Recognition of Nastalique Urdu Ligatures

Gurpreet Singh Lehal
Department of Computer Science
Punjabi University, Patiala
India
gslehal@gmail.com

Ankur Rana
ACTDPL
Punjabi University, Patiala
India
ankurrana628@gmail.com

Abstract
There has been considerable work on Arabic OCR. However, all that work is based on Naskh style. Urdu script is based on Arabic alphabet, but uses Nastalique style. The Nastalique style makes OCR in general and character segmentation in particular, a highly challenging task, so most of the researchers avoid the character segmentation phase and go in for higher unit of recognition. For Urdu, the next higher recognition unit considered by researchers is ligature, which lies between character and word. A ligature is a connected component of one or more characters and usually an Urdu word is composed of 1 to 8 ligatures. There are more than 25,000 Urdu ligatures, out of which top 4567 ligatures account for 99% of coverage. From OCR point of view, a ligature can further be segmented into one primary connected component and zero or more secondary connected components. The primary component represents the basic shape of the ligature, while the secondary connected component corresponds to the dots and diacritics marks and special symbols associated with the ligature. To reduce the class count, the ligatures with similar primary components are clubbed together. In this paper, we have presented a system to recognize 9262 ligatures formed from 2190 primary and 17 secondary components. Various combinations of DCT, Gabor filters and zoning based features along with kNN, HMM and SVM classifiers have been tried and a recognition accuracy of 98% has been reported on pre-segmented ligatures.

Keywords
Urdu OCR, Nastalique, Ligature identification, DCT, SVM

1. INTRODUCTION
There also has been considerable work on Arabic OCR also but however very little work has been done for Urdu OCR. The Nastalique style makes OCR in general and character segmentation in particular, a highly challenging task, so most of the researchers avoid the character segmentation phase and go in for higher unit of recognition. For Urdu, the next higher recognition unit considered by researchers is ligature, which lies between character and word. A ligature is a connected component of one or more characters and usually an Urdu word is composed of 1 to 8 ligatures. Most of the papers in literature related to Urdu OCR, deal with a very limited number of Urdu ligatures [1, 2] or recognize only isolated Urdu characters [3-4]. The only work reported which deals with a sizable number of ligatures are by Javed el [5]. The authors have trained their system for recognition of 1282 ligatures. As the actual number of ligatures is very high and it is difficult to develop a high accuracy recognition system for all the possible Urdu ligatures, we have worked for recognition of 99% most frequently occurring Urdu ligatures. Experiments with different features and classifiers have led to development of a system capable of recognizing the pre-segmented ligatures with more than 98% recognition accuracy.

MOCR '13, August 24 2013, Washington, DC, USA

http://dx.doi.org/10.1145/2505377.2505379

2. OVERVIEW OF URDU
Urdu is an important South Asian language spoken by nearly 250 million people in India, Pakistan and other neighbouring countries. Urdu is the national language of Pakistan, and one of the official languages of India. Urdu is written from right to left just like Arabic and Persian. Urdu has 40 basic letters and 10 diacritical marks, and most of these letters are from Arabic and a small quantity from Persian. Urdu is written in the Nastalique style of Persian calligraphy, whereas Arabic is generally written in the Naskh style. Urdu is highly cursive and context sensitive in nature and as a result the recognition techniques for Arabic script are not directly applicable to Urdu. From OCR point of view Urdu is a very tough script.

The main challenges are:

- Urdu script is written diagonally from top right to bottom left with stacking of characters (Figure 1). The ligatures are tilted at a certain angle towards the right side. Numerals add to the complexity as they are written from left-to-right Due to this diagonal nature the Nastaliq consumes less horizontal space as compared to Naskh, but from OCR point of view it creates problems in word and character segmentation.

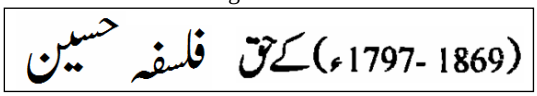

Figure 1. A Sample Urdu text

- Context Sensitive nature of nastalique compels the letters to adapt different shapes. The shape of a character depends on the succeeding characters. Usually an Urdu character acquires one of these four shapes, namely isolated, initial, medial and final, but a character may have up to 45 different shapes.
- Line and word segmentation are non-trivial tasks. We have frequently merging lines and words in adjacent lines merged. For word segmentation, there is not much difference between inter-word and intra-word vertical gap and it is very difficult to judge if the two adjacent ligatures belong to same or different words. As for example, the text in Figure 2 is composed of six ligatures and three words, but it is very difficult to make out from vertical gap, which ligature belongs to which word. There is vertical overlapping both within ligatures and among ligatures, which further complicates word segmentation (Figure 3).

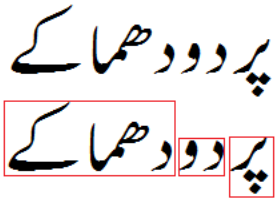

Figure 2. Confusing Inter-ligature and inter-word gap

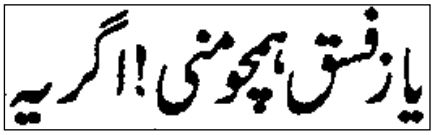

Figure 3. Vertically over lapping Urdu ligatures

- Character segmentation poses another big challenge, as the characters are completely merged and it is very difficult to detect the character segmentation points. Machine printed Arabic (Naskh style) text usually follows a horizontal base-line where most characters, irrespective of their shape, have a horizontal segment of constant width. If we can separate these horizontal constant-width segments from a ligature, the remaining components of the ligature could be recognized. However, this is not the case with Nastaliq which has multiple base-lines, horizontal as well as sloping, making Nastaliq a complex style for character segmentation.
- The presence of diacritic marks and dots in the Urdu characters also creates lots of confusion. The dots often get merged with each other or touch the ligatures or adjacent diacritic marks resulting in distorted shapes.

2.1 Urdu Ligature

A ligature is a connected group of Urdu characters. A word in Urdu is a collection of one or more ligatures. As for example, the word (ہندستان) (*hindustan*) is composed of three ligatures : ہند ، ستا and single character ligature ن. The two ligatures are further composed of three characters (ہند = ہ + ن + د and ستا = س + ت + ا). From OCR point of view, a ligature consists of one primary connected component and zero or more secondary connected components. The primary component represents the basic shape of the ligature, while the secondary connected component corresponds to the dots and diacritics marks and special symbols associated with the ligature. As for example, the ligature (Figure 4) has one primary connected component and five secondary connected components.

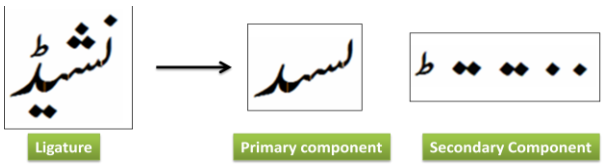

Figure 4. Primary and Secondary Components

3. URDU LIGATURE RECOGNITION PHASES

The Urdu ligature recognition is carried out in two phases: Offline and online phase

3.1 Offline Phase

The major steps in this phase are:

1 Identifying the recognizable Urdu ligatures
2 Creation of training images
3 Creation of code book

3.1.1 Identifying the Recognizable Urdu ligatures

There is no exact count available on the number of Urdu ligatures available. We have used the results presented in [6], to finalise the 2190 primary components for recognition, which cover 99% of corpus. In addition 17 secondary components are recognized by our system (Figure 5).

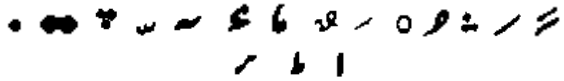

Figure 5. Secondary Components for recognition

3.1.2 Creation of Training Images

The next important task is creating the training data corresponding to 2190 primary components, for training and testing the OCR. The Urdu text printed in books and newspapers can be divided into two generations. The books printed before 1995 are all hand written while majority of the books published after 1995 use computer generated Nastalique fonts such as Noori Nastalique. Thus for our experimentation, we collected a corpora consisting of two sets of images to cover both the generations. The primary components are extracted from ligatures and at least 15 samples are taken for each class. A sample of our training data is shown in Figure 6.

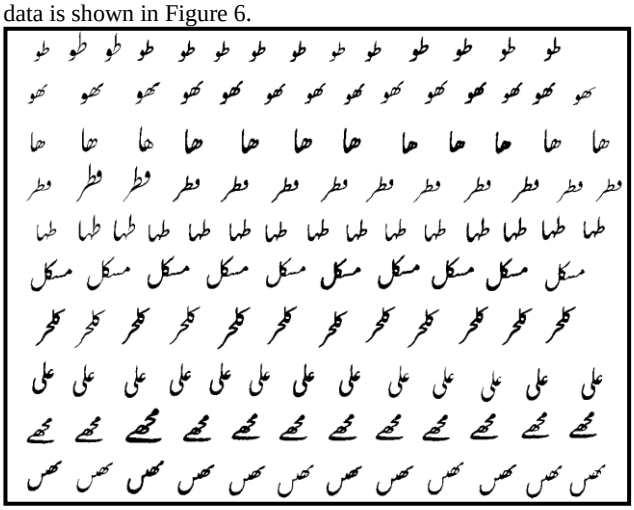

Figure 6. Samples of Training Data

3.1.3 Code Book Generation

The primary and secondary components have to be joined back to form the ligatures. The 2190 primary and 17 secondary components form 9262 ligatures. For each ligature, an entry has been made in the code list, which contains primary code, secondary code and the Unicode string corresponding to the ligature.

Separate codes are assigned to each secondary component, depending upon its position in the primary component for the formation of the ligature. For example

1 Single dot is coded as ‘a’ if it lies below the primary component, ‘A’ if it lies above the primary component or ‘p’ if it lies in the middle.

2 Double dots are coded as ‘b’ if they lie below the primary component, ‘B’ if they lie above the primary component or ‘q’ if they lie in the middle.

3 Three dots are coded as ‘c’ if they lie below the primary component, ‘C’ if they lie above the primary component or ‘r’ if they lie in the middle.

Secondary string is formed by concatenating the codes of secondary components in the order they occur in ligature from right to left. For example, the secondary string for the ligature shown in the Figure 7 is “CBp”

Figure 7. Sample of Urdu Ligature

3.2 Online phase

The major steps in online phase are:

1 Segmentation of ligature from Urdu text image: A hybrid approach, which uses top down technique for line segmentation and bottom up design for segmenting the line into ligatures, has been employed [7].

2 Splitting the ligature into primary and secondary connected components by connected component analysis.

3 Recognizing the primary and secondary components.

4 Combining the recognition results of connected components to form Unicode string.

The third and fourth steps are discussed in detail in following sections.

4. RECOGNIZING PRIMARY AND SECONDARY COMPONENTS

We have experimented with different feature extraction techniques and classifiers to select the most efficient recognition pair. For secondary components, the selection was simpler but due to large number of classes in primary components, selecting the most efficient features and classifier required lots of experimentations.

4.1 Experiments with Various Feature Sets

We have experimented with Gabor filters, DCT features and zoning based features for the primary components. For the SVM and kNN classifier, all the three types of features are extracted. For the HMM classifier, we have used sliding windows principle discussed in [8] and extracted DCT and zoning based features for each of the primary components of the ligatures. The details of the feature sets used are in following sub sections.

4.1.1 Gabor Filters

The word image is normalized to 32x32 pixels and partitioned into four equal non overlapping subregions of size 16x16. These subregions are further partitioned into four equal non overlapping sub-subregions of size 8x8 and thus we obtain 16 small regions in different parts of the image. We then convolve these 21 images with odd symmetric and even symmetric Gabor filters in nine different angles of orientation of 20 degrees, to obtain a feature vector of 189 values.

4.1.2 DCT Features

DCT is a technique to convert data of the image into its elementary frequency components. DCT clusters high value coefficients in the upper left corner and low value coefficients in the bottom right of the array (m, n) where m and n is the width and height of image respectively. DCT coefficients f(u, v) of f(m, n) are computed by equation(1) :

$$f(u,v) = a(u)a(v)\sum_{m=0}^{M-1}\sum_{N=0}^{N-1} f(m,n)\cos\left[\frac{(2m+1)\pi u}{2M}\right]\sin\left[\frac{(2n+1)\pi u}{2N}\right] \quad (1)$$

Where

$$a(u) = \begin{cases} \frac{1}{\sqrt{M}}, & u = 0 \\ \sqrt{\frac{2}{M}}, & 1 \le u \le M-1 \end{cases}$$

$$a(v) = \begin{cases} \frac{1}{\sqrt{N}}, & v = 0 \\ \sqrt{\frac{2}{N}}, & 1 \le v \le N-1 \end{cases}$$

The higher value DCT coefficients are then extracted in a zigzag fashion and stored in a vector sequence, which is shown in the Figure 8.

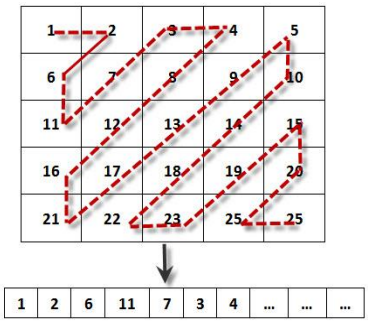

Figure 8. Rearranging DCT coefficients from zigzag order into one vector

By applying DCT, an image of a character is represented by this vector. Thus, one of the main characteristics of DCT is its ability to convert the energy of the image into a few coefficients. We have selected first 100 values of DCT in a zigzag manner and generated a feature vector of size 100 for SVM and kNN.

4.1.3 Zoning Features

For the extraction of zoning features, we have divided image into 3x3, 5x5 and *7x7* zones and the percentage of black pixels in each zone is calculated and various experiments were conducted. A unique feature of Urdu is that it is written in right to left and top to bottom directions, so it can be observed that majority of the character information lies along the diagonals (Figure 9). To extract the diagonal information, the percentage of black pixels in diagonal zones in the image are also calculated and added to the feature vector.

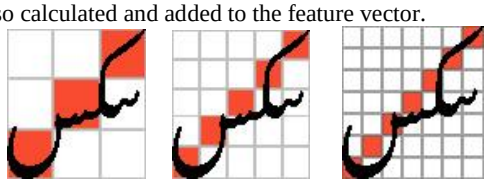

Figure 9. Diagonal Zone in 3x3, 5x5, 7x7 Zoning

4.1.4 Feature Extraction for HMM

We have used sliding window principle and extracted DCT features and 16 features for each frame discussed by Muhtaseb et al. [8]. We have chosen frame size width of 2 pixels as shown in Figure 10.

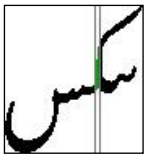

Figure 10. Sliding window for feature extraction for HMM

Muhtaseb et al. [8] divide each frame into 8 zones by dividing the height of the image pixel by 8 and finding the sum of black pixels in each of the zone. After calculating the F_i where $1 \leq i \geq 16$ feature they find other 8 features by summing the already calculated features as:

$F_9 = F_1 + F_2$
$F_{10} = F_3 + F_4$
$F_{11} = F_5 + F_6$
$F_{12} = F_7 + F_8$
$F_{13} = F_9 + F_{10}$
$F_{14} = F_{11} + F_{12}$
$F_{15} = F_9 + F_{10}$
$F_{16} = F_{13} + F_{14}$

In our feature extraction for the Urdu Nastalique script features, we have made height of the image multiple of 8 by padding 0 to the bottom of the image. We have also divided the sum of black pixels of zones by the total number of black pixels in each of the frame zones. Other 8 (F_9 to F_{16}) feature areas are calculated as discussed in [8]. Similarly, we have calculated DCT feature values for each frame. We have extracted 64 Features for each of the primary components in a zigzag fashion.

5. EXPERIMENTS

5.1 Primary Component Recognition

For the training of SVM, kNN and HMM classifies, we have taken maximum 15 training samples for each primary component of the ligature. We extracted the Gabor, DCT and zoning features and tested them with SVM, kNN and HMM classifiers. We have trained SVM for Linear and polynomial type kernel.

In testing set, we have 4380 primary component test images, which are not part of the training data set. We can observe that DCT with SVM (Linear kernel) classifier has recognition accuracy of 98.01%, while for kNN classifier DCT has 96.78% accuracy. Gabor features and zoning features have much lesser accuracy for both the classifiers. SVM and kNN classification results with different features are shown in Table 1.

Table 1. SVM and kNN classification results with different features

Features	Linear SVM	Polynomial SVM	kNN K=1
5x5 Zoning (Feature Vector Size 25)	93.92%	90.01%	85.47%
7x7 Zoning (Feature Vector Size 49)	94.74%	91.22%	88.96%
7x7 Zoning + 3x3 Zoning + diagonals (Feature Vector Size 68)	95.11%	91.06%	89.26%
DCT (Feature Vector Length 100)	98.01%	87.36%	96.78%
Gabor (Feature vector length 189)	94.95%	94.98%	92.71%

We trained HMM for all the odd states i.e. from 3 states to 33 states. Some of the primary components cannot be trained for the higher number of states. So we have used largest lower state HMM model for the image ligatures, which were successfully trained. We have achieved a maximum accuracy of 81.48% for the features extracting using zoning at 33 states* and 92.01% accuracy for the DCT at 23* states (*It also contain HMM model for lower states for which it cannot be trained). We observed that HTK failed to train ligature whose primary components has lesser number of frames than the number of states for which it is trained.

A complete result of HMM at different states with DCT and frame based zoning features is shown in Table 2.

Table 2. HMM classification results with different features

States	DCT (64 Features per frame)	Zoning Features (16 Features per frame)
3	69.33	32.44
5	74.58	30.84
7*	79.53	49.61
9*	83.72	60.40
11*	86.17	67.48
13*	87.48	72.09
15*	89.31	73.94
17*	89.90	75.15
19*	90.83	77.08
21*	91.68	77.62
23*	92.02	78.42
25*	91.27	79.94
27*	91.81	79.89
29*	91.50	80.45
31*	90.96	80.84
33*	90.55	81.48

Another issue we faced during recognition, is the high execution times due to large number of classes. The SVM classifier took 270 seconds to recognize an A4 size page containing 400 primary components on Intel i5 processor, while HMM with 23 states took 116 seconds and kNN took 30 seconds to for recognition.

To increase the efficiency of the recognition and reduce the recognition time, we divided the 2190 classes into 9 sub classes based on combination of area (small, medium, large) and aspect ratio (less than 1, between 1 and 1.2 and greater than 1.2). Some of the classes are present in more than one sub class, as for example if some training samples of a class had aspect ratio 1.27 and others had aspect ratio 1.13, then they would be present in both sub class for aspect ratio greater than 1.2 and aspect ratio between 1 and 1.2 . Now instead of a single SVM or kNN for 2190 classes, we have nine SVMs and kNNs for much smaller number of classes each. For recognition of a primary component, its relevant classifier is invoked. If say, a primary component has small area and its aspect ratio is 1.3, then it will be sent to the classifier meant for recognizing small components and aspect ratio greater than 1.2. Even though there was not substantial increase in recognition accuracy, but there was substantial decrease in execution time. As for example, the SVM classifier now took 77 seconds to recognize the 400 primary components while kNN classifier took 26 seconds.

5.2 SECONDARY COMPONENT RECOGNITION

The secondary components have only 17 classes in comparison to 2190 primary component classes. For the secondary component recognition, we have experimented with zoning, Gabor filters and DCT features. In the training

data, we have used a minimum of 100 samples for each class and tested 10 samples, which are not part of the training set. Table 3 shows the result of secondary component recognition with different features and classifiers. As we can see, the combination of DCT and SVM (polynomial) classifier attains 99.91% recognition accuracy.

Table 3. Experimnetal results of diacritics recognition

Features	Linear SVM	Polynomial SVM	kNN K=1
5x5 Zoning (Feature Vector Size 25)	94.37%	90.62%	91.30%
7x7 Zoning (Feature Vector Size 49)	91.25%	90%	90.06%
DCT (Feature Vector Length 100)	96.87	99.91%	92.54%
Gabor (Feature vector length 189)	94.37	95%	91.92%

6. FORMATION OF LIGATURE UNICODE STRING

Once the primary and secondary components have been recognized, their results are combined to form the Unicode string representing the ligature. For this purpose, a Binary search Tree is created from the code book file. Nodes in the BST contain the primary component codes and the associated secondary components are stored in a linked list. Nodes of the linked list contain secondary component string and ligature as shown in Figure 11.

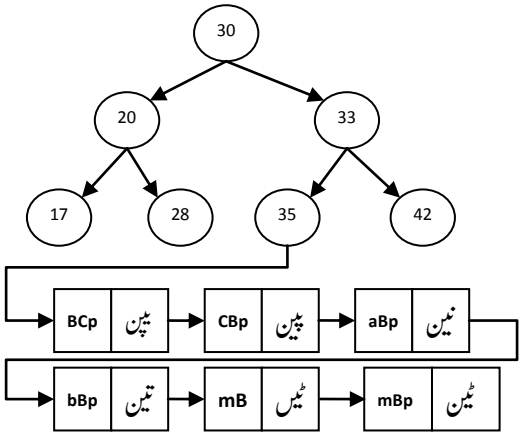

Figure 11. A sample node of BST Representation of Code book

We search node of primary component in the BST and then search the secondary string in the linked list attached to that node, to get the Unicode string. For example if the primary component has code 35 and secondary string of the ligatures is 'CBp', then to get the Unicode string, first the code 35 is searched in the BST and then the corresponding linked list is searched for string 'CBp' to get the Unicode string پین (Figure 11).

7. CONCLUSION

In this paper an Urdu ligature recognition system has been presented, which recognizes 9262 Urdu ligatures with more than 98% recognition accuracy using DCT and SVM (linear kernel) classifier. The ligatures are assumed to be pre-segmented and noise free. We are now working for recognizing broken ligatures and ligatures containing touching secondary components. We are also working on reducing the recognition time.

8. ACKNOWLEDGMENT

This research work is sponsored by Ministry of Communications and Information Technology under the project: Development of Robust Document Analysis and Recognition System for Printed Indian Scripts.

9. REFERENCES

[1] Husain, S. A. and Amin,, S. H. 2002. A Multi-tier Holistic Approach for Urdu Nastaliq Recognition. IEEE INMIC, (Karachi, December 2002), DOI=http://dx.doi.org/10.1109/INMIC.2002.1310191

[2] Sattar, S. A., Haque, S., Haque, Shamsul and Pathan,, M. K. 2008. Nastaliq Optical Character Recognition. Proceedings of the 46th Annual Southeast Regional Conference on XX, 329-331.

[3] Ahmad, Zaheer, Orakzai, Jehanzeb Khan, Shamsher, Inam, and Adnan, Awais. 2007. Urdu Nastaleeq Optical Character Recognition. Proceedings of World Academy of Science, Engineering and Technology, Volume 26, 249-252.

[4] Mukhtar, Omar, Setlur, Srirangaraj and Govindaraju, Venu. 2009. Experiments on Urdu text recognition. In Guide to OCR for Indic Scripts, Advanced in Pattern Recognition, Springer, 163-171.

[5] Javed, S. T., Hussain, S., Maqbool , A., Asloob, S., Jamil, S. and Moin, H. 2010. Segmentation Free Nastalique Urdu OCR. Proceedings of World Academy of Science, Engineering and Technology 2010, 456-461.

[6] Lehal, G. S. 2012, Choice of recognizable units for Urdu OCR. In Proceedings of the Workshop on Document Analysis and Recognition (Mumbai, India, DAR 2012), Publisher ACM, USA. 79-85.

[7] Lehal, G. S. 2013. Ligature Segmentation for Urdu OCR. In Proceedings ICDAR 2013, in press

[8] Muhtaseb, Husni A. Al, A. Mahmoud, Sabri and S. Qahwaji, Rami. 2008. Recognition of off-line printed Arabic text using Hidden Markov Models. In the journal of Signal Processing, 2902-2912. DOI=http://doi.acm.org/10.1145/2431211.24312 22