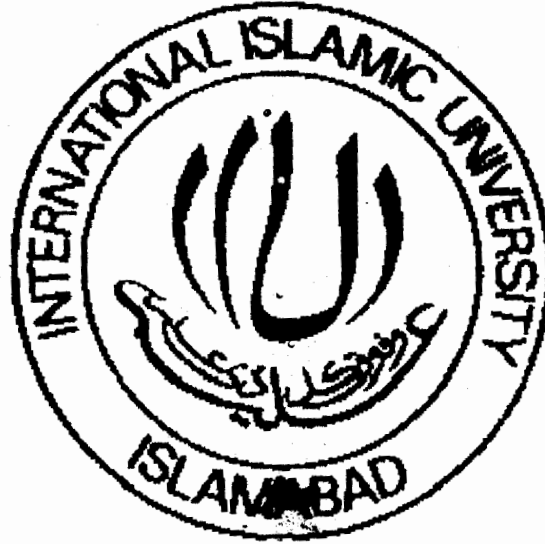


**A SOM Neural Network based Offline Recognition
System for Segmented Urdu Characters in Nasakh Font**



T03189

Developed by

Safdar Zaman
212-FAS/MSCS/F-04

Muhammad Ayub
207-FAS/MSCS/F-04

Supervised by

Dr. Syed Afaq Hussain

**Department of Computer Science
Faculty of Basic and Applied Sciences
International Islamic University, Islamabad
2007**

13-7-2010

MS
006.3
SAS

Neural networks (computer science)

EN 



Accession No 7-03189

D.F.

20-12-10



Project in Brief

Project Title: **A SOM Neural Network based Recognition System for Segmented Urdu Character written in Nasakh Font**

Organization: International Islamic University, Islamabad (IIUI)

Undertaken by: **Safdar Zaman**
212-FAS/MSCS/F-04

Muhammad Ayub
207-FAS/MSCS/F-04

Supervised by: **Professor Dr. Syed Afaq Hussain**
Head:
Department of Electronics Engineering
International Islamic University, Islamabad

Starting Date: September, 2006

Completion Date:

Tool Used: MATLAB 7.0
C++
Microsoft Visual Studio.net (C#)

Operating System: Microsoft Windows XP 2006 Professional

System Used: Pentium III (797 MHz Genuine Intel)
RAM 128
HD 80GB

**International Islamic University, Islamabad
Faculty of Basic and Applied Sciences
Department of Computer Science**

Date: 28/7/.....2007

Final Approval

It is stated that we have read the thesis/research report, entitled as "A SOM Neural Network based Offline Recognition System for Segmented Urdu Character in Nasakh Font" submitted by Safdar Zaman (212-FAS/MSCS/F-04 and Muhammad Ayub (207-FAS/MSCS/F-04. It is our judgment that this project is of standard to warrant its acceptance by the International Islamic University, Islamabad, for the MS Degree in computer science.

Project Evaluation Committee

External Examiner

Dr. Muhammad Afzal,
House # 537, F-2, St# 18,
Westridge-III, Alabad,
Rawalpindi.

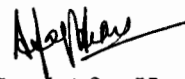

Muz
29/7/07

Internal Examiner

Prof. Dr. M. Sikander Hayat Ichiq
H.No. 1309, Gali 12,
F-10/2, Islamabad

Sikander

Supervisor



Professor Dr. Syed Afaq Hussain,
Head,
Department of Electronics Engineering,
International Islamic University, Islamabad.

*A SOM Neural Network based Offline Recognition System
for Segmented Urdu Character in Nasakh Font*

Declaration

We, Safdar Zaman (212-FAS/MSCS/F-04), Mohammad Ayub (207-FAS/MSCS/F-04), hereby declare and assure that this thesis neither as a whole nor as a part thereof has been copied out from any source. It is further declared that we have developed this software and accompanied thesis entirely on the basis of our personal efforts, made under the sincere guidance of our teachers. If any part of this thesis is proved to be copied out or found to be a reproduction of some other thesis, we shall stand by the consequences.

The work presented in this paper is motivated by the fact that there is no reported effort at developing Optical Character Recognition System for Nasakh script using Kohonen Self Organizing Map Feed (SOM) Neural Network

Safdar Zaman
212-FAS/MSCS/F-04

Muhammad Ayub
207-FAS/MSCS/F-04

DEDICATED TO,

Our honorable and most respectful Parents, Teachers and Friends

Acknowledgement

We bestow all praises to Almighty Allah, The Most Merciful and compassionate, The Most Gracious and Beneficent, Who bestowed us good health, courage and knowledge to carry out and complete work.

We consider it a good privileged to express our deepest gratitude and deep sense obligation to our reverend supervisor Prof. Dr. S. Afaq Hussain, who kept our moral high by his suggestion and appreciation. It will not be out of place to express our profound admiration for our deer teacher and Head of computer science department, Prof. Dr. Sikander Hayat Khiyal.

Finally we must mention that it was mainly due to our family's moral support and financial help during our entire academic career that enabled us to complete our work dedicatedly. We once again would like to admit that we dedicate all our achievements to our most loving parents, for their prayers are more precious then any treasure on the earth.

In the end we are also thankful to our truly, sincere and most loving brothers, friends and class fellows whose prayers have always been a source of determination for us.

Safdar Zaman
212-FAS/MSCS/F-04

Muhammad Ayub
207-FAS/MSCS/F-04

**A dissertation submitted to the
Department of Computer Science,
International Islamic University, Islamabad
as a partial fulfillment of the requirements
for the award of the degree of
MS in Computer Science**

Table of Contents

	Page No
1. INTRODUCTION	
1.1 OCR	1
1.2 Types of OCR	2
1.3 Urdu Fonts	3
1.4 OCR for Urdu Scripts	3
1.5 Challenges in Urdu Scripts	3
1.6 Classification Methods	5
1.6.1 Matrix/Template matching	5
1.6.2 Structured Pattern Recognition	5
1.6.3 Neural Network Pattern Recognition	5
1.6.3.1 Supervised Training	5
1.6.3.2 Unsupervised Training	6
1.7 Organization of Thesis	6
2. LITERATURE SURVEY	
2.1 A Multi-tier Holistic approach for Urdu Nastaliq Recognition	7
2.2 Ligature Based Optical Character Recognition of Urdu, Nastaleeq Font	8
2.3 Arabic Handwriting Recognition	9
2.4 Neural network Analysis of Hand-printed Characters, Proc. 7th International workshop on Statistical and Structural Pattern Recognition	10
2.5 Automatic Recognition of Handwritten Arabic Characters Using Their Geometric Features	11
2.6 Extracting Features from Arabic Characters	12
2.7 Arabic Character Recognition, Handbook of Character Recognition and Document. Image Analysis, Pp.397-420	12
2.8 Preprocessing and Structural Features Extraction for Multi-Fonts Arabic / Persian	14
2.9 Problem Definition	15
2.10 Proposed Solution	15

3	CLUSTERING	
3.1	Introduction	17
3.2	KOHONEN SELF-ORGANIZING MAPS	18
3.3	Clustering Comparisons	31
3.4	Effect of Cluster Topology	32
3.5	Effect of Cluster Size	33
3.6	Effect of number of Epochs	33
3.7	Final Cluster Table	33
3.8	Summary	34
4	PROPOSED METHODOLOGY	
4.1	Introduction to System	34
4.2	Proposed System	34
4.3	System Phases	36
4.3.1	Input Phase	36
4.3.1.1	Acquisition of image	36
4.3.1.2	Conversion from RGB to Gray level	36
4.3.1.3	Conversion from Gray level to Binary	36
4.3.2	Preprocessing Phase	36
4.3.2.1	Thinning	36
4.3.2.2	Chain Code Formation	36
4.3.2.3	Smoothing	37
4.3.3	Feature Extraction Phase	38
4.3.3.1	Features Extracted	38
4.3.4	Recognition Phase	39
4.3.4.1	Classification-I	39
4.3.4.1.1	Weight Matrix Formation	39
4.3.4.1.2	Clustering	39
4.3.4.2	Classification-II	39
4.3.4.2.1	Additional Features	39
4.4	Implementation	40

5	TESTING	41
5.1	Analysis	41
5.2	Methodology used for testing	41
5.3	Tested Data in the form of sentences	41
5.4	Tested Data in the form of Samples	44
5.4.1	TA of START Sample	44
5.4.2	YA of MIDDLE Sample	44
5.4.3	GHAEN of END Sample	45
5.4.4	QAAF of START Sample	45
5.4.5	JEEM of END Sample	46
5.4.6	SHEEN of END Sample	46
5.4.7	LAAM of MIDDLE Sample	47
5.5	Recognition Rate	47
6	CONCLUSION AND FUTURE WORK	
6.1	Basic Input	49
6.2	Results of Clustering	51
6.3	Results of Testing	52
6.4	Constraints	53
6.5	Future Recommendations	53
	APPENDIX A	54
	REFERENCES	

Abstract

Character recognition is one of the oldest fields of research since the advent of computers. However it is an open field for researchers due to the challenging nature of segmentation and recognition. Different languages like English, Chinese, Arabic and Persian have tremendously attracted the attention of character recognition researchers. In contrast, research in the field of character recognition for Urdu script face major problems mainly related to its characteristics like cursive nature, multiple fonts and shapes of character depending upon the different positions in the word and their attachments on the base line.

Proposed work addresses problems recognizing Nasakh script of Urdu Language. The work consists of five major phases. After getting segmented character as input the original image is preprocessed which involves various steps like binarization/smoothing etc. It is followed by thinning phase, which makes the image one pixel wide. After thinning the most important phase upon which much of the concentration has been paid, Feature Extraction takes place. In this phase more than twenty five different features are extracted from the thinned image which help the system recognize the segmented character. At the last Classification phase occurs which classifies the segmented character on the bases of obtained 26 features from previous phase. This phase begins with Classification-I which adopts clustering technique to classify segmented characters by using 18 features. In the second classification phase, extra features are used to distinguish characters in a cluster by Kohonen Self Organizing Map (SOM) Neural Network to perform the final recognition. The work presented in this paper is motivated by the fact that there is no reported effort at developing Optical Character Recognition System for Nasakh script using Kohonen Self Organizing Map (SOM) Neural Network.

CHAPTER NO 1

INTRODUCTION

1. INTRODUCTION

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 [1]. It is a cursive language even in its printed form. Like Arabic, recognizing Urdu script presents challenges of cursive orthography and context sensitive letter shape. However, in contrast to Arabic text, in which connected characters follows a base line, the joined characters in Nastaliq and Nasakh are positioned according to their preceding character as well as a vertical justification of the ligature [1].

English and Chinese languages have tremendously attracted interests of character recognition researchers. In contrast, research in the recognition for Urdu /Arabic scripts face major problems mainly related to the unique characteristics like being cursive, context sensitive of a character in a word and connectivity of characters on the baseline.

1.1 OCR

Character recognition as one of the most important fields of pattern recognition, has been the center of attention for researchers in the last four decades. Often abbreviated OCR, Optical Character Recognition refers to the branch of computer science that involves reading text and translating it into a form that the computer can manipulate (for example, into UNICODE).

OCR is the automatic transcription by the computer from the available image of the text. On recognition of English and Arabic characters, much work has been done on OCR, which not only covers the separate script. Urdu speaking people such as Pakistanis, Indians, Bengalis and people in some other parts of the world frequently use Urdu characters in writing; even though they pronounce it differently, need handwritten recognition of Urdu cursive script [2].

The popularity of OCR has been increasing each year with the advent of fast microprocessors providing the vastly improved recognition techniques. This can be shown in OCR wands (Handheld optical character readers) now reading print that, over 10 years ago, large batch readers would have rejected. There has been a tremendous improvement in increasing both effective read rates and accuracy. Data Entry through OCR is faster, accurate, and generally efficient than keystroke data entry. Desktop OCR scanners can read typewritten data into a computer at rates up to 2400 words per minute [2].

Urdu Alphabet set, as shown in Table 1-1, is commonly used for writing any widespread languages (e.g. Urdu, Arabic, Persian), yet there is much less research in progress for recognizing Urdu script than there is for Roman and Chinese text [3].

End	Middle	Start	Isolated	End	Middle	Start	Isolated
ص	ص	ص	ص	ا	ا	ا	ا
ض	ض	ض	ض	ب	ب	ب	ب
ط	ط	ط	ط	پ	پ	پ	پ
ظ	ظ	ظ	ظ	ت	ت	ت	ت
ع	ع	ع	ع	ث	ث	ث	ث
غ	غ	غ	غ	ج	ج	ج	ج
ف	ف	ف	ف	چ	چ	چ	چ
ق	ق	ق	ق	ح	ح	ح	ح
ک	ک	ک	ک	خ	خ	خ	خ
ل	ل	ل	ل	د	د	د	د
م	م	م	م	ذ	ذ	ذ	ذ
ن	ن	ن	ن	ر	ر	ر	ر
و	و	و	و	ز	ز	ز	ز
ی	ی	ی	ی	س	س	س	س
				ش	ش	ش	ش

Table 1-1: Urdu Alphabet Set in Nasakh Script

1.2 Types of OCR (Optical Character Recognition)

OCR can be classified into the following two types based on their input method [4].

- Offline OCR
- Online OCR

Offline OCR

In this method, the text is written on some source and then that source is given as input to the system in the form of image(s). Then that image is preprocessed and made easily capable for extraction of the features that help the system classify the text in the form of image. Offline OCR can either recognize whole word or can segment a word into its constituent characters. In the latter case segmented characters are then preprocessed and features are extracted from them in order to recognize them and hence the whole word is recognized.

Online OCR

Online recognition involves the automatic conversion of text as it is written on a special digitizer or PDA, where a sensor picks up the pen-tip movement $X(t)$, $Y(t)$ as well as pen-Up/pen-Down switching. That type of data is called digital ink and can be regarded as dynamic representation of handwriting. The obtained signal is converted into letter codes which are usable within computer and text-processing application.

1.3 Urdu Fonts

Urdu Script contains 36 characters for which Urdu script uses different Fonts. Font is a style of writing with more than 10 commonly used Fonts in Urdu namely, Nastaliq, Nasakh, Noori Nasakh, Noori Nastaliq, Koofi, etc. Nastaleeq and Nasakh are the two most popular fonts as discussed below.

- **Nastaleeq** is a special calligraphic way of writing Urdu. It does not have a baseline rather the text is centre justified and it is very difficult to recognize this style of writing because of its complexity.
- **Nasakh** is another writing style of Urdu which follows one baseline. This way of writing is simpler than Nastaleeq so it is easy to recognize this style because of its simplicity.

1.4 OCR for Urdu Scripts

Much of work has been done in different languages like English, Latin, Chinese, Roman, Indian, Arabic, Persian etc. Arabic and Persian scripts are similar to Urdu scripts but Recognition System applied for them does not apply to Urdu because Urdu has some properties different from Arabic and Persian. It is very important to have a slightly different OCR for Urdu because for the last few decades Urdu Nasakh style was used to type the books, news papers and other materials. So it is necessary to have a recognition system in order to convert the old printed material documents into computer files.

1.5 Challenges in Urdu Script

Urdu is an unconstrained written language by nature. Because of its cursive scripts, it poses the following difficulties in recognition.

1.5.1 Direction of Writing

Horizontal-justification of Urdu text is from Right to Left.

1.5.2 Cursive Nature

Urdu text is cursive in nature i.e. all the characters are connected to each other within word. Unlike English where each alphabet is written individually, in Urdu alphabets are connected together to form words.

1.5.3 Baseline

Urdu Nasakh text has a baseline. Baseline is a horizontal line which runs through the text, cutting all the words at some point.

1.5.4 Word Overlapping

Urdu characters some times overlap vertically without touching each other. Connected words overlap each other. So, there are no defined boundaries for words. It is difficult to tell where one finishes and the other starts.

1.5.5 Diacritics

Diacritics like Hamza, Jazm, Khari Zabar, Khari Zer etc are special symbols which are very important in the proper pronunciation of the word.

1.5.6 Ligatures

Ligatures are the single or combination of the characters. Ligatures collectively make words. In a word ligatures are separated by one another via half space whereas full space separates one word from other.

1.5.7 No Case for characters

Urdu characters have neither Upper nor Lower case property. Unlike English where each alphabet has only two possible shapes, small and capital, in Urdu shapes of alphabets are context dependent. So they change shapes depending upon their position in the word. Each alphabet may have many shapes.

1.5.8 Shapes of the characters

Shape of the characters depend upon both the writing style as well as position in the word e.g. In Nasakh writing style character has 4 different shapes of character depending upon the position. Whereas in Nastaleeq style each character assumes different shapes in a word depending on the context in which it is used. Unlike English where each alphabet has only two possible shapes, small and capital, in Urdu shapes of alphabets are context dependent. So they change shapes depending upon their position in the word. Each alphabet may have many shapes.

1.5.9 Strokes

Every Urdu character has one main stroke and zero or one secondary stroke. Urdu character has a longest continuous portion of the character which is written before lifting the pen. This is called main stroke. Some characters have some dot(s) with this main stroke and some have another secondary stroke. Secondary stroke is the next longest stroke after main stroke.

1.5.10 Association and Positioning of Special Symbols

Special symbol like dots, hamza, hey, mud etc carry lot of information. Depending upon their number and position, word change meaning. So two words with same base shape can have many meaning depending upon how many special symbols it have and where they have been placed. Also because of word overlapping, association of special symbols with their base shape is very big problem. Some times it becomes impossible to tell if a certain special symbol between shape A and B, belongs to A or B. There is no visual way to use to associate symbols. Prior knowledge about language some times becomes necessary for symbols association.

1.6 Classification Methods

There are three basic classification methods used for OCRs:

1.6.1 Matrix/Template matching

1.6.2 Structured Pattern Recognition

1.6.3 Neural Network Pattern Recognition

1.6.1 Matrix/Template Matching

Matrix matching is the simpler and more common method. This method compares unknown character with a library of character templates or prototypes. When an image matches one of these prescribed matrices of dots within a given level of similarity, the computer labels that image as the corresponding ASCII character. Here printed characters give suitable results as compared to handwritten characters.

1.6.2 Structured Pattern Recognition

Feature Extraction is the process of without strict matching to templates prescribed as matrices. Also known as Intelligent Character Recognition (ICR), or Topological Feature Analysis, this method varies by how much "computer intelligence" is applied by the manufacturer. The computer looks for general features such as open areas, closed shapes, diagonal lines, line intersections, etc. This method is much more versatile than matrix matching.

1.6.3 Neural Network Pattern Recognition

Neural Network is an information-processing system that has certain performance characteristics in common with biological neural networks. A Neural Network consists of nodes which are connected to each other via directed communication links, each with an associated weight. The values of weights are trained in neural net to find the output values. We shall distinguish two types of training -- Supervised and Unsupervised -- for a neural network. These two training methods are as following [5].

1.6.3.1 Supervised Training

This training is accomplished by presenting a sequence of training vectors, each associated with a target output-vector. The weights are then adjusted according to a learning algorithm.

Some of the simplest neural nets are designed to perform pattern classification, i.e., to classify an input vector as either belonging or not belonging to a given category. In this type of neural net, the output is a bivalent element, say, either 1 (if the input vector belongs to the category) or -1 (if it does not belong). We consider several simple single-layer nets that are designed for pattern classification. These nets (i.e. AND, OR and NOT Functions) are trained using a supervised algorithm (Hebb rule, delta rule and Perceptron Learning rule) [5].

1.6.3.2 Unsupervised training

Self-organizing neural nets group similar input vectors together without the use of training data to specify what a typical member of each group looks like or to which group each vector belongs. A sequence of input vectors is provided, but no target vectors are specified. Self-organizing neural nets classify similar input vectors without the use of training data. The net modifies the weights so that the most similar input vector is assigned to the same output unit [9].

1.7 Organization of Thesis

There are six chapters and one appendix in the thesis/research report. The first chapter introduces Optical Character Recognition (OCR), its types and different classification methods. A brief introduction on Urdu and its OCR is also given in this chapter. The chapter-2 gives brief overview of some previous work done in the field of OCR. The chapter-3 describes how the clustering was done. The chapter-4 presents our proposed methodology. It describes the working of each module of the system. The chapter-5 describes the testing of different samples. The chapter-6 gives brief discussion of the conclusion and concludes the discussion with future recommendations.

CHAPTER NO 2

LITERATURE SURVEY

2. LITERATURE SURVEY

This chapter discusses some of the previous works done in the field of Optical Character Recognition. Following are some research papers, which have been reviewed in order to understand what has been done in the past.

2.1 A Multi-tier Holistic approach for Urdu Nastaliq Recognition [1]

This research work describes an offline recognition approach that recognizes the cursive Urdu text for Noori Nastaliq script. Identification here is of ligature based instead of character based. Every word consists of number of the ligatures and this work uses a multi-tier holistic approach in order to recognize those ligatures. This paper utilizes segmentation free methodology where word is not segmented into characters rather it is recognized as a connected ligature. In the first stage, special symbols diacritics (Hamza, Mad, Tau, Dots, Ha) are identified. In the second tier, more features are extracted and a back propagation Neural Network is used to recognize the ligature. The Neural Network used is Feed Forward Back Propagation Neural Network with 15 inputs, 25 hidden and 25 output neurons to recognize the special ligatures. The different stages involve:

Preprocessing

The preprocessing stage involves Smoothing, Skew detection and correction, Document decomposition, Slant normalization, Binarization etc.

Segmentation

This research utilizes Connected Component Labeling which 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. 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.

Feature Extraction I

At the initial stage only those features are extracted that help in the recognition of special ligatures. These features used were Solidity, Number of Holes, Axis Ratio, Eccentricity, Moments, Normalized segment length, curvature, ratio of bounding box width and height.

Special Ligature Identification

This is a primary classification scheme for identifying special ligatures. Feed Forward Back propagation neural network with 15 inputs, 25 hidden and 25 output neurons was used. The feature vectors obtained from Feature extraction I stage of the system are fed to this neural network. It then identifies the ligatures as either special ligatures or base ligatures.

Feature Extraction II

In this stage, association of special ligatures with the base ligatures takes place. 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.

Neural Network based Classification

In this stage, the final feature vector consisting of 34 features is fed into Feed Forward Back propagation neural network. The network architecture consists of 34 inputs, 65 hidden neurons and 45 output neurons.

Results

The performance of this system was 100% on trained ligatures. The untrained ligatures were given the closest match. Future enhancements in this paper can be made by increasing the number of trained ligatures and diacritics.

2.2 Ligature Based Optical Character Recognition of Urdu, Nastaleeq Font [2]

This paper work is meant to recognize the type-written Nastaleeq script of Urdu language. It is ligature based recognition system because it extracts the line from the image and then isolates every ligature extracted in that line. For classification of the characters, template matching technique is used in this paper.

It discusses the characteristics of Urdu script like Direction of the writing, Cursive nature, presence of Baseline, Overlapping, Diacritics, ligatures, different shapes and Strokes.

The basic methodology of this OCR system is:

Image Acquisition and Storage

The OCR system needs an image file containing Urdu text as its input. For simplicity we are assuming that the user inputs a monochrome image file. After the image is acquired, its contents are stored as ones and zeros in a 2 dimensional array.

Separation of lines from obtained image

Now the file is in ready state, and as a first step, we have to separate the lines of text. Simple methods for separating lines are those based on projection profiles. A projection profile is a histogram giving the ON pixels accumulated along parallel lines. Thus a horizontal projection profile is a one dimensional array where each element denotes the number of ON pixels along a row in the image. And the same have been used using additional information of line width in order to include secondary strokes too.

Isolation of ligatures

The first step towards ligature isolation is to label every connected component in the line. Then mathematical features of every connected component in a line are calculated which include height, width, Aspect Ratio and Bounding Box of every component. The Aspect Ratio of the main body also forms a part of the Feature Matrix.

Distinguish between Main Bodies and Visible Features

Visible features include: diacritics, secondary strokes and Nuktas (dots). In this step horizontal projection to construct a hypothetical band across a line of text is used. Further, a two-pass method for recognition of visible features is used. In the first pass, the visible feature set is recognized and in the second pass the visible feature recognition strategy is applied to the main body set, so as to extract the visible features coming in the region of the base band. The visible features are recognized using template matching.

Get Constituent Unicode characters

After identifying the class to which a ligature belongs, its constituent Unicode characters are retrieved from the Database and appended in a text file. If a class has more than one ligature, then the main body of the unknown ligature is matched with the templates of this class and the ligature with maximum similarity is selected. Its constituent Unicode characters are looked up and appended in a file.

In this paper formatting like spaces and tabs are not handled in the converted text. Further a font size of 36 performs well when aspect ratio is used as a feature vector. Changing the font size causes problems in identifying some ligatures when used with aspect ratio. This paper has potential for future enhancement.

2.3 Arabic Handwriting Recognition [3]

This thesis explores different kinds of strategies used for recognition of handwritten Arabic, like Baseline-finding algorithm, Segmentation-free methods, Nearest Neighbor and Tangent features extracting strategies and also Centric and Moments finding strategies. This work helped us in understanding the whole scenario of the feature extraction

Preprocessing

As well as the actual recognition stage, paper assures that a handwriting recognition system must perform a variety of preprocessing tasks to ensure the text is in a suitable form. These steps include:

- Thresholding - distinguishing foreground pixels from background pixels in the handwriting image. Typically, a grayscale value is chosen, and any pixels darker than this are assigned to the foreground.
- Noise Removal - identifying pixels which represent noise and correcting them.
- Line and Word Segmentation - splitting the page image into areas representing lines of text and then words.
- Baseline Finding - finding the writing line, i.e. the line upon which the text sits

In the next step, this paper uses the baseline method, Principal Components Analysis (PCM).

Principal Components Analysis (PCM)

Is used to find the principal axis of the foreground distribution. This gives an angle for the baseline but not necessarily its vertical position. Use of this method on Latin text is investigated by Steinherz et. al.. They found a higher performance when using the background pixels for PCA rather than the foreground.

Principal Components Analysis is a way of finding the directions along which a distribution exhibits the greatest variation. These directions are termed the Principal Components of the distribution. They correspond to the most significant eigenvectors of the covariance matrix of the data points.

This paper describes the Nearest Neighbor Method for feature extraction that produced 81% results.

2.4 Neural Network Analysis of Hand-printed Characters [4]

This paper introduces feature extraction method and then introduces the neural network system which recognizes the Latin characters on the basis of feature extracted. This paper states that the character has to be segmented and features are extracted in each segment. It takes sample of Latin character and then digitizes the character. After digitization, characters are cleaned and thinned using the parallel algorithm. For character recognition it determines the primitive feature of each character. Two main primitives it involves are straight line and curve primitives. In order to find the primitives, a Binary tree construction strategy is used which is an algorithm implementing a 3×3 window is used to trace along the path of the skeleton, recording the structural information of the trace path. This path is described as a tracing path and is stored in a binary tree.

Following are the structural information

- 1-Freeman Code (8-directional code)
- 2-Positional. (Coordinates information)
- 3-Loop (Closed Path) •

Primitives encountered are

- Break point (Separator)
- Straight Line
- Curve

After having enough features, this paper uses Neural Network technique to recognize the characters.

2.5 Automatic Recognition of Handwritten Arabic Characters Using Their Geometric Features [7]

This paper is meant for offline handwritten Arabic words. This paper not only discusses the problems that could be faced in this recognition but also it discusses a system capable of recognizing handwritten characters of only single writer. It initially concentrates on the extraction of main geometrical features of each of the extracted handwritten Arabic characters. Then it discusses some processes on images like slant correction, slope correction, base line estimation and etc. Then it discusses the segmentation process to segment the characters in the words. The geometrical features like endpoints, junction, turning points, loops, frames and strokes are briefly discussed. Here it is also discussed how to pass those geometrical features to the neural network in order to recognize the characters. This work has achieved 69.7 percent recognition rate for character frames of data.

Preprocessing

The phase consists of image Loading, Slope Correction, Slant Correction, and Thinning. The slope correction is achieved by application of the Shear transform parallel to the y-axis.

Finding Handwriting Features

A number of useful features have been found from the processing that has already been performed on the writing: endpoints, junctions, complementary characters, loops, and turning points. The methods for the detection of intersection points, endpoints, and loops, are all operating on skeletonized bit maps.

Character Classification

The character classification is done in this research using feed forward error back propagation neural network. The network has a single hidden layer of standard perceptions with nonlinear activation functions. The mapping process is from input, represented by features extracted for the Arabic character, to the output, that represents an indication to that character.

The recognition process has known two trials. The neural network has three layers. In both trials, the number of output neurons and the number of hidden neurons was same. The achieved accuracy in first trial is 53%. In the second trial, the recognition accuracy increases to 69.7%.

2.6 Extracting Features from Arabic Characters [8]

This paper concentrates on thinning process and feature extraction for character set such as Arabic. Not only thinning is discussed but it also discusses the problems encountered in producing satisfactory thinned form and also, to some extent, it describes solutions to those problems and difficulties. It also discusses that after thinning an image to one pixel many features can easily be found like intersection points (pixels that have more than two neighbors pixels), end points (pixels that have exactly one neighbor pixel) and all other pixels that have exactly two neighbor pixels thinned image also helps to easily get numbers of holes in the image. This paper then describes some thinning problems like

- * Distortions at intersection points
- * Thinning algorithm may not produce 1 pixel thick representation.
- * Ambiguity can also be found in thinned forms.
- * Irregular edges can also be found when thinning the image.

In order to avoid all the above problems this paper discusses how to properly thin an image, remove hole, remove the dot and remove the trail.

Nearest Neighbor

One of the simplest classifiers we can use is the Nearest Neighbor classifier. This takes a test point in vector form, and finds the Euclidean distance between this and the vector representation of each training example. The training example closest to the test point is termed its Nearest Neighbor. Since this example is in some sense the one most similar to our test point, it makes sense to allocate its class label to the test point. This exploits the 'smoothness' assumption that points near each other are likely to have the same class.

2.7 "Arabic Character Recognition", Handbook of Character Recognition and Document. Image Analysis [9]

This reference discusses both the techniques Off-line as well as On-line of handwritten Arabic recognition. It gives references of too many papers and research works done on this topic. After discussing some properties of Arabic writing system it compares it with some other languages like English. It also discusses different shapes of a characters depending upon their occurrences in different places (in the beginning, end, middle and isolated form). It discusses some strategies like IRAC (Interactive Recognition of Arabic Characters) system for On-line recognition that how does IRAC do sampling and smoothing process on the characters.

The writer proposed a system for online Arabic word recognition. The hand drawing is directly segmented in to characters on the basis of certain heuristic criteria. For every

word, the characters are connected to each other by horizontal segments from right to left. Statistically, these connections appear:

1. Almost always after an intersection point.
2. Often after a cusp point.
3. Sometimes simply after a change of curvature.

These points serve remarkably as separators in the segmentation process and directly permit one to obtain a list of characters of the word component. With the help of each separator, portions of the hand writing are extracted and transmitted to the character recognition module for identification. If no character has been recognized, the corresponding segmentation is canceled and a new tentative try is carried out with the separator of the next lower priority.

The character recognition module is similar to the system which recognizes isolated characters. Finally, three hypotheses at most, i.e. the three best score candidates provided by the character recognition module are associated with each of the characters which are extracted from the handwritten Arabic word by segmentation module. The set of hypothesis form a lattice. Identification of the word consists of traversing the lattice to find the path of the best score corresponding to a word in the dictionary. Binary diagrams are used for resolving ambiguities and eliminating all a candidates which are not present in the dictionary of known words.

The second method is global without segmentation in to characters. This method uses the notion of stroke instead of character. A vector defining the main parameters of word (number of secondary strokes, size and position of groups of dots, number of intersection points, number of cusp points) makes it possible to recognize a word and each of its constituent strokes. Whenever the same vector yields several possibilities, they are classified according to a score computed from secondary parameters of the stroke (the start point, form and angular variation of the main stroke).

Furthermore to enhance the recognition rate of a syntactical and semantically analyzer that verifies the grammatical structure and the meaning of Arabic sentence is used.

It also discusses different properties like open curves, closed curves, vertical strokes, Horizontal strokes, curps points, inflection point, group of dots and secondary strokes and pencil lift. Then it discusses Off-line character recognition, for that it first discusses segmentation process, then extraction of the features from the segmented characters. Also it discusses the vertical as well as horizontal scanning of the characters for features. This material helped us to understand how character recognition is achieved.

2.8 Preprocessing and Structural Features Extraction for Multi-Fonts Arabic / Persian [12]

This paper discusses Multi-Fonts Arabic/Persian character recognition system. Character recognition as one of the most important fields of pattern recognition has been the center of attention for researchers in the last four decades. The modern version of OCR appeared in the middle of 1940's with the development of the digital computers. Since then several character recognition systems for English, Chinese and Japanese characters have been proposed. However, development of character recognition systems for other languages such as Arabic and Persian hasn't had such a fast trend. Arabic / Persian are the all written using modified Arabic alphabets. This recognition system consists of three major phases.

Digitization

Here Image is obtained, first of all, by scanning the text. File is converted into Gray scale image. Gray Scale image easily makes it possible to distort the noise or some features e.g. Loop etc.

Preprocessing

Here Preprocessing includes Global Threshold, connected components recognition, grouping of connected components, skew deletion and correction.

Feature Extraction

This phase consists of contour tracing, contour analysis, sub-words detection, deleting a sub word's complementary characters and then producing the output file

The proposed work has been tested on some printed Arabic words with 14 of 19 different fonts available in Arabic for Windows (famous fonts like Naskh, Andalus, Kufi and traditional Arabic). The system has been developed by Microsoft Visual C++ on Windows NT platform. Some Arabic fonts and other complex features of Arabic script have caused poor results for the recognition of 2 and 3 dots. In these fonts contour's length and area for 2 or 3 dots are different from most of them, so finding correct values for thresholds would be difficult and in some cases impossible. Another reason for rather poor recognition of dots is the attaching dots to main bodies in some fonts. In these cases no contours would be traced for dots, therefore they wouldn't be recognized as dots at all. Also the behaviors of the system's algorithms are predictable on noisy documents. The system recognizes all the existent loops of the words.

2.9 Problem Definition

A number of researches have focused on recognition of Arabic or Arabic script based languages. However, none of them is comprehensive and a complete solution. A commercial application for Urdu Script Recognition is still far from sight because:

1. It has some extra characters in addition to Arabic characters.
2. Its character changes its shape with respect to context.
3. Some of its characters are Overlapping.
4. Its words have even parts (ligatures) with significant meaning.
5. Diacritics used in it, also play important role in word manipulation.

The scope of our system is Offline, character-based Recognition as well as segmentation free. The Recognition of the segmented character of Urdu language written in Nasakh Font is the target of proposed research. The writing style is selected as Nasakh because characters in Nasakh can be easily segmented contrary to Nastaleeq in which a ligature based approach is better. Also Urdu Nasakh script is closer to Arabic script, so the system can be extended to recognize the Arabic script. The recognition system is Offline with A Neural Network (Self-organizing map) based classifier trained on extracted features.

2.10 Proposed Solution

We have proposed a multi tier approach in order to achieve the desired recognition. Initially Character Acquisition takes place to get the segmented character's image which is the real philosophy of the Offline Character Recognition. Then Preprocessing phase applies some techniques like Thinning, to make the image more appropriate for further processing. After this, Feature Extraction phase extracts all necessary features which help our system recognize the shape of the character. Then Classification-I finds the cluster in which character fits best. One cluster consists of many characters so Classification-II finally classifies the character from the other characters of the cluster and hence the given image is recognized as character. The deep discussion of proposed solution is given in subsequent chapter and shown in the Fig2-1 below.

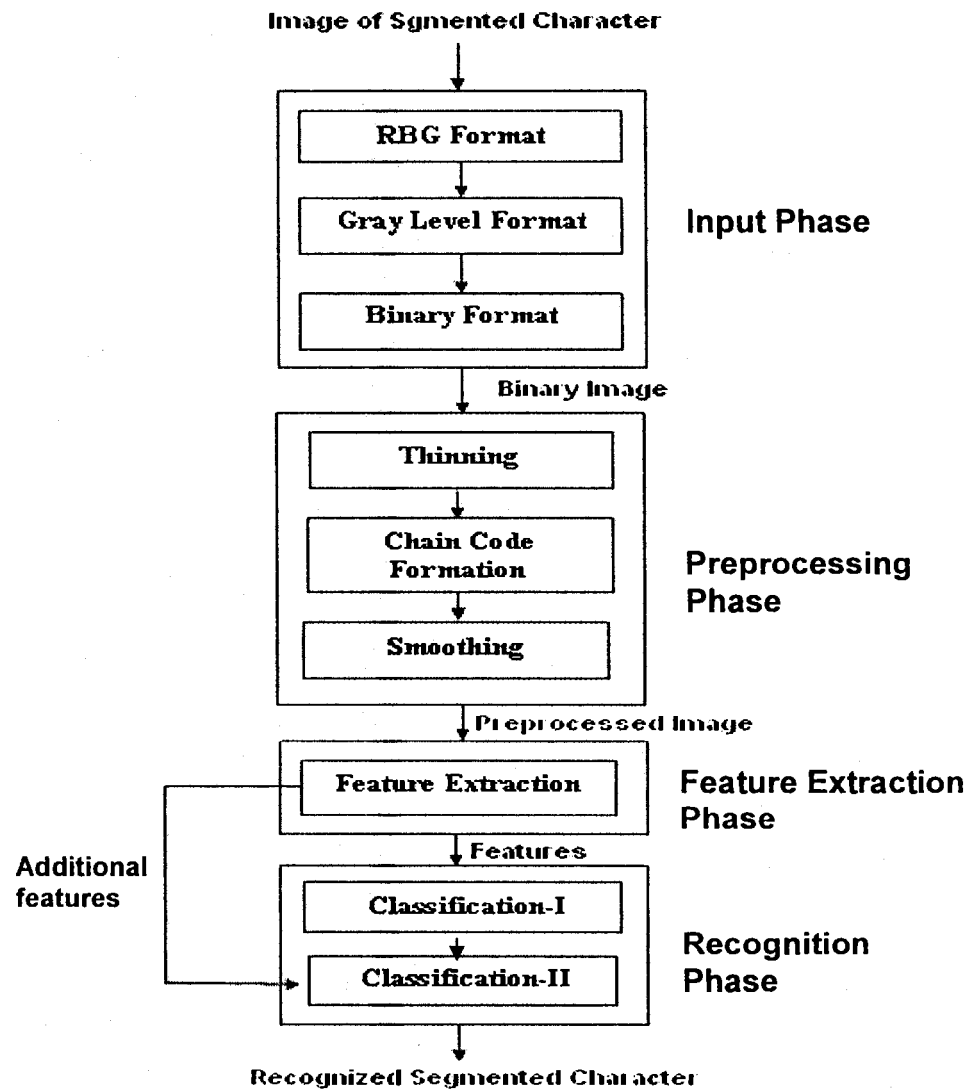


Fig 2-1

CHAPTER NO 3

CLUSTERING

3. CLUSTERING

This chapter introduces clustering which is the main objective of Classification-I of proposed system. Different clustering results on the basis of various topologies and epochs are discussed one by one and at the end overall comparison is given in the form of table.

3.1 Introduction

Clustering is the concept of grouping similar patterns together. In this text, the term “clustering” is used only for unsupervised learning problems in which the desired groupings are not known in advance [9].

Clustering can be considered the most important unsupervised learning problem; so, as every other problem of this kind, it deals with finding a structure in a collection of unlabeled data. A loose definition of clustering could be “the process of organizing objects into groups whose members are similar in some way”[11]. A cluster is therefore a collection of objects which are similar to each other but are dissimilar to the objects belonging to other clusters as shown by the Figure3-1 given below.

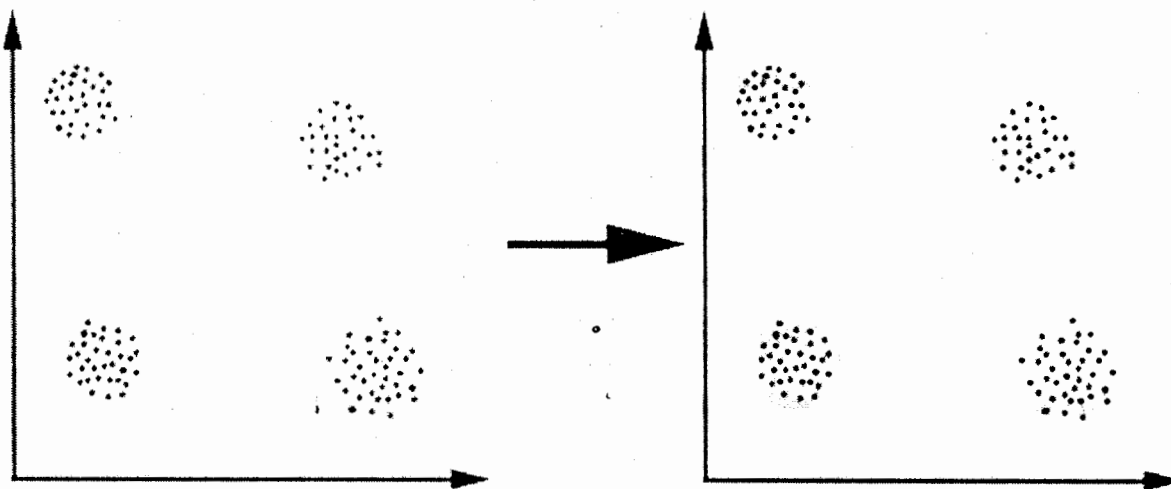


Figure 3-1

In the above case four clusters can easily be identified into which the data can be grouped. The similarity criterion between the objects is distance i.e. two or more objects belong to the same cluster if they are ‘close’ according to a given distance (in this case geometrical distance). This is called distance-based clustering [11].

Another kind of clustering is conceptual clustering where two or more objects belong to the same cluster if they possess a common concept. In other words, objects are grouped according to their fitness to descriptive concepts (Features), not according to simple similarity measures. This kind of clustering can be implemented using methods like Statistical, Syntactical and Neural Network. The method we adopted is Neural Network. Neural Network further has different types like SELF-ORGANIZING MAPS (SOM), Learning Vector Quantization (LVQ1, LVQ2) etc. We have used SOM Neural Network known as KOHONEN SELF-ORGANIZING MAPS [11].

3.2 KOHONEN SELF-ORGANIZING MAPS

We have considered the self-organizing Map (SOM) developed by Kohonen given in Figure 3-2, which groups the input data into clusters, a common use of unsupervised learning. In Kohonen learning, the algorithm begins training by using squared Euclidean distance between input vector and weight vector and chooses the unit whose weight vector has the smallest Euclidean distance from the input vector such as: [9]

$$D(j) = \sum (w_{ij} - x_i)^2$$

Where, 'D' is the total distance after summing the values, 'j' is the input vector. w_{ij} is the weight from weight matrix depending on the i & j values (i being index for values of input vector, and j is number of vector), and x_i is the vector values.

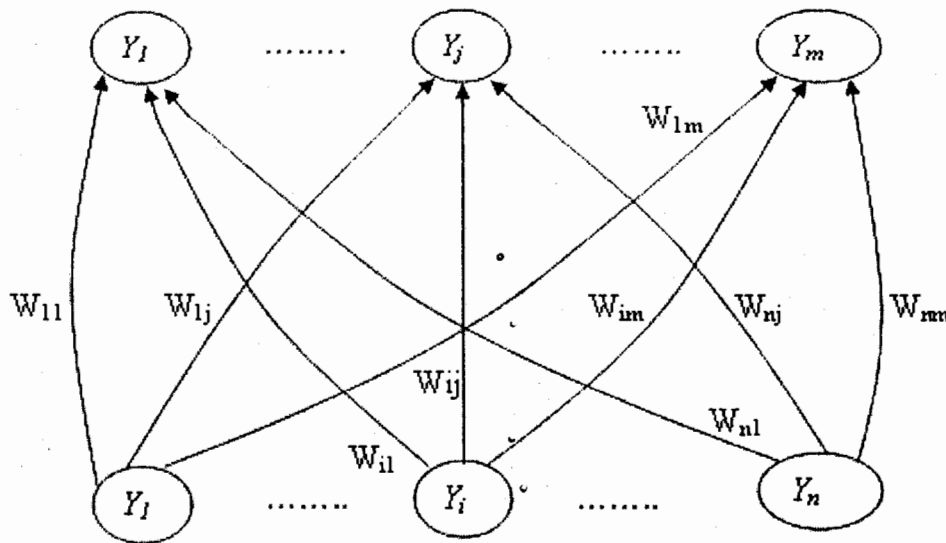


Figure 3-2 Kohonen Self-organizing map

The units that update their weights do so by forming a new weight vector that is a linear combination of the old weight vector and the current input vector.

The weight update for output unit j is given as:

$$W_j(\text{new}) = W_j(\text{old}) + \alpha [x - W_j(\text{old})]$$

$$= \alpha x + (1 - \alpha) W_j(\text{old})$$

Where x is the input vector, W_j is the weight vector for unit j (which is also the j^{th} column of the weight matrix) and the learning rate (α) that decreases as learning proceeds.

The Alphabet Set given in Table 1-1 contains 106 characters which are actually the input patterns. Out of these 106 patterns some of them are very similar to each other e.g. characters: پ, ب, ث, ت have same main region irrespective of the number and position of the dots, so by observing we can count these four characters as one class ب. This way decreases the number of total input patterns from 106 to 54 as shown in Table 3-1 given below. This decrease in number of input patterns has advantage that it is easier to classify less number of input patterns.

Supported Characters	Class	S.No	Supported Characters	Class	S.No
ع	ع	28	ا	ا	1
ع	ع	29	ا	ا	2
ع	ع	30	پ ب ث ت	ب	3
ف	ف	31	ب ب ب ب ب	ب	4
ف	ف	32	ب ب ب ب ب	ب	5
ف	ف	33	ب ب ب ب ب	ب	6
ف	ف	34	ب ب ب ب ب	ب	7
ف	ف	35	ب ب ب ب ب	ب	8
ف	ف	36	ب ب ب ب ب	ب	9
ک	ک	37	ب ب ب ب ب	ب	10
ک	ک	38	د	د	11
ک	ک	39	د	د	12
ک	ک	40	ر	ر	13
ل	ل	41	ر	ر	14
ل	ل	42	س	س	15
ل	ل	43	س	س	16
ل	ل	44	س	س	17
م	م	45	س	س	18
م	م	46	ص	ص	19
م	م	47	ص	ص	20
م	م	48	ص	ص	21
ن	ن	49	ص	ص	22
ن	ن	50	ط	ط	23
و	و	51	ط	ط	24
و	و	52	ط	ط	25
ی	ی	53	ط	ط	26
ی	ی	54	ع	ع	27

Table 3-1 General Classes

The Clustering was applied on the 54 classes as in Table 3-1. Initially, we trained the classifier with topology of '6 × 5' grid, the result obtained for Epochs 75 is shown in Table 3-2. This gives 23 clusters. Cluster # 17 has 'م-ع-م' characters, which are totally distinct characters. Similarly cluster# 4,19,22,26 etc. have totally dissimilar characters.

Epochs 75, grid 6×5			
#	Characters	#	Characters
1	و-ص-و	19	ل-ا-ا-م
3	ط-س-ه	20	ل
4	ح-ز-ا-ط	21	ع
5	ع	22	ق-ب
7	د-ر-ط	23	ی-ب
10	آ-ک-ا	24	ن
11	و-ه	25	ق-ک-ص
12	ط-د	26	ح-ح-س-ع
13	ل	27	س
15	ع-س	28	س-س
16	ص-ح	30	ف-ص-ک
17	م-ع-م		

Table 3-2 Clustering Table formed for grid 6×5 with 75 Epochs

Then keeping the same topology of '6 × 5' grid, but changing the number of Epochs from 75 to 100. Fourteen 14 clusters were formed which are shown in Table 3-3 given below. In this case result obtained was better for some of the clusters e.g. cluster#26 which consists of five characters و-ص-و-ق-ق all of them have loop at their start and only one end point. But some of the clusters did not show good result e.g. cluster#5 which consists of five characters د-ر-ل-ط-ط combining having looped and non looped characters. Also the characters having loop in the middle, are scattered in different clusters. Table 3-3 gives the result below.

Epochs 100, grid 6×5	
#	Characters
2	ح ح س ع
5	د ر ل ط
6	ا ل م
8	و د ص
9	ر
12	ب ی ش م
15	ع ح ل د س ه
16	س ن ی ی ی ب
19	ط
22	ص ص ع ی
25	آ
26	ف و و و
28	ل
30	ع س ل

Table 3-3 Clustering Table formed for grid 6×5 with 100 Epochs

Increasing Epochs to 150 for the same topology of '6 × 5' grid, gave 12 clusters. The result is shown in Table 3-4 given below. Its result is not better because different shaped characters are in the same cluster e.g. cluster#10 containing characters ک ک ل ص ص ص . Similarly in cluster#12 etc.

Epochs 150, grid 6×5	
#	Characters
1	د ر ل
2	ط ط ع ی ی م و ر ص
3	د
4	ط ط
5	ا ل ن
6	ا م
7	ح ل س ع ه
8	ب ب س س ی ی
9	آ ا ع ک ل
10	ص ص ی ی ک ل
11	و و و
12	ح ح س ع

Table 3-4 Clustering Table formed for grid 6×5 with 150 Epochs

Epochs 200 with topology of '6 × 5' grid, gave 13 clusters as shown in Table 3-5 below. Cluster#1 has ح-ع-س-ك-ل-ي characters which are not similar w.r.t their features and also Cluster#13 contains characters ب-ب-ب-ب-ب-ب, all of them are disimilar. Cluster#9 contains both looped and non looped characters.

Epochs 200, grid 6×5	
#	Characters
1	ح-ع-س-ك-ل-ي
3	و-و-و
5	ع-ز-ح
7	ح-ح-س-ع-ع
9	د-ص-ط-ط-ه-و ر
11	ل-س-ه
13	ب-ب-ب-ب-ب-ب
15	ص-ص-ط-ط
17	ا-ا
19	س-س-ف-ف-ك-ك
26	د-ر-ر
28	ا-ل-ل
30	ف-ف

Table 3-5 Clustering Table formed for grid 6×5 with 200 Epochs

Then we changed the scenario by changing the topology from '6 × 5' grid to '8 × 7' grid. For Epochs 75, we got 27 clusters whose result is given in Table 3-6 shown on the next page. This clustering gave many clusters as compared to previous results. Also number of characters per cluster has decreased for this grid. Most of the clusters here have similar characters but some still show not better result e.g. cluster#37 has characters م-ا-ل all of them are totally different. Result is given in the following Table 3-6.

Epochs 75, grid 8×7			
#	Characters	#	Characters
1	ک-ک-ل	32	ک
4	ح-ع	35	ب
7	د-ح-ع	37	ل-ا-م
10	ر	39	ح-ع
11	ر-ط	40	ک-ز
13	ع	41	ص-و
14	ح-س-د	43	ن-ل
16	ل-س-د	44	ی
17	ص-ص	46	ب-ل-م-ی
19	ط	50	و
23	و-د-ر	53	س-س
26	ف-ق	55	ط
28	ا-ر	56	ط
30	ع-ص-ح		

Table 3-6 Clustering Table formed for grid 8×7 with 75 Epochs

Table 3-7 is result of 100 Epochs with topology of ' 8 × 7 ' grid. This result gave 27 clusters. Like previous result it has also more clusters and less characters per cluster.

Epochs 100, grid 8×7			
#	Characters	#	Characters
2	ط-ط	32	ص
4	م-ی	33	ک-ک-ل
7	و-ر	35	ن-ی-ب
8	د	37	ح-ع
9	س	39	ف-ق
11	ر	42	ص
14	ع-ص-ح	44	د-ح-ع
17	س	46	ل-س-د
19	ر-ر	49	ص
21	ب-ل	51	ط-ط
24	و-و	53	ک-ک
26	ن-ل-ا	55	ح-ع
28	ا-م	56	ر
30	ح-س-د		

Table 3-7 Clustering Table formed for grid 8×7 with 100 Epochs

Table 3-8 is result of 150 Epochs for topology of ' 8 × 7 ' grid. It gave 23 clusters. Cluster#30 has ی-ن-ل-م-ی all of them are dissimilar.

Epochs 150, grid 8×7			
#	Characters	#	Characters
2	ل-ر-ر	30	م-ف-ن-ی
4	ا-ل	32	ص-ص
7	ق	34	و-و
9	ط-ط	37	ط-ط
11	ا-م	39	ک-ک
14	ن-ن	44	ک-ک
16	ف-ص	46	د-س-ن-و
18	و-د-و	48	ل
20	ق-م	49	ح-ع
23	س-س	53	ر-ح-ع
25	ح-س-د-و	55	ع-ح-د
27	و-ص-و		

Table 3-8 Clustering Table formed for grid 8×7 with 150 Epochs

Table 3-9 gives result of 200 Epochs for topology of '8 × 7' grid. This clustering gave 19 clusters. Its result is not better e.g. cluster#33 'و-د-ک-ع-ر-ح' contains different shaped characters similarly cluster#35 has characters 'د-س-ن-و' which are totally different.

Epochs 200, grid 8×7			
#	Characters	#	Characters
1	ف-م-ق	33	و-د-ک-ع-ر-ح
3	و-ص-و	35	د-س-ن-و
6	ح-س-د-و	37	ن-ن
8	ح-ع	40	ص-و-و
17	م-ن-ی	44	ط-ص-ص
19	ط-ط	47	ر-ک-ل
21	د-و	51	ع-ح
26	س-س	53	ط
28	ل-ر-ر	56	ف-ق
30	ل-ا-ا-م		

Table 3-9 Clustering Table formed for grid 8×7 with 200 Epochs

Then increasing topology from '8×7' to '9×8' grid. Applying clustering for 75 Epochs we got result shown in Table 3-10 given below. It gave 29 clusters. Its result is better except in some clusters e.g. cluster#12 contains د-س-و which all are dissimilar.

Epochs 75, grid 9×8			
#	Characters	#	Characters
2	ک-ز	36	ص-و-ح
3	ح-ع-ک	38	ص-و-و
5	ر-ر	40	ف-ق
6	ر	43	ح-س-د
9	ح-ع-د	45	ا-م
12	د-س-و	47	ص
15	ط	49	س-س
16	ط	52	ا-ل
18	و-د	55	ک
19	ر	59	ب-ن
22	ق-م	62	ل
25	ع	64	ک
29	م	65	م-ن-ی
33	ح	69	ط-ط
		71	ص

Table 3-10 Clustering Table formed for grid 9×8 with 75 Epochs

Table 3-11 is the result of 100 Epochs for topology of '9×8' grid. It gave 28 clusters. Its result is better but some of the clusters gave quite unsatisfactory result as in cluster no.45 'ح-ز-ع' and similarly in cluster#35 which consists of د-س-و. Also characters per cluster were also less i.e. at most four characters per cluster.

Epochs 100, grid 9×8			
#	Characters	#	Characters
2	ا-ا	35	د-د-د
4	ب-ب-ب	38	ط-ط
6	س	40	ح-ح
7	ف-ف	45	ح-ر-ع
9	س	49	و
11	د-ر-ا	51	ک-ک
16	ح-س-د	54	ص
17	س	56	ل
21	د-ر	57	و-و
23	ح	60	د
26	ب-ل-ن	63	ک-ک
28	ط-ط	66	ح-ح
31	ص-ص	70	ص
33	ف-ف	72	و

Table 3-11 Clustering Table formed for grid 9×8 with 100 Epochs

Table 3-12 is the result of 150 Epochs for topology of '9×8' grid. It gave 25 clusters. It is slightly better but some of the clusters give totally unsatisfactory result like cluster#52.

Epochs 150, grid 9×8			
#	Characters	#	Characters
4	ص-ک-ک	42	ح-ح
6	ب-ب	45	د
9	ص	48	د-ر-ا
13	ل	49	د
16	س	52	د-د-د
18	ط	54	ط-ط
23	ح-ع-ا	59	ح-س-د
28	ص	61	ا-ا
30	و-و	64	ا-ل-ن
32	ط-ک-ک	67	ص-ص-ح
33	و	68	ب-ب-ب
35	ف-ف	70	ب
39	ح-ر-ع		

Table 3-12 Clustering Table formed for grid 9×8 with 150 Epochs

Table 3-13 is result of Epochs 200 for topology of '9×8' grid. It gave 22 clusters. Resultwise it is almost same as the previous result of Table 3-12.

Epochs 200, grid 9×8			
#	Characters	#	Characters
3	ح-ع	33	ط-ط
5	ی-ی	35	ر-د
8	ف-ف	40	ح-ز-ع
9	ج-س-د-ع	50	و-ل
15	ص	52	ا
19	س-س	54	ح-ع
22	ف-ف-م	57	ر-ر
24	ص-ص	61	ک-ک
26	ص-د-ع	64	و-ل-س-ک
29	ب-ل-ن	67	ا-ب-ل-م
31	ط-ط	71	د-س-و

Table 3-13 Clustering Table formed for grid 9×8 with 200 Epochs
Changing topology from '9 × 8' to '10×10' grid. Taking 75 Epoches, we got result shown in Table 3-14. It gave 32 clusters.

Epochs 75, grid 10×10			
#	Characters	#	Characters
1	ی-ی	49	ح-ع
5	ص-و	55	ا
9	س-س	61	ح-ع
12	ب-ب	63	د
17	ف-ف	68	ر-د
20	ک-ک-ل	75	ل
21	ف	76	ن
24	ط	78	و
26	ط	80	ر-ر
28	ص	83	و
29	ص	84	س
31	م	87	ب
33	ح-ز-ع	89	ط-ط
42	ل	95	ج-س-د-ع
44	ک-ک	98	ص-د-ع
46	ا	100	ل

Table 3-14 Clustering Table formed for grid 10×10 with 75 Epochs

Table 3-15 is the result of 100 Epoches for topology of 10×10 grid. It gave 32 clusters. Its result is almost same as previous result of Table 3-14

Epochs 100, grid 10×10			
#	Characters	#	Characters
1	و	49	ح - س - ع
4	ط - ط	53	ب
6	ق - ف	55	ک - ک
8	ص - ع - ه	60	ح - ز - ع
10	ح - ع	62	ل
15	م	69	د
19	ح - ع	71	ا
21	ر	74	ن - ی
23	د - ر	77	ل
27	ع	85	س - س
34	ط	88	و - و
35	ط	90	س - ه
38	ر	92	ن
40	ک - ک - ل	94	ب
41	م - ا	97	ق
46	ص - ص	99	ص - ف

Table 3-15 Clustering Table formed for grid 10×10 with 100 Epochs

Table 3-16 gives result of 150 Epoches for topology of 10×10 grid. This result gave 27 clusters. This clustering also gave a bit better.

Epochs 150, grid 10×10			
#	Characters	#	Characters
1	ل-ق	50	م-ن-و-ح-ص
4	ک-ک-ف-جس	53	س
7	ا	56	ص
10	ل-ل	58	د
16	ص	67	ک-ک
18	ر	70	ل-ر-ر
22	د	74	ع-س-س-ح
27	م-ا	78	ن-ع-س-ر-ح
29	ی-ن-ن	82	س
33	ط	86	و-و
35	ط	95	ع-ح
38	م-ب	97	د
41	ع-ح	100	ط-ط
44	و		

Table 3-16 Clustering Table formed for grid 10×10 with 150 Epochs

Table 3-17 is result of 200 Epoches topology of '10 × 10' grid. This Clustering gave 29 clusters. This result is also satisfactory as compared to other results of the 100 clusters of different Epoches.

Epochs 200, grid 10×10			
#	Characters	#	Characters
1	م-ا	56	ص
3	ل-ا	58	ی-ں
7	ہ	60	و-و
9	ط-ط	62	ص
12	ط	64	د
14	ل-ر-د-د	67	ع-ح
18	ہ-ف-ع-ص	73	ق
21	ط	79	ع-ح
30	س-ک	86	ن-س
33	ک-ک	88	د
36	ن-ب	90	ع-س-ح
42	ص	92	ف
44	و-و	94	ع-ز-ح
47	س-س	97	ک-ک
53	ل		

Table 3-17 Clustering Table formed for grid 10×10 with 200 Epochs

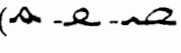

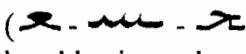
3.3 Clustering Comparison

On the next page, Table 3-18 shows overall results obtained by previous clustering. 'Topology' column gives the grid used. 'Epochs' is the number of epochs applied on the grid for clustering. 'Clusters Formed' is the total number of clusters obtained. 'Minimum' is the least number of characters in a cluster whereas 'Maximum' is the maximum number of characters present in a cluster. It can be easily seen that the greater the Topology (grid) the more will be the number of clusters and also number of characters per cluster decreases.

S.No	Measurements			Characters in One Cluster	
	Topology	Epochs	Clusters Formed	Minimum	Maximum
1	6 × 5	75	23	1	4
2	6 × 5	100	14	1	6
3	6 × 5	150	12	1	10
4	6 × 5	200	13	2	8
5	8 × 7	75	27	1	4
6	8 × 7	100	27	1	3
7	8 × 7	150	23	1	4
8	8 × 7	200	29	1	7
9	9 × 8	75	30	1	4
10	9 × 8	100	28	1	3
11	9 × 8	150	25	1	4
12	9 × 8	200	22	1	5
13	10 × 10	75	33	1	3
14	10 × 10	100	33	1	3
15	10 × 10	150	27	1	5
16	10 × 10	200	29	1	4

Table 3-18 Results of Clustering

In the light of above experimental results, we preferred clustering of 10 × 10 grid with 100 epochs (Table 3-15) because it gave much satisfactory results because all of its clusters have almost similar patterns e.g.

- The cluster # 8 () has characters that have 3 end points and a loop in the middle.
- Its cluster # 19 () has characters that have 4 end points and a curve towards right.
- Its cluster # 49 () has characters that have 4 end points and having start and end horizontal.

3.4 Effect of Cluster Topology

We applied different size of topology of 6 × 5, 8 × 7, 9 × 8 and 10 × 10 order grids. Change in topology put significant impact on the clustering. We observed that

- Increase in topology increases the number of Clusters but ratio of the clusters w.r.t topology is greater.
- Increase in Topology decreases number of characters per cluster.

3.5 Effect of Cluster Size

Cluster of greater size means cluster having more characters. If a cluster contains more characters then it becomes difficult to classify each of them. More the characters per cluster more features will be required to classify them in the same cluster.

3.6 Effect of number of Epochs

We applied different number of epochs 75,100,150 and 200 on each of the topology used. The effect of number of epochs on the clustering was that

- Increase in number of epochs for the same topology, decrease the number of clusters formed.
- Increase in number of epochs for the same topology, increases number of characters per cluster.

3.7 Final Cluster Table

The clusters found by 10×10 grid with 100 epochs (Table 3-15) gave a weight matrix of order 18×100 . But we faced problem that for the same new Input vector, an unwanted cluster was obtained. e.g. ط is in cluster# 4, if ط is given with the same features, it used to give cluster#3 rather than cluster#4. To overcome this problem we took only those columns of the Weight Matrix which corresponded to the clusters of Table 3-15. So we got a matrix of 18×33 order. This matrix was then applied with 54 classes each having 18 features. The final clusters obtained are given in Table 3-19 as shown below.

Final Clusters	
#	Characters
1	د-ز-ح-ع-س-ب ک-ک-ک-ک-ط-د-و
2	ه-و-ط-ط
4	ل-و
5	ح-ح-ع-ط-س-ب
6	ج-س-د-ع
8	ف-ق
9	ص-ص
10	ص-ه
12	ط-ز-ر-ر
14	م-ا-د
15	ع
18	س-س-س-ن
21	ل-ا-ی
22	ف-ب-ج-ه
29	ن-ب

Table 3-19 Finally Obtained Clusters by manipulating Clusters found by 10×10 grid with 100 Epochs

3.8 Summary

106 characters of Alphabet Set are manually converted into 54 classes on the basis of their same main regions. On these 54 classes, we applied Kohonen SOM Neural Network with 18 features. We applied different size of topologies of 6×5 , 8×7 , 9×8 , 10×10 order grids. For each topology we used different number of epochs 75, 100, 150 and 200. The clusters formed after 100 epochs with topology of 10×10 grid was selected. It gave 33 clusters, which are then converted into 15 clusters by decreasing the order of Weight Matrix from 18×100 to 18×33 . These 15 clusters as shown in Table 3-19 are used by Classification-I to find cluster for an input pattern. The Clustering was very important and its final result is the basic need of proposed system because Classification-I phase totally depends upon the final result of Clustering.

CHAPTER NO 4

**PROPOSED
METHODOLOGY**

4. PROPOSED METHODOLOGY

This chapter gives description about proposed system and its phases thoroughly. Each Phase along with its sub stages is discussed subsequently.

4.1 Introduction

For recognition, two types of recognizer can be used offline or online. Offline recognizer takes text as input in the form of image(s). Then that image is preprocessed and made easily capable for extraction of the features that help the recognizer classify the text. This recognizer can either recognize whole the word or can segment a word into its constituent characters. In the latter case segmented characters are then preprocessed and features are extracted in order to recognize them and hence the whole word is recognized whereas online recognizer involves the automatic conversion of text as it is written on a special digitizer, where a sensor picks up the pen-tip coordinates $[X(t), Y(t)]$ as well as pen-Up/pen-Down switching. The obtained signal is converted into letter codes which are usable within computer and text-processing application.

Every recognizer uses one of three basic classification methods which are Matrix/Template matching Recognition, Structured Pattern Recognition, Neural Network Pattern Recognition. Matrix/Template matching Recognition method compares unknown character with a library of character templates or prototypes. Structured Pattern Recognition uses Feature Extraction which is the process of without strict matching to templates prescribed as matrices. Neural Network Pattern Recognition can be either supervised or unsupervised. Supervised Training is accomplished by presenting a sequence of training vectors, each associated with a target output-vector. Unsupervised training, a sequence of input vectors is provided, but no target vectors are specified

Methodology used for our system is offline character recognition. As recognition can be either word, ligature or character based. Our system uses character based recognizer. As character based recognizers depend upon segmentation. Therefore our recognizer is segmentation based rather than segmentation free. As far as classification method is concerned, our recognizer is unsupervised Neural Network Pattern Recognizer.

4.2 Proposed System

Our methodology described in Figure 4-1, assumes only segmented characters and hence it can be used with any of the host system (Ligature/word based recognition). Initially system gets the segmented character as image and passes it through some steps which convert it into gray level and then into binary image. System applies some preprocessing techniques as in [9], on binary image and thins the image because thinned form of the image is more appropriate for the extraction of features. Then another preprocessing technique takes the thinned image and removes the edges which are developed during thinning and may cause features to be invalid. After these preprocessing techniques

system gives the preprocessed image to feature extraction phase [8], during this phase image goes through a number of procedures which find the features from the image. After getting enough features of the image, we apply different weights already fixed, to give more satisfactory results for clustering. Then classification phase activates its sub phase classification-I, which uses weight matrix of order 18×33 and gives only one of 15 clusters for 106 segmented characters. As each cluster has more than one characters therefore the sub phase classification-II uses additional features to classify each character from all other characters of the same cluster.

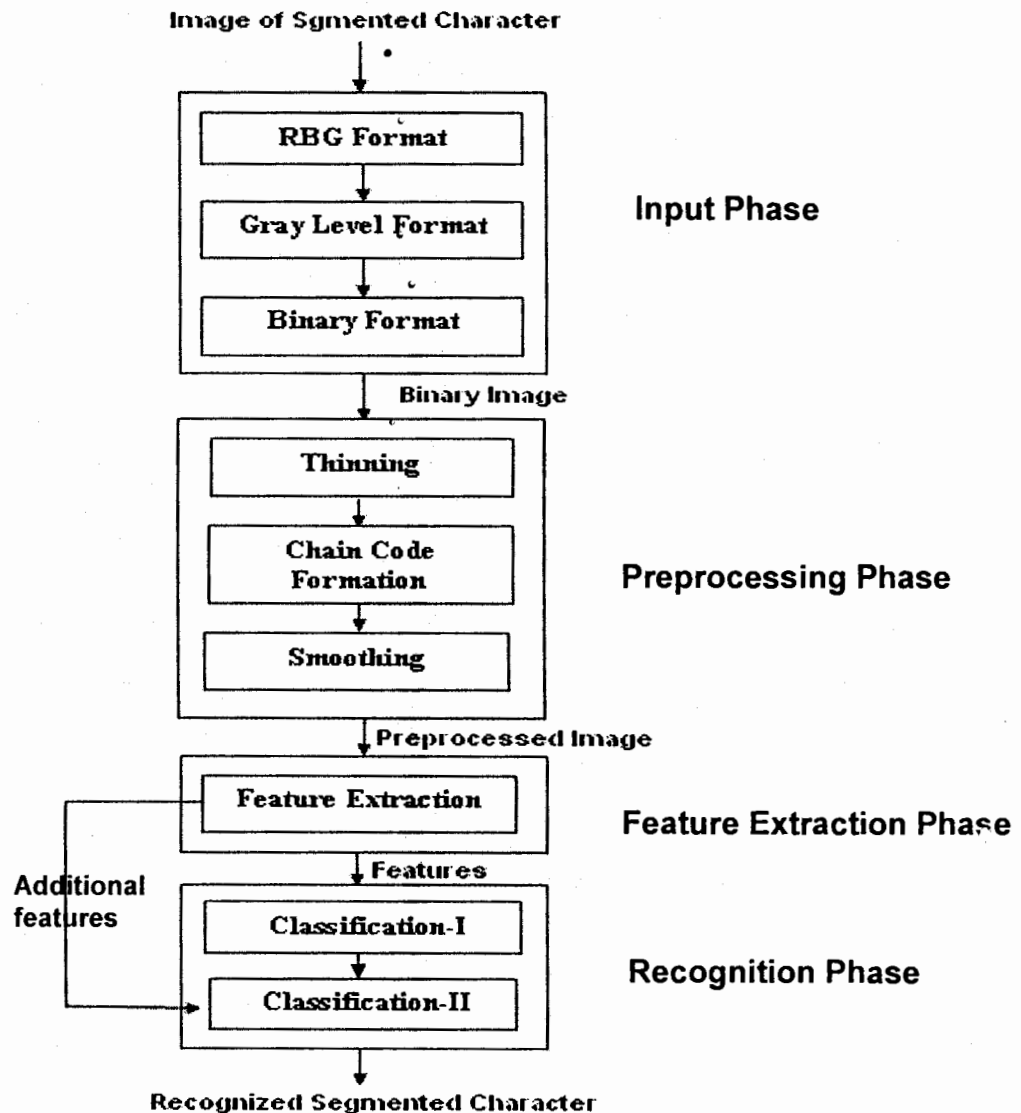


Figure 4-1 Block Diagram of the Proposed System

4.3 System Phases

Our system contains four phases discussed below.

4.3.1 Input Phase

The image obtained is converted into binary format which is appropriate for the feature extraction, this phase involves following steps:

4.3.1.1 Acquisition of image: System takes already saved image in JPEG/JPG format. This image is RGB but it may be a Gray level image.

4.3.1.2 Conversion from RGB to Gray level: This step converts the RGB (colored) format of image into Gray level, the levels are in between '0' and '255'. If image is already Gray level then this step does not put any impact on it.

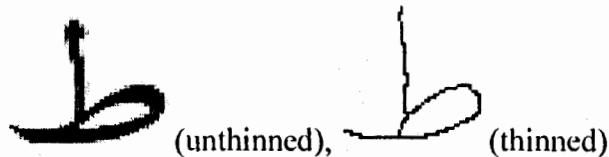
4.3.1.3 Conversion from Gray level to Binary: This step converts the Gray level image into Binary format which contains only two levels 0 and 1. For background, value 0 is used while for foreground value 1 is used.

4.3.2 Preprocessing Phase

This phase deals with various preprocessing required to enable Feature Extraction. This phase takes image in the form of binary matrix and applies some necessary operations on this matrix. The steps involved here are Thinning, Chain Code Formation and Smoothing which are discussed below.

4.3.2.1 Thinning

This preprocessing uses the binary image and makes the image one pixel wide. Thinned image plays vital role in the feature extraction phase because features can easily be extracted from it and it also reduces the complexities faced in feature extraction phase. Thinning is achieved using the well known Zang Susan algorithm.



4.3.2.2 Chain Code Formation

Chain Code Formation is the process of converting a one pixel wide image into linear vector of unit values. Chain Code used in the proposed system is 8-directional i.e. system assumes codes associated with eight possible directions.

If 'c' is current pixel being processed, then eight possible directions around it are given as

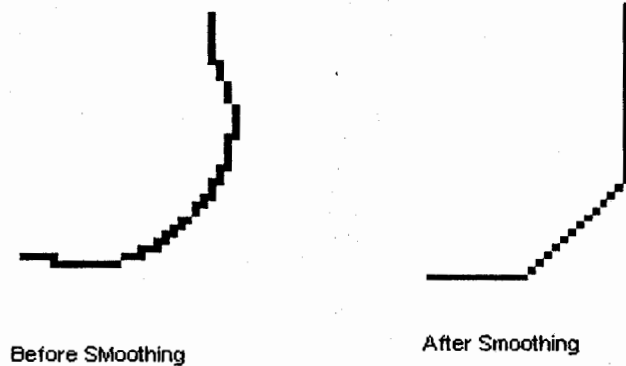
$$\text{Chain Code} = \begin{matrix} & 3 & 2 & 1 \\ & 4 & c & 0 \\ & 5 & 6 & 7 \end{matrix}$$

Chain Code formed for 'Ra isolated' is

6,6,6,6,6,6,7,6,7,6,6,7,6,6,6,5,6,6,6,5,6,5,6,5,5,6,5,4,5,5,5,5,4,5,4,5,4,4,4,4,4,4,3,4,4,4,4.

4.3.2.3 Smoothing

Smoothing is somehow filtering process. It is based on Chain Code obtained in the previous preprocessing step. Proposed system uses one pixel smoothing i.e. if a single (not more than one) value appears in the code sequence then it is converted into value same as the value just preceding it. Following are the images and codes for 'Ra isolated' showing chain code before and after the smoothing process.



Original Chain Code

6,6,6,6,6,6,7,6,7,6,6,7,6,6,6,5,6,6,6,5,6,5,6,5,5,6,5,4,5,5,5,5,4,5,4,5,4,4,4,4,4,4,3,4,4,4,4.

Chain Code after Smoothing

6,5,5,5,5,5,5,5,5,5,5,5,4,4,4,4,4,4,4,4,4,4,4.

Preprocessed (thinned as well as smoothed) image is now ready for feature extraction phase.

4.3.3 Feature Extraction Phase

4.3.3.1 Features Extracted

Features are very important for recognition. In this phase different procedures are used to extract the features with the help of which system recognizes the characters.

- **Height:** This feature is set when the character under observation has height greater than width e.g. ل, خ, م
- **Width:** This feature is set when the character under observation has width greater than height e.g. ب, ف, ک
- **Loop:** Loop is closed path. It is set for all those characters which have closed path e.g. ط
- **Loop_M:** This feature is set for only those characters which contain loop but in the middle of the character e.g. ظ, م
- **Loop_S:** This feature is set for only those characters which contain loop but in the beginning of the character e.g. ص, ق
- **Cross:** this is set for character which contains crossover, the point from which four different ways exit e.g. ق, و
- **Curve_R:** This feature is selected when the character has curve toward rights side e.g. ج, ع
- **Curve_U:** This feature is selected when the character has curve toward up side e.g. ن, ص, س
- **Start_H:** It is set for characters that have horizontal start e.g. ط, ب
- **Start_V:** It is set for characters that have vertical start e.g. ن, ش
- **End_H:** It is set for characters that have horizontal end e.g. ذ, و
- **End_V:** It is set for characters that have vertical end e.g., ی, ا
- **Endp_1:** This feature is selected when the character has only one end point e.g. و, ف
- **Endp_2:** This feature is selected when the character has two end points e.g. پ, ع
- **Endp_3:** This feature is selected when the character has 3 end points e.g. ع, ک
- **Endp_4:** This feature is selected when the character has 4 end points e.g. ح, م
- **Joint:** It is point from which two or three ways exit as in characters. e.g. ظ, ک, ح
- **Joint_1:** It is selected when character has only one joint e.g. ذ, ف
- **Joint_2:** It is selected when character has two joints e.g. ط, ح
- **Joint_3:** It is selected when character has three joints e.g. ط, ظ

4.3.4 Recognition Phase

4.3.4.1 Classification-I:

4.3.4.1.1 Weight Matrix Formation:

Well structured Weight Matrix plays vital rôle in the classification. In order to get the proper weight matrix we invoke a learning process that updates matrix each time it learns about new input vector. After getting enough features of the image through Features Extraction, we apply Kohonen Neural Network known as Self-Organized Map (SOM). With a rectangular topology of grid 10×10 , Neural Network trains the system. After 100 epochs SOM-Neural Network finds the Weight Matrix W .

4.3.4.1.2 Clustering:

A cluster is the group whose members are similar in some way. The Weight Matrix formed in the last stage, is used to form the clusters of given 54 Input Vectors, which result in 15 clusters each of which contains similar Input Vectors. In our case, one Input Vector is collection of the features of one segmented character. Therefore each of 15 clusters contains group of similar segmented characters.

Classification-I uses weight matrix of order 54×18 and gives only one of 33 clusters for each segmented character. Generally, sub phase Classification-I takes the features and finds the cluster in which the segmented character being observed fits best.

4.3.4.2 Classification-II:

As Classification-I only find the cluster for the character, and some clusters have more than one character so some additional features are also required to distinguish the character in the same cluster.

4.3.4.2.1 Additional Features

- **Dots:** this feature is set when the character being processed contains dots
e.g. ب, ف, ج, غ, ٲ
- **Dots_1:** it is set when number of dots in the character is only one
e.g. ب, ج, ن, ف
- **Dots_2:** it is set when number of dots in the character is two e.g. ت, ق, ٲ
- **Dots_3:** it is set when number of dots in the character is three e.g. پ, ش, ج
- **Dots_A:** this feature is set when the dot(s) in the character are above the main region of the character e.g. ذ, ق, ت
- **Dots_B:** this feature is set when the dot(s) in the character are below the main region of the character e.g. ٲ, ج, ٲ
- **Dots_M:** this feature is set when the dots in the character are in the middle of the character e.g. ج, ج

On the basis of above features,106 character shapes gave 54 classes, as shown in Table4-1. These 54 classes are then converted into 15 clusters,after which character is classified.

Supported Characters	Class	S.No	Supported Characters	Class	S.No
ع غ	ع	28	ا	ا	1
ع ج	ع	29	ا	ا	2
ع خ	ع	30	پ ب ث ت	ب	3
ف	ف	31	ب ت ی ژ ز	ر	4
و	و	32	ب ی ت ت ش ز	س	5
ف ق	ف	33	ب ا ت ث	س	6
ف ح	ف	34	ح	ح	7
ق	ق	35	ج ح ذ	ح	8
ق ح	ق	36	ج ح ی ز	ح	9
ک	ک	37	ح	ح	10
ک	ک	38	ذ	د	11
ک	ک	39	ذ	د	12
ک	ک	40	ر ز	ر	13
ل	ل	41	ر ز	ر	14
ل	ل	42	س ش	س	15
ل	ل	43	س ش	س	16
ل	ل	44	س ش	س	17
م	م	45	س ش	س	18
ن	ن	46	ص ض	ص	19
ن	ن	47	ص ض	ص	20
ن	ن	48	ص ض	ص	21
ن	ن	49	ص ض	ص	22
س	س	50	ط ظ	ط	23
و	و	51	ط ظ	ط	24
و	و	52	ط ظ	ط	25
ی ی	ی	53	ط ظ	ط	26
•••••	•	54	ع غ	ع	27

Table 4.1

4.4 Implementation

Implementation was carried out in MATLAB 7.0 as this software is rich enough to provide all the necessary built-in functions and techniques used in the proposed system. Not only had this but MATLAB also provided facility to create standalone application related to system. Whole the Code for Testing was written in MATLAB programming and Graphical User Interface was also created in it.

CHAPTER NO 5

TESTING

5. TESTING

This chapter describes the testing procedure for the proposed system. Not all but some of the examples showing sentences with segmented characters are given here. Also some Samples are given in order to understand the testing data.

5.1 Analysis

As our system is for recognition of segmented Urdu characters, therefore, for testing, we need already segmented characters. For this purpose we took different sentences written in Urdu Nasakh font. Each sentence contained number of words and each word comprised number of characters which collectively made this word meaningful. Initially, sentence was made by using the Urdu Software 'In page 2.0', and then the words from the sentence were separated through software MS-Paint. Finally, for the input of our system we segmented the separated words to form segmented characters. After that each of the segmented characters was converted into '.JPEG/.JPG' formatted image using MS-Paint software. This image was used for further processing in the recognition system.

5.2 Methodology used for testing

Actually, we had to test the segmented character such as *ض, چ, ع, ق, ت*, so we cut the character from start, middle or from the end of the word such as *مقا, (ق) قن*, *(ق) جق, (ق)*. For this purpose we applied more than sixty samples and fifteen different sentences in order to experience the result. A character, if cut from the same place (from start) in different words as shown below, may give different results. However, we had handled this technical problem and after that we reached to the final satisfactory results. We obtained almost 85% of accuracy given in Table 5-1. Now we are giving different examples of sentences, which show the different technical aspects, which occur in the recognition of segmented characters:

5.3 Tested Data in the form of sentences

The following examples explain testing in detail.

Example#1

In the example given below, a sentence is given in the Urdu Nasakh Font.

موت کو یاد کرو
موت کو یاد کرو

Figure 5-1 Sentence with four Words

This sentence contains four words. Each word is then decomposed into its constituent characters. These characters when separated are called segmented characters. These segmented characters were used as input into the system for testing purpose. Its result was 90%, means only one character was falsely recognized while the rest were recognized correctly.

Example#2

This example also contains four different words but it is slightly complex

ذرا نرم باتیں بتانا

ذرا نرم باتیں بتانا

Figure 5-2 Sentence with Four words

sentence due to the presence of more dotted characters. Its segmented characters are 16 in number, three of which 'ذ - ب - ا' are repeated. Here also result remained 90%.

Example#3

The sentence in this example contains five words out of which three contains loop (closed path). Its words when segmented give 15 segmented characters shown below the Urdu sentence.

تم ظالم مت بننا یار

تم ظالم مت بننا یار

Figure 5-3 Sentence with Five Words

Its result was 85% due to character meem which was recognized falsely.

Example#4

This example involves a bit complicated words which contain complex and more characters. These three words give 16 segmented characters.

پروفیسر لیکچر دیتا

پروفیسر لیکچر دیتا

Figure 5-4 Sentence with Three Words

When these segmented characters were applied for recognition. We got 100% result for none of the characters was unrecognized.

Example#5

The following example is well familiar collection of the words often used in different places. Its distinct characters are 25 in number.

چار اصول صداقت
امانت شجاعت دیانت

چار اصول صداقت دیانت
شجاعت دیانت

Figure 5-5 Sentence with Six Words

Testing of these segmented characters also gave 90% results.

*A SOM Neural Network based Offline Recognition System
for Segmented Urdu Character in Nasakh Font.*

5.4 Tested Data in the form of Samples

5.4.1 TA of START Sample

In this sample, combination of ت 'TA of START' with different characters is shown. Under the sample words each TA of START is shown when segmented, because segmentation may change the length of character. Its result was 90%.

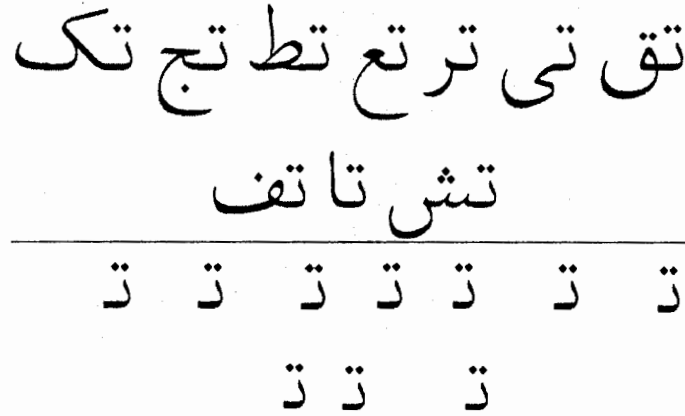


Figure 5-6 Sample for TA of START

5.4.2 YA of MIDDLE Sample

This sample describes combination of ے 'YA of MIDDLE' with different characters. When YA is segmented from the below samples, it is obtained in the form shown below

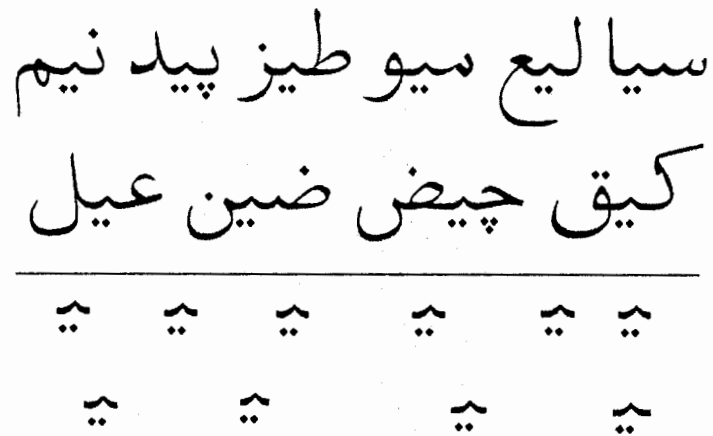


Figure 5-7 Sample for YA in the MIDDLE

the samples. Its recognition rate is 95%.

5.4.3 GHAEEN of END Sample

The following sample shows the association of غ 'GHAEEN of END ' with different characters in a word. Its segmentation may also create complication while segmenting it from the different character in the words. It provided 90% of the results.

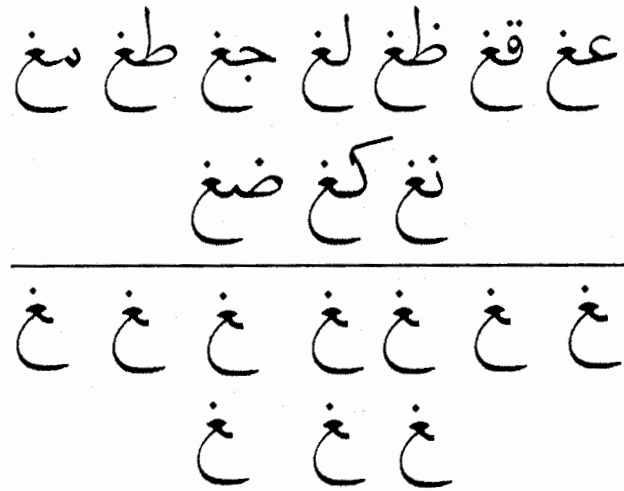


Figure 5-8 Sample for GHAEEN at the END

5.4.4 QAAF of START Sample

Here ق 'QAAF of START ' is described by the different samples. Its segmentation may also make it changed lengthwise. Its achieved result was about 80% because of segmentation variation from word to word.

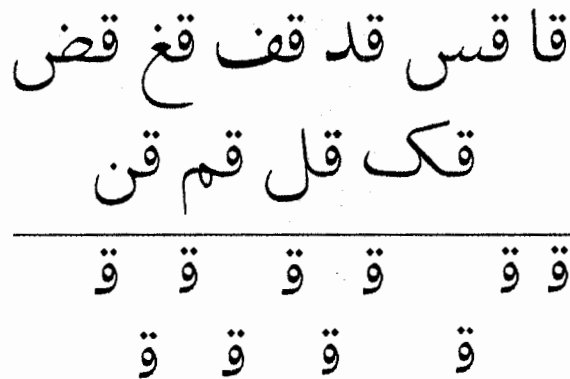


Figure 5-9 Sample for QAAF of START

5.4.5 JEEM of END Sample

This sample is for ج 'JEEM of END'. All segmented JEEMs are shown below the sample. It gave satisfactory result of 90%.

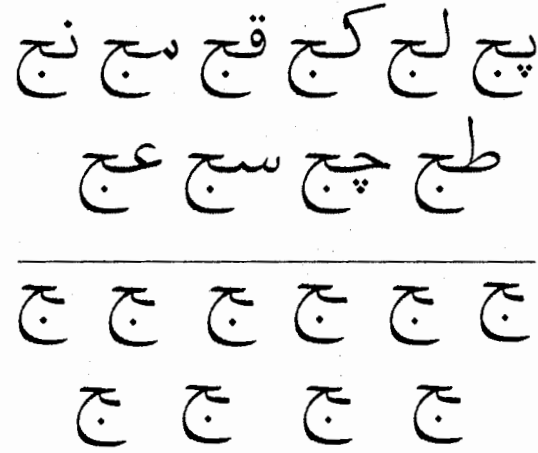


Figure 5-10 Sample for JEEM at the END

5.4.6 SHEEN of END Sample

This sample shows the association of ش 'SHEEN of END' with different characters. Its result was about 85%.

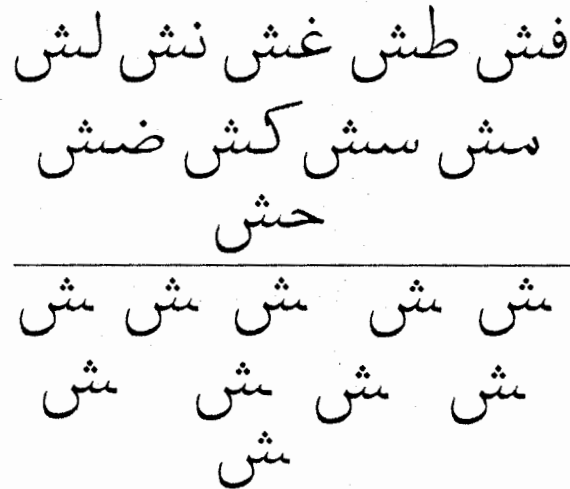


Figure 5-11 Sample for SHEEN at the END

5.4.7 LAAM of MIDDLE Sample

The following sample shows the association of ل 'LEEM of MIDDLE ' with different characters in a word. Its segmentaion may also create complication while segmenting it from the different characters in the words. It provided 80% of the results.

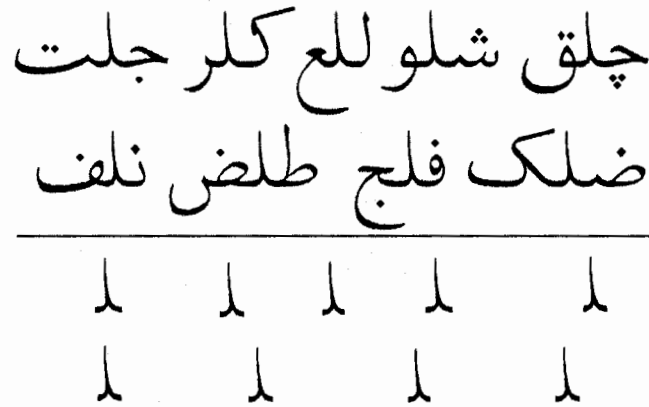


Figure 5-12 Sample for LAAM in the Middle

5.5 Recognition Rate

In the examples given above, only a few sentences were presented whereas all sentences collectively gave more than 120 different segmented characters. As far as samples are concerned, we applied more than 40 samples for different characters. The overall average rate of recognition achieved was 85 % as shown in the Recognition Rate Table 5-1

Recognition Rate w.r.t. samples	Total Samples	Status	Class	No
90%	10	Isolated	ا	01
90%	13	Ending	ب	02
90%	10	Starting	پ	03
90%	10	Ending	ت	04
90%	10	Middle	ث	05
80%	10	Ending	ج	06
90%	10	Middle	چ	07
90%	10	Starting	ح	08
70%	10	Ending	خ	09
80%	10	Ending	د	10
80%	10	Ending	ڈ	11
90%	10	Ending	ڈ	12
90%	10	Middle	ڈ	13
90%	10	Starting	ڈ	14
90%	10	Middle	ڈ	15
80%	10	Ending	ڈ	16
70%	10	Middle	ڈ	17
90%	10	Starting	ڈ	18
90%	10	Middle	ڈ	19
70%	10	Starting	ڈ	20
90%	10	Starting	ڈ	21
90%	10	Ending	ڈ	22
90%	10	Middle	ڈ	23
90%	10	Middle	ڈ	24
90%	10	Ending	ڈ	25
60%	10	Starting	ڈ	26
90%	10	Middle	ڈ	27
90%	10	Ending	ڈ	28
90%	10	Middle	ڈ	29
80%	10	Isolated	ی	30

Table 5-1 Recognition Rate

CHAPTER NO 6

CONCLUSION AND FUTURE WORK

6. CONCLUSION AND FUTURE WORK

This chapter concludes the overall proposed methodology and presents the work recommended for the future. We tested different samples and examples of data (Urdu character), so we are giving the final results with their percentage. The chapter will conclude the thesis by showing the final conclusion. Constraints are also included in this chapter.

6.1 Basic input

The following table shows the alphabet character set containing 106 characters that are used for input in our system.

End	Middle	Start	Isolated	End	Middle	Start	Isolated
ص	ص	ص	ص	ا	.	.	ا
ض	ض	ض	ض	ب	ب	ب	ب
ط	ط	ط	ط	پ	پ	پ	پ
ظ	ظ	ظ	ظ	ت	ت	ت	ت
ع	ع	ع	ع	ث	ث	ث	ث
غ	غ	غ	غ	ج	ج	ج	ج
ف	ف	ف	ف	چ	چ	چ	چ
ق	ق	ق	ق	ح	ح	ح	ح
ک	ک	ک	ک	خ	خ	خ	خ
ل	ل	ل	ل	د	.	.	د
م	م	م	م	ذ	.	.	ذ
ن	ن	ن	ن	ر	.	.	ر
ں	.	.	ں	ز	.	.	ز
و	.	.	و	س	س	س	س
ی	ی	ی	ی	ش	ش	ش	ش

Table 6-1 Input Alphabet Set for Urdu Nasakh Script

On the basis of above alphabet set table, we constructed general classes of the alphabets by observing the similar features and characteristics, which is given on next page.

Supported Characters	Class	S.No	Supported Characters	Class	S.No
ع	ع	28	ا	ا	1
ع	ع	29	ا	ا	2
ع	ع	30	پ ب ث ت	ب	3
ع	ع	31	ب ت ی ڈ ز	د	4
ع	ع	32	ب ت ی ڈ ز	د	5
ع	ع	33	ب ت ی ڈ ز	د	6
ع	ع	34	ح	ح	7
ع	ع	35	ج د ذ	ح	8
ع	ع	36	ج د ذ	ح	9
ع	ع	37	ج د ذ	ح	10
ع	ع	38	ذ	د	11
ع	ع	39	ذ	د	12
ع	ع	40	ر ز	ر	13
ع	ع	41	ر ز	ر	14
ع	ع	42	س ش	س	15
ع	ع	43	س ش	س	16
ع	ع	44	س ش	س	17
ع	ع	45	س ش	س	18
ع	ع	46	ص ض	ص	19
ع	ع	47	ص ض	ص	20
ع	ع	48	ص ض	ص	21
ع	ع	49	ط ظ	ط	22
ع	ع	50	ط ظ	ط	23
ع	ع	51	ط ظ	ط	24
ع	ع	52	ط ظ	ط	25
ع	ع	53	ط ظ	ط	26
ع	ع	54	ع غ	ع	27

Table 6-2 General Classes

For the above general classes we attempted to form the clusters. For clustering we applied different size of topology (grids of 6×5 , 8×7 , 9×8 , 10×10) and each topology is applied for number of epochs 75, 100, 150 and 200. The overall conclusion of the clustering is given by the following table.

6.2 Results of Clustering

Measurements				Characters in One Cluster	
S.No	Topology	Epochs	Clusters Formed	Minimum	Maximum
1	6 × 5	75	23	1	4
2	6 × 5	100	14	1	6
3	6 × 5	150	12	1	10
4	6 × 5	200	13	2	8
5	8 × 7	75	27	1	4
6	8 × 7	100	27	1	3
7	8 × 7	150	23	1	4
8	8 × 7	200	29	1	7
9	9 × 8	75	30	1	4
10	9 × 8	100	28	1	3
11	9 × 8	150	25	1	4
12	9 × 8	200	22	1	5
13	10 × 10	75	33	1	3
14	10 × 10	100	33	1	3
15	10 × 10	150	27	1	5
16	10 × 10	200	29	1	4

Table 6-3 Results of Clustering

❖ Above table concludes that

Topology 6 × 5 gave 23 clusters for epochs 75
 Topology 8 × 7 gave 27 clusters for epochs 75
 Topology 9 × 8 gave 30 clusters for epochs 75
 Topology 10 × 10 gave 33 clusters for epochs 75

As topologies 6 × 5, 8 × 7, 9 × 8 and 10 × 10 gave as many as 30, 56, 72 and 100 clusters respectively. Therefore 23 / 30 is greater than 27 / 56, 30 / 72 and 33 / 100 each. Hence small topology gave more number of clusters as compared to greater size of topology.

- ❖ As we increased the size of topology the number of maximum characters in one cluster decreased as shown by the last column of the above Table 6-3
- ❖ Keeping the same number of epochs, increase in size of topology increased the number of clusters formed.

6.3 Results of Testing

It is concluded that recognition phase depends upon clustering, because the segmented character is first found in its corresponding cluster. Then Classification-II with additional features, recognized the segmented characters in the cluster.

Depending upon the data and samples given for testing purpose (chapter 5), the system gave 85% of overall recognition result shown in following table

Recognition Rate w.r.t. samples	Total Samples	Status	Class	No
90%	10	Isolated	ا	01
90%	13	Ending	ل	02
90%	10	Starting	ب	03
90%	10	Ending	ت	04
90%	10	Middle	ث	05
80%	10	Ending	ج	06
90%	10	Middle	ح	07
90%	10	Starting	خ	08
70%	10	Ending	د	09
80%	10	Ending	ر	10
80%	10	Ending	ز	11
90%	10	Ending	س	12
90%	10	Middle	ش	13
90%	10	Starting	ص	14
90%	10	Middle	ظ	15
80%	10	Ending	ط	16
70%	10	Middle	ق	17
90%	10	Starting	ف	18
90%	10	Middle	غ	19
70%	10	Starting	ک	20
90%	10	Starting	گ	21
90%	10	Ending	ک	22
90%	10	Middle	گ	23
90%	10	Middle	چ	24
90%	10	Ending	چ	25
60%	10	Starting	ز	26
90%	10	Middle	ز	27
90%	10	Ending	و	28
90%	10	Middle	و	29
80%	10	Isolated	ی	30
85%	302	Total		

Table 6-4 Recognition Rate

6.4 Constraints

- We assume that the characters that system takes as input are already segmented.
- Special/complementary characters are not treated.
- Numeric values are not handled.
- Only dots as diacritics are used in this system
- Not handling multiple languages or multiple fonts.

6.5 Future Recommendations

The recognition result of our system is almost 85%. The following directions are recommended for future work.

- Urdu Nasakh script is closer to Arabic script; therefore the system can be extended for the recognition of Arabic script.
- The system may be extended for more segmented characters for any Urdu type language
- Only one diacritic is used by the system, so complete set of diacritics may be used to extend the system.
- The system is flexible and contains structured nature of programming; it may be converted from Offline to Online Recognizer.
- Our System uses Neural Network, but other techniques may be used e.g. Markov Hidden Model and a comparative study may be conducted.
- MATLAB 7.0 is used here for Testing and Implementation purposes. C++/Visual C 6.0 and C# can be used for implementation to get more efficient software.

APPENDIX A

A-1 Main Interfacing Window

GUI provides a means to the non technical to interact and understand the system better. Before making GUI, our system had already been implemented in the MatLab programming. This program was merely a collection of more than 40 functions. It was well operational technically as well as logically, yet we felt the need for GUI for the end user. The main Interfacing Window of GUI is in the following Figure A-1.

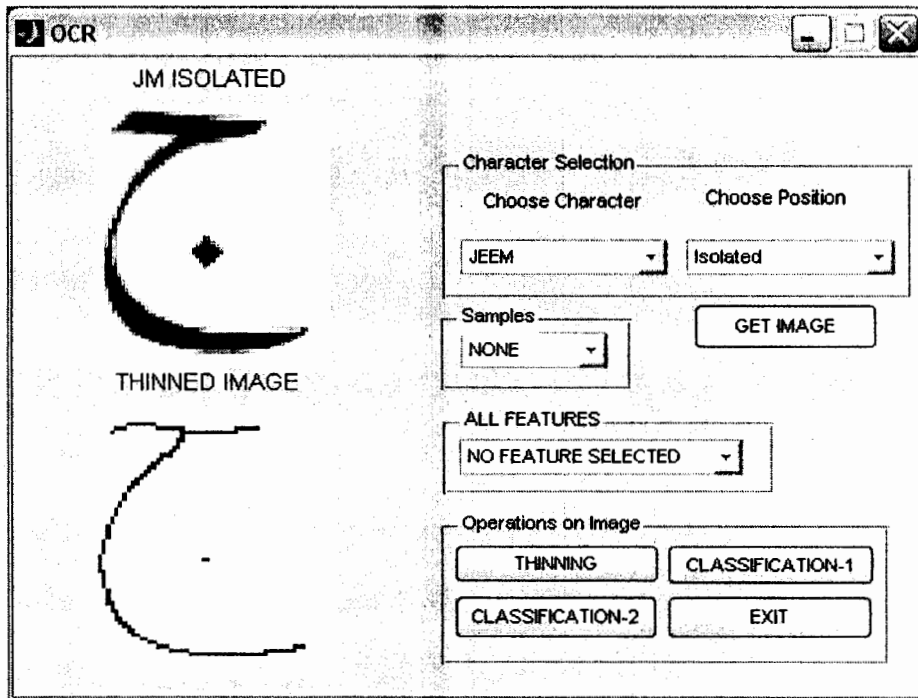


Figure A-1 Main GUI Window

A-2 Main Objects

A-2.1 Image1

The image1 is the box which shows the segmented character in its original form as input. It is obtained when GET IMAGE button is pressed. Here only the image selected by combo-boxes appears.

A-2.2 Image 2

The thinned version of the input image of the segmented character is shown in this box. The thinned image is only one pixel wide. Here image displays only if THINNING button is pressed.

A-2.3 Combo Box for Character

This box provides a way to select the character for input. The character is taken as input and displayed in GUI.

A-2.4 Combo Box for Position of Character

A character can appear in one of the four positions either Isolated, at Start, in the Middle, or at the End of word. This object provides these four different options, Isolated, Start, Middle and End.

After having selected character and its position, it is displayed on Image1 box in GUI when GET IMAGE button is pressed.

A-2.5 Combo Box for Features

This combo box makes the user select a feature to view its result about the selected input characters. For example one of the features is Loop, if user selects this option then result 'YES' appears in message box if input character contains Loop otherwise 'NO' appears.

A-2.6 Combo box for Samples

It provides option for more than fifty samples used for testing. Any sample if selected, is shown in the separate figure box with its all words and segmented characters.

A-2.7 GET IMAGE Button

This button is the source for input to the system. It takes the character selected by the user using the combo boxes. This button takes character already stored in the JPG formatted image.

A-2.8 THINNING Button

This button applies thinning operation on the selected input character and displays its thinned form in the Image2 box. Thinned Image is one pixel wide image and used for feature extraction.

A-2.9 CLASSIFICATION-I Button

This button applies different subroutines on the selected image and extracts number of features like loop, dots, crossover etc. Using these features, the function behind this button, finds the cluster in which the input segmented character falls.

A-2.10 CLASSIFICATION-II Button

This button completely recognizes the Input segmented character. The function behind this button takes character in the form of image as input and invokes different routines which digitally recognize the input image for computer usage.

A-3 Screen Shots

A-3.1 Selection for Input Character

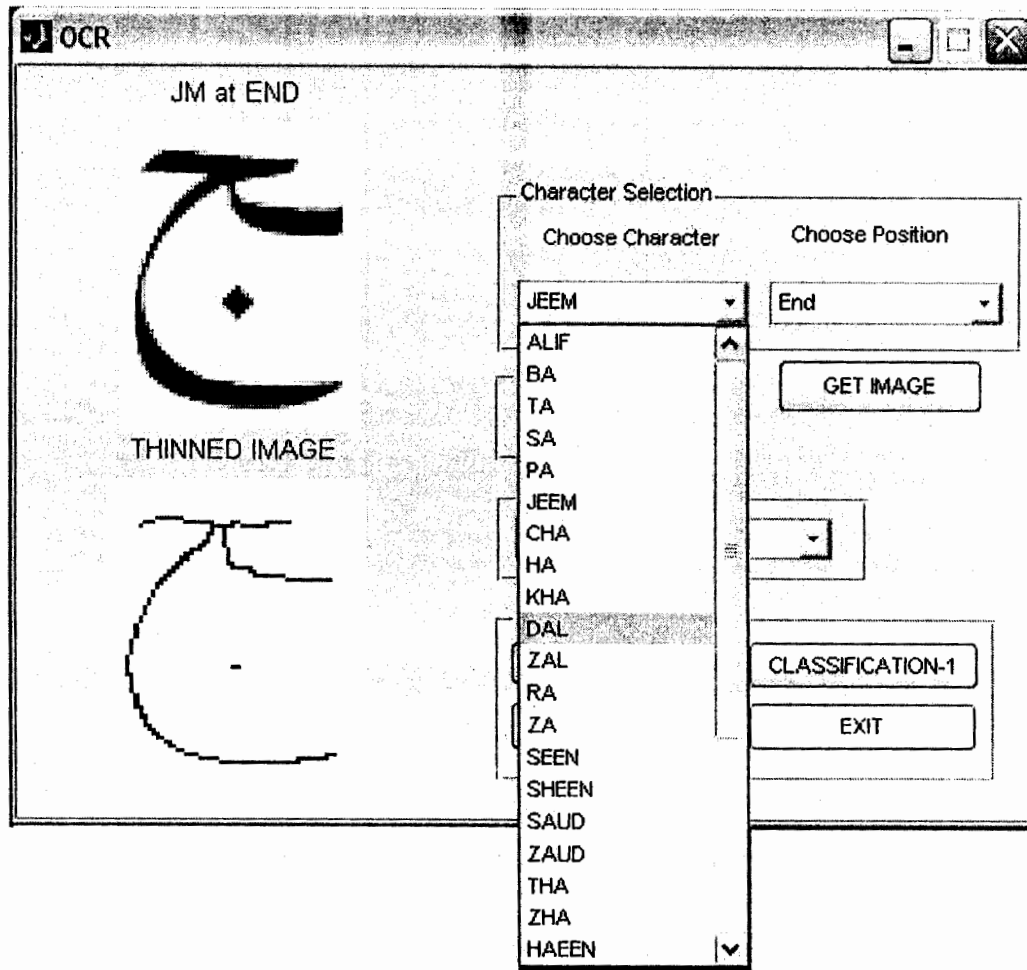


Figure A-2 Window showing the Selection of Character

This figure shows how the input character is selected. In the above case Combo box for choosing the character is popped down and 'DAL' option is highlighted. ج is already displayed on the screen. For input selection of character not only this but position of the

character in the word has to be selected from its Combo box. For this purpose next Figure A-3.

A-3.2 Selection for the Position of Input Character

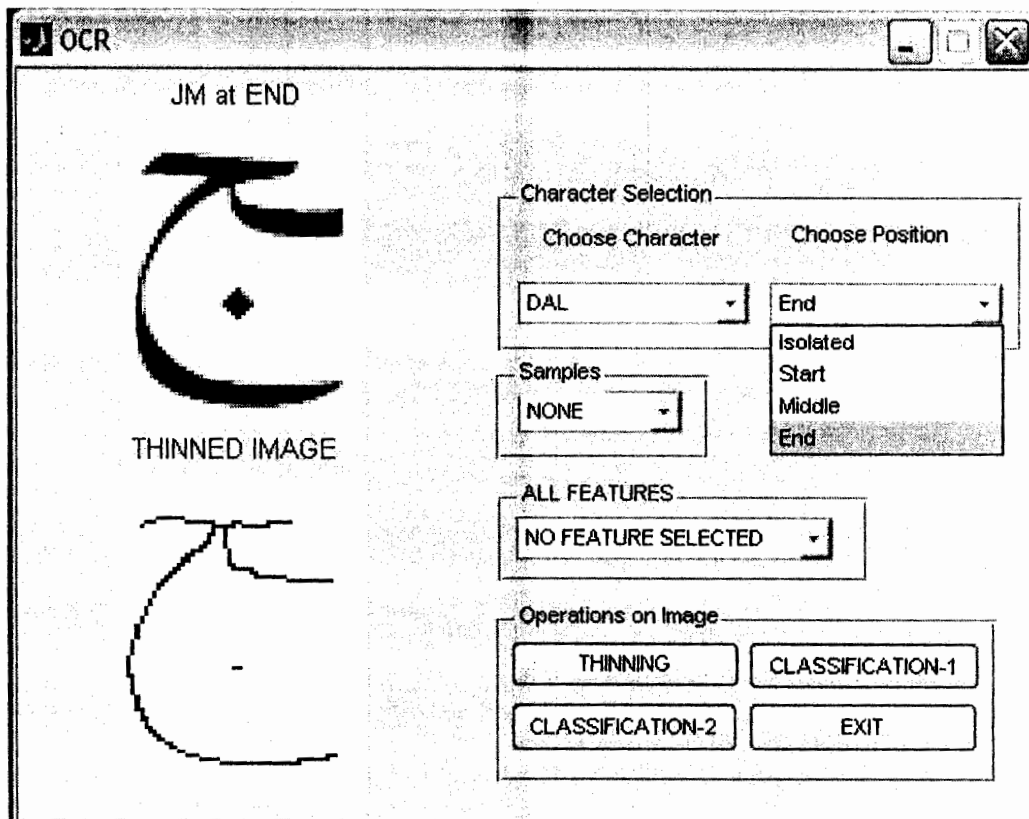


Figure A-3 Window showing the Selection of Position for the Character

The above figure shows the selection of the position for the input character. 'DAL' is already selected and now its position is being selected which is 'End'. Now by pressing the GET IMAGE button will take 'DAL of End' as Input Image. The next figure describes this.

A-3.3 Displaying the Input Character

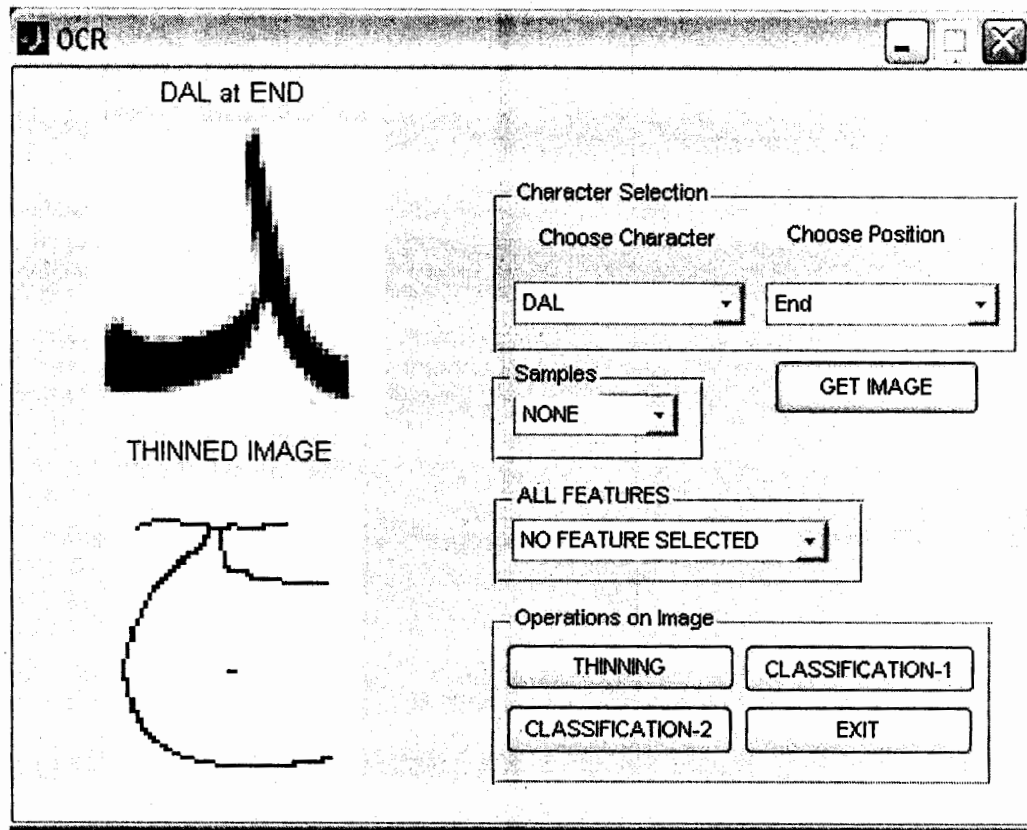


Figure A-4 Window showing Input of the Character

The above Figure 6-4 now describes the displaying of desired input character 'DAL of End' in this case. The displayed image ensures that the desired character has been obtained as input for the system. Now the subsequent figure will show the thinning operation on the input character.

A-3.4 Thinning Process

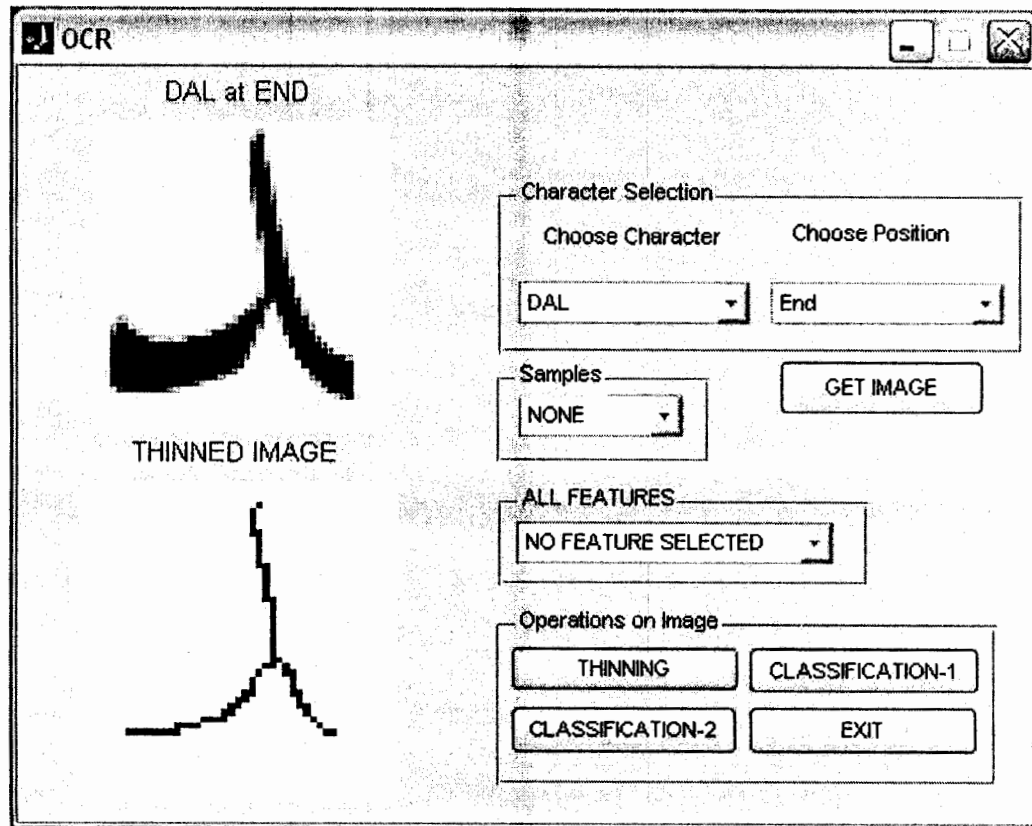


Figure A-5 Window showing the Thinning process

This figure shows the processing of THINNING button. When this button is pressed, it invokes all the necessary routines to thin the image and displays it under the original image. Thinned image is one pixel wide image which is essential in feature extraction process.

A-3.5 Clustering the Input Character

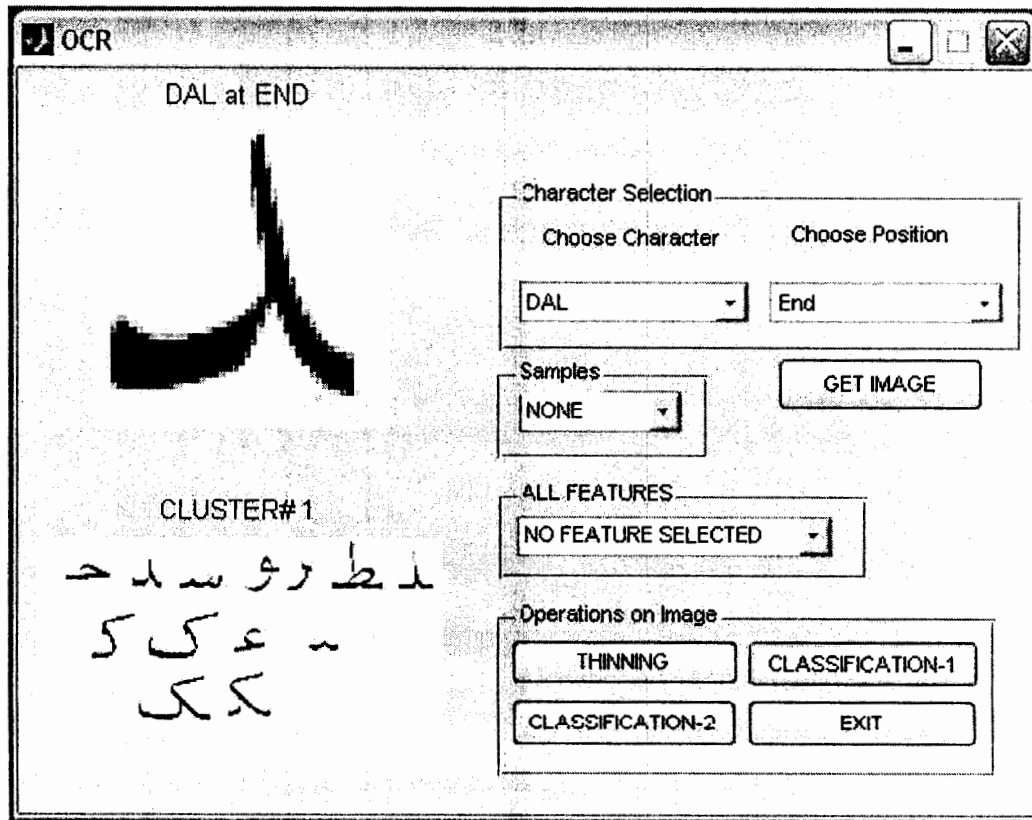


Figure A-6 Window showing Classification-I (clustering)

This figure shows Classification-I. Classification-I finds the cluster to which input image belongs. So in the above case 'DAL of End' lies in cluster number 1 who's all the characters are displayed under the original image. The functionality behind this button is to find all the features of the input image and to find cluster for the image on the basis of obtained features.

A-3.6 Recognition of the Input Character

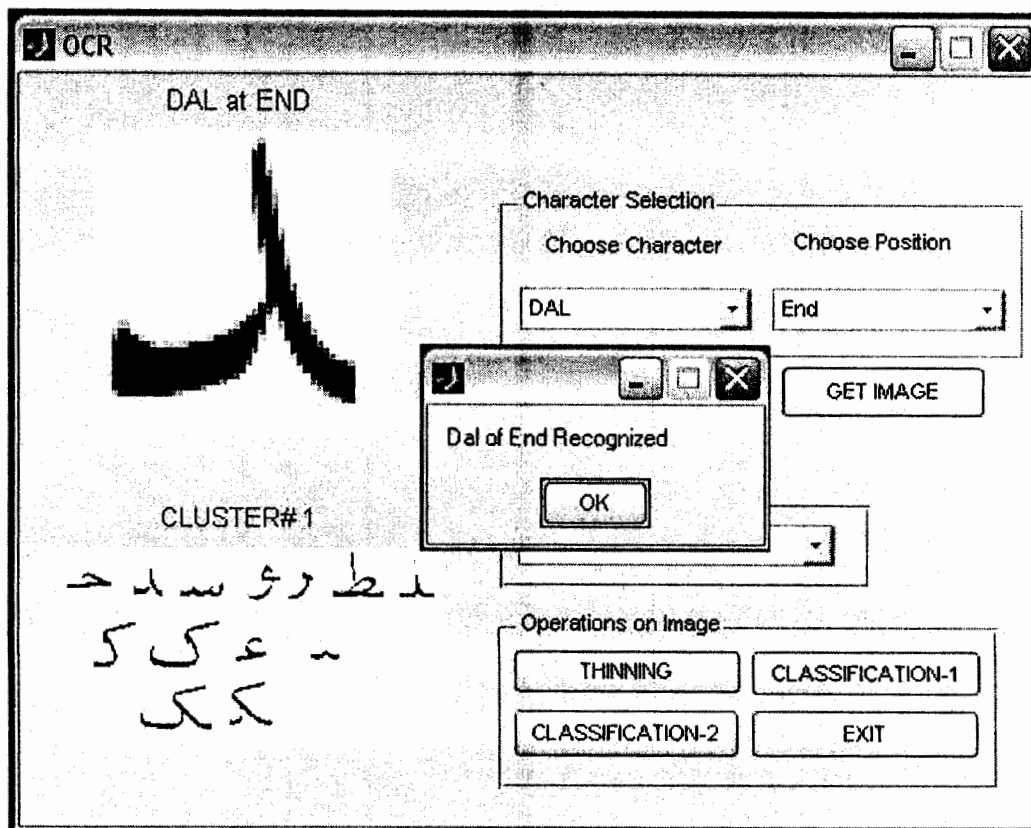


Figure A-7 Window showing the Final Recognition of the Character

This figure explains final recognition of the input image. Classification-II button classifies the input image on the basis of its features. In this case the final recognition message 'Dal of End Recognized' is displayed in the message box also shown in the Figure A-7.

A-3.7 Feature Extracted from the Input Character

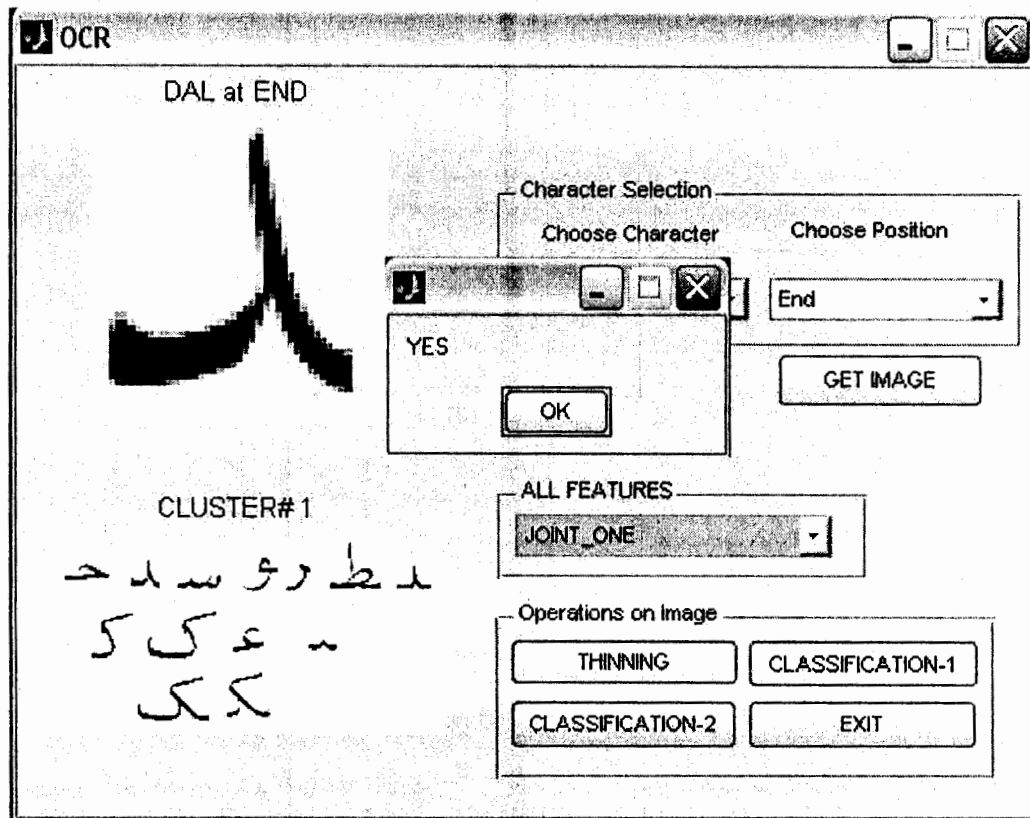


Figure A-8 Windows showing the result of selected feature

The above figure shows how user can view the result of selected feature of input image. In the above case 'JOINT_ONE' feature was selected and its result 'YES' is displayed in the message box. Its result is 'YES' because the input character 'د' contains only one joint. In addition to 'JOINT_ONE' there are as many 25 features like Loop, Dots, and Endpoints etc.

A-3.8 Sample

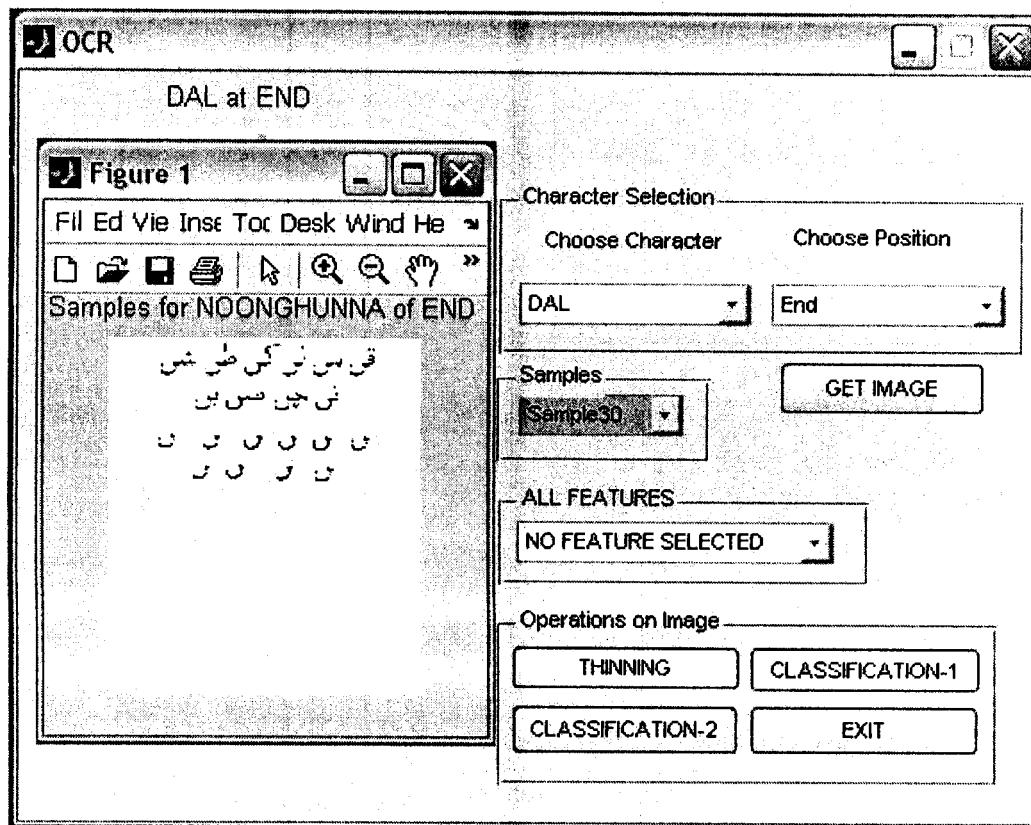


Figure A-9 Window showing a Sample

The above figure shows one sample displayed in a mini figure inside the main window. The sample shows the sentence, words and segmented characters. In this case Sample 30 is selected from the combo box and displayed in the figure. There are 59 samples whose data is examined through our system.

A-3.9 Exiting

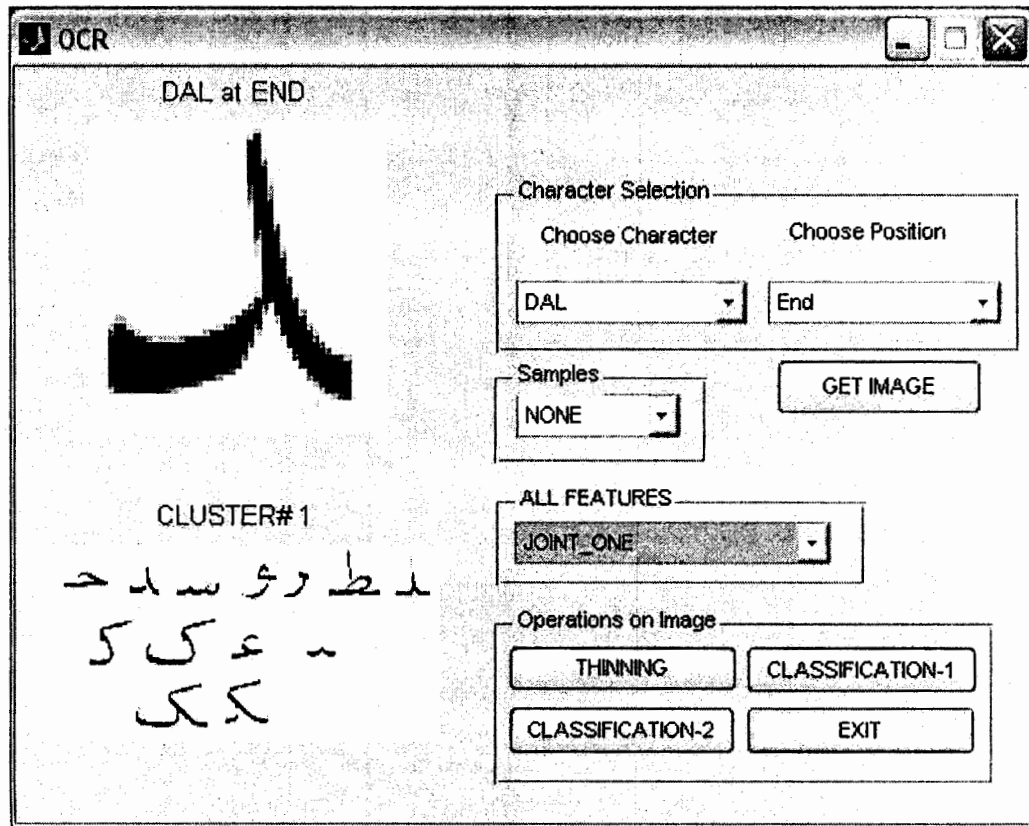


Figure A-10 Window showing the Exit Process

This figure just describes how to quit the system. EXIT button is used for this purpose.

REFERENCES

- [1] Syed Afaq Hussain, Syed Hassan Amin. A Multi-tier Holistic approach for Urdu Nastaliq Recognition, proc. Of IEEE INMIC Karachi 2002.
- [2] Zahra A Shah, Farah Saleem. "Ligature Based Optical Character Recognition of Urdu Nastaleeq Font", proc. Of IEEE INMIC Karachi 2002.
- [3] Peter Burrow. Arabic Handwriting Recognition. Thesis, Master of Science, School of Informatics, University of Edinburgh (2004)
- [4] A. Amin and S. Singh. Optical Character Recognition: Neural network Analysis of Hand-printed Characters, Proc. 7th International Workshop on Statistical and Structural Pattern Recognition, Sydney, Australia, Lecture Notes in Computer Science 1451, Springer, pp. 492-498 (August 1-13, 1998)
- [5] Laurene Fausett, "Fundamentals of Neural Networks", Florida Institute of Technology, Published by Prentice Hall; Us Ed edition (January 15, 1994)
- [6] <http://www.dataid.com/aboutocr.htm>
- [7] Maged Mohamed Fahmy, Maged Mohamed. Automatic Recognition of Handwritten Arabic Characters Using Geometric Features
- [8] John Cowell, Fiaz Hussain. "Extracting Features from Arabic Characters",
- [9] Edsh.Banke,S.Pwang .Chapter "Arabic Character Recognition" .Hand Book Of Character Recognition And Document. Image Analysis, Pp.397-420
- [10] J. R. Parker, "Practical computer Vision using C", Published by John Wiley & Sons; Pap/Dsk edition (October 28, 1993).
- [11] http://www.elet.polimi.it/upload/matteucc/Clustering/tutorial_html/
- [12] Mandana Kairanifar, Adnan Amin. Preprocessing and Structural Features Extraction for Multi-Fonts Arbic/Persian. Proc. 5th International Conference on Document Analysis and Recognition,p.213 ICDAR ' 99 .