more than one user. Also if we can take the stroke information and give it to our system, then it will be possible to recognize even cursive script also.

The recognized characters are stored in the text file. We can add words to the sound files and invoke them through the program, so that the recognized words can be read aloud. Thus we can make the computer read the handwritten document.

**6.0 Constrains of The Project :**

Every letter has to be more strongly differentiated than is customary in type design. Most of the   
principle for designing type for OCR remain same as Latin, while special care need to be taken   
for Devanagari because of the difference in character segmentation. However, many constraints   
which were present while designing OCR-B are not applicable now because of the advancement   
in technology for example previously OCRs were only able to detect monospaced font but now   
because of the development in the OCR system it can also recognize proportional fonts with accu-  
racy. Some of the things that should be taken care while designing type for OCRs are as follows

**One Character should Never be Contained in Another Character**

No character when overlapped with another should be completely inside the other letter. This is very important for correct recognition. To do this certain additional feature or elements are added to differentiate it from the other characters as shown in figure 5.1. We can also have different counter size of similar looking characters like प and ष.

Figure 6.1 Addition of elements in OCR-B to differentiate two characters

**Font should be Monolinear**

Monolinear fonts are the fonts that have same visual weight of the vertical and horizontal strokes.   
If a font has different stroke width then there is a possibility of the breaking of the thin stroke at   
small point size while scanning or during the process of binarization thus creating problems while   
recognizing.

**Font should be Sans Serif**

Serif is a small decorative line added at the end of some of the strokes that make up thee basic form   
of a character as shown in figure 5.2. A typeface with serifs is called a serif and a typeface without   
serifs is called sans serif. Sans serif typefaces are preferred for OCR because serifs increases the   
common coverage area of the characters therefore increasing the similarity between characters.



Figure 6.2 Serifs in a typeface (grey serifs)

**Generous Character Spacing**

Character spacing is the distance between two characters. White space between two characters help in character segmentation but it should not be comparable to the space bar (' '). If the character spacing is not enough, the characters can end up touching each other because of the noise added while scanning; then this would create problem in character segmentation.

Shadow characters should also be avoided. A character is said to be under the shadow of another character if they do not physically touch each other but it is not possible to separate them merely by drawing a vertical line. Example of shadow characters is shown in figure 5.3.

Although the algorithm takes care of shadow characters, it reduces the accuracy in some cases.

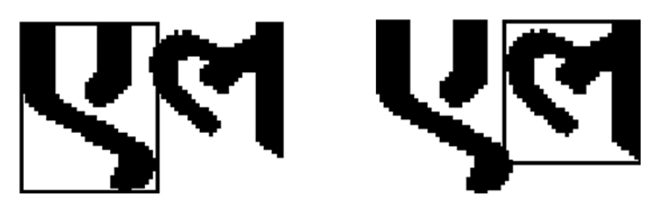


Figure 6.3 Shadow Characters

**Big Closed Counters**

The enclosed or partially enclosed circular or curved negative space (white space) of some letters such as d, o, and s is the counter as shown in figure 5.4.



Figure 6.4 Counters is the circular negative space (grey)

While designing for OCRs, counter size needs to be kept huge so that they don't get completely filled because of noise while scanning or they can also get filled while printing. This can result in faulty recognition as a character can be confused for other characters, for example if the counter of ढ is filled it can be confused for द by the OCR.

**Bold Strokes**

Stroke width is another feature of a font which is very important for recognition as thin strokes can get smudged and get broken because of poor quality of printing and scanning. Bold stroke is also helpful in the process of binarization.

**5.2 Special Care for Devanagari**

Apart from the precautions stated above some special care has to be taken for Devanagari because of the complicated segmentation process. For character segmentation the script is divided in three parts: top, core (or middle) and bottom and all these parts are recognized separately. This increases the complication because unlike Latin script, descenders and ascenders of the characters (in core strip) won't be treated as the part of the character in Devanagari script. So no differentiating feature can be present in the ascender or descender of the character. These special precautions that need to be taken care of are discussed below.

**Removal of Shiro Rekha and the Top Strip**

Removal of Shiro Rekha is the second step in character segmentation (as shown in figure 3.7). When Shiro Rekha is removed, all the features of the character at the level of Shiro Rekha or above it are also removed from the core strip as shown in figure 5.5.



Figure 6.5 Example of few characters after the removal of Shiro Rekha

When some important features of the character are at the level of Shiro Rekha or above it gets removed resulting in no recognition or recognizing a different character. For example भ has a curve at the level of Shiro Rekha which when removed results in looking like म. Similarly ध looks like घ when the Shiro Rekha is removed. This can be seen in figure 5.6



Figure 6.6 Similarities between different characters once the Shiro Rekha is removed

Also the differentiating characteristic between the kana (ा) and purna viraam (।) is the presence of Shiro   
Rekha above the kana. Once the Shiro Rekha is removed there is no differentiating features between   
theses two characters and one character can be confused for other. So while designing some differen-  
tiating features have to be added in either of two characters so that they can be recognized accurately.

**Removal of Bottom Strip**

The step after the removal of top strip in character segmentation is the removal of bottom strip. Bottom strip is the strip which contains the lower matras, halanta and descenders of the letters in the core strip. The most difficult part of this step is to determine where the core strip ends and the bottom strip begin because in Devanagari script the lower matras are connected to the characters in the core strip.

Also a few characters like इ झ has characteristic features extending to the bottom strip. When these   
features are removed the character might closely resemble other characters as shown in figure 5.7.

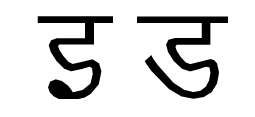


Figure 6.7 Similarities between different characters once the bottom strip is removed

Also in some cases the descender resembles a particular lower matra or a diacritical mark. While recognizing the lower matras in the bottom strip, the descender can be confused for the lower matra which would result in incorrect recognition of both the character and the lower matra as shown in figure 5.8.

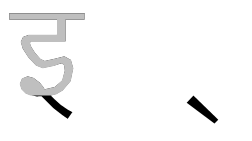


Figure 6.8 Similarities between descender and Halanta

**Recognition of Characters**

Recognition of characters is much more complicated in Devanagari than in Latin because of the graphical similarities in the letters. The graphical similarities in the letters in Devanagari is much more than that in Latin. Some of the letters have just a difference of a stroke like ष just has an additional diagonal stroke as compared to प. While there are others which differ from each other only because of the presence of vertical line like न and म.

Also unlike Latin script, Devanagari has letters which are disjoint horizontally. This should be avoided in the characters in which this can be avoided for example रव can also be designed as ख. This results in inaccurate recognition.

Also the open counters in the letters should be designed carefully. Open counter is the curved part of the character that encloses curved parts (counter) of some letters as shown in figure 5.9.



Figure 6.9 Example of open counter (grey)

While designing the counters, special care need to be taken so that the strokes forming these curves don't get connected because of noise or smudging. This results in the algorithm to confuse between two letters. For example if the strokes of ल connects together they can be recognized as न.

**7.0 Proposed System :**

Our system will recognize the Devanagari Script Characters from the scanned digital image . Instead of giving the direct input of digital image to the OCR Engine system will preprocess the image. This Preprocessed image is then Segmented . System will extract the features from each and every segmented part of the image . After Feature Extraction the processed data is then passed towards the OCR Engine which will perform the Character Recognition.

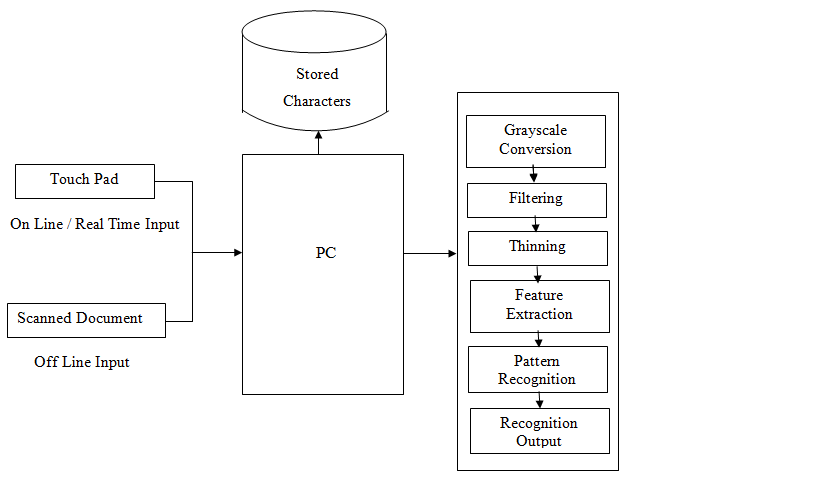


Figure 7.0 Proposed OCR System

The most important principle of automatic pattern recognition is training the machine what kind of pattern may be present and what they look like. In OCR the patterns are letters, numbers and punctuations. Machine is trained to recognize the pattern by showing it all the kind of characters present in the script. This period is referred as the training period. On the basis of these examples the machine builts a prototype of all the characters. Then during recognition the machine compares the unknown character to the prototype and assigns the character which is the closest match. The four steps in recognition shown in figure 3.2 are as follow:

1. Preprocessing

2. Segmentation

3. Recognition

4. Post Processing

**Preprocessing**

The text document is generally scanned at 300 or 400 DPI. Preprocessing is also done to improve the accuracy of the recognition algorithm. Main steps in preprocessing are noise removal, binarization and skew correction.

Noise Removal or De-Noising

The main sources of noise in the input image are as follows:

• Noise due to the quality of paper on which the printing is done.

• Noise induced due to printing on both sides of paper or the quality of printing

• Noise added due to the scanner source brightness and sensors.

All this noise results in reduction of accuracy of OCR system. As a result of this having a noise   
correction routine in place becomes inevitable. To reduce the amount of noise, image is passed   
through a mean filter; in this filter the intensity of the each pixel is replaced by the average intensity   
of pixels surrounding it. After de-noising the image is subjected to binarization and skew (or tilt)   
correction.

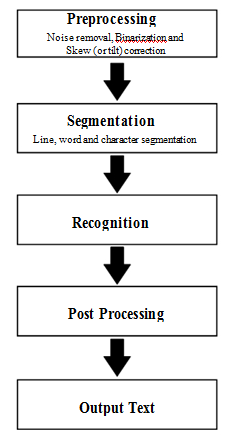


Figure 7.2 Procedure of Devanagari Recognition

**Binarization**

Printed documents generally are black text on white background. Hence most of the OCR algo-  
rithms are designed to interpret bi-level images (an image that has only two possible value of pixel

i.e. black and white). This process of converting colored or grayscale images to bi-level image is often known as binarization or thresholding



Figure 7.3 Image before binarization (left); Image after binarization (right)

**Segmentation**

Segmentation is the process of the dividing the page into its constituent element. The aim of segmentation is to extract out all the character from the text in the image. This is needed to recognize these characters.

Segmentation phase is a very crucial stage since this is where most of the errors occur. Even in good quality documents, sometimes adjacent characters touch each other due to inappropriate scanning resolution or the design of characters. This can create problems in segmentation. Incorrect segmentation leads to incorrect recognition. Segmentation phase includes line, word and character segmentation. Segmentation in OCR occurs in three steps: line segmentation, word segmentation and character segmentation. While the precise algorithm for segmentation can be found in [6] and [7], an overview of segmentation process is given below.

Line Segmentation

In line segmentation our aim is to separate out the line of text from the image. For this global horizontal projection profile method is used which constructs a histogram of all the black pixels in every row as shown in figure 3.4. Based on the peak/valley points of the histogram, individual lines are separated. The steps for line segmentation are as follow:

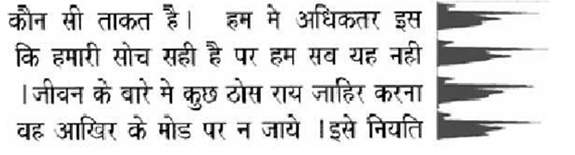
1. Horizontal projection profile for the image is created.

2. Using the projection profile, the points from which the line starts and ends are found.

3. For a line of text, upper line is drawn at a point where we start finding black pixels and lower   
 line is drawn where we start finding absence of black pixels. And the process continues for next   
 line and so on.

Word Segmentation

After line segmentation the boundary of the line (i.e. the top and bottom of the line) is known. Word

 Figure 7.4 Horizontal Projection Profiles of a document for line segmentation[8]

segmentation is extracting out the boundary of the words from these segmented lines. Word seg-  
mentation is done in the same way as line segmentation but in place of horizontal profiling, vertical   
projection profiling is done as shown in figure 3.5. The steps for line segmentation are as follow:

1. Vertical projection profile for the image is created.

2. Using the projection profile, points from which the word starts and ends are found.

3. Then we create vertical lines at the start and end of each line. And the process continues for   
 next word and so on.



Figure 7.5 Vertical Projection Profiles of a document for word segmentation[8]

**Character Segmentation**

Once the words are segmented, the next step is to extract out the characters form these words. A word in Devanagari script is further divided into three parts: as shown in figure 3.6:

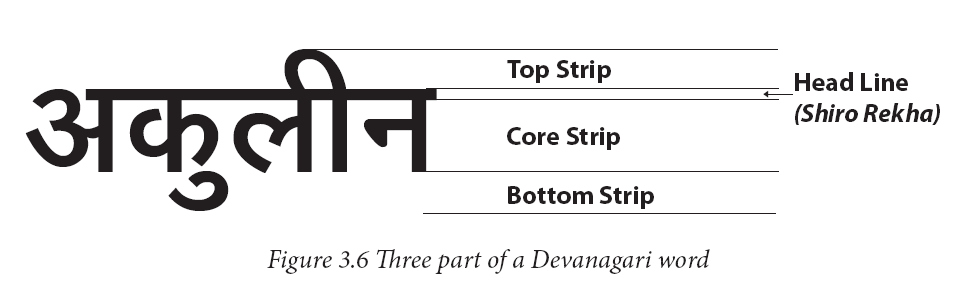
1. Top

2. Core (or Middle)

3. Bottom

The top strip and the core part are separated by the Header Line or the Shiro Rekha. But there is   
no separation between the core strip and the bottom strip. The top strip contains the top matras   
and the bottom strip contains the bottom matras or the descenders of some on the characters. The

Shiro Rekha is a unique feature of Devanagari script and helps to identify Devanagari in multilingual document. It also helps in the identification of the baseline of the text.



The steps of character segmentation shown in figure 3.7 are as follows:

1. Shiro Rekha is identified and the top tip is separated from the core and bottom strip. So now the text is divided in two parts

1. The shiro Rekha and tha top matra
2. The core-bottom part of the text

2. Core strip and bottom strip from the core-bottom part of the text, is identified and lower   
 matras are extracted.

3. Core strip is segmented into different letters or characters which may include conjuncts, punc-  
 tuation or numerals.

4. Conjuncts are segmented into single characters.

5. Shiro Rekha is removed form the extracted top strip and top matras are extracted.

6. Once the segmentation of the core character is done, Shiro Rekha is put back on the top of indi-  
 vidual characters.

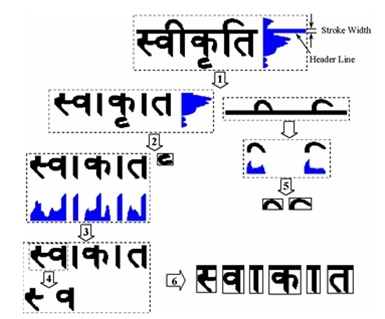


Figure 7.7 The procedure of Hindi character segmentation[6]

## 8.0 Methodology :

## For OCR, after performing the Feature Extraction we need to train and rin oue Neural Network.

### What is an Artificial Neural Network?

An Artificial Neural Network (ANN) is an information-processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurones) working in unison to solve specific problems. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurones. This is true of ANNs as well.

### Why use neural networks?

Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A trained neural network can be thought of as an "expert" in the category of information it has been given to analyse. This expert can then be used to provide projections given new situations of interest and answer "what if" questions.  
Other advantages include:

1. Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience.
2. Self-Organisation: An ANN can create its own organisation or representation of the information it receives during learning time.
3. Real Time Operation: ANN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which take advantage of this capability.
4. Fault Tolerance via Redundant Information Coding: Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage.

### Neural networks versus conventional computers

Neural networks take a different approach to problem solving than that of conventional computers. Conventional computers use an algorithmic approach i.e. the computer follows a set of instructions in order to solve a problem. Unless the specific steps that the computer needs to follow are known the computer cannot solve the problem. That restricts the problem solving capability of conventional computers to problems that we already understand and know how to solve. But computers would be so much more useful if they could do things that we don't exactly know how to do.

Neural networks process information in a similar way the human brain does. The network is composed of a large number of highly interconnected processing elements(neurones) working in parallel to solve a specific problem. Neural networks learn by example. They cannot be programmed to perform a specific task. The examples must be selected carefully otherwise useful time is wasted or even worse the network might be functioning incorrectly. The disadvantage is that because the network finds out how to solve the problem by itself, its operation can be unpredictable.

On the other hand, conventional computers use a cognitive approach to problem solving; the way the problem is to solved must be known and stated in small unambiguous instructions. These instructions are then converted to a high level language program and then into machine code that the computer can understand. These machines are totally predictable; if anything goes wrong is due to a software or hardware fault.

Neural networks and conventional algorithmic computers are not in competition but complement each other. There are tasks are more suited to an algorithmic approach like arithmetic operations and tasks that are more suited to neural networks. Even more, a large number of tasks, require systems that use a combination of the two approaches (normally a conventional computer is used to supervise the neural network) in order to perform at maximum efficiency.

These features of Neural Network motivates us for carrying out character recognition using Neural network architectures.

**Training Algorithms for Neural Network**

Once a network has been structured for a particular application, that network is ready to be trained. To start this process the initial weights are chosen randomly. Then, the training, or learning, begins.

There are two approaches to training - supervised and unsupervised. Supervised training involves a mechanism of providing the network with the desired output either by manually "grading" the network's performance or by providing the desired outputs with the inputs. Unsupervised training is where the network has to make sense of the inputs without outside help.

The vast bulk of networks utilize supervised training. Unsupervised training is used to perform some initial characterization on inputs. However, in the full blown sense of being truly self learning, it is still just a shining promise that is not fully understood, does not completely work, and thus is relegated to the lab.

### Supervised Training.

In supervised training, both the inputs and the outputs are provided. The network then processes the inputs and compares its resulting outputs against the desired outputs. Errors are then propagated back through the system, causing the system to adjust the weights which control the network. This process occurs over and over as the weights are continually tweaked. The set of data which enables the training is called the "training set." During the training of a network the same set of data is processed many times as the connection weights are ever refined.

The current commercial network development packages provide tools to monitor how well an artificial neural network is converging on the ability to predict the right answer. These tools allow the training process to go on for days, stopping only when the system reaches some statistically desired point, or accuracy. However, some networks never learn. This could be because the input data does not contain the specific information from which the desired output is derived. Networks also don't converge if there is not enough data to enable complete learning. Ideally, there should be enough data so that part of the data can be held back as a test. Many layered networks with multiple nodes are capable of memorizing data. To monitor the network to determine if the system is simply memorizing its data in some nonsignificant way, supervised training needs to hold back a set of data to be used to test the system after it has undergone its training. (Note: memorization is avoided by not having too many processing elements.)

If a network simply can't solve the problem, the designer then has to review the input and outputs, the number of layers, the number of elements per layer, the connections between the layers, the summation, transfer, and training functions, and even the initial weights themselves. Those changes required to create a successful network constitute a process wherein the "art" of neural networking occurs.

Another part of the designer's creativity governs the rules of training. There are many laws (algorithms) used to implement the adaptive feedback required to adjust the weights during training. The most common technique is backward-error propagation, more commonly known as back-propagation. These various learning techniques are explored in greater depth later in this report.

Yet, training is not just a technique. It involves a "feel," and conscious analysis, to insure that the network is not overtrained. Initially, an artificial neural network configures itself with the general statistical trends of the data. Later, it continues to "learn" about other aspects of the data which may be spurious from a general viewpoint.

When finally the system has been correctly trained, and no further learning is needed, the weights can, if desired, be "frozen." In some systems this finalized network is then turned into hardware so that it can be fast. Other systems don't lock themselves in but continue to learn while in production use.

### Unsupervised, or Adaptive Training.

The other type of training is called unsupervised training. In unsupervised training, the network is provided with inputs but not with desired outputs. The system itself must then decide what features it will use to group the input data. This is often referred to as self-organization or adaption.

At the present time, unsupervised learning is not well understood. This adaption to the environment is the promise, which would enable science fiction types of robots to continually learn on their own as they encounter new situations and new environments. Life is filled with situations where exact training sets do not exist. Some of these situations involve military action where new combat techniques and new weapons might be encountered. Because of this unexpected aspect to life and the human desire to be prepared, there continues to be research into, and hope for, this field. Yet, at the present time, the vast bulk of neural network work is in systems with supervised learning. Supervised learning is achieving results.

One of the leading researchers into unsupervised learning is Tuevo Kohonen, an electrical engineer at the Helsinki University of Technology. He has developed a self-organizing network, sometimes called an auto-associator, that learns without the benefit of knowing the right answer. It is an unusual looking network in that it contains one single layer with many connections. The weights for those connections have to be initialized and the inputs have to be normalized. The neurons are set up to compete in a winner-take-all fashion.

Kohonen continues his research into networks that are structured differently than standard, feedforward, back-propagation approaches. Kohonen's work deals with the grouping of neurons into fields. Neurons within a field are "topologically ordered." Topology is a branch of mathematics that studies how to map from one space to another without changing the geometric configuration. The three-dimensional groupings often found in mammalian brains are an example of topological ordering.

Kohonen has pointed out that the lack of topology in neural network models make today's neural networks just simple abstractions of the real neural networks within the brain. As this research continues, more powerful self-learning networks may become possible.

## Backpropagation Learning Algorithm

The backpropagation algorithm trains a given feed-forward multilayer neural network for a given set of input patterns with known classifications. When each entry of the sample set is presented to the network, the network examines its output response to the sample input pattern. The output response is then compared to the known and desired output and the error value is calculated. Based on the error, the connection weights are adjusted. The backpropagation algorithm is based on Widrow-Hoff delta learning rule in which the weight adjustment is done through mean square error of the output response to the sample input. The set of these sample patterns are repeatedly presented to the network until the error value is minimized.

Refer to the figure 2 below that illustrates the backpropagation multilayer network with $ M$layers. $ N_j$represents the number of neurons in $ j$th layer. Here, the network is presented the $ p$th pattern of training sample set with $ N_0$-dimensional input $ X_{p1}, X_{p2},
... , X_{pN_0}$and $ N_M$-dimensional known output response $ T_{p1}, T_{p2},
... , T_{pN_M}$. The actual response to the input pattern by the network is represented as $ O_{p1}, O_{p2},$$ ... , O_{pN_M}$. Let $ Y_{ji}$be the output from the $ i$th neuron in layer $ j$for $ p$th pattern; $ W_{jik}$be the connection weight from $ k$th neuron in layer $ (j-1)$to $ i$th neuron in layer $ j$; and $ \delta_{ji}$be the error value associated with the $ i$th neuron in layer $ j$.

|  |
| --- |
| \begin{figure} \centerline {\epsfysize=4.0in \epsfbox{./figures/figBackprop.epsi}}\end{figure} |
| **Figure 8.1:** Backpropagation Neural Network |

Steps to follow until error is suitably small

Step 1: Input training vector.  
Step 2: Hidden nodes calculate their outputs.  
Step 3: Output nodes calculate their outputs on the basis of Step 2.  
Step 4: Calculate the differences between the results of Step 3 and targets.  
Step 5: Apply the first part of the training rule using the results of Step 4.  
Step 6: For each hidden node, n, calculate d(n).  
Step 7: Apply the second part of the training rule using the results of Step 6.

Steps 1 through 3 are often called the *forward pass*, and steps 4 through 7 are often called the *backward pass*. Hence, the name: back-propagation.

**9.0 REQUIREMENTS GATHERING AND PLANNING :**

**9.1 REQUIREMENTS ELICITATION**

**9.1.1 USE CASE DIAGRAMS AND DESCRIPTIONS**

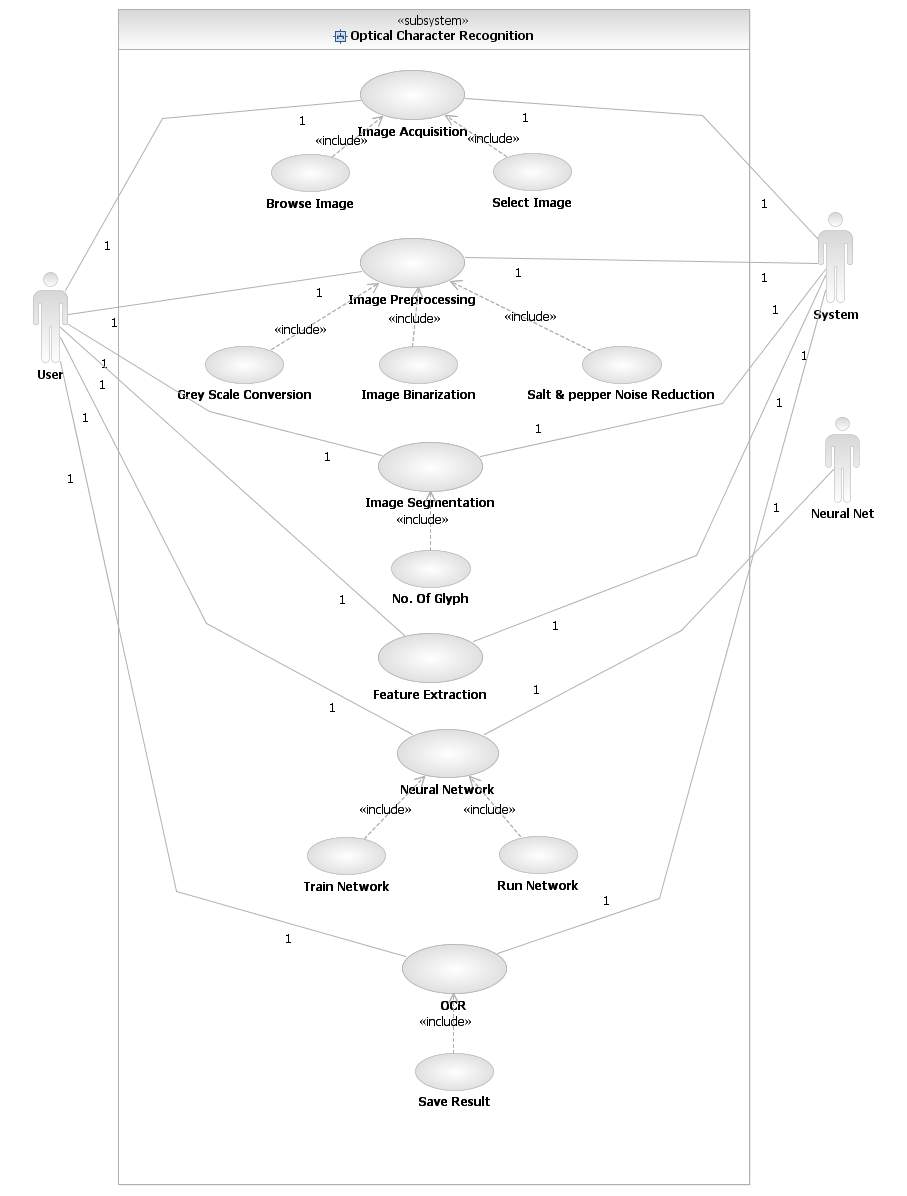


Fig. Use Case Diagram

|  |  |  |
| --- | --- | --- |
| **Actor** | An Actor, as mentioned, is a user of the system, and is depicted using a stick figure. The role of the user is written beneath the icon. Actors are not limited to humans. If a system communicates with another application, and expects input or delivers output, then that application can also be considered an actor. | http://res.dotnetcoders.com/images/uml/usecase-actor.png |
|  | A Use Case is functionality provided by the system, typically described as verb+object (e.g. Register Car, Delete User). Use Cases are depicted with an ellipse. The name of the use case is written within the ellipse. | http://res.dotnetcoders.com/images/uml/usecase-usecase.png |
| **Association** | Associations are used to link Actors with Use Cases, and indicate that an Actor participates in the Use Case in some form. Associations are depicted by a line connecting the Actor and the Use Case. | http://res.dotnetcoders.com/images/uml/usecase-association.png |

**Include**

An include relationship is a relationship between two use cases

http://sourcemaking.com/files/sm/images/uml/img_95.jpg

**Extend**

It indicates that the use case to which the arrow points is included in the use case on the other side of his makes it possible to reuse a use case in another use case.

In UML modelling, you can use an extend relationship to specify that one use case (extension) extends the behaviour of another use case (base). This type of relationship reveals details about a system or application that are typically hidden in a use case.

C:\Users\System Acc\Downloads\24.PNG

**9.2 FEASIBILITY STUDY**

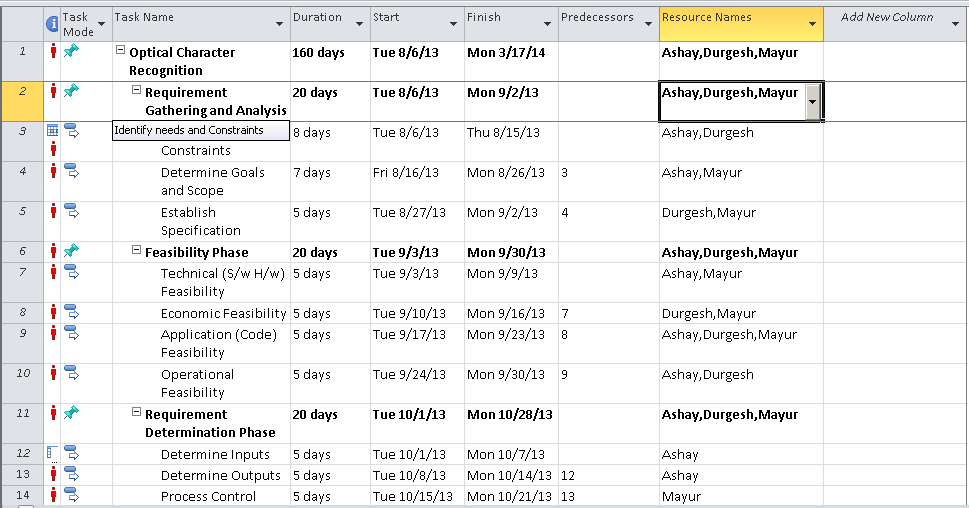
**9.2.1 TECHNICAL FEASIBILITY**

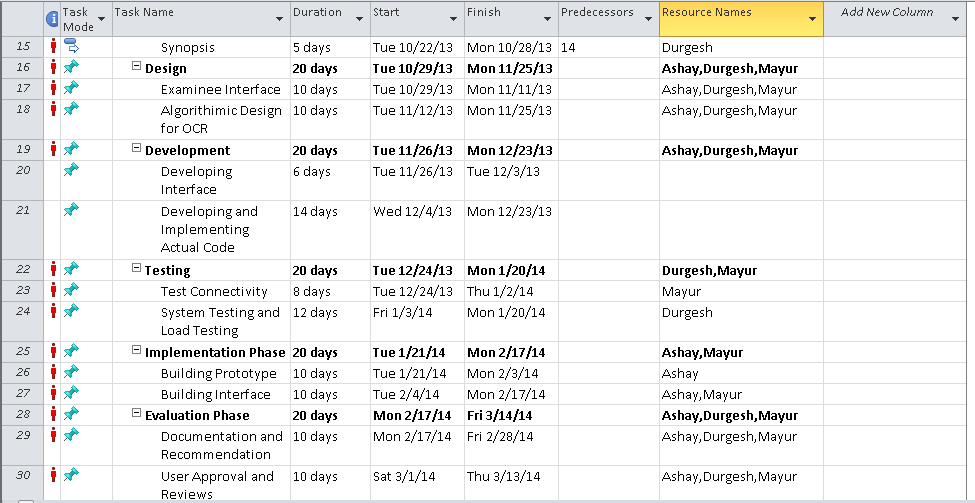
**Hardware Requirements:**

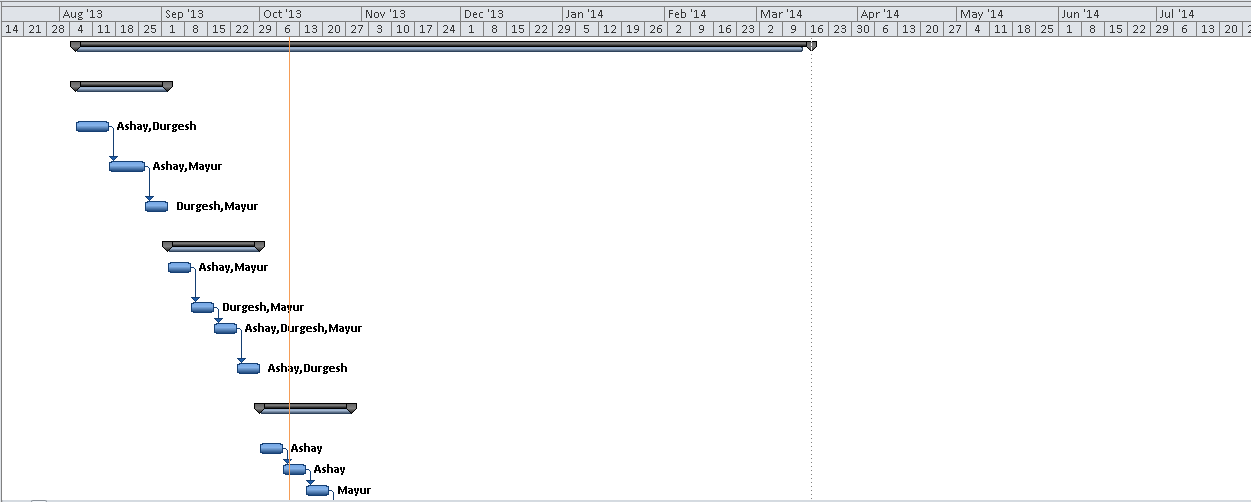
* 40 GB of HDD space
* Minimum 2GB RAM

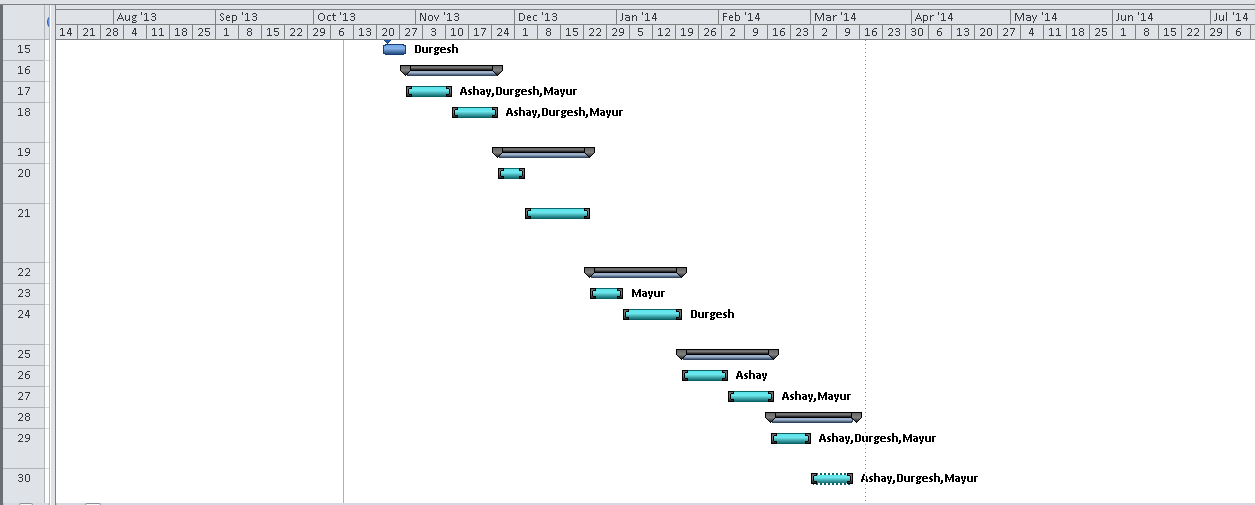
**Software Requirements:**

* Operating System : Window 7/XP
* MATLAB R2012b
  1. **TIME LINE CHART**





****

****

**9.5 W.B.S. CHART**

1.5.7 Implementation and Maintainence

1.5.6 Black Box Testing

1.5.5 White Box Testing

1.5.4 Test Execution

1.4.5 Code Character Recognition Module

1.4.4 Code Feature Extraction Module

1.1.8 Specify Project Goals

1.1.7 UML Diagrams

1.1.6 Character Recognition

1.1.5 Feature Extraction

1.3.4 Development Methodology

1.1.4 Segment Image

1.2.3 Black Book

1.3.3 Design Module in Detail

1.4.3 Code Segment Module

1.5.3 Setup Test Environment

1.1.3 Preprocess Image

1.2.2 Software Requirement Specification

1.3.2 Identify Design Modules

1.4.2 Code Preprocess Module

1.5.2 Create Test Data

1.1.2 Load Image

1.2.1 Synopsis

1.5.1 Preparation of Test Plan

1.4.1 Frontend and Backend

1.3.1 Identify Design Pattern

1.1 Analysis

1.2 Documentation

1.5 Testing

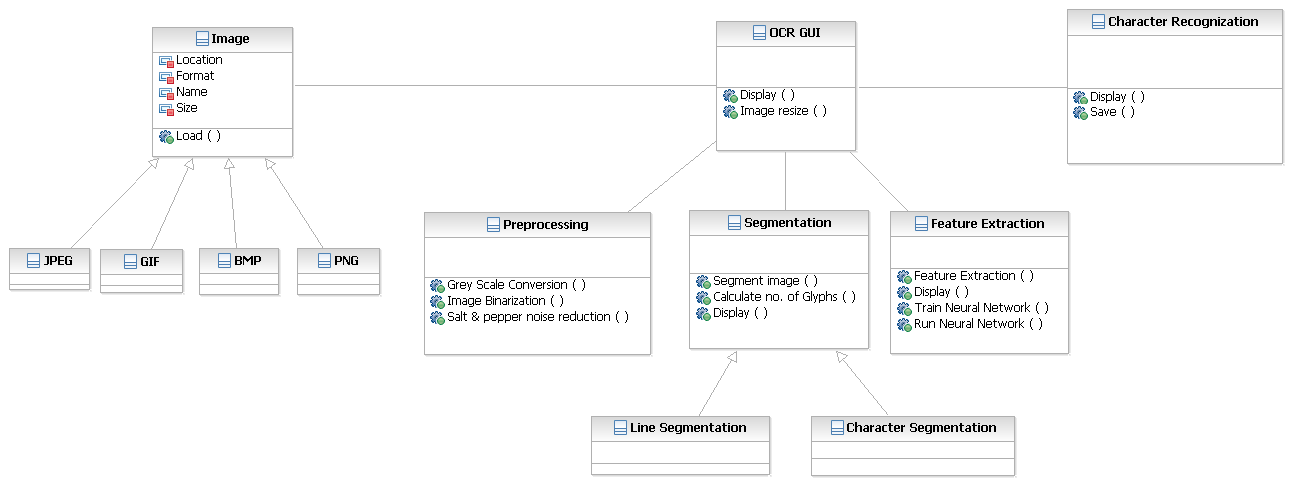
1.4 Coding

1.3 Design

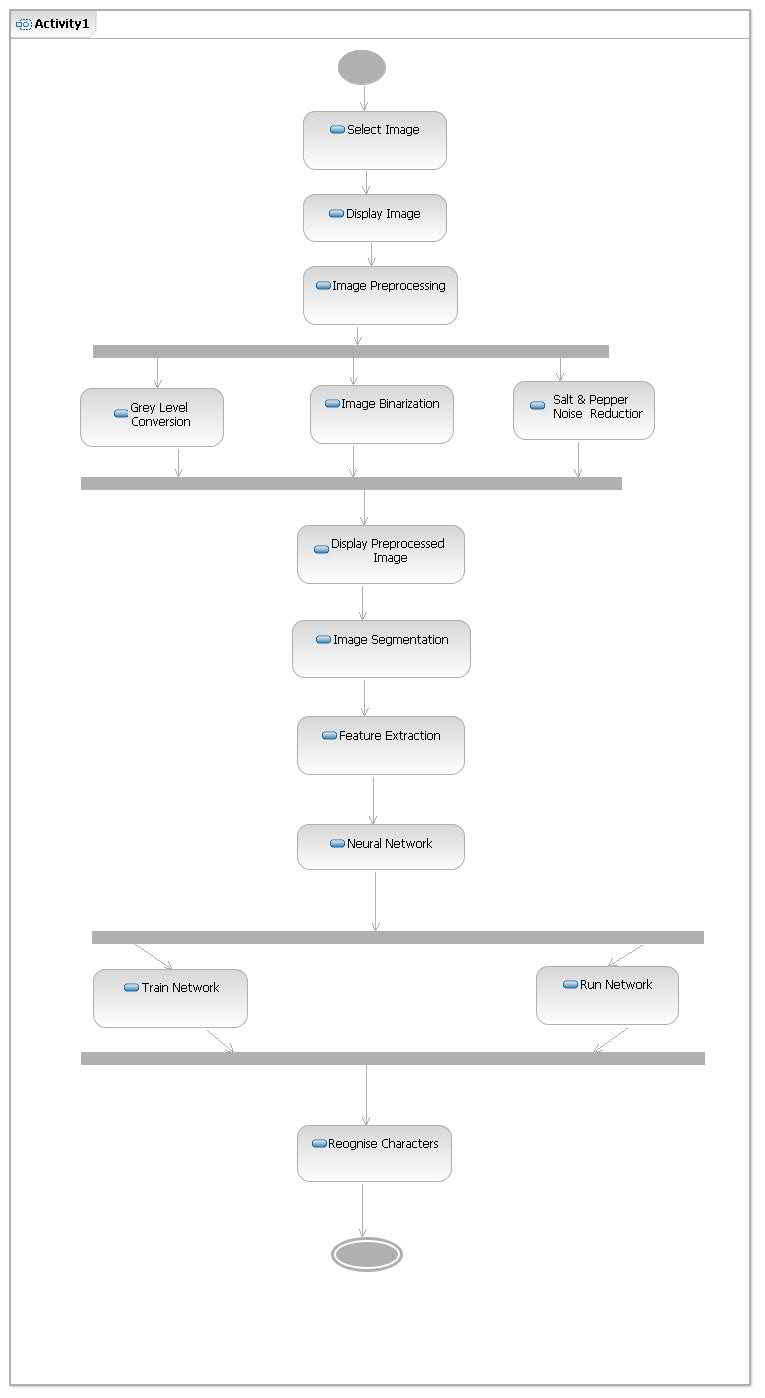
1.Effective Pattern Discovery For

Text Mining

1. **ANALYSIS**
   1. **CLASS DIAGRAM**



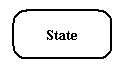
**10.2 ACTIVITY DIAGRAMS**



**COMPONENTS:**

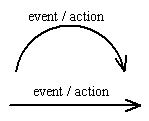
**States**

States represent situations during the life of an object. You can easily illustrate a state in SmartDraw by using a rectangle with rounded corners.



**Transition**

A solid arrow represents the path between different states of an object.Transition with the event that triggered it and the action that results from it.



**Initial State**

A filled circle followed by an arrow represents the object's initial state

Initial State

**Final State**

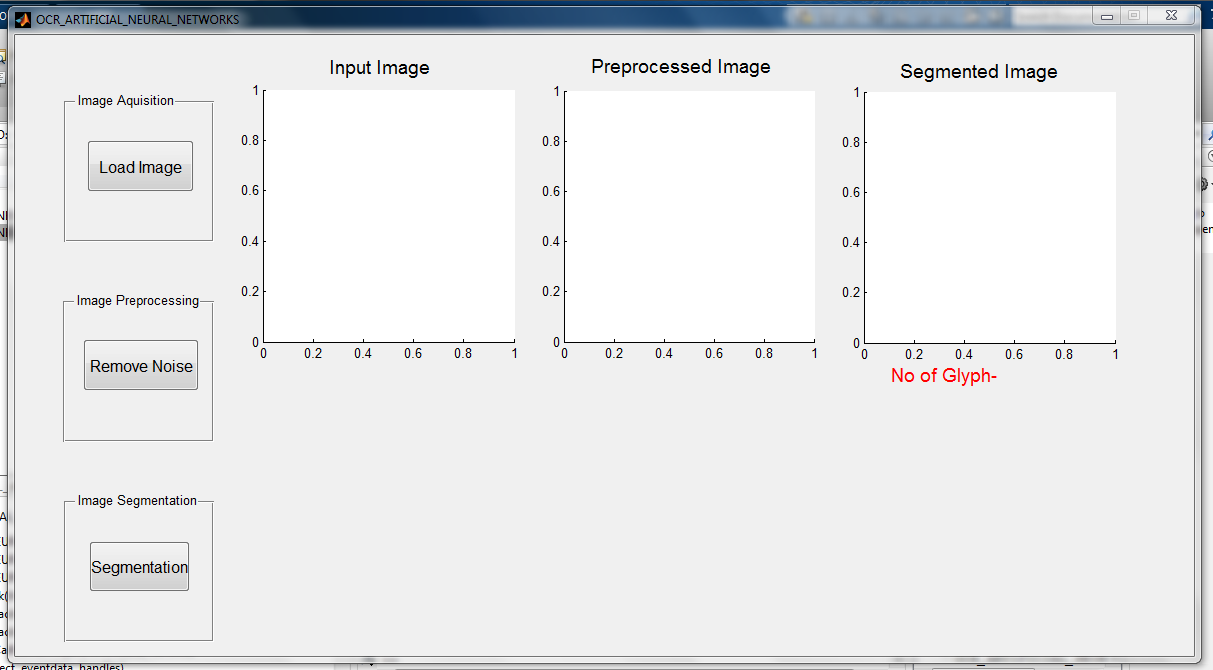
An arrow pointing to a filled circle nested inside another circle represents the object's final state.

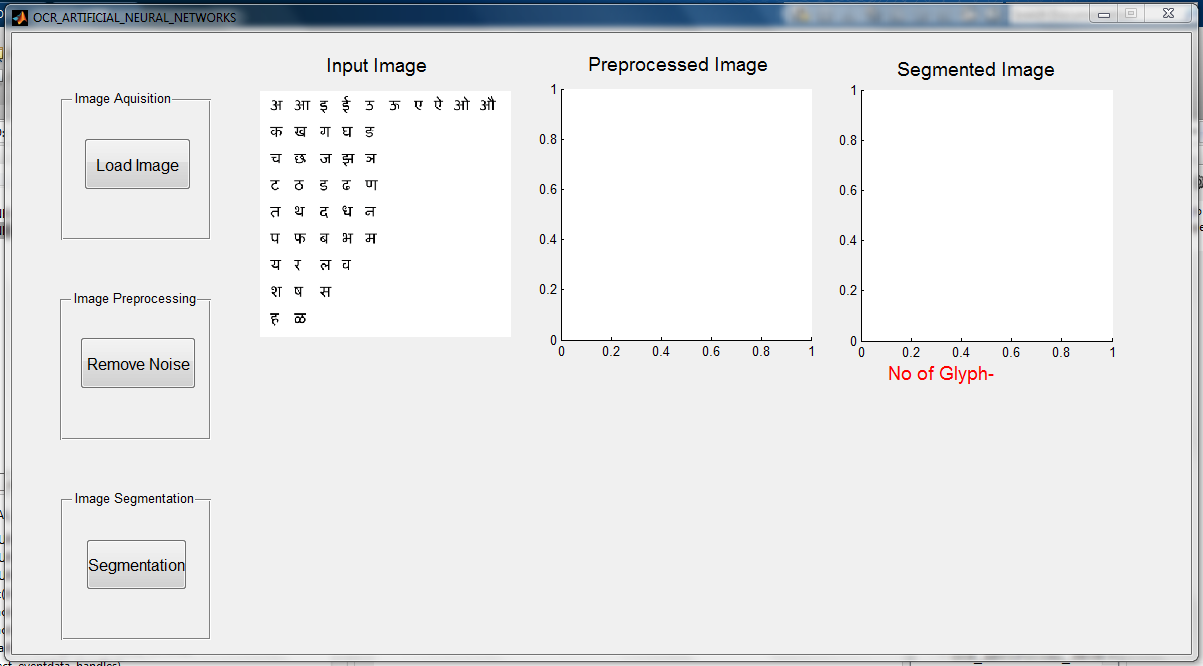
Final State

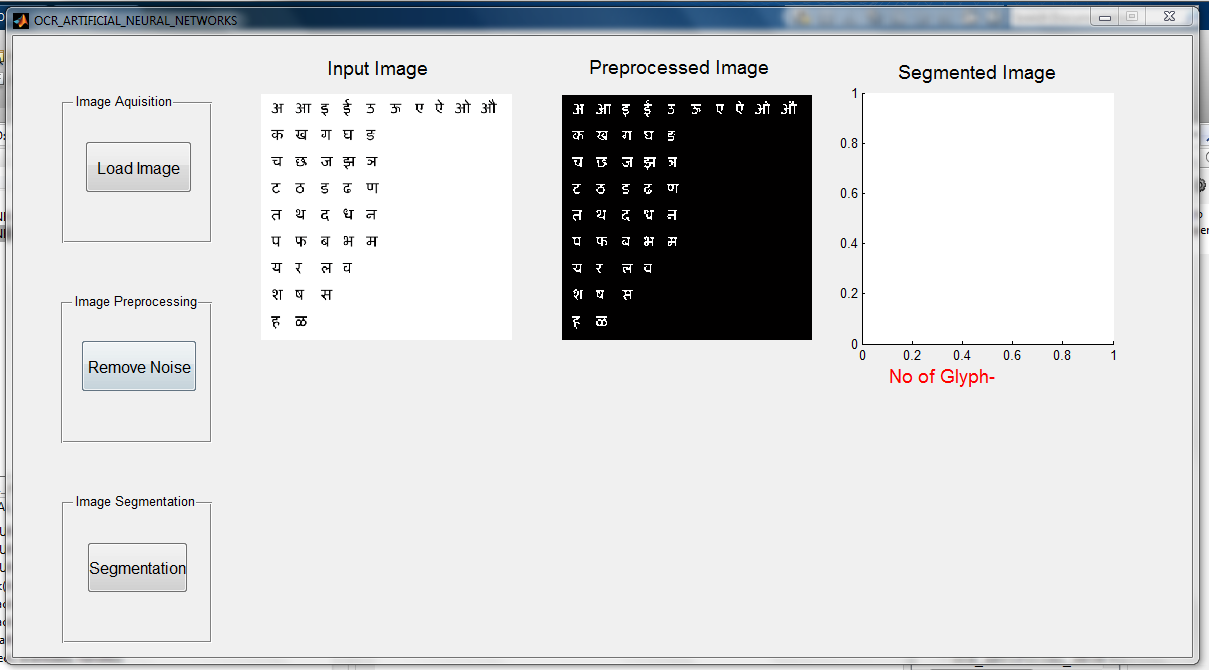
**10.3 Data Flow Diagram**

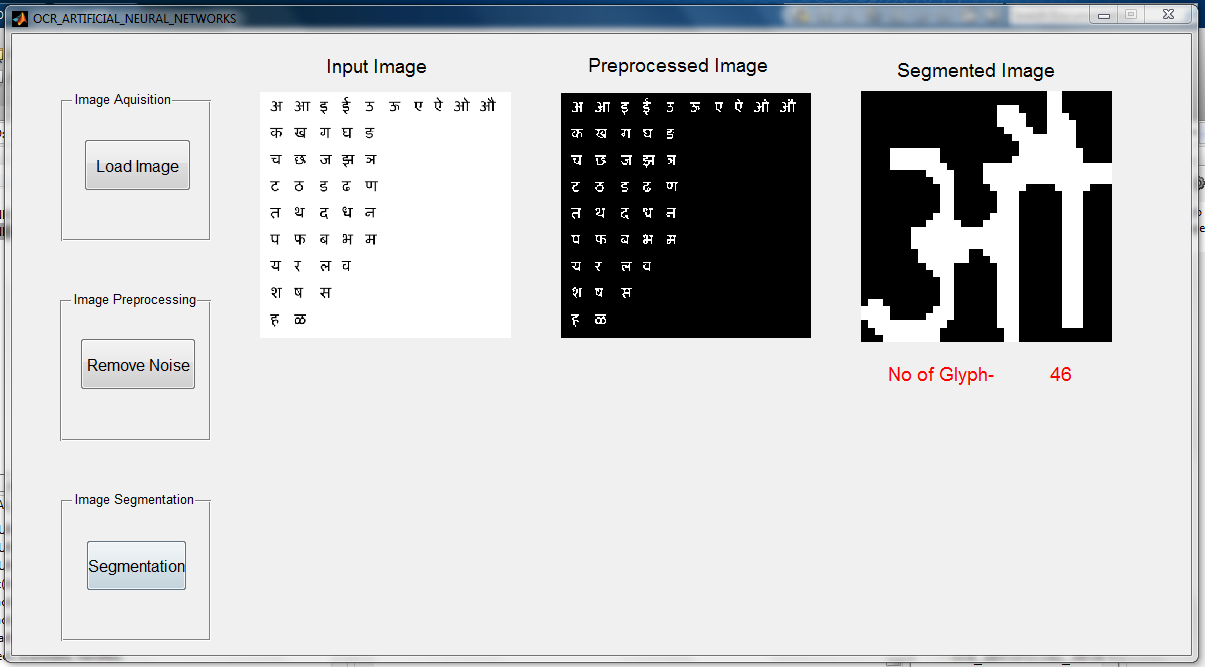
**11. DESIGN**

**11.1 UI DESIGN**









**12.0 Appendix A: Glossary**

**Abbreviations:**

|  |  |
| --- | --- |
| ANN | Artificial Neural Networks |
| ANSI | American National Standards Institute |
| BPM  DPI | Bitmap  Dots Per Inch |
| ECMA | European Computer Manufacturers Association |
| FFNN | Feedforward Neural Network |
| Gb | Giga byte |
| GHZ | Giga Hertz |
| GIF  ISO | Graphic Interchange Format  International Standards Organization |
| JPEG/JPG  MATLAB | Joint Photographic Expert Group  Matrix Laboratory |
| M-files | Matlab Files |
| NN | Neural Networks |
| NNT | Neural Networks Toolbox |
| OCR  OS | Optical Character Recognition  Operating System |
| PDF | Portable Document  Format |
| RTF | Rich Text Format |
| TIFF | Tagged Image File Format |
|  |  |
|  |  |

**13.0 REFERENCES :**

[1] Optical Character Recognition (OCR) for Printed Devnagari Script Using Artificial Neural Network - International Journal of Computer Science & CommunicationVol. 1, No. 1, January-June 2010, pp. 91-95

[2] OCR for Script Identification of Hindi (Devnagari) Numerals using Feature Sub Selection by Means of End-Point with Neuro-Memetic Model by Banashree N. P., and R. Vasanta - International Journal of Electrical and Computer Engineering 2:7 2007

[3] Recognition of Handwritten Devnagari Characters through Segmentation and Artificial neural networks - International Journal of Engineering Research & Technology (IJERT) Vol. 1 Issue 6, August - 2012 ISSN: 2278-0181

[4] Text book on Neural Network for Pattern Recognition by – Christopher M. Bishop