PEHCHAN:

OPTICAL CHARACTER RECOGNITION FOR NOORI NASTALEEQ

FINAL PROJECT REPORT

Project Code:

PGSH03-2004

Project Advisor:

Dr. Sarmad Hussain

Project Manager:

Mr. Masood Ahmad

Project Team:

Shoaib Amjad Khan	[1237]
Mansoor Ahmad	[1223]
Usman Naeem	[1247]

Centre for **R**esearch in **U**rdu **L**anguage **P**rocessing
National **U**niversity of **C**omputer and **E**merging **S**ciences

Table of Contents

Chapter 1

Abstract

Character recognition is one of the oldest fields of research in computer science. And even to date researchers find themselves a long way before they may begin to exhaust the challenges faced in simulating the human ability of perception and recognition. Human understanding of shapes and patterns has associated with it a huge amount of information that is gathered during our interaction with these patterns or shapes. We thus find it quite easy to read different fonts and styles with an immaculate accuracy even when characters are broken or overlapping. It is however, quite another thing to mimic the workings of human perception on computers.

Optical character recognition (OCR) is the process of converting an image of text, such as a scanned paper document or electronic fax file, into computer-editable text [1]. The text in an image is not editable: the letters are made of tiny dots (pixels) that together form a picture of text. During OCR, the software analyzes an image and converts the pictures of the characters to editable text based on the patterns of the pixels in the image. After OCR, the converted text can be exported and used with a variety of word-processing, page layout and spreadsheet applications [2]. One of the main aims of OCR is to emulate the human ability to read at a much faster rate by associating symbolic identities with images of characters. Its potential applications include Screen Readers, Refreshable Braille Displays [3], reading customer filled forms, reading postal address off envelops, archiving and retrieving text etc. OCR’s ultimate goal is to develop a communication interface between the computer and its potential users.

Urdu is the national language of Pakistan. It is a language that is understood by over 300 million people belonging to Pakistan, India and Bangladesh. Due to its historical database of literature, there is definitely a need to devise automatic systems for conversion of this literature into electronic form that may be accessible on the worldwide web.

Although much work has been done in the field of OCR, Urdu and other languages using the Arabic script like Farsi, Urdu and Arabic, have received least attention. This is due in part to a lack of interest in the field and in part to the intricacies of the Arabic script. Owing to this state of indifference, there remains a huge amount of Urdu and Arabic literature unattended and rotting away on some old shelves. The proposed research aims to develop workable solutions to many of the problems faced in realization of an OCR designed specifically for Urdu Noori Nastaleeq Script, which is widely used in Urdu newspapers, governmental documents and books. The underlying processes first isolate and classify ligatures based on certain carefully chosen special, contour and statistical features and eventually recognize them with the aid of Feed-Forward Back Propagation Neural Networks. The input to the system is a monochrome bitmap image file of Urdu text written in Noori Nastaleeq and the output is the equivalent text converted to an editable text file.

Chapter 2

Background & Justification

A Brief History

Current Advancements

OCR Systems for Urdu

2.1. A Brief history of OCR

The engineering attempts at automated recognition of printed characters started prior to World War II. But it was not until the early 1950's that a commercial venture was identified that justified necessary funding for research and development of the technology. The American Bankers Association and the Financial Services Industry provided this impetus. They challenged all the major equipment manufacturers to come up with a "Common Language" to automatically process checks. After the war, check processing had become the single largest paper processing application in the world. Although the banking industry eventually chose Magnetic Ink Recognition (MICR), some vendors had proposed the use of an optical recognition technology. However, OCR was still in its infancy at the time and did not perform as acceptably as MICR. The advantage of MICR was that it is relatively impervious to change, fraudulent alteration and interference from non-MlCR inks. The "eye" of early OCR equipment utilized lights, mirrors, fixed slits for the reflected light to pass through, and a moving disk with additional slits. The reflected image was broken into discrete bits of black and white data, presented to a photo-multiplier tube, and converted to electronic bits.

The "brain's" logic required the presence or absence of "black" or "white" data bits at prescribed intervals. This allowed it to recognize a very limited, specially designed character set. To accomplish this, the units required sophisticated transports for documents to be processed. The documents were required to run at a consistent speed and the printed data had to occur in a fixed location on each and every form.

The next generation of equipment, introduced in the mid to late 1960's, used a cathode ray tube, a pencil of light, and photo-multipliers in a technique called "curve following". These systems offered more flexibility in both the location of the data and the font or design of the characters that could be read. It was this technique that introduced the concept that handwritten characters could be automatically read, particularly if certain constraints were utilized. This technology also introduced the concept of blue, non-reading inks as the system was sensitive to the ultraviolet spectrum.

The third generation of recognition devices, introduced in the early 1970's, consisted of photo-diode arrays. These tiny little sensors were aligned in an array so the reflected image of a document would pass by at a prescribed speed. These devices were most sensitive in the infra-red portion of the visual spectrum so "red" inks were used as non-reading inks. That brings us to this generation of hardware.

In 1951 M. Sheppard invented GISMO-a Robert, which gave birth to the modern OCR technology. A prototype machine was developed in 1954, by J. Rainbow, which was able to read uppercase typewritten output at the fantastic speed of one character per minute. This OCR was marketed by quite a few companies like IBM, Inc, Controle Data etc in 1967. After many dramatic developments took place in technology during the late 1960s, the OCR system was considered to be exotic and futuristic. Such systems were being used by government agencies or large corporations only. Today OCR systems are faster, reliable and less expensive with higher accuracy. More fonts can be recognized with OCR systems today. Some systems advertise themselves as omni fonts, which are able to read any machine printed font. Today one PC based OCR system for less than $50.

Handwritten recognition and form reading are among the current research areas in OCR, which are further being explored and worked upon. Reliable recognition of handwritten cursive script is now under intensive investigation. Research is also being conducted in reading forms. This will help formulate an interpretation of the document by using all available information.

2.2. Current Advancements in OCR

The advent of the array method of scanning, coupled with the higher speeds and more compact computing power, has led to the concept of "Image Processing". Image processing does not have to utilize optical recognition to be successful. For example, the ability to change any document to an electronically digitized item may effectively replace microfilm devices. This provides the user a much more convenient method of sorting images compared to handling actual documents or microfilm pictures.

Image processing relies on larger more complex arrays than early third generation OCR scanners. When these image scanners are coupled with OCR logic, they provide an extremely powerful tool for users. Image recognition can be done in an "off-line" mode rather than in "real time" - a tremendous advantage over earlier versions of OCR devices. This allows a much more powerful logic system to work over time and requires less rigorous demands on both the location of the information and the font design of the characters to be scanned. An example of this is found in the coupling of "image with convenience amount recognition" planned for the Financial Services Industry for check processing - still the world's largest paper processing application. This will be the first viable marriage of MICR with optical technology.

2.3. OCR system for URDU

Character recognition is an active area of research with numerous applications including web publishing, document analysis and text to speech conversion. In the past, a lot of research has been done on automatic recognition of text written in languages based on Roman, Chinese text, Arabic and Persian but no serious research has ever been published on Urdu text recognition. Arabic and Persian, which are based on similar basic characters and writing styles as Urdu, have seen quite worthwhile research in the past decade. However, those solutions are not valid to Urdu due to a number of inherent differences in the script and styles of Urdu text.

Little or negligible amount of work has been done in the development of a practical Urdu character recognition system. Fuzzy Logic and Template Matching are two approaches that have been used so far for Urdu OCRs. However, fuzzy logic has only been successful in recognizing just 36 characters of Urdu [4]. Where as template matching is a crude way of character recognition and it is computationally an expensive task. Few efforts have been made to improve the working of the OCR by incorporating Neural Networks, but to no significant avail. The best that has been done is a ligature based OCR that has been designed to recognize Nastaleeq Script at point size 36 and gives an average accuracy of 79%. Out of around 25000 estimated ligatures that have been identified for Urdu [5], the work only considered 3000. These were then isolated into 323 separate classes based on some structural and statistical features [6], but this number is quite significant and greatly compromises the efficiency of Neural Nets [7]. In a nutshell then, only a very limited number of solutions have been proposed to date for an Urdu OCR and none of them provides a workable practical output.

The primary aim of the proposed research then is to probe into the problems faced in developing a practical OCR system for Urdu and seek solutions to them. A practical solution should focus on small font sizes (e.g. 8 or 10) which are normally used for printing and publishing books, newspapers, magazines and governmental documents, as well as handle the difficulties of recognizing formatted text (e.g. division into columns). Moreover, it should cover all the possible ligatures present in Nastaleeq and isolate them into a lesser number of distinct classes so that the efficiency of Neural Nets is not compromised. In order to ensure an accuracy of somewhere near 90%, this isolation into groups or classes should be achieved through a careful study of certain structural and statistical features of the script.

Chapter 3

Problem Statement

3.1. The Problem at Hand

The primary aim of the proposed research is to probe into the problems faced in developing a practical OCR system for Urdu and seek solutions to them. A practical solution should focus on small font sizes (e.g. 8 or 10) which are normally used for printing and publishing books, newspapers, magazines and governmental documents, as well as handle the difficulties of recognizing formatted text (e.g. division into *columns*). Moreover, it should cover all the possible ligatures present in Nastaleeq and isolate them into a lesser number of distinct classes so that the efficiency of Neural Nets is not compromised. In order to ensure an accuracy of somewhere near 90%, this isolation into groups or classes should be achieved through a careful study of certain structural and statistical features of the script.

3.2. The Intrinsic Complexities of Nastaleeq

Nastaleeq Script which is the focus of the current research is indeed one of the most elegant and complex scripts. It is a combination of two different writing styles i.e. 'Naskh' and 'Taaleeq'. It was originally created in Iran by the calligrapher Mir Ali Tabrezi, and has been refined over the past 600 years. Some of the more intricate features of Nastaleeq are:

- Every letter of the alphabet set assumes different shapes in a word depending on the context in which it is used, as shown in Table 1 [12]. This is unlike Naskh and complicates the computerization of this script.

CHARACTER	NUMBER OF SHAPES WITH RESPECT TO POSITION			
	Isolated	Initial	Medial	Final
Bay	1	24	24	2
Jeem	1	18	18	1
Ray	1	-	-	2
Dal	1	-	-	1
Seen	2	19	19	2
Suad	1	19	19	1
Toa	1	18	18	1
Ain	1	18	18	1
Fay	1	18	18	1
Qaf	1	18	18	1
Kaf	1	19	18	1
Lam	1	19	18	1
Meem	1	18	18	2
Noon	1	24	24	1
Vao	1	-	-	1
Hey	1	18	18	1
Hey (do chashmi)	1	18	18	1
Yay (bari)	1	24	24	1
Yay	1	24	24	2

Table 3.1: Number of Shapes for Urdu Characters w.r.t position in Nastaleeq

- Nastaleeq is written diagonally from top to bottom. Due to this property, Nastaleeq does not have a single base line. This script has a problem of having multiple base lines as it is different for all characters within a ligature.

Naskh: درد دل کے واسطے پیدا کیا انسان کو

Nastaleeq: درد دل کے واسطے پیدا کیا انسان کو

- In Nastaleeq, the overlapping problem is seen not only in the characters but in the ligatures as well. The Calligrapher always tries to produce a piece of script that is beautiful and catches the eye. Nastaleeq is written in such a way that the ligatures may also overlap each other without touching.

- Nastaleeq has complex Mark placement rules e.g. for Dots (Nukta's) and diacritics.

- Another problem seen commonly in this calligraphic script is that of the occurrence of multiple shapes for justification purposes. In order to justify the text in a line, the calligrapher tends to stretch or contract characters within a word (Kashida Kari). This feature is seen only in handwritten manuscripts.

3.3. Further Problems at Practical Font Sizes

In addition to the aforesaid intricacies, which are intrinsic to the Nastaleeq way of writing, there arise some very intriguing problems when we descend down to the small font sizes. A detailed treatment of some of these problems at point size 10 now follows:

- Breaking down of ligatures: At resolutions less than or equal to 200 dpi, quite a significant amount of ligatures showed signs of discontinuity and ***the main bodies*** appear considerably distorted. This makes it impossible to separate ligatures using the existing method of ***8-connected components.***

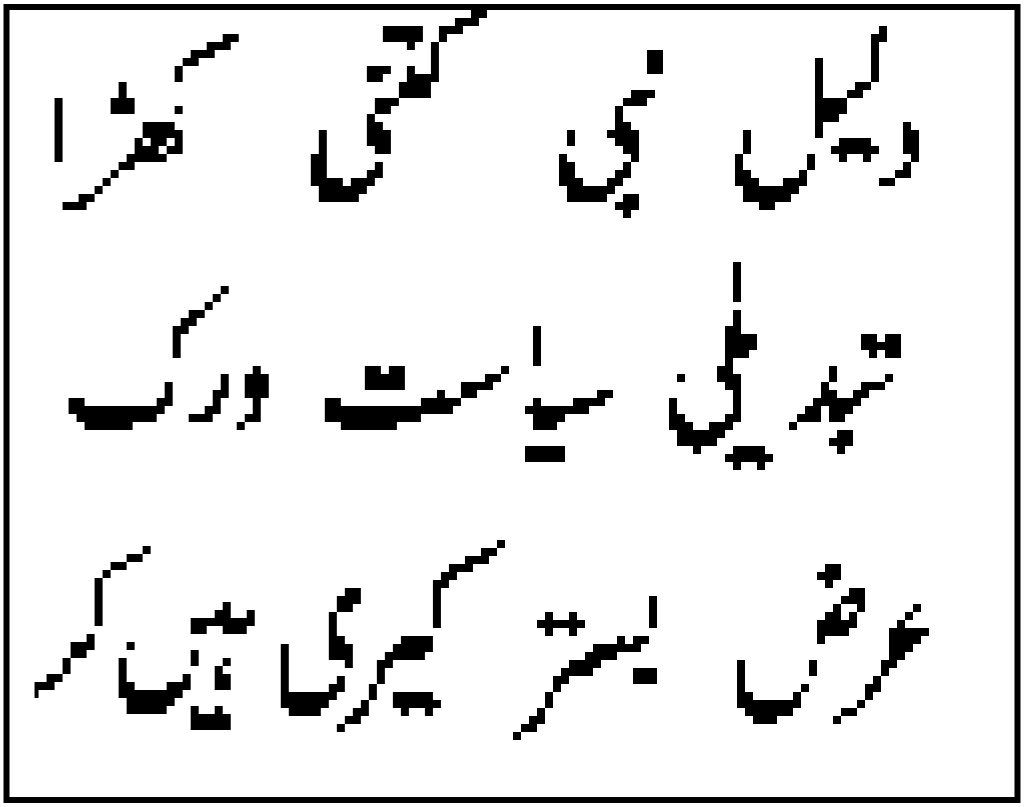


Fig. 3.1. Magnified image to elucidate the random breaking down of ligatures at font size 10 with resolutions <= 200dpi.

- Inability to distinguish between the special features (dots and diacritics) associated with the ligature e.g. the differentiation of one dot from two dots. This in turn leads to extreme difficulties in properly classifying ligatures based on special features.

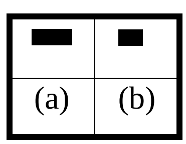


Fig 3.2. (a) is an image of two dots and (b) is that for a single dot at font size 10 with resolutions <= 200dpi.

- Special features merging into main body of the ligatures. The problem gives rise to a dangerously high and variable number of new and unpredictable classifications of ligatures which otherwise do not exist. These classes are normally comprised of only a single token and tend to make the entire process of Character Recognition rather dubious.

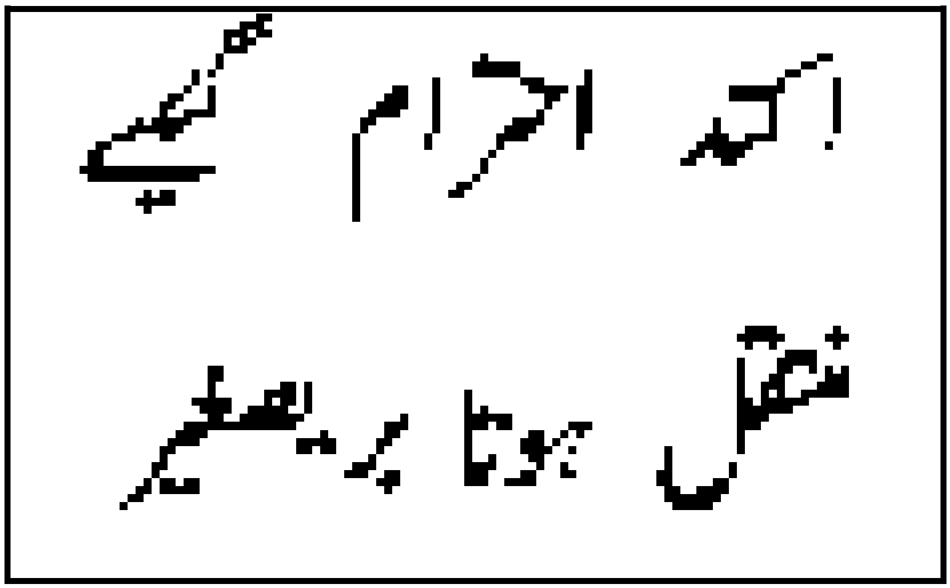


Fig. 3.3. Special features merging into main bodies at font size 10 with resolutions <= 200 dpi

- Special features of adjacent ligatures merging into the main bodies. The problem is as grave as the one mentioned above. It gives rise to a dangerously high and variable number of new and unpredictable classifications of ligatures which otherwise do not exist. These classes are normally comprised of only a single token and tend to make the entire process of Character Recognition rather dubious.

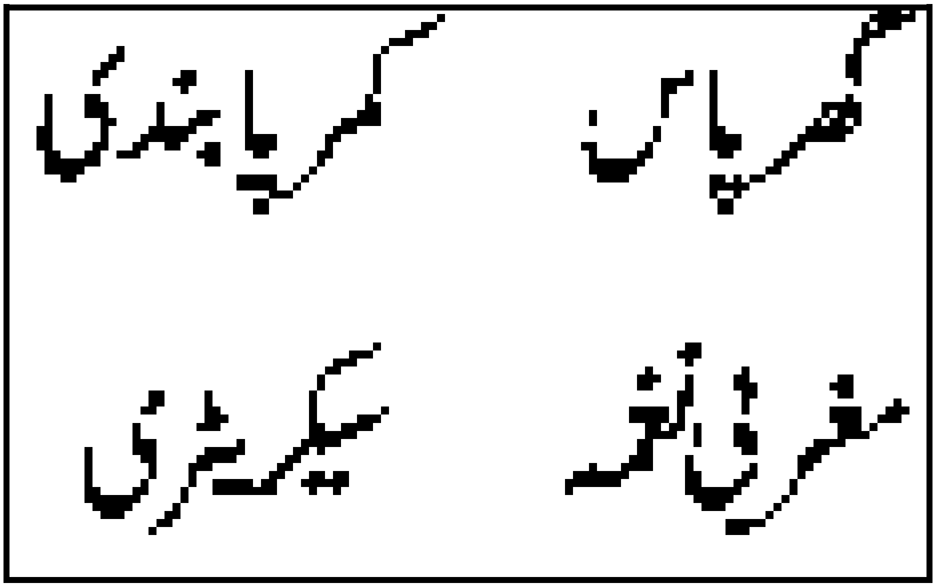


Fig. 3.4. Special features of adjacent ligatures merging into main bodies at font size 10 with resolutions <= 200 dpi

- Main bodies of the ligatures striking and merging with adjacent ligature main bodies, thus degenerating the process of ligature separation and ultimately of ligature classification.

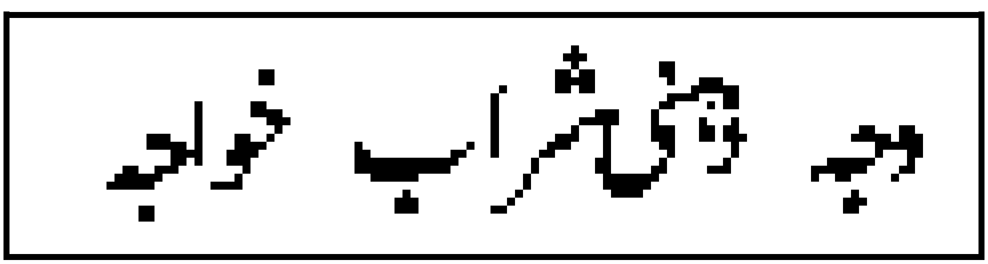


Fig. 3.5. Main bodies of ligatures merging into main bodies of adjacent ligatures at font size 10 with resolutions <= 200 dpi

- ***The above mentioned problems are also observed at resolutions as high as 300 dpi. However, the frequency of such erroneous tokens is a fraction less than that observed at resolutions which are less than or equal to 200 dpi.***

- The degree of distortion of main bodies of ligatures that is observed at resolutions up to 300 dpi is significantly high to effect the eventual calculation of the seven moments and other statistical features. Thus, encumbering the process of recognition, which is achieved through the Neural Networks.

Chapter 4

Literature Review

4.1 A Categorical Overview

After an extensive review of the field of offline cursive word recognition, we have classified the field into three categories [14][15][16]:

i) Segmentation Based Systems: which look for the best match between consecutive sequences of primitive segments and letters of a possible word.

ii) Segmentation-Free Systems: which compare a sequence of observations derived from a word image with similar references of words in the lexicon.

iii) Perception Oriented Method: that relates to methods that perform a human-like reading technique, in which anchor features found all over the word are used to bootstrap a few candidates for a final evaluation phase.

A detailed description now follows.

4.1.1. Segmentation Based Systems

In Segmentation based systems, each word is further divided into a number of subparts. In segmentation-based methods the recognition process is based on an attempt to find the best complete bipartite match between blocks of primitive segments and a word's letters. These primitive segments are created by some segmentation algorithm, which might be imperfect, and therefore cause over- or under-segmentation. We refer to this as a pre-segmentation stage. The segmentation-based systems are further subdivided into four categories:

1. Isolated/Pre-segmented characters.
2. Segmenting a word into characters.
3. Segmenting a word into primitives.
4. Integration of recognition and segmentation.

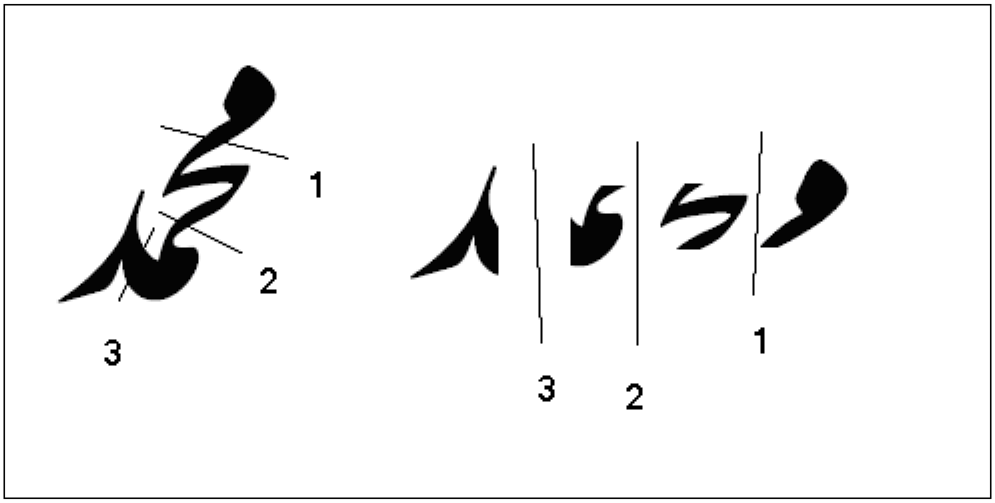


Fig.4.1. Complexities involved in segmentation of Nastaleeq ligatures to primitive characters

These systems are either impractical because they try to recognize digits and isolated characters or they have low recognition rate because of segmentation errors [15]. As you may see in the example above that how complex is it to segment a ligature. This complexity slows down the efficiency of OCR.

Segmentation of merged characters is an old problem. For document text, character touching/merging is either due to font style, font size, xeroxing, etc., or due to scanner's finite resolution or the use of a high Binarization threshold as to avoid breaking characters. Originated by Casey and Nagy [35], conventional approaches to solving merged characters are classification-baaed segmentation [36][37][38][39]. In those schemes, the presence of merged characters is hypothesized by the rejection from OCR; cutting attempts are then made to decompose the pattern, and the segmentation is confirmed by reclassification of the decomposed components.

4.1.2. Segmentation Free Systems

In a segmentation-free method, one should find the best interpretation possible for an observation sequence derived from a word or part of a word (ligature) image without performing a meaningful segmentation first. In these systems, the word or part of a word (ligature) is recognized as a whole without trying to segment and recognize characters or primitives [40]. ***One approach for such systems is to calculate a single feature vector for each word;*** this feature vector is then used to recognize the word.

An observation sequence can be classified into three categories according to the representation level of the word it stands for. The first category relates to observations that are based on low-level features taken directly from the word image. Such features include smoothed traces (quantized/normalized fragments) of the word contour, pieces of strokes between anchor points, edges of a polygonal approximation of the image skeleton, etc. The second category aggregates such low-level features to serve as primitives. For example, neighboring strokes can be merged into a smoothed pattern that will constitute a primitive. The main difference between the current category and the former one is in the nature of the relevant feature space - continuous in contrast with discrete. The last category involves methods that use even higher-level features of a word image. The most popular features are the most irregular, i.e., holistic features that are hard to miss and are invariant with respect to all the different writing styles. In this case holistic means global with respect to a whole word resolution; meaning that features of this kind, such as ascenders, descenders, loops, dots, strokes, etc., are prominent even in an image of a complete word. These features may be sub-classified according to size, location or orientation. These special features are also referred to as symbols.

4.1.3. Perception Oriented method

Since human are able to hand written text with apparent ease, it seems appropriate to base automatic printed text reader on human reading model. Though several models have been proposed to explain the process of mental lexicon access while reading [49] such as Logogen Model [50], the Verification Model [51], and the Interactive Activation model [52], they all rely on printed text. Nevertheless their underlying concepts can be adapted to the recognition of cursive script [53][54][55].

Although it is not as popular as the two former approaches, we find this approach significant as it seems to resemble a good working model, namely the human reading scheme. However, there are only a few methods that prefer this approach, perhaps because of implementation issues. Unlike the two different methods that were discussed earlier, a perception-oriented method does not work sequentially. Sequential recognition methods attempt to match an ordered list of observations/ primitive segments with a word pattern (either specific or variable) from left to right. When using perception oriented methods, efforts are made, in a bottom-up manner, to identify letters anywhere in the observation sequence derived from a word image. Next, a decision procedure is run, resulting in an interpretation which consists of the best non-overlapping set of letters, and the most likely interpretation of possible gaps that were left between them.

In Edelman et al., [56], recognition is based on the alignment of letter prototypes, given an image of a cursive word. We perceive this method to be one of the most successful word recognition methods in general and in perception oriented in particular. The main stages of the recognition process are as follows:

4.1.3.1. Anchor point extraction

This is performed by tracing the vertical and horizontal extrema of the contour and the line endings (including T-junctions).

4.1.3.2. Stroke detection

Strokes are recognized by prototype alignment, using affine transformations computed from anchor-point correspondences.

4.1.3.3. Letter hypothesization

Potential instances of each of the 26 letters are detected. Every instance at this stage has a score that reflects its closeness to the part of the contour with which it is aligned. This score takes into account the distance between the relevant part of the contour and the respective prototype. The amount of distortion undergone by the stroke's prototype and intrusion to forbidden zones defined by each prototype reduce the score.

4.1.3.4. Instance filtering

The previous stage frequently results in the detection of several overlapping instances of the same letter. The set of all detected instances of a letter is filtered to discard those that are clearly superfluous, based on the concept of *domination*. A character instance is said to dominate another one if it has a higher score and the overlap between the two is sufficiently large.

4.1.3.5. Interpretation

At this stage, a best-first search is used to assemble the interpretation string out of the set of all detected letter instances. The algorithm reminds the one used by Bozinovic et al., [57][58][59] - both relate to the famous A^* algorithm. At each step, one expands the most promising node among all those situated along the current expansion frontier. The expansion is performed by a function that returns all one letter continuations to the right of the current string. Each one of the new expanded strings is checked for validity. For example the distance between the additional letter and the rightmost letter of the original string should not exceed a threshold. The entire cycle is repeated a preset number of iterations, after which all strings that reached the right end of the word image are sorted.

4.1.3.6. Application of lexical knowledge

The fast spell-checking function available in the Symbolics Lisp environment is used in order to synthesize the lexical neighborhood and statistical Englishness of a string.

4.2. The Proposed Method: A Segmentation Free System

Any segmentation-free OCR would generally have following stages [13]:

1. Image acquisition
2. Preprocessing
3. Document Decomposition
4. Feature extraction
5. Classification
6. Recognition
7. Post processing

A detailed treatment now follows.

4.2.1. Image acquisition

OCR or Script Recognition depends on the quality of the image that is the quality of text recognition is dramatically improved for the high quality and less noisy images. The image quality depends on the paper quality and scanner performance and quality and backgrounds of the document.

4.2.2. Pre Processing

In preprocessing following functions are performed:

1. Binarization
2. Skew detection and removal
3. Noise removal
4. Normalization (Scaling and Slant Normalization)

4.2.2.1. Binarization

The digitized text images are first converted into two-tone images (black and white). Here the object (black) pixels are represented by 1 and background (white) pixels by 0. The two-tone image generally shows protrusions and dents in the characters as well as isolated object pixels over the background, which is cleaned by a logical smoothing approach [32]. There are two different approaches for Binarization of text images [22]:

1. Binarization using the image histogram
2. The fuzzy C-means clustering algorithm

4.2.2.1.1. Binarization using the Image Histogram

Generally, image Binarization has been accomplished by looking for an optimal threshold, Φ, of the normalized histogram, pixel's distribution, of the image [23][24][25]. A pixel is considered white if the value of the pixel is greater than Φ, otherwise is black. The decision for a pixel to be white or black depends on a "hard" value**.** It has been proved that Binarization using histogram is a practical and reliable technique, but it does not mean that the development of different techniques is not possible [26].

4.2.2.1.2. Binarization using the fuzzy C-means clustering algorithm

Using a fuzzy clustering algorithm to make the Binarization is based on the idea that human vision works more in the way of "soft" limits, membership degrees, than in a "hard" limit, "the closer one looks at a real-world problem, the fuzzier becomes its solution" [27]. The fuzzy c-means algorithm [28][29]. The algorithm uses only local information. Two different functions are used for the defuzzification process [22], maximum and median with the purpose to improve the resulting image. The parameters used to binarize the images are; level of fuzziness m = 1.5, error allowed for the objective function 0.1. In order to get a measurement of comparison it is implemented a binarization technique using the optimal threshold to binarize the image [30].

4.2.2.2. Skew detection and removal

One of the most commonly occurring problems addressed under the heading of preprocessing is that the document to be read is not always placed correctly on a flat-bed scanner. This means that the document may be skewed on the scanner bed, resulting in a skewed image. This skew has a detrimental effect on document analysis, document understanding, and character segmentation and recognition. Consequently, detecting the skew of a document image and correcting it are important issues in realizing a practical document reader. Casual use of the scanner may lead to skew in the document image.

Skew angle is the angle that the text line of the document image makes with the horizontal direction. Skew correction can be achieved by:

1. Estimating the skew angle, and
2. Rotating the image by the skew angle in the opposite direction.

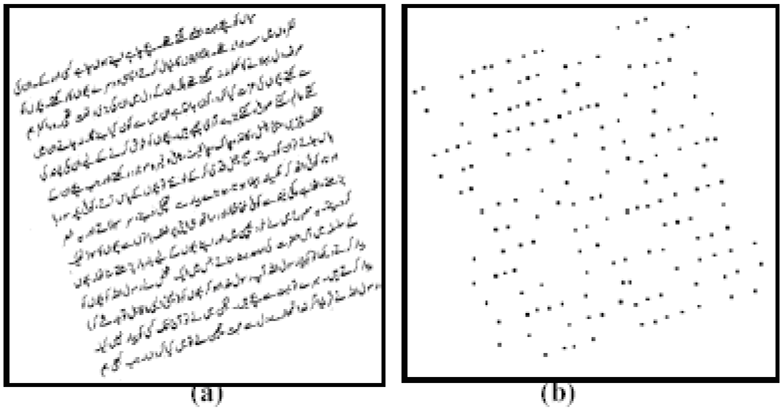
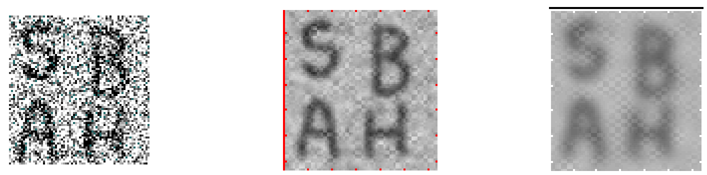


Fig.4.2. (a): Example of Urdu skewed text (b): Candidate points for Hough transform are shown

Hough transform based techniques can be used to estimate skew angle. To reduce the amount of data to be processed by the Hough transform, several candidate points are computed considering some selected components from the image. For component selection, mean width bm of the bounding boxes of the connected components should be computed and components having bounding box width greater than 0.5 × bm should be selected. Thus, small and irrelevant components like dots, punctuation marks, small modifiers, etc. will be mostly filtered out of the skew estimation process.

For skew angle detection, usual Hough transform is used on these candidate points. After skew angle detection the image is rotated according to the detected skew angle. It has been noted that font style and size variations do not affect this skew estimation method. Also, the proposed method can handle documents with skew angle between +45° to - 45°. Experiments show that in about 97.4% of the cases, the proposed method can compute the skew angles with a tolerance of ± 0.5 degree [31].

4.2.2.3. Noise removal & Smoothing



A Noisy image Smoothed Continued smoothing

Fig.4.3. showing the process of Noise removal and smoothening

One goal in image restoration is to remove the noise from the image in such a way that the "original" image is discernible. Of course, "noise" is in the eye of the beholder; removing the "noise" from a Jackson Pollack painting would considerably reduce its value. Nonetheless, one approach is to decide that features that exist on a very small scale in the image are noise, and that removing these while maintaining larger features might help "clean things up".

One well-traveled approach is to smooth the image. The simplest such version is replace each pixel by the average of the neighboring pixel values. If we do this a few times we get the image in the middle above; if we do it many times, we get the image on the right.

On the plus side, much of the spotty noise has been muted out. On the downside, the sharp boundaries that make up the letters have been smeared due to the averaging. While many more sophisticated approaches exist, the goal is the same: to remove the noise, and keep the real image sharp. The trick is to not do too much and to "know when to stop"[43].

At this stage we are assuming that the image provided for OCR is noise free so we have not yet put any extensive effort to remove noise due to time constraints.

4.2.2.4. Normalization

The normalized image is robust to translation, rotation and scaling. This allows recognition of subjects based on their normalized images, since the variance of the normalized images of the same subject is much lower than the variance of the original images of that subject.

Local normalization algorithm uniformizes the local mean and variance of an image. This is especially useful for correct non-uniform illumination or shading artifacts.

The local normalization of $f(x,y)$ is computed as follows [33]:

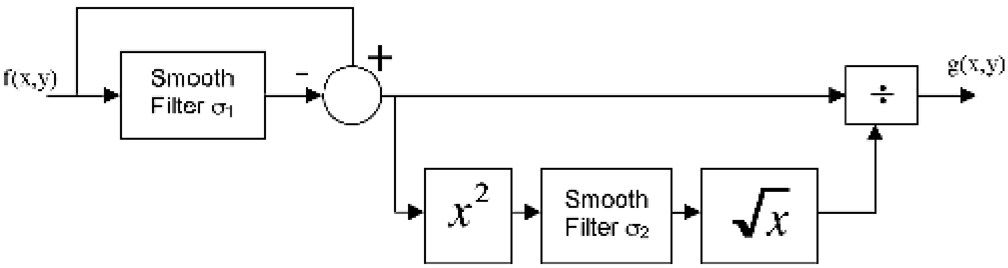
$$g(x,y) = \frac{f(x,y) - m_f(x,y)}{\sigma_f(x,y)}$$

where:

- $f(x,y)$ is the original image
- $m_f(x,y)$ is an estimation of a local mean of $f(x,y)$
- $\sigma_f(x,y)$ is an estimation of the local standard deviation
- $g(x,y)$ is the output image

The estimation of the local mean and standard deviation is performed through spatial smoothing.

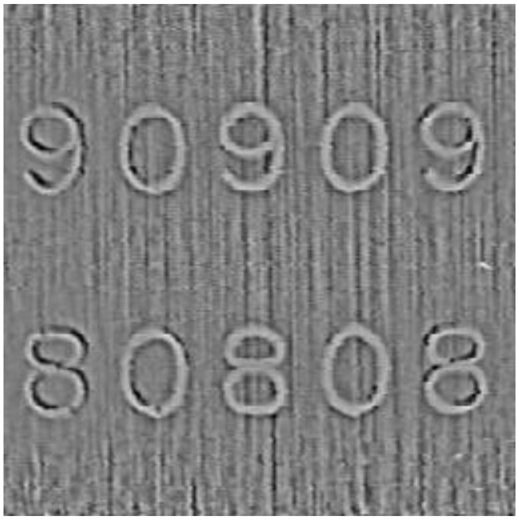
Diagram block



The parameters of the algorithm are the sizes of the smoothing windows, σ_1, and, σ_2, which control the estimation of the local mean and local variance, respectively.

Example

Input image



Output image

Fig.4.4

4.2.3. Document decomposition

Documented decomposition is done in two steps:

1. Line segmentation
2. Ligature separation

4.2.3.1. Line Segmentation

The proposed OCR system automatically detects individual text lines and then segments the ligatures in each line. The lines of a text block are segmented by finding the valleys of the projection profile computed by counting the number of black pixels in each row [22][31]. The trough between two consecutive peaks in this profile denotes the boundary between two text lines. A text line can be found between two consecutive boundary lines. An Urdu text with its projection profile is shown in Fig.6. Line segmentations are shown by dotted lines in this figure.

Fig.4.5. Horizontal projection profile of an Urdu text and its line segmentations are shown

Due to dots and diacritical marks in Urdu Nastaleeq Script horizontal projection may give confusing results, which can be handled using probabilistic calculations.

4.2.3.2. Ligature separation

In document image analysis, four commonly used algorithms are connected components labeling, X-Y tree decomposition, run-length smearing and Hough transform. Considering both its simplicity and efficiency, the proposed OCR implements 8-connected component labeling to image of Urdu text.

The technique assigns to each connected component of binary image a distinct label. The labels are usually natural numbers from 1 to the number of connected components in the input image. The algorithm scans the image from left-to-right and top-to-bottom. On the first line containing black pixels, a unique label is assigned to each contiguous run of black pixels. For each black pixel, the pixels in its eight neighborhoods are examined, if any of these pixels has been labeled the same label is assigned to the current pixel, otherwise a new label is assigned to it. The procedure continues to the bottom of the image [13]

1	2	3
8	A Central pixel	4
7	6	5

Fig.4.6. 1-to-8 neighbours of a pixel

4.2.3.3. Fuzzy Run length

From our analysis of the complex document images we found some general properties of text lines and other graphics. We noticed that for both handwritten and printed text, the relative distances between characters in same lines are generally smaller than the distances between text lines. Human identification of the text lines utilizes these differences efficiently. We can also tolerate touching or connection between text lines by finding a background path between each pair of text lines. The touching or connections between text lines are usually made by oversized ligatures or ligatures with long descendents running through the neighboring lines. Therefore if we could ignore some bridging strokes across lines, we may see the background paths between the lines.

We may use the run length smearing approach for building our background runs. “Smearing” is same as skipping or ignoring some foreground pixels. Hopefully it will break the touching between text lines. But setting up the threshold for skipping the small foreground runs is difficult. If we set the value too small, we may not be able to break the crossing strokes connecting the text lines. If we set the value too big, then it may be bigger than the stroke width of most of the characters and end up erasing the text areas.

To solve the problem a new kind of run length is proposed – fuzzy run length [34]. The algorithm is to *trace* a background run starting from a background pixel along two directions; to its left and right (this is for horizontal runs, otherwise the up and down directions for vertical runs). On the way of the tracing skip some foreground pixels. When the accumulated number of skipped pixels exceeds a pre-set threshold, stop the tracing and do the same along the other direction. The total number of the traced positions is the length of the run associating to the position where the algorithm starts the tracing. Intuitively, the fuzzy run length at a pixel is how far we can see from standing at the pixel along horizontal (or vertical) direction. Like a human standing in a forest looking for a path out of the forest, the length that he can see along a direction may not be the distance from where he is standing to the first tree in front of him. Rather he may be able to “see through” a few trees to get a longer view. But he may not be able to see through.

4.2.4. Feature extraction

One approach for such systems is to calculate a single feature vector for each word; this feature vector is then used to recognize the word. The best way to identify ligatures is that we proceed in two pass method in first pass we extract only those ligatures which will help us in extracting special features. These features are

1. Solidity
2. Number of holes
3. Axes ratio,
4. Eccentricity,
5. Invariant moments
6. Normalized segment length,
7. Curvature

8. Global vision modeling
9. Local vision modeling
10. Topological and contour based features
11. Statistical features
12. Discrete cosine transforms (DCT)
13. Ratio of bounding box.
14. Special features

4.2.4.1 Solidity

Solidity is a scalar quantity. It is defined as the proportion of the pixels in the convex hull that are also in the region. It is computed as

Solidity = Ligature Area/ Convex Hull Area

Where,

Ligature Area = $\sum\sum f(x, y)$

For all x, y in the binary image of the ligature

Convex Hull Area = $\sum f(x,y)$

For all x, y in the convex hull of the ligature

4.2.4.2 Number of holes

This feature gives total number of holes in a ligature. If feature points of ligature are considered as a set of vertices V, and segments as a set of edges E, of a graph G(V, E), then total number of holes in the ligature can be found using graph theory as following:

Number of Holes = E - Est

Where,

E = Number of edges in G

Est= Number of edges in the spanning tree of G.

A graph with N vertices has N-1 edges in its spanning tree.

4.2.4.3 Axes ratio

It is the ratio of the major axis to the minor axis of the best-fit ellipse of the ligature.

Axis Ratio = a/b

Where a and b are the lengths of semi-major axis and semi-minor axis of the best-fit ellipse.

4.2.4.4 Eccentricity

It is the ratio of the distance between the foci of the best-fit ellipse to its major axis.

Eccentricity = distance btw foci / 2b

4.2.4.5 Invariant moments based features

These refer to certain functions of moments, which are invariant to geometric transformations such as, translation, scaling, and rotation [41]. Such features are useful in identification of objects with unique shapes, regardless of their location, size and orientation. The recognition of geometrical patterns independent of position, size, and orientation can be accomplished using moment invariants. These moments uniquely determine a piecewise continuous function f(x,y) which has non-zero values only in a finite part of the x-y plane. If f(x,y) be a digital image in two dimensional space, then the moments of order (p+q) can be defined by [42]:

$$M_{pq} = \sum_x \sum_y x^p \; y^q f(x,y) \qquad (1)$$

for p,q= 0,1,2... (9)

The central moments can be expressed as:

$$\mu_{pq} = \sum_x \sum_y (X-X_c)^p \; (Y - Y_c)^q f(x,y) \qquad (2)$$

$$\text{where } X_c = M_{10} / M_{00} ,\; Y_c \text{ ---- } M_{01}/M_{00} \qquad (3)$$

The normalized central moments Npq can be defined as:

$$Npq = \mu pq / (\mu_{00})^{\gamma} \qquad (4)$$

$$\text{where } \gamma = (p+q)/2 + 1 \text{ for } p+q = 2,3,.. \qquad (5)$$

Hu [44] has shown that a set of seven moments can be derived which are invariant to translation, rotation, and scaling transformations. These seven moments are:

$$\Phi 1 = N_{20} + N_{02}$$

$$\Phi 2 = (N_{20} - N_{02})^2 + 4N_{11}^2$$

$$\Phi 3 = (N_{30} - 3N_{12})^2 + (3N_{21} - N_{03})^2$$

$$\Phi 4 = (N_{30} + N_{12})^2 + (N_{21} + N_{03})^2$$

$$\Phi 5 = (N_{30}-3N_{12})(N_{30}+N_{12})[(N_{30}+N_{12})^2 - 3(N_{21}+N_{03})^2] + (3N_{21}N_{03})(N_{21}+N_{03})[3(N_{30}+N_{12})^2-(N_{21} + N_{03})^2]$$

$$\Phi 6 = (N_{20} - N_{02})[(N_{30} + N_{12})^2 - (N_{21} + N_{03})^2] + 4N_{11}(N_{30} + N_{12})(N_{21} + N_{03})$$

$$\Phi 7 = (3N_{21}-N_{30})(N_{30}+N_{12})[(N_{30}+N_{12})^2 -3(N_{21}+N_{03})^2] + (3N_{12}-N_{30})(N_{21}+N_{03})[3(N_{30}+N_{12})^2-(N_{21} + N_{03})^2]$$

A statistical program based on these moments can be developed and it can recognize numerals but the problem is that if noise is introduced into the image, the results are not reliable. It is known that neural networks are not sensitive to noise so if moment invariants are used as preprocessed inputs to neural networks, results are expected to be more reliable.

4.2.4.6 Normalized segment length

First the normalized length of a segment i is calculated relative to other segment lengths in the same word. Then normalized length of the ligature is calculated as:

$$\textit{Normalized Length} = \sum L(i)$$

4.2.4.7 Curvature

In a similar fashion, first the curvature of a segment is measured by simply dividing the Euclidean distance between the two feature points of that segment by its actual length. This feature equals zero when the segment is a loop and 1 when the segment is a straight line.

$$C(i) = \textit{(Euclidean distance between endpoints) / segment length}$$

Then curvature feature of the ligature is calculated as a sum of curvature features of all of its segments.

$$\textit{Curvature Feature} = \sum C(i)$$

4.2.4.8. Local and global vision modeling

In order to meet industrial requirements local and global modeling high processing speed, robustness, and extremely low error rates, specific transparent neural network (TNN) can be chosen. TNN with local representation is interesting when the amount of data to be represented is small and when the data can be described with simple relations. The behavior of this type of network can be explained step by step. Thus, it falls into the category of “transparent” systems. This kind of NN has been applied on handwritten Latin by M.Côté [45], [46]. We have adapted it to the recognition of handwritten Arabic words. The particularity of our TNN is that it needs a simple step of feature extraction and a normalization post-processing step used to reduce variability of handwriting. In deed, feature extraction is one of the most difficult and important problems of Arabic script recognition, due to the variability of the handwriting. So, the selected set of features should be a small set the values of which efficiently discriminate between patterns of different classes, but are invariant for pattern within the same class. The kind of features depends also on the recognition system. Our recognition system tries to simulate human reading. It proceeds by GVM structural features of the word in order to do a first classification. A second step of classification is done after a LVM of zones without structural features by the use of normalized Fourier Descriptors [47].

4.2.4.10. Topological and contour based features

Individual characters are recognized by combination of topological features, contour based features and features obtained from the concept of a water reservoir.

4.2.4.10.1. Water Reservoir Principle

The water reservoir principle is as follows. If water is poured from one side of a component, the cavity regions of the component where water will be stored are considered as reservoirs [48]. By top (bottom) reservoirs we mean the reservoirs obtained when water is poured from top (bottom) of the component. (A bottom reservoir of a component is visualized as top reservoir when water will be poured from top after rotating the component by 180°). Similarly if water is poured from left (right) side of the component, the cavity regions of the components where water will be stored are considered as left (right) reservoirs. For an illustration see Fig.4.7. Here top, bottom, left and right reservoirs of some Urdu characters are shown. Water flow direction from a full reservoir is also shown in this figure.

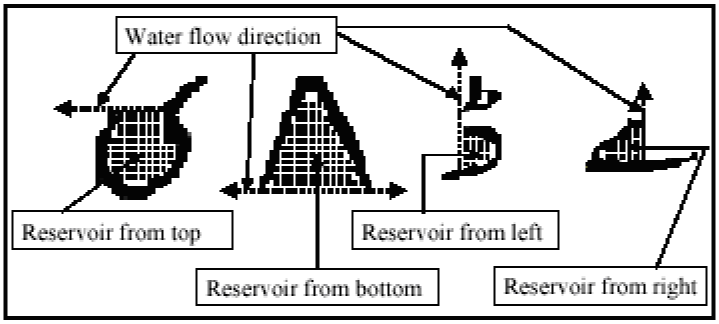


Fig.4.7. Different reservoirs and their water flow directions are shown in four characters. Water flow directions are shown by dotted arrow.

Contour features include characteristics of different profiles obtained from a portion of character's contour. The main water reservoir based features used in the recognition scheme are (a) number of reservoirs from different sides of a component, (b) position of a reservoir with respect to its character bounding box, (c) height of a reservoir, (d) water flow level of a reservoir, (e) direction of water overflow, (f) ratio of reservoir height to component height etc.

4.2.4.14. Special features/ ligatures

For identifying special ligatures, a Feed Forward Back propagation neural network can be used template matching can also be used. The feature vectors obtained from Feature extraction 1 stage (all those features that have been identified above) of the system are fed to this neural network. It then identifies the ligatures as either special ligatures or base ligatures these ligatures contain diacritical marks, dots, small toa second strokes of gaaf and gol hai etc.

Fig.4.8. Some special legatures

4.2.5. Classification

In this stage, we associate special ligatures with the base ligatures. We associate special ligature with the base ligature whose Centroid-to-Centroid distance is minimum or the special ligatuires who comes with in the horizontal dimensions of main body legatuer. A number of lines are grown from the centre of each special ligature, when one of these lines touches a base ligature, then the special ligature is associated with that base ligature. In this stage, due to association of special ligatures with the base ligatures many new features are added to the feature vector of the base ligature.

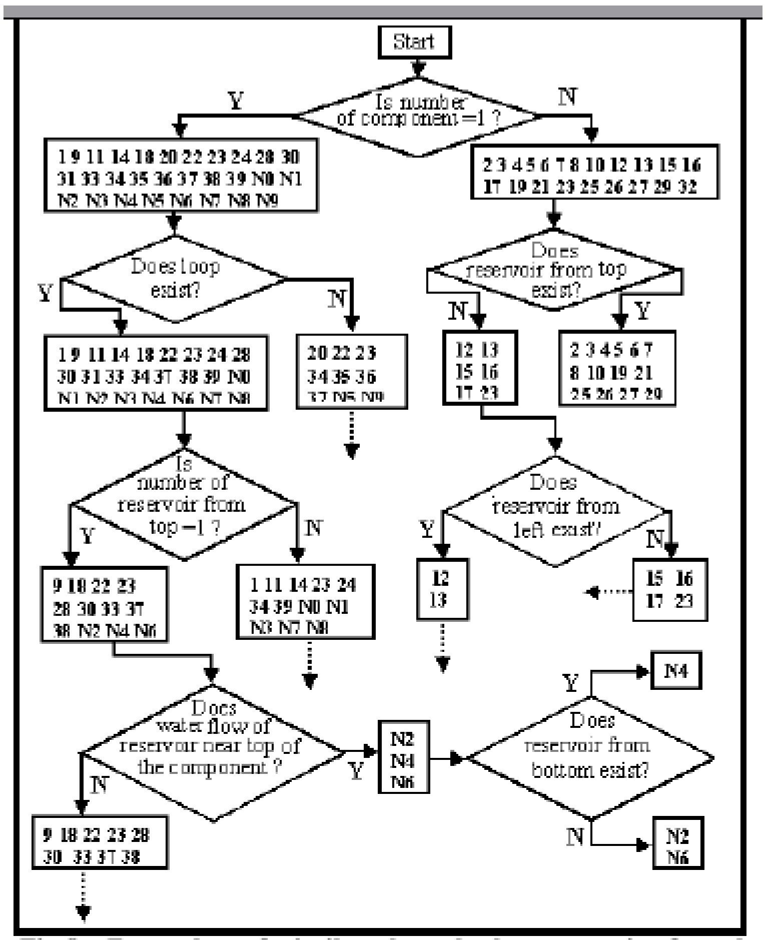


Fig.4.9. Example of a binary classification tree

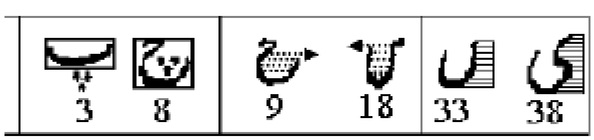


Fig.4.10. Recognition techniques of similar-shaped characters.

4.2.6. Recognition

A variety of techniques of Pattern Recognition [17] such as: (a) Template Matching, (b) Neural Networks, (c) Syntactical Analyses, (c) Wavelet Theory, (d) Hidden Markov Models, (e) Bayesian theory etc. have been explored to develop robust OCRs for different languages such as Latin, Chinese (Kanji) [18], [19], Hangul (Korean) scripts [20], Arabic script [21] also. In case of segmentation-free systems, the feature vector is fed to a multi layer neural network for final identification of ligature.

4.2.6.1. Template Matching

Unlike feature extraction techniques, template matching compares pattern pixel by pixel with a set of pattern templates. A pattern is classified to the template class to which it is most similar. The match between the template and the input image is computed by using a distance function or dissimilarity measure between the two. The method used to measure the similarity or distance between the input and each template is crucial. The challenge in template matching is in making the matching process robust against distortions due to variation in character size, minor rotation, or changes in position and style. Also, it is essential that the distance measure be calculated rapidly. For most applications, each input image needs to be compared to numerous templates involving the comparison of a large number of pixels.

4.2.6.2. Neural Networks

An artificial neural network is an information processing system that has certain functional characteristics in common with biological neural networks. Artificial neural networks have been developed as generalizations of mathematical models of human cognition and neural biology, based on the assumptions that:

1. Information processing occurs at many simple elements called neurons.
2. Signals are passed between neurons over connection links.
3. Each connection link has an associated weight, which in a typical neural network, multiplies the signal transmitted.
4. Each neuron applies an activation function (usually nonlinear) to its net input (sum of weighted input signals) to determine its output signal.

A neural network is characterized by (a) its architecture or pattern of connections between the neurons, (b) its method of determining the weights of the connections (also called its *learning algorithm*), and (c) its activation function.

Each neuron has an internal state, called its *activation* or *activity level*, which is a function of the inputs it has received. Typically, a neuron sends its activation as a signal to several other neurons. It is important to note that a neuron can send only one signal at a time, although that signal is broadcast to several other neurons [62].

For example, consider a neuron Y, illustrated in *Fig. 4.11* that receives inputs from neurons X_1, X_2 and X_3. The activations of these neurons are x_1, x_2 and x_3 respectively. The weights on the connections from X_1, X_2 and X_3 to neuron Y are w_1, w_2 and w_3 respectively. The net input, y_in, to the neuron Y is the sum of the weighted signals from neurons X_1, X_2 and X_3 i.e.

$$y\_in = w_1 x_1 + w_2 x_2 + w_3 x_3$$

The activation y of neuron Y is given by some function of its net input, $y = f(y\_in)$, e.g., the *Sigmoid Function*:

$$f(x) = 1/(1 + exp(x))$$

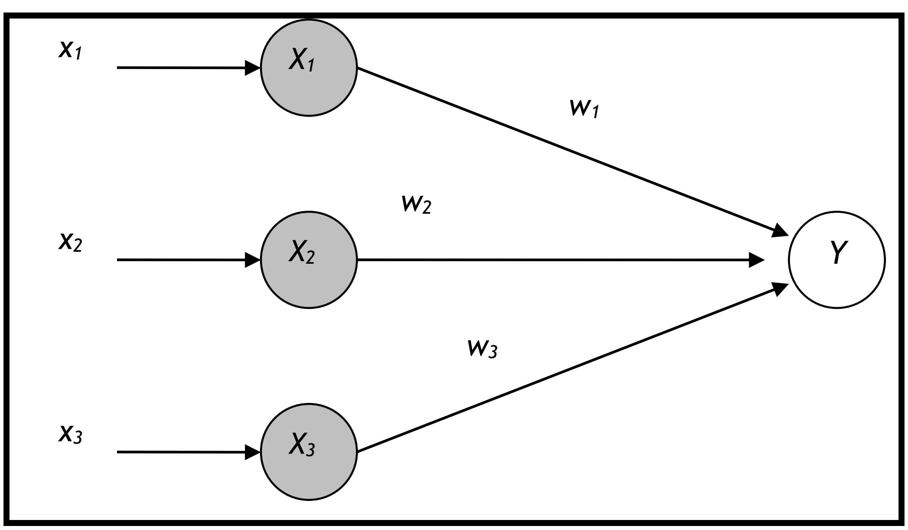


Fig. 4.11. A simple neural network

Now suppose further that neuron *Y* is connected to neurons Z_1 and Z_2, with weights v_1 and v_2, respectively, as shown in *Fig. 4.12*. Neuron *Y* sends its signal *y* to each of these units. However, in general, the values received by neurons Z_1 and Z_2 would depend on inputs from several other neurons, not just one, as shown in this simple example. Although, the neural network in the *Fig. 4.12* is very simple, the presence of a hidden unit, together with a nonlinear activation function, gives it the ability to solve many and more problems than can be solved by a net with only input and output units.

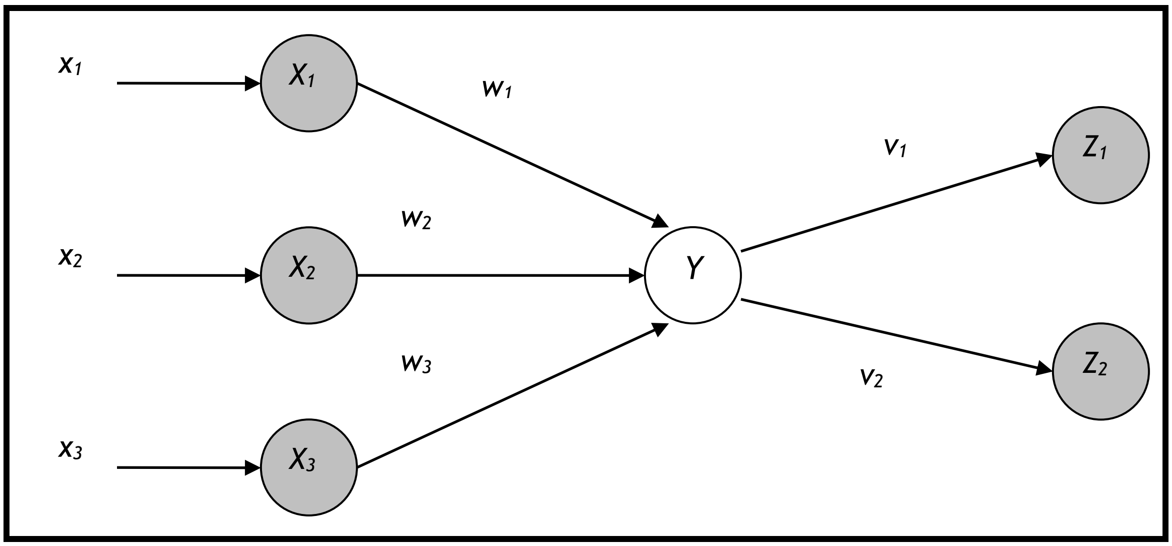


Fig. 4.12. Multilayer Perceptron

4.2.7. Post Processing

The post-processing of an OCR text consists of two phases: error detection and error correction. A straightforward way to post-process an OCR text could be to detect incorrectly recognized words by a dictionary-based spell checker, and present those as well as words with unrecognized or mis-recognized characters to users for their correction. In this strategy, however, users are considerably burdened by the amount of errors and finding the correct words for the mistaken words.

4.2.7.1. Unrecognized Characters vs. Recognition Errors

The symbol representing an unrecognized character can be located and replaced with the appropriate character in significantly less time than it takes to detect .and correct a recognition error

(Fig.). Hence, a variable-threshold OCR device should be operated so that recognition errors are less frequent than unrecognized characters unless this would significantly increase the total number of characters requiring correction. Having fewer recognition errors than unrecognized characters should not only reduce the amount of time needed to prepare a finished document but should also help to reduce residual errors [60].

UNRECOGNIZED CHARACTER ("#" substituted for a character that could not be recognized)

quick --------> q#ick
fastest ------> fast#st

RECOGNITION ERRORS (One or more characters were incorrectly recognized or deleted)

Substitution error (Non-word):
optical -----> optjcal
original ----> oriqinal

Substitution error (Word):
card --------> cord

Insertion error (Non-word):
word --------> vvord

Deletion error (Non-word):
recognize ---> recognze
climb -------> dimb
running -----> ruming

Deletion error (Word):
where -------> here

Fig.4.11. Common types of "errors" associated with optical character recognition (OCR).

4.3. Conclusion

After such an exhaustive study, research and brain storming sessions we decided that for our proposed OCR project we will be using Ligature Based Segmentation Free Holistic approach with Feed Forward and Back Propagation Neural Network. We shall be using horizontal projection for line separation and 8-connected method for labeling identifying different ligatures. And after having a tree classification we shall use Neural Networks for final identification.

Up till now we believe that this is the fastest and most reliable way for any practical OCR, dealing with Noori Nastaleeq (especially for small fonts i.e. point 10 and above).

Chapter 6

Results & Analysis

A Scientific Investigation of Pehchan

The Variable Set

Categories of Ligatures

Results

Graphical Analysis

Discussion

6.1. A Scientific Investigation of Pehchan

In order to evaluate the accuracy of our system and assess its performance, we chalked out a set of independent and dependent variables after much deliberation and thought. Empirical data on these variables was then gathered by processing image samples assimilated in a way so as to give us maximum information on any correlation that may exist between the members of the set. This collection of data was a precise process with much emphasis on accuracy.

The samples that were chosen to test and evaluate the system were designed by selecting ligatures from among the list of ligatures in a corpus of 10,000 most frequently occurring words [63]. There were on the whole six categories of ligatures and correspondingly, six image samples were generated for the investigation at different resolutions. Each of these samples was a collection of a subset of ligatures on which the system had been already trained. Every ligature in the subset had exactly three occurrences randomly positioned in the entire sample.

6.2. The Variable Set

The set of variables for which data was gathered consisted of the following members:

1. Number of Tokens (or number of ligatures).
2. Number of Classes (or number of types of ligatures).
3. Number of Tokens Recognized.
4. Number of Classes Recognized.
5. Error Rate in Token Recognition.
6. Error Rate in Class Recognition.
7. Resolution.

6.3. Categories of Ligatures

Based on the number of characters in each ligature, the ligatures were categorized into the following six groups:

1. Single Character Ligatures.
2. Two Character Ligatures.
3. Three Character Ligatures.
4. Four Character Ligatures.
5. Five Character Ligature.
6. Six & Seven Character Ligatures.

6.4. Results

The following statistics were gathered when the samples were processed by the system:

6.4.1. Single Character Ligatures

Number of Tokens	120
Number of Classes	9
Number of Tokens Recognized	103
Number of Classes Recognized	8
Percent Tokens Recognized	85.8
Percent Classes Recognized	88.9
Error Rate in Token Recognition	14.2
Error Rate in Class Recognition	11.1
Resolution (dpi)	300

Table 6.4.1.1

Number of Tokens	120
Number of Classes	9
Number of Tokens Recognized	107
Number of Classes Recognized	9
Percent Tokens Recognized	89.2
Percent Classes Recognized	100
Error Rate in Token Recognition	10.8
Error Rate in Class Recognition	0.0
Resolution (dpi)	450

Table 6.4.1.2

Number of Tokens	120
Number of Classes	9
Number of Tokens Recognized	113
Number of Classes Recognized	9
Percent Tokens Recognized	94.2
Percent Classes Recognized	100
Error Rate in Token Recognition	5.8
Error Rate in Class Recognition	0.0
Resolution (dpi)	600

Table 6.4.1.3

6.4.2. Two Character Ligatures

Number of Tokens	1200
Number of Classes	131
Number of Tokens Recognized	983
Number of Classes Recognized	123
Percent Tokens Recognized	81.9
Percent Classes Recognized	93.9
Error Rate in Token Recognition	18.1
Error Rate in Class Recognition	6.1
Resolution (dpi)	300

Table 6.4.2.1

Number of Tokens	1200
Number of Classes	131
Number of Tokens Recognized	1022
Number of Classes Recognized	125
Percent Tokens Recognized	85.2
Percent Classes Recognized	95.4
Error Rate in Token Recognition	14.8
Error Rate in Class Recognition	4.6
Resolution (dpi)	450

Table 6.4.2.2

Number of Tokens	1200
Number of Classes	131
Number of Tokens Recognized	1053
Number of Classes Recognized	126
Percent Tokens Recognized	87.8
Percent Classes Recognized	96.2
Error Rate in Token Recognition	12.2
Error Rate in Class Recognition	3.8
Resolution (dpi)	600

Table 6.4.2.3

6.4.3. Three Character Ligatures

Number of Tokens	1350
Number of Classes	179
Number of Tokens Recognized	1039
Number of Classes Recognized	163
Percent Tokens Recognized	77.0
Percent Classes Recognized	91.2
Error Rate in Token Recognition	23.0
Error Rate in Class Recognition	8.3
Resolution (dpi)	300

Table 6.4.3.1

Number of Tokens	1350
Number of Classes	179
Number of Tokens Recognized	1114
Number of Classes Recognized	166
Percent Tokens Recognized	82.4
Percent Classes Recognized	92.7
Error Rate in Token Recognition	17.6
Error Rate in Class Recognition	7.3
Resolution (dpi)	450

Table 6.4.3.2

Number of Tokens	1350
Number of Classes	179
Number of Tokens Recognized	1163
Number of Classes Recognized	168
Percent Tokens Recognized	86.1
Percent Classes Recognized	93.9
Error Rate in Token Recognition	13.9
Error Rate in Class Recognition	6.1
Resolution (dpi)	600

Table 6.4.3.3

6.4.4. Four Character Ligatures

Number of Tokens	1125
Number of Classes	187
Number of Tokens Recognized	849
Number of Classes Recognized	169
Percent Tokens Recognized	75.5
Percent Classes Recognized	90.4
Error Rate in Token Recognition	24.5
Error Rate in Class Recognition	9.6
Resolution (dpi)	300

Table 6.4.4.1

Number of Tokens	1125
Number of Classes	187
Number of Tokens Recognized	908
Number of Classes Recognized	172
Percent Tokens Recognized	80.7
Percent Classes Recognized	92.0
Error Rate in Token Recognition	19.3
Error Rate in Class Recognition	8.0
Resolution (dpi)	450

Table 6.4.4.2

Number of Tokens	1125
Number of Classes	187
Number of Tokens Recognized	926
Number of Classes Recognized	173
Percent Tokens Recognized	82.3
Percent Classes Recognized	92.5
Error Rate in Token Recognition	17.7
Error Rate in Class Recognition	7.5
Resolution (dpi)	600

Table 6.4.4.3

6.4.5. Five Character Ligatures

Number of Tokens	975
Number of Classes	163
Number of Tokens Recognized	728
Number of Classes Recognized	149
Percent Tokens Recognized	74.7
Percent Classes Recognized	91.4
Error Rate in Token Recognition	25.3
Error Rate in Class Recognition	8.6
Resolution (dpi)	300

Table 6.4.5.1

Number of Tokens	975
Number of Classes	163
Number of Tokens Recognized	769
Number of Classes Recognized	151
Percent Tokens Recognized	78.9
Percent Classes Recognized	92.6
Error Rate in Token Recognition	21.1
Error Rate in Class Recognition	7.4
Resolution (dpi)	450

Table 6.4.5.2

Number of Tokens	975
Number of Classes	163
Number of Tokens Recognized	791
Number of Classes Recognized	151
Percent Tokens Recognized	81.1
Percent Classes Recognized	92.6
Error Rate in Token Recognition	18.9
Error Rate in Class Recognition	7.4
Resolution (dpi)	600

Table 6.4.5.3

6.4.6. Six/Seven Character Ligatures

Number of Tokens	990
Number of Classes	193
Number of Tokens Recognized	728
Number of Classes Recognized	175
Percent Tokens Recognized	74.7
Percent Classes Recognized	90.7
Error Rate in Token Recognition	25.3
Error Rate in Class Recognition	9.3
Resolution (dpi)	300

Table 6.4.6.1

Number of Tokens	990
Number of Classes	193
Number of Tokens Recognized	789
Number of Classes Recognized	178
Percent Tokens Recognized	79.7
Percent Classes Recognized	92.2
Error Rate in Token Recognition	20.3
Error Rate in Class Recognition	7.8
Resolution (dpi)	450

Table 6.4.6.2

Number of Tokens	990
Number of Classes	193
Number of Tokens Recognized	817
Number of Classes Recognized	183
Percent Tokens Recognized	82.5
Percent Classes Recognized	94.8
Error Rate in Token Recognition	17.5
Error Rate in Class Recognition	5.2
Resolution (dpi)	600

Table 6.4.6.3

6.5. Graphical Analysis

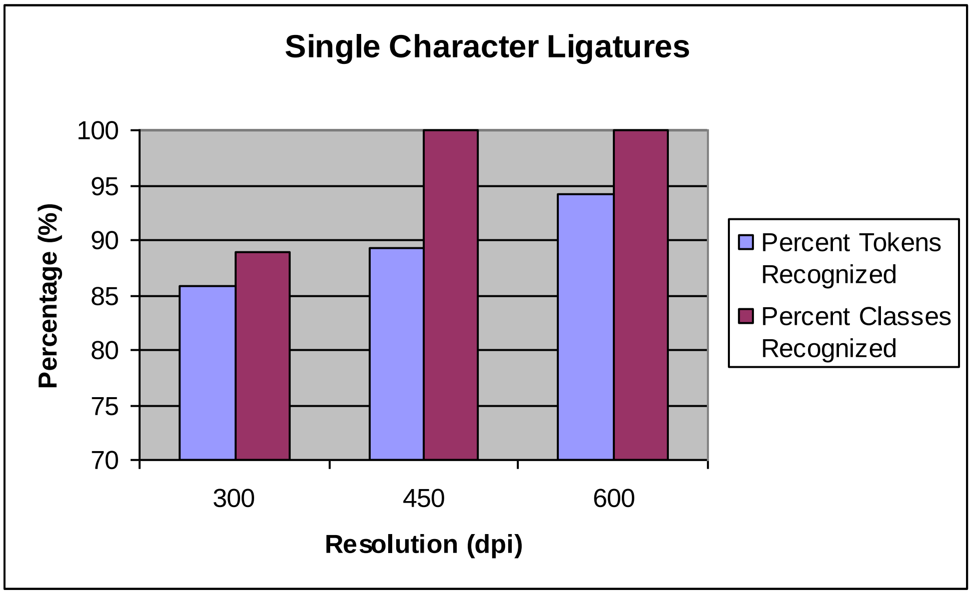


Fig. 6.5.1

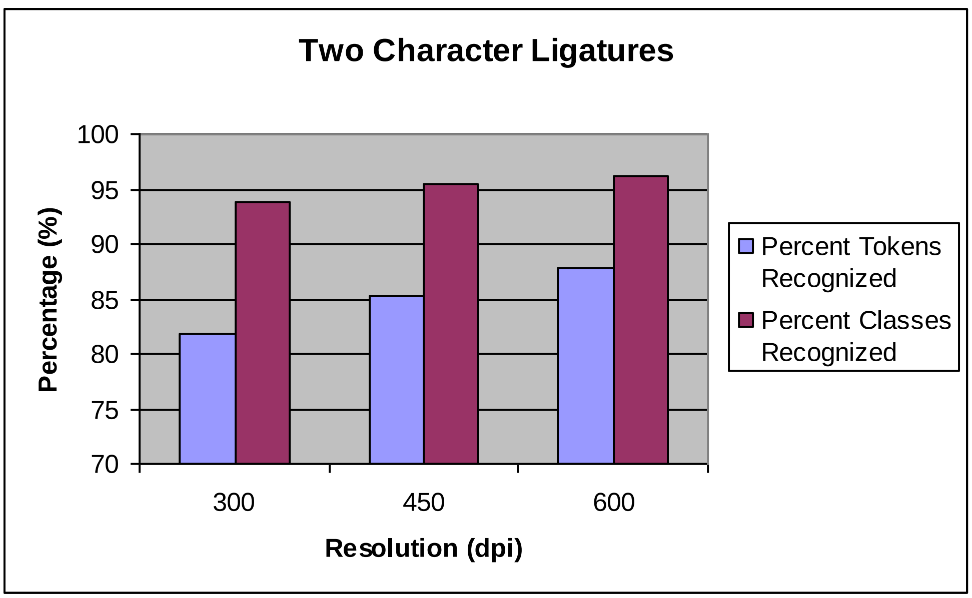


Fig. 6.5.2

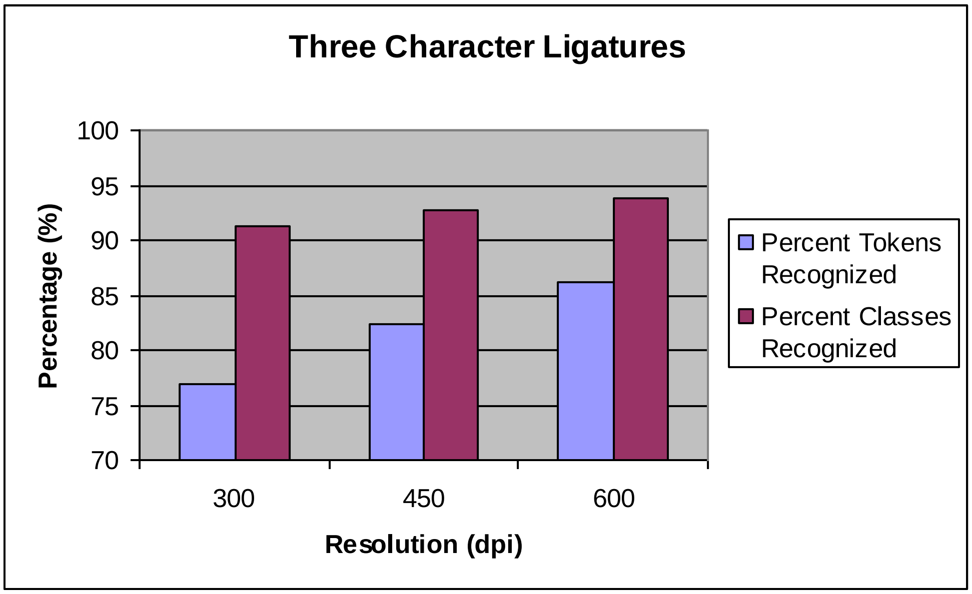


Fig. 6.5.3

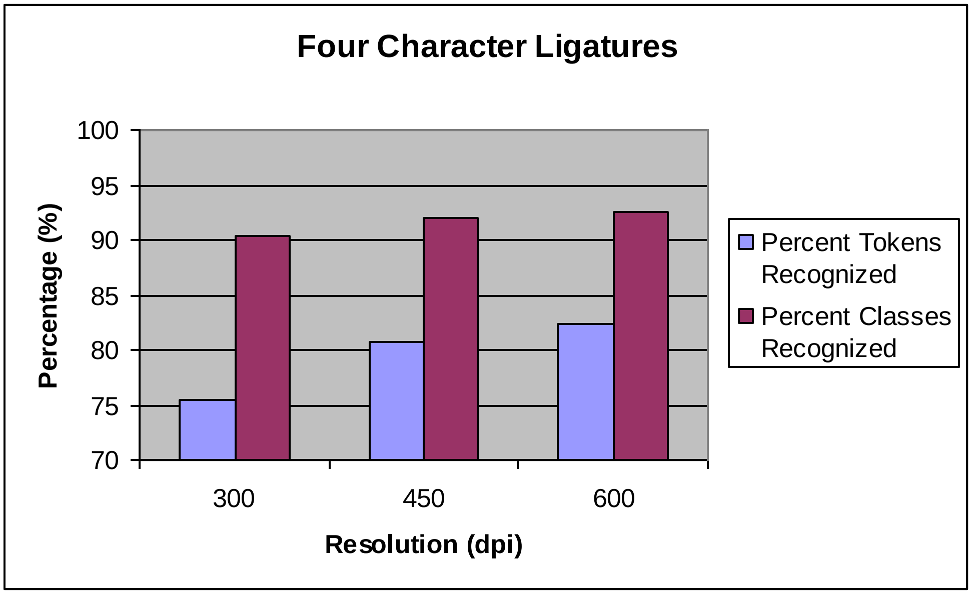


Fig. 6.5.4

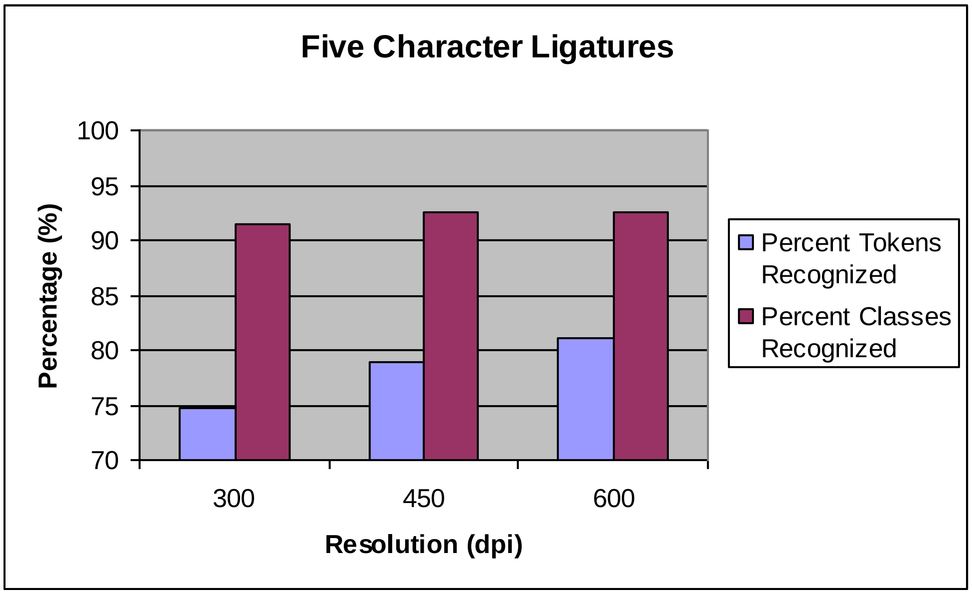


Fig. 6.5.5

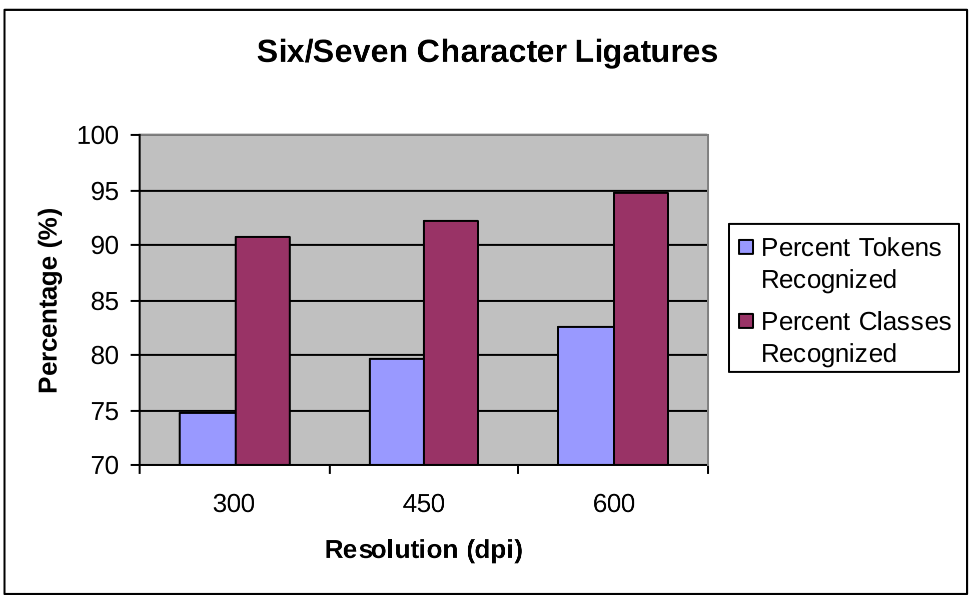


Fig. 6.5.6

6.6. Discussion

From the statistics that have been gathered and tabulated under section 6.4 and graphically analyzed in section 6.5 of this chapter, there appears to be a clear positive correlation that exists between the percent of tokens recognized (the dependent variable) and the resolution of the image sample (the independent variable). In simpler words, on higher resolutions, comparatively a greater percentage of tokens are properly and accurately recognized than at lower resolutions. The same is true for the percentage of classes recognized. However, the error rate is much less for class recognition than is for token recognition. One of the possible reasons could be the fact that class recognition is achieved through template matching and token recognition is based on neural networks. The inputs to the networks are the seven invariant moments, and possibly, the training process needs to be made more precise and rigorous. These results are depicted in Fig. 6.6.1.

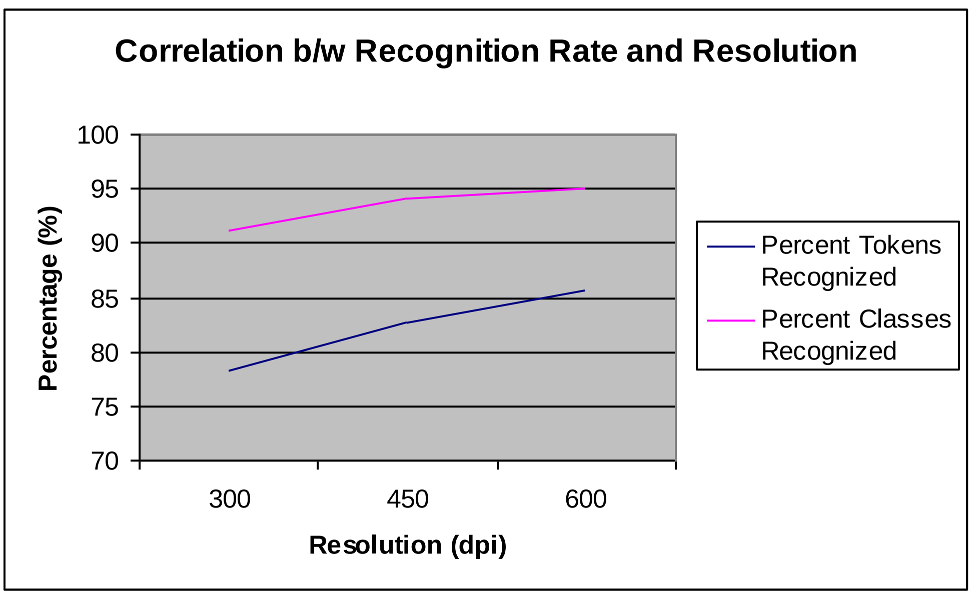


Fig. 6.6.1

Furthermore, by looking at the results of the first four categories, it appears as though the token recognition rate also has a positive correlation with the categorizations that were established in section 6.3 of this chapter. That is to say that up to four character ligatures, ligatures that are more complex have a lesser probability of being recognized than the simpler ligatures. This however is not true for the last two categories. Therefore, we cannot as yet establish a certain correlation between the said variables. However, as we probed deeper into the results, we realized that the recognition rate has a direct relationship with the number of ligatures in a class. This is a positive correlation.

We might've been able to dilute the classes by using the water reservoir principle mentioned in Chapter 4, but only at the expense of significantly increasing the number of classes. That would ensure a more accurate system but one that is slower in performance than Pehchan.

References

[1] Sami Lais, "*Optical Character Recognition*", Computer World, July 29th 2004. (http://www.computerworld.com)

[2] Susan Haigh, "*Optical Character Recognition (OCR) as a digitization technology*", Information Technology services, National Library of Canada, November 15th 1996.

[3] "*Optical Character Recognition / Reading Software*", Adaptive Technology Resource Centre, University of Toronto, September 6th 2004. (http://www.utoronto.ca/atrc/reference/tech/ocr.html)

[4] D.B. Megherbi, S.M. Lodhi and J.A. Boulenouar, "*A Fuzzy Logic based technique for Urdu Character Representation and Recognition*", Proceedings of the IASTED International Conference, Intelligent Systems and Control, October 28-30, 1999, Santa Barbara, California, USA.

[5] Ahmed Mirza Jamil, "*Noori Nastaleeq Computerized Urdu Calligraphy, List of Ligatures*", Government of Pakistan, National Awards, Independence Day 1982.

[6] Faiza Mughal, and Fariha Zaman, "*Ligature based OCR for Nastaleeq Script using Neural Networks*" , Internal Document, Kinnaird College for Women, Lahore, Pakistan and Centre for Research in Urdu Language Processing, NUCES, Lahore, Pakistan.

[7] Yasir Niaz Khan; Member of visiting faculty NUCES.
e-mail: yasir.niza@nu.edu.pk

[8] Aamir Wali, Atif Gulzar, Ayesha Zia, Muhammad Ahmad Ghazali, Muhammad Irfan Rafiq, Muhammad Saqib Niaz, Sara Hussain, and Sheraz Bashir *"Contextual Shape Analysis of Nastaliq"*.

[9] *"Features for Noori Nastalique"* by Aamir Wali, Atif Gulzar, Ayesha Zia, Muhammad Ahmad Ghazali, Muhammad Irfan Rafiq, Muhammad Saqib Niaz, Sara Hussain, and Sheraz Bashir

[10] Aamir Wali; Member of permanent faculty NUCES, department of Computer Science.
e-mail: aamir.wali@nu.edu.pk

[11] Sarmad Hussain and Muhammad Afzal, "Urdu Computing Standards: Urdu Zabta Takhti (UZT) 1.01", Proceedings of INMIC 2001, Lahore University of Management Sciences, Lahore Pakistan.

[12] Aamir Wali, Sara Hussain, Noman Nazar, "*Logical model of Nastaleeq Script Style*", Internal document, Centre for Research in Urdu Language Processing, NUCES, Lahore, Pakistan.

[13] Mohammad S. Khorsheed, "*Automatic recognition of words in Arabic manuscripts*", PhD Dissertation, Churchill College, University of Cambridge, June 2000.

[14] H. Bunke, P. Wang, "*Handbook of character recognition and document image analysis*", World Scientific, 2000.

[15] Syed. Afaq Husain, and Syed. Hassan Amin, "*A Multi-tier Holistic approach for Urdu Nastaliq Recognition*", Proceedings of IEEE-INMIC 2002, Karachi Institute of Information Technology, Karachi, Pakistan.

[16] Tal Steinherz, Ehud Rivlin, and Nathan Intrator "*Offline cursive script word recognition - a survey*", International Journal on Document Analysis and Recognition, September 1999.

[17] Arun K. Pujari, C. Dhanunjaya Naidu, and B C Jinaga "*An Adaptive Character Recognizer for Telugu Scripts using Multiresolution Analysis and Associative Memory*"

[18] A.-B. Wang, K.-C. Fan. "*Optical recognition of handwritten Chineses character by hierarchical radical matching method*" Pattern Recognition, 34, pp 15-35, 2001.

[19] R. Casey, G. Nagy. "*Recognition of printed Chinese character*", IEEE tr Electron. Comp, EC-15 (1966) pp 91-101.

[20] H.- Y. Kim, J.H. Kim, "*Hierarchical random graph represetation of handwritten and its application to Hangul recognition.*" Pattern Recognition, Vol 34, 2001.

[21] A Cheung, M Bennamoun, N. W. Bergmann, "*An Arabic OCR system using ecognitionbased segmentation*", Pattern Recognition, Vol 34, pp 215-233, 2001.

[22] Mario I. Chacon M "*Fuzzy Binarization and Segmentation of Text Images for OPCR*"

[23] Gonzalez R., and Woods R., "*Digital Image Processing*", Addison-Wesley Publishing Company, 1993.

[24] Glasbey C. ," *An Analysis of Histogram-Based Thresholding Algorithms*", 'CVGIP:Graphical Models and Image Processing, Vol 55,No. Nov. 1993, pp 532-

[25] Papamarkos N.,& Gates B. ,"*A New Approach for Multilevel Threshold Selection*", Graphics Models & Image Processing,Vol 55 No 5, Sep. 1994, pp 357-370.

[26] BenHajel N., Stamen G.,& Lemoine J. ,"*Fuzzy Iterative Image Segmentation with Recursive Merging*" ,SPIE Visual Communication and Image Processing, Vol 18181992, pp 1271-1281.

[27] Zadeh L., "*Outline of a New Approach to the Analysis of Complex Systems and Decision Processes*, " ,IEEE Tran. on System, Man, & Cybernetics, Vol. SMC-3,No1,January 1973, pp28-44.

[28] Ezdek J., "*Pattern Recognition with Fuzzy Objective Function Algorithms*", Plenum Press 1981.

[29] Bezdek J.,& Pal K., "*Fuzzy Models for Pattern Recognition, methods that search for structures in data*", IEEE Press 1991.

[30] Tou J., Cheng J., Lu D.,& McCarthy J., "*Automatic Type Font Identification for Falsified Documents" ,* Proceedings 6th International Conference on Pattern Recognition, Vol 2, IEEE Computer S. Press, October 1982.

[31] U. Pal and Anirban Sarkar "*Recognition of Printed Urdu Script*"

[32] B. B. Chaudhuri and U. Pal, "*A complete printed Bangla OCR system*", Pattern Recognition, vol.31, pp.531-549, 1998.

[33] "*Local Normalization*", Biomedical Imaging Group, October 15th 2004. (http://bigwww.epfl.ch/demo/jlocalnormalization/)

[34] D.B. Megherbi, S.M. Lodhi and J.A. Boulenouar, "A Fuzzy logic based techniwue for Urdu character representation and recognition", Proceedings of the IASTED international Conference, INTELEGENT SYSTEMS AND CONTROLE, October 28 – 30, 1999, Santa Barbara, California, USA.

[35] R. G. Casey and G. Nagy, "*Recursive segmentation and classification of composite character pattern*," 1982 International Joint Conference on Pattern Recognition, pp. 1023-1026.

[36] S. Kahan, T. Pavlidis, and H. S. Baird, "*On the recognition of printed characters of any font and size*," *IEEE Trans. on Pattern Analysis and Ma*chine Intelligence, Vol. PAMI-9, No. 2, March 1987, pp. 274288.

[37] M. Maier, "*Separating characters in scripted documents*," 1986 International *Conference on Pattern* Recognition, pp. 1056-1058.

[38] S. Harmalkar and R. M. K. Sinha, "*Integrating word level knowledge in text recognition",* 1990, International Conference on Pattern Recognition, pp. 758-760.

[39] O. Matan, H. S. Baird, J. Bromley, C. J. C. Burges, J. S., Denker, L. D. Jackel, Y. LeCun, E. P. D. Pednault, W. D. Stterfield, C. E. Stenard, and T. J. Thompson, "*Reading handwritten digits: A zip code recognition system*," Computer, July 1992, pp. 59-63.

[40] I.Guyon, J.Bromley, N.Matic, etc, "*A neural network system for recognizing on-line handwriting*", Models of Neural network, Springer Verlag, 1996.

[41] H.Guo, X.Q.Ding, "*The development of high performance Chineses/English bi-lingual OCR system*", proc. CMIN '95, Beijing, China, March 95, 248-253.

[42] R.C. Gonzalez and P. Wintz, "*Digital Image Processing"*, pp 419-425, Addison-Wesley Publishing Co., 1987.

[43] "*Noise Removal From Images*", Department of Mathematics, University of California Berkley, October 15th 2004. (http://math.berkeley.edu/~sethian/Movies/Movienoiseremoval.html)

[44] Ming-Kuei Hu, "*Visual pattern recognition by moment invariants*", IRE Trans. Inform. Theory, vol IT-8, pp 179-197, Feb. 1962.

[45] M.Côté, M.Lecolinet, E.Cheriet, M.Suen, " *Automatic Reading of Cursive Scripts Using a Reading Model and Perceptual Concepts*", IJDAR, Vol1, pp:3-7, 1998.

[46] JL.McClelland, DE.Rumelhart, " *An interactive activation model of context effects in letter perception*", Psychological Review 88, pp: 375-407, 1981.

[47] S. Snoussi Maddouri, H. Amiri, "*Combination of Local and Global Vision Modeling for Arabic Handwritten Words Recognition*", IWFHR' 2002.

[48] U. Pal, A. Belaïd and Ch. Choisy "*Touching numeral segmentation using water reservoir concept*", Pattern Recognition Letters, vol.24, pp. 261-272, 2003.

[49] M.traft editor. "*Reading and mental lixicon*" Lawrence Earlbaum new Jersey, 1991

[50] J. Morton "*Interaction of information in word recognition*", psychological review 76:165-178, 1969

[51] C.A becker, "*Allocation of attention during visual word recognition*", Journal of experimental psychology, Human perception and performance, 2; 556-566, 1976

[52] J.I McClellend, and D.E. Rumalhert, “*An interaction activation model of context effect in letter perception*”, psychological review 88:375-407, 1981

[53] C.A.Higgins and P.E Bramal. “*Hand written recognition based on human reading process*”. In C.Feure, P. Keuss, G Lorette and A.Vinter editors, Advances in Handwriting and Drawing. A Multidisciplinary Approach, pages 155-167, Europe, Paris, 1994.

[54] S.H lee and J.H Kim. “*On-line cursive script recognition by a letter spotting technique based on HMM’s”*. In proc. Of the Second Korean Nation conference on Character Recognition, pages 93-104, Korea, 1994,

[55] E.Lecolinet. “*Cursive Script Recognition by backward maching*”. In C.Feure, P. Keuss, G Lorette and A.Vinter editors, Advances in Handwriting and Drawing. A Multidisciplinary Approach, pages 155-167, Europe, Paris, 1994.

[56] S. Edelman, T. Flash, and S. Ullman. “*Reading cursive handwriting by alignment of letter prototypes”*. International Journal of Computer Vision 5(3):303{331 (1990)

[57] R. M. Bozinovic and S. N. Srihari. “*Offline cursive script word recognition*”. IEEE Transactions on Pattern Analysis and Machine Intelligence 11(1):68{83 (1989)

[58] S. N. Srihari and R. M. Bozinovic. “*A multi-level perception approach to reading cursive script*”. Artificial Intelligence 33(2):217{255 (1987)

[59] S. N. Srihari and R. M. Bozinovic. “*A multi-level perception approach to reading cursive script”*. In Proceedings Int. Joint Conf. on Artificial Intelligence, pages 844{847, 1987

[60] William H. Cushrnan, Purnendu S. Ojha, and Cathleen M. Daniels, “*Usable OCR: What Are The Minimum Performance Requirements?”* , CHI ’90 Proceedings, April 1990.

[61] Dr. Sarmad Husain and Muhammad Afzal, "Urdu Computing Standards: Urdu Zabta Takhti (UZT) 1.01", Proceedings of INMIC 2001, LahoreUniversity of Management Sciences Lahore, Pakistan.

[62] Laurene Fausett, *"Fundamentals of Neural Networks: Architectures, Algorithms, and Applications"*, Prentice Hall International, Inc. 1994.

[63] Tahira Khizar: Team Lead *"Urdu Lughat – Urdu Localization Project",* Project funded by Electronic Government Directorate, Pakistan. E-mail: tahira.khizar@nu.edu.pk

Bibliography

1. A.-B. Wang, K.-C. Fan. "*Optical recognition of handwritten Chineses character by hierarchical radical matching method*" Pattern Recognition, 34, pp 15-35, 2001.

2. A Cheung, M Bennamoun, N. W. Bergmann, "*An Arabic OCR system using ecognitionbased segmentation*", Pattern Recognition, Vol 34, pp 215-233, 2001.

3. Ahmed Mirza Jamil, "*Noori Nastaleeq Computerized Urdu Calligraphy, List of Ligatures*", Government of Pakistan, National Awards, Independence Day 1982.

4. B. B. Chaudhuri and U. Pal, "*A complete printed Bangla OCR system*", Pattern Recognition, vol.31, pp.531-549, 1998.

5. BenHajel N., Stamen G.,& Lemoine J. ,"*Fuzzy Iterative Image Segmentation with Recursive Merging*" ,SPIE Visual Communication and Image Processing, Vol 18181992, pp 1271-1281

6. Bezdek J.,& Pal K., "*Fuzzy Models for Pattern Recognition, methods that search for structures in data*", IEEE Press 1991.

7. C.A.Higgins and P.E Bramal. "*Hand written recognition based on human reading process*". In C.Feure, P. Keuss, G Lorette and A.Vinter editors, Advances in Handwriting and Drawing. A Multidisciplinary Approach, pages 155-167, Europe, Paris, 1994.

8. C.A becker, "*Allocation of attention during visual word recognition*", Journal of experimental psychology, Human perception and performance, 2; 556-566, 1976

9. D.B. Megherbi, S.M. Lodhi and J.A. Boulenouar, "*A Fuzzy Logic based technique for Urdu Character Representation and Recognition*", Proceedings of the IASTED International Conference, Intelligent Systems and Control, October 28-30, 1999, Santa Barbara, California, USA.

10. E.Lecolinet. "*Cursive Script Recognition by backward maching*". In C.Feure, P. Keuss, G Lorette and A.Vinter editors, Advances in Handwriting and Drawing. A Multidisciplinary Approach, pages 155-167, Europe, Paris, 1994.

11. Ezdek J., "*Pattern Recognition with Fuzzy Objective Function Algorithms*", Plenum Press 1981

12. Glasbey C. ," *An Analysis of Histogram-Based Thresholding Algorithms*", 'CVGIP:Graphical Models and Image Processing, Vol 55,No. Nov. 1993, pp 532-

13. Gonzalez R., and Woods R., "*Digital Image Processing*", Addison-Wesley Publishing Company, 1993.

14. H. Bunke, P. Wang, "Handbook of character recognition and document image analysis", World Scientific, 2000.

15. H.Guo, X.Q.Ding, "*The development of high performance Chineses/English bi-lingual OCR system*", proc. CMIN '95, Beijing, China, March 95, 248-253.

16. H.- Y. Kim, J.H. Kim, "*Hierarchical random graph represetation of handwritten and its application to Hangul recognition.*" Pattern Recognition, Vol 34, 2001.

17. Guyon, J.Bromley, N.Matic, etc, "*A neural network system for recognizing on-line handwriting*", Models of Neural network, Springer Verlag, 1996.

18. J.I McClellend, and D.E. Rumalhert, "*An interaction activation model of context effect in letter perception*", psychological review 88:375-407, 1981

19. J. Morton "*Interaction of information in word recognition*", psychological review 76:165-178, 1969

20. M.Côté, M.Lecolinet, E.Cheriet, M.Suen, " *Automatic Reading of CursiveScripts Using a Reading Model and Perceptual Concepts*", IJDAR, Vol1, pp:3-7, 1998.

21. Ming-Kuei Hu, "*Visual pattern recognition by moment invariants*", IRE Trans. Inform. Theory, vol IT-8, pp 179-197, Feb. 1962.

22. M. Maier, "*Separating characters in scripted documents*," 1986 International *Conference on Pattern* Recognition, pp. 1056-1058.

23. Matan, H. S. Baird, J. Bromley, C. J. C. Burges, J. S., Denker, L. D. Jackel, Y. LeCun, E. P. D. Pednault, W. D. Stterfield, C. E. Stenard, and T. J. Thompson, "*Reading handwritten digits: A zip code recognition system*," Computer, July 1992, pp. 59-63.

24. Papamarkos N.,& Gates B. ,"*A New Approach for Multilevel Threshold Selection*", Graphics Models & Image Processing,Vol 55 No 5, Sep. 1994, pp 357-370.

25. R. Casey, G. Nagy. "*Recognition of printed Chinese character*", IEEE tr Electron. Comp, EC-15 (1966) pp 91-101.

26. R. G. Casey and G. Nagy, "*Recursive segmentation and classification of composite character pattern*," 1982 International Joint Conference on Pattern Recognition, pp. 1023-1026.

27. R. M. Bozinovic and S. N. Srihari. "*Offline cursive script word recognition*". IEEE Transactions on Pattern Analysis and Machine Intelligence 11(1):68{83 (1989)

28. S. Edelman, T. Flash, and S. Ullman. "*Reading cursive handwriting by alignment of letter prototypes"*. International Journal of Computer Vision 5(3):303{331 (1990)

29. S.H lee and J.H Kim. "*On-line cursive script recognition by a letter spotting technique based on HMM's"*. In proc. Of the Second Korean Nation conference on Character Recognition, pages 93-104, Korea, 1994,

30. S. Kahan, T. Pavlidis, and H. S. Baird, "*On the recognition of printed characters of any font and size*," *IEEE Trans. on Pattern Analysis and Ma*chine Intelligence, Vol. PAMI-9, No. 2, March 1987, pp. 274288.

31. S. N. Srihari and R. M. Bozinovic. "*A multi-level perception approach to reading cursive script*". Artificial Intelligence 33(2):217{255 (1987)

32. S. Snoussi Maddouri, H. Amiri, "*Combination of Local and Global Vision Modeling for Arabic Handwritten Words Recognition*", IWFHR' 2002.

33. U. Pal, A. Belaïd and Ch. Choisy "*Touching numeral segmentation using water reservoir concept*", Pattern Recognition Letters, vol.24, pp. 261-272, 2003.

34. William H. Cushrnan, Purnendu S. Ojha, and Cathleen M. Daniels, "*Usable OCR: What Are The Minimum Performance Requirements?"* , CHI '90 Proceedings, April 1990.

35. Zadeh L., "*Outline of a New Approach to the Analysis of Complex Systems and Decision Processes*", IEEE Trans. on System, Man, & Cybernetics, Vol. SMC-3,No1,January 1973, pp28-44.