A Finite State Model for Urdu Nastalique Optical Character Recognition

Sohail Abdul Sattar **Shams-ul Haque** **Mahmood Khan Pathan**

Department of Computer Science and Information Technology,
NED University of Engineering and Technology, Karachi, Pakistan

Abstract
Finite state technology is being used since long to model NLP (Natural Language Processing) applications specially it has very successfully applied to machine translation and speech recognition systems. Character recognition in cursive scripts or handwritten Latin script also have attracted researchers' attention and some research is also done in this area. Optical character recognition is the translation of optically scanned bitmaps of printed or written text into digitally editable data files. OCRs developed for many world languages are already under efficient use but none exist for Nastalique – a calligraphic adaptation of the Arabic script, just as Jawi is for Malay. Urdu has 39 characters against the Arabic 28. Each character then has 2-4 different shapes according to their position in the word: initial, medial, final and isolated.

In Nastalique, word and character overlapping makes optical recognition more complex. Optical character recognition of the Latin script is relatively easier. This paper based on research on Nastalique OCR discusses a proposed finite state model for the optical recognition of Nastalique printed text.

Key words:
Nastalique, Script, Cursiveness, Feature, Automata, OCR

1. Introduction

A single script with its basic character shapes is adapted for writing in multiple languages e.g. the Latin script for English, German and French while the Arabic for Persian, Urdu, Sindhi, Kurdish, Uygur and Malay.

In Urdu many character shapes have multiple instances. The shapes are context sensitive too – character shapes changing with changes in the antecedent character or the precedent one. At times even the 3rd or the 4th character may cause a similar change depicting an n-gram model in a Markov chain.

Urdu uses the Arabic script for writing, with the most prevalent style being Nastalique. Research in Urdu text recognition is almost non-existent, however a considerable research has been done on Arabic text recognition which uses the Naskh style of writing. The Arabic language is considered to be a difficult one with a much richer alphabet than the Latin, the form of the letter is the function of its position in the word: initial, medial, final or isolated, it changes it's shape depending upon it's position, shape has multiple instances, words are written from left to right. Arabic characters have features that make direct application of algorithms for character classification in other languages difficult to achieve as the structure of Arabic is very different [2].

The most difficult case in character segmentation is the cursive script. The scripted nature of Arabic written language poses some high challenges for automatic character segmentation and recognition. The problem of Arabic character recognition has not received as much attention as Latin or Chinese characters in spite of the fact that Arabic characters serve as scripts for several languages such as Arabic, Farsi, Urdu, Sindhi, Malay, Kurdish and Uygur [1]

In the last years Hidden-Markov-Model (HMM) based methods are very successfully applied to e.g. English or German cursive script recognition. All these methods need preprocessing and normalization of the scanned word images before the features can be extracted. During this preprocessing step as many individual characteristics as possible are removed, to make the feature extraction and recognition as simple as possible. A preprocessing step which is always needed for every language, cursive writing recognition is the estimation of the writing line or reference line called base line. This line represents a first orientation in a word and is essential for many further tasks [7].

Mohammad S Khorsheed and William. F. Clocksin [9], have presented a global approach for recognizing Arabic cursive words from scanned images of text. The approach is segmentation-free, and is applied to four different

Manuscript received September 5, 2009
Manuscript revised September 20, 2009

Arabic type faces, where ligatures and overlaps pose challenges to segmentation-based methods. They transform each word into a normalized polar image, then they apply a two dimensional Fourier transform to the polar image. The resultant spectrum tolerates variations in size, rotation or displacement. Each word is represented by a template that includes a set of Fourier coeficients. The recognition is based on a normalized Euclidean distance from those templates.

The Latin, Chinese and Japanese scripts have received ample research and work has been done on the optical recognition of these scripts. Compared to this only few papers have specifically addressed the recognition of Arabic text and languages using the Arabic script like Urdu, Farsi and Malay for various reasons. One of them is the complexity of the Arabic script itself while a lack of interest in this regard accounts for another [3]. Mohammad S. Khorsheed et al [4] presented an approach in which the system recognizes an Arabic word as a single unit using Hidden Markov Model. The system highly depends on a predefined lexicon which acts as a look-up dictionary. All the segments in a word are extracted from its skeleton then each of the segment is transformed into a feature vector. Then each of the feature vector is mapped to the closest symbol in the codebook. The resulting sequence of observations presented to a Hidden Markov Model for recognition. Shumaila Malik et al [6] proposed a system which takes online input from user by writing the Urdu character with the help of stylus pen or mouse and converts user handwriting information into Urdu text. The process of online hand written text recognition is divided into six phases, each of the phase is implemented using a different technique depending upon the speed of writer and the level of accuracy.

Michel Fanton [5] has discussed the features which the Arabic writing has and identified the fact that the features of Arabic writing impose computational overload for any arabicized software. He also noted that the way in which Arabic is printed imitates handwriting. He pointed it out that the Finite State Automata give efficient solution for the translated problems which can be formalized as regular languages.

Qing Chen et al [7] addressed the problem of automatic recognition of an image pattern without any consideration of its size, position and orientation. In this regard the extracted image features are made to have invariance properties against image transformation including scale, translation and rotation. They approximated the transformation by affine transformation to preserve collinearity and ratios of distances.

Arabic characters have features that make direct application of algorithms for character classification in other languages difficult to achieve as the structure of Arabic is very different. [9]

2. Nastalique Character Set

Urdu uses an extended Arabic adapted script, it has 39 characters as against Arabic 28. Each character then has 2-4 different shapes depending upon its position in the word; initial, medial or final. Table 1 shows first ten letters of the Urdu alphabet with different shapes. When a character shape is written alone it's called isolated character shape. Each of these initial, medial and final character shapes can have multiple instances, the character shape changes depending upon the change in the antecedent or the precedent character. This characteristic of having multiple instances of these character shapes is called context sensitivity. A complete language script comprises of an alphabet and style of writing. Urdu uses an extended Arabic script for writing. It has two main styles, Naskh and Nastalique. Nastalique is a calligraphic, beautiful and more aesthetic style and is widely used for writing Urdu.

Table 1. Subset of Urdu Alphabet

Iso	Ini	Med	Fin
ا	-	-	ا
ب	بـ	ـبـ	ـب
پ	پـ	ـپـ	ـپ
ت	تـ	ـتـ	ـت
ٹ	ٹـ	ـٹـ	ـٹ
ث	ثـ	ـثـ	ـث
ج	جـ	ـجـ	ـج
چ	چـ	ـچـ	ـچ
ح	حـ	ـحـ	ـح
خ	خـ	ـخـ	ـخ
⋮	⋮	⋮	⋮

3. Nastalique Script Complexities

Here we describe the complexities of Nastalique script with particular reference to optical recognition of printed Urdu text written in Nastalique style.

3.1 Cursive Nature

Nastalique with its oriental cursive nature makes a complex script. A single word in the script can comprise of several ligatures formed in turn by combining several characters cursively joined together along with isolated characters.

3.2 Context Sensitive

Ligature in Nastalique are unique combinations or units of characters that change their shape according to their position within the unit. e.g. an initial "BAY" which is the second character in the alphabet is quite different from medial, final or an isolated one, added to this is the dependence of each character on the preceding or succeeding characters it joins with. A character might take as many as 20 different shapes according to the character it is joining with. Sometimes even the 3rd, 4th or 5th antecedent or precedent character may initiate a similar change in shapes.

3.3 Position and Number of Dots

Several Urdu characters (17 out of 39) are differentiated by the presence of dots placed over, below or within them.

Fig 1: (a, b, c) Dots

Three situations of ambiguity arise because of this. In the first instance, one character may have a dot while the other does not Fig. 1(a). In the second case, two similar characters have different numbers of dots to distinguish their different sounds Fig. 1(b). Lastly, two characters may be different only because of the difference in the position of dots Fig. 1(c).

3.4 Kerning

For the better visual appeal the space between pairs of characters is usually reduced or "Kerned". If the natural space between two characters is attempted to be reduced it causes a slight overlapping on the characters making them less identifiable and more difficult to be differentiated. When scanned the lack of white pixels between the two characters makes them read as a single continuous character shape.

Although the problem rears most infrequent in the Roman scripts, it is a common one in Nastalique (because kerning implies further cursiveness, overlapping characters in numerous possible ways).

4. Phases in Optical Character Recognition

A generic OCR model has the following phases for Optical recognition of printed text as shown in Fig. 1.

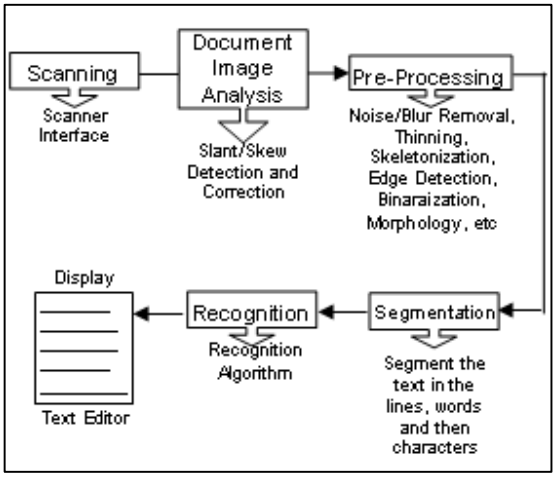

Figure 2. Different phases of OCR

4.1 Scanning

A flat-bed scanner is usually used at 300dpi which converts the printed text into a bitmap image.

4.2 Document image Analysis

The bitmap image of the text is analyzed for the presence of skew or slant and consequently these are removed.

4.3 Pre-Processing

In this phase several processes are applied to the text image like noise and blur removal, binarization, thinning, skeletonization, edge detection and some morphological processes.

4.3 Segmentation

The bitmap image obtained from the scanner is first segmented into text and non-text regions. Then the text region is segmented into separate lines of text. These lines

are then segmented into words and finally words into individual letters.

4.4 Recognition

This is the most vital phase in which recognition algorithm is applied to the images present in the text image segmented at the character level.

As the recognition process proceeds from the smallest bounding box around a character image to the word and then the line, as a result of recognition, character codes corresponding to the images are returned by the system which are then passed to a word processor and then the rendering engine to be displayed on the screen, where the text can now be edited, modified and saved in a new file format.

5. Ligature in Urdu

A ligature in Urdu is a complex unit of several characters bound together cursively to give a single fluid form. A ligature is not a complete word but can be considered as a compound character. A word in Urdu may be composed of one or two ligatures as well as isolated characters. For example the word صلاحیت has two ligatures, Fig. 3 and 4 .

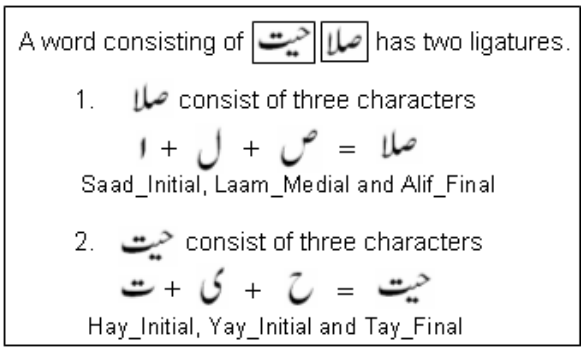

Figure 3. Ligature in Urdu

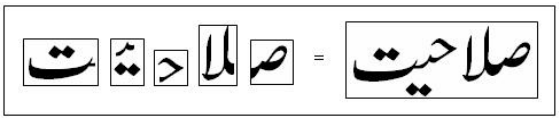

Figure 4. Word Segmentation

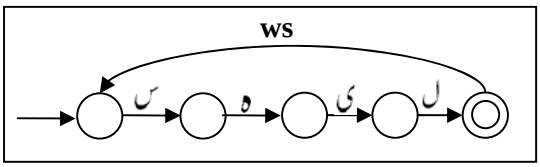

Figure 5. Simple Finite State Nastalique Word Recognizer

6. Finite State Nastalique Text Recognizer

A finite state Nastalique text recognizer can be modeled as a finite state automata shown in Fig. 5 and 6:

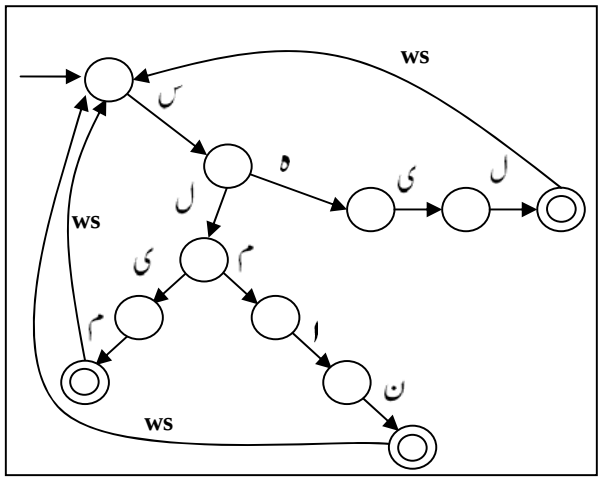

Figure 6. A Finite State Nastalique Text Recognizer

A finite state automaton A, is defined by a five tuple as follows [8]:
A = (Q, ∑, δ, q_o, F), where
Q: a finite set of states
∑: a finite set of input symbols also called alphabet
F: a finite set of accepting states
δ: the transition function, which is specified as

$$\delta: (Q \times \sum) \rightarrow Q$$

6.1 Nastalique Character Shapes

In Nastalique we have 2-4 basic character shapes for each character in the alphabet: Isolated, Initial, Medial and Final.

Except Isolated last three are position dependent shapes and can have several forms depending on the precedent as well as the antecedent characters.

Each of these character shapes has multiple instances, for example Bay ب, which is the second letter in the Urdu alphabet has 13 different shapes for its initial form. These character shapes are context sensitive – character shape change if the antecedent or the precedent character changes. At times even the 3rd or the 4th character may cause a similar change depicting an n-gram model in a Markov chain. The first order Markov model is

$$\mathbf{C_i \mid C_{i-1}}$$

6.2 Nastalique Feature Set

We call a set comprising features of Nastalique character shapes as a Nastalique feature set which can have the following features:

Height, Thickness, Angle, Rotation

These features are also termed as attributes, thus the Nastalique feature set is given as:

FS = { Height, Thickness, Angle, Rotation }

Word forming in Nastalique is not as simple as in Roman script. In Nastalique a word is composed of ligatures and isolated character shapes. Sometimes a word consists of only a single ligature e.g.بہت

Two or more character shapes join cursively to form a ligature. While forming a ligature a character shape can join the other from left as well as the right side. When a character shape joins another from its right side with its own left side then at the joining point the two feature sets must match.

So we have features defined at one or both of the two possible joining ends of each character shape as left features (LF) and right features (RF).

If only LFs are available, then character is Initial, if only RFs are available then character is final, if both LFs and RFs are available then character is Medial.

7. Components of Nastalique Text Recognizer

A finite state Nastalique text recognizer will have the following components:

7.1 Character Shape Recognizer

This component gets the input from the segmentation phase as a character image and checks whether it has LFs only or RFs only or it has both LFs and RFs or it does not have any. On the basis of it following decisions are made:
Initial Character Shape, if it has LFs only
Final Character Shape, if it has RFs only
Medial Character Shape, if it has both LFs and RFs
Isolated Character Shape, if it has Non of LFs or RFs

7.2 Next-State Function

Having decided the shapes of characters in a ligature as initial, medial, final or isolated, the system checks for the multiple instances of these character shapes. In Nastalique we have multiple instances of these character shapes encoded in the font file, the Next-state function compares the LFs and RFs on one-to-one basis and decides the correct instance of the character shape present at a particular position in a ligature and returns its character code that the system stores in a text file and moves to the next state.

As the recognition process proceeds the character codes found in a ligature are stored in a text file concatenated in sequence as they are found in a ligature. After the completion of the recognition process all the character codes stored in a text file are given to the rendering engine which displays the recognized Nastalique word on the screen.

The Finite State Recognizer will always be in a state to expect a new input character image except that it receives a white space character that results in either an error state indicating an incomplete word or an accepting state where all the character codes found in a ligature are stored in a text file and leaving a white space the system gets ready to read new inputs, shown in Fig. 7.

The Finite State Nastalique Text Recognizer is

$A = (Q, \sum, \delta, q_0, F)$

States of the Finite State Recognizer

$Q : \{ q_0, q_1, q_2, q_3, q_4 \}$

Transition or Next state function: δ

q_0 = starting sate and final state
q_1 = input is an initial character shape
q_2 = input is a medial or final character shape
q_3 = error state / incomplete word
q_4 = input is a white space

The alphabet or set of input symbols

$\sum$: { initial, final, medial, isolated, white space }
$\sum$ = { in, f, m, is, ws }
F: set of accepting states = $\{ q_0 \}$

Table 2. Transition Function Definition

δ	in	m	f	is	ws
q_0	q_1	q_3	q_3	q_0	q_3
q_1	q_3	q_2	q_4	q_3	q_3
q_2	q_3	q_2	q_4	q_3	q_3
q_3	q_3	q_3	q_3	q_3	q_3
q_4	q_3	q_3	q_3	q_3	q_0

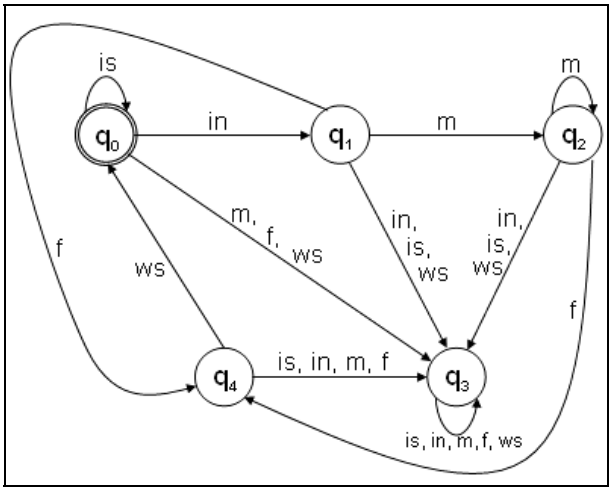

Figure 7. Transition Diagram

8. Results and Discussion

Although the Nastalique character recognition system is developed in Microsoft Soft VC++ 6.0 we used the software Matlab for rapid prototyping and experimentation.

We performed experiments on a subset of Nastalique words keeping same the font size and the results are very encouraging.

Our Nastalique character recognition system requires a character based True Type Nastalique font and an image of Nastalique printed text written with the character based True Type Nastalique font.

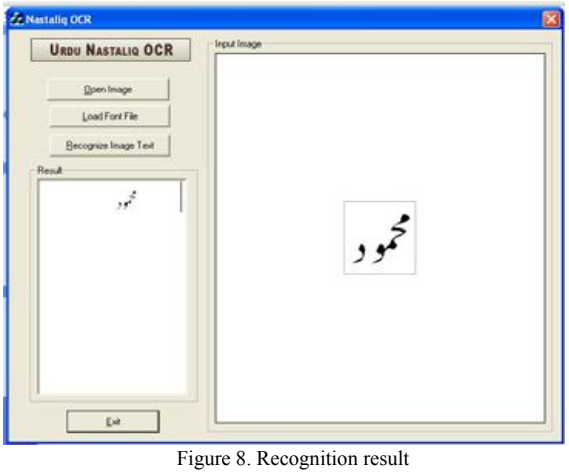

Figure 8. Recognition result

After the segmentation is complete the text image appears in segmented character shapes and the isolated character shapes can be identified separately. Next is the recognition phase in which the Finite State Nastalique Text Recognizer reads and recognizes each of the character shapes in the segmented text image, line by line, and writes their character codes in the sequence the character is found, into a text file.

As the recognition process is completed the character codes in the text file are transferred to the rendering engine which displays the recognized text in a text region, shown in Fig. 8.

9. Conclusion and Future Work

In this paper we discussed our finite State Model for Nastalique Text recognition with its results on a small subset of Urdu Nastalique words. The Nastalique script has a complex nature, each character has 2-4 shapes then each shape has multiple instances, depending upon the antecedent or the precedent one, a character changes it's shape. This characteristic of the script is called context sensitivity and makes optical recognition of Nastalique script more complex.

For the future we plan to extend our system to include a larger set of Urdu Nastalique words.

References

[1] Sari T.,Souici L. and Sellami M., "Off-Line Handwritten Arabic Character Segmentation Algorithm: ACSA", Proc. of Internationl Workshop on Frontiers in Handwriting Recognition, pp. 452-457, 2002.

[2] Majid M. Altuwaijri and Magdy A. Bayoumi, "Arabic Text Recognition Using Neural Networks", 1994, IEEE International Symposium on Circuits and Systems, ISCAS'94, London, UK, 30 May- 02 June,1994, Vol. 6, pp. 415-418.

[3] Muhammad Sarfaraz, Syed Nazim Nawaz and Abdulaziz Al-Khuraidly. "Offline Arabic Text Recognition System". 2003, Proceedings of the International Conference on Geometric Modeling and Graphics (GMAG'03), IEEE COMPUTER SOCIETY, London, UK, July 16-18, 2003, pp. 30-35.

[4] Mohammad S. Khorsheed and William F. Clocksin, "Structural Features of Cursive Arabic Sscript", 1999, Proceeding of the 10th British Machine Vision Conference, Nottingham. pp: 422-431.

[5] Michel Fanton, "Finite State Automata and Arabic Writing", COLING-ACL `98 Workshop on Computational approaches to Semitic Languages. Aug 16, 1998, University of Montreal, Canada.

[6] Shumaila Malik and Shoab A. Khan, "Urdu Online Handwriting Recognition:, 2005, Proceedings of the IEEE International Conference on Emerging Technologies, Sept 17-18, 2005 Islamabad, Pakistan, pp: 27-31.

[7] Pechwitz M., Margner V., "Baseline Estimation For Arabic Handwritten Words", IEEE Transactions on PatternAnalysis and Machine Intelligence, May 2006, Volume: 28, Issue: 5, page(s): 712- 724, ISSN: 0162-8828.

[8] John E. Hopcroft, Rajiv Motwani and Jeffrey D. Ullman. "Introduction to Automata theory, Languages and Computation", 2nd edition, Addison-Wesley, 2000.

[9] Mohammad S Khorsheed and William. F. Clocksin. "Multi-Font Arabic Word Recognition Using Spectral Features". Proceedings of 15th International Conference on Pattern Recognition, Sept 3-7, 2000, Barcelona, Spain, Volume: 4, page(s): 543-546, ISBN: 0-7695-0750-6.

Sohail A. Sattar received BE in mechanical engineering from NED University of Engineering and Technology, Karachi, Pakistan in 1988. He is a life member of Pakistan Engineering Council and Institution of Engineers (Pakistan) and a member of International Association of Engineers. Having served the industry for 12 years he switched his career to computer science getting an MCS degree in computer science in 2001 from University of Karachi and an MSc in computer Science in 2002 from NED University, where now he serves as assistant professor of computer science. He is also pursuing a doctorate degree in computer science. His research interests include computer vision, image processing and computational linguistics.