Indian Journal of Science and Technology, *Vol 8(35), DOI: 10.17485/ijst/2015/v8i35/86807, December 2015*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Offline Urdu OCR using Ligature based Segmentation for Nastaliq Script

Ankur Rana[1*] and Gurpreet Singh Lehal[2]

[1]Department of Computer Engineering, Punjabi University, Patiala - 147002, Punjab, India; ankurrana628@gmail.com
[2]Department of Computer Science, Punjabi University, Patiala - 147002, Punjab, India; gslehal@gmail.com

Abstract

There are two most popular writing styles of Urdu i.e. Naskh and Nastaliq. Considering Arabic OCR research, ample amount of work has been done on Naskh writing style; focusing on Urdu, which uses Arabic character set commonly used Nastaliq writing style. Due to Nastaliq writing style, Urdu OCR poses many distinct challenges like compactness, diagonal orientation and context character shape sensitivity etc., for OCR system to correctly recognize the Urdu text image. Due to compactness and slanting nature of Nastaliq writing style, existing methods for Naskh style would not give desirable results. Therefore, in this paper, we are presenting ligature based segmentation OCR system for Urdu Nastaliq script. We have discussed in detail various unique challenges for the Urdu OCR and different feature extraction techniques for Ligature recognition using SVM and kNN classifier. The system is trained to recognize 11,000 Urdu ligatures. We have achieved overall 90.29% accuracy tested on Urdu text images.

Keywords: Feature Extraction (DCT, Directional, Gabor and Gradient), K-Nearest Neighbor, SVM, Urdu OCR

1. Introduction

Optical character recognition is technique used to convert images data into editable text format. Optical character recognition is used for digitization of the printed text material like books, newspapers and old literature book, blind book reader, banks etc. A lot of research is being done for Arabic languages. Urdu also shares its character set with the Arabic character set. Generally Arabic language is written from left and right in Naskh writing style whereas Urdu is written in Nastaliq writing style from left to right. Nastaliq writing style is cursive writing style. Therefore very little work has been done for Urdu language because of Nastaliq writing style. For the cursive script like Urdu, segmentation and segmentation free approaches were being used by the researchers. Most of the researchers either worked on isolated Urdu character recognition or the small set of the Urdu ligature.

Shamsher et al.[1] and Ahmad et al.[2] developed OCR for printed Urdu script. They used feed forward neural network for training and classification of the Urdu character. Shamsher et al.[1] extracted the extreme points of all characters as feature vector for the neural network. They reported 98.3% accuracy in recognizing individual Urdu character. Ahmad et al.[2] experimented with the structural features of the machine printed ligatures. They reported 70% accuracy. Pathan[3] worked on the handwritten isolated Urdu characters.

Al Muhtaseb et al.[4] used HMM for developing Urdu OCR. They used vertical sliding with three pixels width and extracted 16 features by dividing window into 8 equal sizes and the sum up number of black pixels in each subwindow. They used 2500 lines for the training and 266 lines for testing. They achieved 99% accuracy for the Arial font. Their system is font style and font size depended.

Hasan et al.[5] used LSTM (Long Short-Term Memory) neural network to recognize the printed Urdu Nastaliq text. They also used sliding window of width 30 for feature extraction. They took pixels values as the feature vector. They reported 94.85% character recognition accuracy.

Lehal and Rana[6] explored ligatures recognition for the Urdu OCR. They have divided ligature into two

*Author for correspondence

components i.e. Primary component (main shape of ligature without diacritics) and secondary component (diacritics). They have achieved 98% recognition rate in primary component and 99.9% recognition rate in diacritics component using SVM as classifier. Lehal[7] and Hussain[8] worked on the segmentation of the Nastaliq Urdu OCR.

We have considered 10082ligatures for the development of Urdu OCR. Our approach is segmentation based approach. We are considering pre-segmented text for the recognition. We have divided our ligatures into two sub components i.e. i) Primary Component and ii) Secondary Component. So our total classes are reduced to 1845 classes for primary component and 19 classes for the secondary components. We have achieved total 98.15% accuracy for primary component and 99.91% accuracy for the secondary components. Total overall accuracy on input of Urdu text image (having 500 ligatures on average) achieved for the Urdu OCR is 90.29% accuracy.

2. Overview of Urdu

Urdu is the national language of the Pakistan and official language of six Indian states Delhi, Jammu and Kashmir, Uttar Pradesh, Bihar, Andhra Pradesh and Telangana. Urdu is the national language of the Pakistan and official language of six Indian states Delhi, Jammu and Kashmir, Uttar Pradesh, Bihar, Andhra Pradesh and Telangana. Urdu language is derived from the Farsi script. Urdu language is particularly important as it has been vastly used by poets for composing their poetry. Urdu is written in Nastaliq style whereas Arabic is written in the Naskh style. Figure 1 shows the same Urdu text in two different writing styles.

Urdu has 38 basic letters. Figure 2 shows all the basic letters of Urdu script. It is written from right to left whereas Urdu numerals are written as roman numerals i.e. from left to right.

Urdu characters are classified as joiner and non-joiner. Those Urdu characters which join with the preceding character but not with the succeeding character termed as non-joiner. There are 12 non joiners in the Urdu as shown in Figure 3. Rest of character exception these 12 non joiners connect with succeeding and proceeding character and change its shape.

Figure 1. Red and Blue color text is in Nastaliq and Naskh writing style respectively.

3. Challenges in Urdu Recognition

The development of OCR for Urdu Script involves many unique complexities which are as follows:

- Urdu is written diagonally from right to left and top to bottom. All ligatures look tilted at some angle from top right to bottom left direction as the different joiners are written in Urdu. Numerals add another level of complexity to the Urdu OCR. As observed from the books having both Roman as well as Arabic Numerals written. It is found that Urdu and Roman numerals are written left to right in the Urdu as show in Figure 4.
- As Urdu is mostly written diagonally from right top to bottom left, there is problem of segmentation. This poses problem for the character and word segmentation. Figure 5 exhibit shows overlapped ligatures marked as in red, green and blue color.

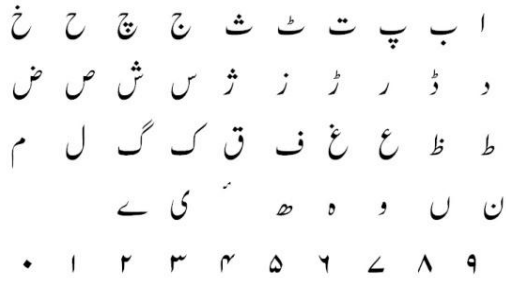


Figure 2. Urdu alphabets and number.

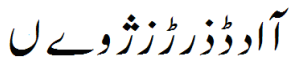


Figure 3. Non-joiner in Urdu.

Figure 4. Urdu writing style in nastalique from right to left and number written from left to right.

Figure 5. Ligature overlapping.

- Urdu is context sensitive script as well. Context sensitive means that the shape of the character depends on the succeeding character. When different characters are written together, the shape of the character depends on the shape of the character it follows as shown in table. Character in red color in the left hand side of the equal to is the character which when written with other characters changes its shape on the right hand side, because of the context sensitive nature of script, where in Naskh each character has its four shapes i.e. isolated, middle, left and right as show in Table. But Urdu in Nastaliq script has more than four shapes for its character set. Naz[9] reported 32 different shapes of the one character in nastaliq script.
- One of the major problems of the Urdu OCR is the broken character in the image. If one of the diacritics is not printed or not correctly scanned then the OCR cannot correctly identify it. In Figure 6 ligature is broken at the end. Consequently, OCR will get confused whether to treat broken ligature as one component or more than one.
- Another problem with the Urdu OCR is the case merged characters. In Urdu different ligatures have same basic shape but different diacritics. From the diacritics of the ligature, we can identify the correct meaning of the ligature. But some time diacritics get merged with the primary shape of the ligature and it is difficult to segment even ligature primary shape and diacritics as shown in Figure 7.

Table 1. Different Shape of tey, tay and meem with different other characters

تن = ت+ن	تب = ت+ب	تو = ت+و
ٹڈ = ٹ+ڈ	ٹن = ٹ+ن	ٹب = ٹ+ب
مر = م+ر	من = م+ن	مل = م+ل

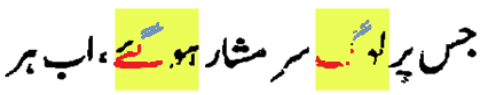


Figure 6. Broken ligatures in urdu text.

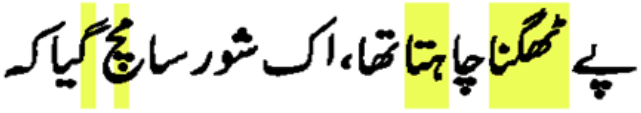


Figure 7. Merged diacritics with ligature in Urdu text.

- There is very little space in between words in Urdu. Different ligatures either have space or non-joiner at end as a last character. As shown in Figure 8, bar in green color shows the boundary between different words where bar in red bar represent the space between two ligatures in one words. From the space between ligature and words, we cannot find the word boundary between different words.
- Line segmentation also pose problem in Urdu OCR. Because of the diagonal writing style of the script, two ligatures in two difference lines get merged as shown in Figure 9.
- It is observed in some Urdu books footer of the page is written in Naskh style which add another complexity to the recognition. Having both Naskh and Nastliq writing style on one page adds another level of challenge in correct recognition of the page. As shown in Figure 10, Urdu text in green color is in nastaliq writing style and red color Urdu text is in Naskh writing style. Having both type of script on page is type of multiscript recognition which further increases the complexity of the system.

4. Urdu Data Preparation

Urdu text printed in books and newspapers can be divided into two generations. The books printed before 1995 are

Figure 8. Urdu text with very small space between ligatures.

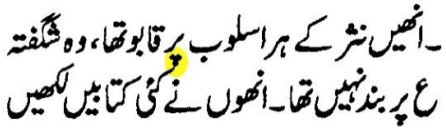


Figure 9. Two ligature in different lines get merged

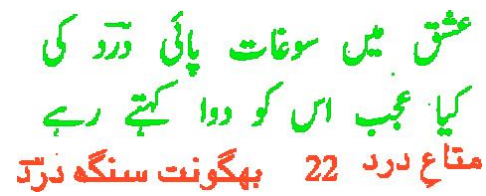


Figure 10. Urdu Text written in two different styles (multiscripts on one page).

all hand written while majority of the books published after 1995 use computer generated Nastaliq fonts such as Alvi Nastaliq or Noori Nastaliq. Shape of the characters in Urdu depends on the shape of the character it follows. That's why we cannot take the character unit as the classification unit for the recognition. Ligature segmentation at character level is shown in Figure 11.

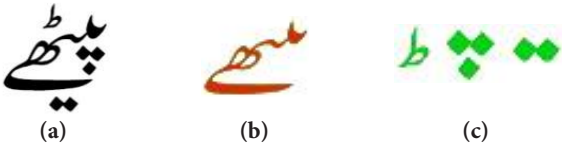
For the development of Urdu OCR we have taken ligatures as a classification unit. Ligature is the connected component and has different Urdu characters and end character is either non joiner or space. Urdu words can have more than one ligature. As for example, the word (بادشاہ) (*badshah*) is composed of four ligatures: two ligatures having multiple characters are (با and شا) and have two single character ligature (د and ہ). The two ligatures with multiple character are further composed of two characters each (ب+ا = با and ش+ا = شا)

Lehal[10] did the statistical analysis of the recognizable unit for Urdu OCR. He has taken 6,533,057 words corpus and identified 25,957 unique ligatures. He identified nearly 10082 ligatures which are used 99% time in the whole corpus.

We have developed our proposed system with these 10082 ligatures. But to collect the training data for 10082 ligatures from the books is very cumbersome task. To decrease the number of recognition classes, Lehal[10] has separated ligature into primary and secondary component as shown in Figure 12.

After removing diacritics from the 10082ligatures and grouping ligatures having same primary component, we have 1845 classes of the primary component and 16 secondary components. When these secondary components are used along with 1845 primary component we get a total of 10082ligatures.

Figure 11. Urdu word with character segmentation.



(a) (b) (c)

Figure 12. **(a)** Urdu Ligature **(b)** Ligature primary component **(c)** Ligature secondary component.

We have gathered training data from various scanned books. Some of the least frequent ligatures, as mentioned by Lehal[10], rarely occur in some books. To generate the training data for those primary components of the ligature, we made some synthetic images with different font size i.e. 35, 38, 40, 45, 50, 55 and different format option like bold or regular. We have removed diacritics marks from these ligatures to get the primary component. We have gathered total 1200 primary component from the scanned books and rest of the primary components were generated synthetically. Even in 1200 primary component, some of primary component samples are less than the required number of samples. To complete the samples we mixed synthetic and scanned books samples. We have also trained Urdu numerals and roman numerals as the primary component. Sample of the primary components are shown in Figure 13.

We have also collected samples of the merged ligatures. Merged ligature are those ligature in which diacritics merge with the primary component shape. As merged characters or touching character pose problem for the recognition in any OCR system. After analyzing scanned images from books, we have found some diacritics components merged with the primary component shape. We have collected total 41 such primary components merged with the existing primary components. Some of the diacritics touching primary components are shown in the Figure 14.

We have collected150 samples for each secondary component from the scanned books. Sample of the secondary component is shown in the Figure 15:

Complete statistics of training data is given in following Table 2.

Figure 13. Primary component training sample data.

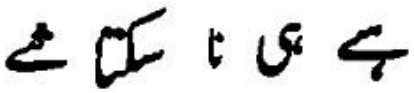


Figure 14. Sample of merged primary component with secondary component.

Figure 15. Secondary component training sample data.

Table 2. Training data statistics

Total Ligatures	10082
Primary Components (After removing diacritics from the ligatures)	1845
English Numerals	10
Urdu Numerals	10
Secondary Components	10
Punctuation marks as Primary Component	9
Punctuation marks as secondary Component	6

5. Features Extraction

For the classification of any pattern, relevant features have to be extracted. For Urdu OCR many researchers use different features. We have recognized primary and secondary components. For classification of the primary component of the ligature, we calculated DCT, Gabor, directional and gradient features.

5.1 Discrete Cosine Transformation (DCT)

DCT is the statistical feature extraction technique. DCT maps ligature image from spatial domain to the frequency domain. DCT maps the entire high frequency component to the upper right corner of the image matrix and low frequency components maps to the bottom right corner of the image. DCT coefficients f(p, q) of image I(m, n) are computed by equation(1) :

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

Where

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

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

We have scaled all the images to 32×32 for the DCT. Whole image DCT gives us 1024 frequency components. DCT has one important property that left top of the DCT matrix gives high frequency component. High frequency component means maximum information about the image is stored on top left the DCT matrix. We have extracted total 100 features from the total of 1024 feature values in zigzag manner as shown in Figure 16.

5.2 Gabor Features

Gabor function G(i,j) is the linear filter. Rajneesh Rani et al.[11] used gabor features for the script identification. It is used for the edge detection in image processing. It is multiplication of harmonic function and Gaussian function.

$$G(m,n)=P(m,n)*C(m,n)$$

This is used both for the orientation and spatial frequency. A Gabor filter is defined as

$$m,n,\theta,\sigma_m,\sigma=\begin{bmatrix}n\\ e^{\left[\frac{-1}{2}\left[\frac{R_1^2}{\sigma_m^2}+\frac{R_2^2}{\sigma_n^2}\right]\right]}\\ G\end{bmatrix}*e^{j\frac{2\pi R1}{\lambda}}$$

where $R_1=m\cos\theta+n\sin\theta$ and $R_2=-m\sin\theta+n\cos\theta$

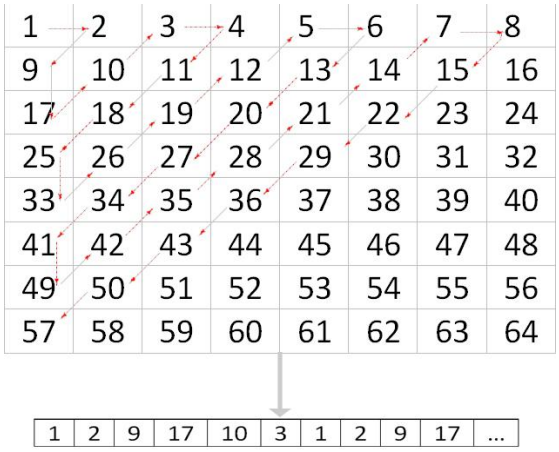


Figure 16. DCT coefficients of image selected in zigzag direction into one vector.

θ is the orientation of sinusoidal plane wave, λ is the wavelength. σ_m and σ_n are the standard deviations. We have taken both the standard deviations equal of the feature extraction.

To calculate the feature of the input, first image is scaled to 32 × 32 pixels. Then it is further partitioned into four equal non overlapping sub regions of size 16 × 16. These sub regions are again further partitioned into 4 non overlapping sub-sub regions of size 8 × 8. After 8x8 sub region division we get total 16 small regions. These 21 images are then convolved with odd symmetric and even symmetric Gabor filters in nine different angles, of orientation θ of 20 degrees, to obtain a feature vector of 189 values.

5.3 Directional Features

Directional features[12] calculated the distance of black and white pixels in eight different directions for each pixel. We have scaled our input image to 36 × 36 pixels. After scaling, directional features vector is calculated in eight different direction i.e. 0°, 45°, 90°, 135°, 180°, 225°, 270° and 315°. It gives directional feature vector of length 16 for each pixel. To down sample 20736 (16*36*36) directional features value, we divided our image into 9 (3 × 3) zones. Then we have generated 16 feature vector lengths from each zone by adding corresponding directional feature vector values of all the pixels in that zone. So, we have obtained directional feature vector of length 144 i.e. 16 feature values * 9 zones.

5.4 Gradient Features

A gradient feature[12,13] calculates the magnitude and direction of the maximum changes in intensity in the neighborhood of the pixels. For the gradient features extraction, first image is normalized to 63 × 63 pixel sizes. After normalizing image, the gradient vector is calculated in both x and y direction at each pixel position using the sobel operator as shown in Figure 17.

$$g_x(x, y) = z(x + 1, y - 1) + 2z(x + 1, y) + z(x + 1, y + 1) - z(x - 1, y - 1) - 2z(x - 1, y) - z(x - 1, y - 1)$$

$$g_y(x, y) = z(x - 1, y + 1) + 2z(x, y + 1) + z(x + 1, y + 1) - z(x - 1, y - 1) - 2z(x, y - 1) - z(x + 1, y - 1)$$

-1	0	1
-2	0	2
-1	0	1

1	2	1
0	0	0
-1	-2	21

Figure 17. Sobel operator.

After calculating gradient vector, we calculated the magnitude and direction as given in equation.

$$Magnitude = \sqrt{g_x(x, y)^2 + g_y(x, y)^2}$$

$$direction = \tan^{-1}\left(g_x(x, y) / g_y(x, y)\right)$$

The direction gradient vector is then decomposed along 8 chain code directions (D0, D1, D2, D3, D4, D5, D6 and D7) as shown in Figure 18. After that, the character image is divided into 9 × 9 blocks (81 blocks). If the gradient vector lies between two directions, then it is decomposed, else, its magnitude of the vector is retained. This results in 63 × 63 × 8 values. Next, the spatial resolution 9 × 9 is reduced to 5 × 5 for the down sampling of every two horizontal and vertical blocks with 5 × 5 Gaussian filter to get the 200 features value per image.

6. Experiments

6.1 Primary Component Recognition

Lehal and Rana[6] reported 98.01% and 96.78% accuracy of the 2190 primary component classes using Support Vector Machine[14] (SVM) and k nearest neighbor respectively. We have found that out of 2190 primary component many primary shapes of the ligatures looks same. Therefore, our primary component count of the ligature is reduced to 1873. We have used DCT, Gabor, directional and gradient for the feature extraction of the primary component. We have used SVM (linear and polynomial kernel) and kNN classifier for the primary component recognition. Results primary component recognition with SVM classifier using linear and polynomial kernel having degree 3 and 4 is given in Table 3.

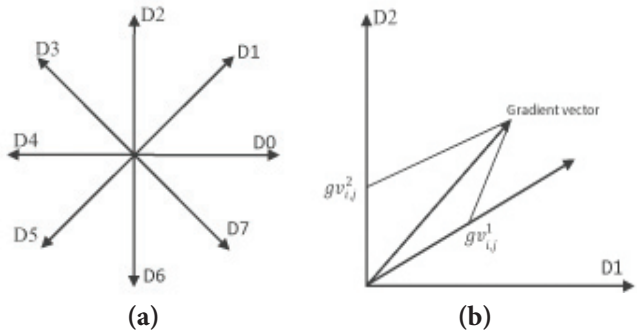


Figure 18. **(a)** Eight (D0–D7) chain code direction. **(b)** Breakdown of gradient vector.

We also experimented with k nearest neighbor classifier with different values of k (1, 3 and 5). Result of the primary component recognition using kNN classifier is shown in Table 4. With increase in value of k we have observed that output goes to 98.30%.

We observed that due to large number of classes SVM takes average 559 seconds to recognize 3746 primary components of ligature where as kNN classifier takes only 170 seconds to recognize the same.

Table 3. Primary component of ligature recognition using SVM classifier

Features	Feature Vector Length	Linear SVM	Polynomial SVM (Degree 3)	Polynomial SVM (Degree 4)
Discrete Cosine Transformation (DCT)	100	98.15	94.62	92.87
Discrete Cosine Transformation (DCT)	32	95.93	92.97	90.57
Discrete Cosine Transformation (DCT)	64	97.03	94.12	92.08
Gabor	189	94.68	94.55	90.15
Directional Feature	144	90.10	89.53	89.50
Gradient Features	200	95.14	91.56	93.96

Table 4. Primary component of ligature recognition using kNN classifier

Features	Feature Vector Length	K=1	K=3	K=5
Discrete Cosine Transformation (DCT)	100	95.17	97.96	98.30
Discrete Cosine Transformation (DCT)	32	93.34	97.23	97.88
Discrete Cosine Transformation (DCT)	64	94.90	97.78	98.27
Gabor	189	88.56	94.93	96.39
Directional Feature	144	86.32	94.07	95.79
Gradient Features	200	93.60	97.25	97.83

6.2 Secondary Component Recognition

We have 19 secondary component classes, for which, we have extracted DCT, Gabor and zoning features. For training samples, we have 100 samples for each class. Table 5 shows the result of secondary component recognition with different features and classifiers. As we can see, the combination of DCT and polynomial SVM (degree = 3) classifier attains 99.50% recognition accuracy.

We also use kNN classifier for the secondary component recognition with different values of k (1 and 3) Experimental result of the secondary component recognition with kNN is given in Table 6.

7. Formation of Ligature

After getting the primary and secondary component, we form ligature from the grouping of these two

Table 5. Secondary component of ligature recognition using SVM classifier

Features	Feature Vector Length	Linear SVM	Polynomial SVM (Degree = 3)
DCT (Feature Vector Length 100)	100	96.87	**99.50%**
Gabor (Feature vector length 189)	189	94.37	95%
Directional Feature	144	95.16	**95.69**
Gradient Features	200	96.77	93.01

Table 6. Secondary component of ligature recognition using kNN classifier

Features	Feature Vector Length	K=1	K=3
Discrete Cosine Transformation (DCT)	100	94.11	98.93
Discrete Cosine Transformation (DCT)	32	93.58	98.93
Discrete Cosine Transformation (DCT)	64	94.11	98.93
Gabor	189	93.04	97.32
Directional Feature	144	94.11	99.99
Gradient Features	200	93.04	99.46

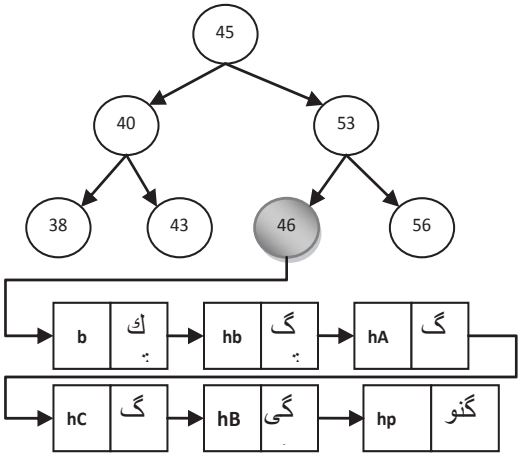


Figure 19. A sample of Binary Search Tree Depiction of Code book.

codes (primary and secondary component code). We manually crafted code book which comprises primary component code and secondary string code. From the combination of primary and secondary components code we extracted the ligature from the primary component code and the secondary string code. To search the combination of primary and secondary code, we implemented Binary Search Tree (BST). Nodes of the binary search tree contain the primary code and linked list of nodes having secondary code and their ligature code in Unicode. Binary search tree and nodes of the linked list of our code book structure is shown in the Figure 19.

Let primary component classifier gives a code of 46 and the secondary string code is hB. Then BST search gives ligature گنو as the identified ligatures.

8. Conclusions and Future Scope

We have used DCT with linear kernel SVM for the primary component. For the secondary component recognition we used DCT features (feature vector length 100) and polynomial kernel SVM (Degree 3). We have tested our system on 110 pages Urdu and got 83% accuracy. Urdu images having no broken or merged primary or secondary component and no Naskh style Urdu text have accuracy nearly 90.29%. New methodology needs to be devised to handle broken or merged ligatures. Also to recognize the Naskh writing style on the same Urdu text page, Naskh recognition OCR and font identification needs to be developed.

9. References

1. Shamsher I, Ahmad Z, Orakzai JK, Adnan A. OCR for printed urdu script using feed forward neural network. Proceeding of World Academy of Science, Engineering and Technology. 2007; 172–5. ISSN – 1307-6884.
2. Ahmad Z, Orakzai JK, Shamsher I, Adnan A. Urdu nastaleeq optical character recognition. Proceedings of World Academy of Science, Engineering and Technology. 2007; 26:249–52.
3. Pathan IK, Ahmed Ali A, Ramteke RJ. Recognition of offline handwritten isolated urdu character. Journal of Advances in Computational Research. 2012; 4(1):117–21
4. Al-Muhtaseb HA, Mahmoud SA, Qahwaji RS. Recognition of off-line printed Arabic text using Hidden Markov models. Journal on Signal Processing. 2008; 2902–12. DOI: 10.1016/j.sigpro.2008.06.013.
5. Ul-Hasan A, Ahmed SB, Rashid SF, Shafait F, Breuel TM. Offline printed Urdu Nastaleeq script recognition with bidirectional LSTM networks. International Conference on Document Analysis and Recongnition; 2013. p. 1061–5. DOI: 10.1109/ICDAR.2013.212.
6. Lehal GS, Rana A. Recognition of Nastalique Urdu Ligatures. Proceedings of 4th International Workshop of Multilingual OCR, ACM; Washington DC, USA. 2013. DOI: 10.1145/2505377.2505379.
7. Lehal GS. Ligature Segmentation for Urdu OCR. Proceedings 12th International Conference on Document Analysis and Recognition (ICDAR). IEEE. 2013. p. 1130–4. DOI: 10.1109/ICDAR.2013.229.
8. Sarmad H, Salman A, Qurat ul Ain Akram A. Nastalique segmentation-based approach for Urdu OCR. International Journal on Document Analysis and Recognition. 2015. DOI: 10.1007/s10032-015-0250-2.
9. Saeeda N, Khizar H, Muhammad Imran R, Mohammad Waqas A, Madani Sajjat A, Same KU. The Optical Character recognition of Urdu-like cursive script. Journal of Pattern Recognition. 2013 Oct; 11:1229–48.
10. Lehal GS. Choice of recognizable units for Urdu OCR. Proceedings of the Workshop on Document Analysis and Recognition (Mumbai, India, DAR 2012), Publisher ACM, USA. 2012; 79–85.
11. Rani R, Dhir R, Lehal GS. Gabor features based script identification of lines within a bilingual/trilingual document. International Journal of Advanced Science and Technology. 2014; 66(2014):1–12. ISSN: 2005-4238. Available from: http://dx.doi.org/10.14257/ijast.2014.66.01
12. Singh J, Lehal GS. Comparative Performance Analysis of Feature(S)-Classifier Combination for Devanagari Optical Character Recognition System. International Journal of Advanced Computer Science and Applications (IJACSA).

2014; 5(6). Available from: http://dx.doi.org/10.14569/IJACSA.2014.050608
13. Aggarwal A, Rani R, Dhir R. Recognition of devanagari handwritten numerals using gradient features and SVM. International Journal of Computer Applications. 2012. DOI: 10.5120/7371-0151.
14. Hsu C-W, Chang C-C, Lin C-J. A Practical Guide to Support Vector Classification. 2010. Available from: http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf
15. Akram Q, Hussain S, Adeeba F, Rehman S, Saeed M. Framework of Urdu Nastalique Optical Character Recognition System. In the Proceedings of Conference on Language and Technology. (CLT 14), Karachi, Pakistan. 2014.
16. Sabbour N, Shafait F. A segmentation-free approach to Arabic and Urdu OCR. SPIE Document Recognition and Retrieval XX, DRR'13; San Francisco, CA, USA. 2013 Feb.