Machines and Algorithms
http://www.knovell.org/mna

Research Article

Optical Character Recognition for Nastaleeq Printed Urdu Text using Histogram of Oriented Gradient Features

Muhammad Awais[1, *], Fatima Yousaf [2] and Tanzeela kousar[3]

[1]Department of Computer Science, Bahauddin Zakariya University, Multan, 60000, Pakistan
[2]Department of Computer Science and Information Technology, University of Chakwal, Chakwal, 48800, Pakistan
[3]Institute of Computer Science and Information Technology, The Women University Multan, 60000, Pakistan
[*]Corresponding Author: Muhammad Awais. Email: awaisahmadd555@gmail.com
Received: 12 October 2023; Revised: 1 January, 2024; Accepted: 5 February 2024; Published: 14 March 2024
AID: 003-01-000032

Abstract: The focus of research on optical character recognition (OCR) has been to digitize text in images. Urdu OCR is a challenging task because of its complexity, where a character can have multiple inflections depending on its position in the word, making it more difficult than English and similar languages. The proposed research aims to detect offline Urdu printed text using a segmentation-free approach, which means a holistic approach is taken. Horizontal histogram projection is used to extract text lines from an image, while connected components labelling is used for ligature segmentation in the extracted image to text line. To train the proposed model, a set of 14 statistical features along with HOG features are extracted for each sub-word/ligature. An open-source dataset UPTI is used to train and test the proposed algorithm, and SVM with RBF kernel function is used for the classification of ligatures. The proposed algorithm achieves a 97.3%-character recognition rate on the given dataset.

Keywords: Urdu language; Optical Character Recognition; HOG features; Connected Components; Support Vector Machine;

1. Introduction

Pattern recognition is a crucial aspect of both data science and computer vision, and its primary aim is to identify specific patterns within data and understand their connections. Optical character recognition (OCR) is a well-known example of this type of problem. Researchers have devoted significant effort to OCR over the last 30 years, but despite many advancements, there is still a requirement for more effective techniques to be developed [1].

Extracting text from printed documents is a difficult task for the Urdu language. With advancements in machine learning, there has been an increased expectation for text extraction from images, leading to the development of various approaches for Urdu optical character recognition (OCR) [2]. The field of OCR has seen significant improvement over the last 20 years, with widespread use in industries such as pharmaceuticals, accounting, and medical. OCR has numerous everyday applications, especially in the banking and financing sectors, such as reading and digitizing handwritten banker checks, verifying signatures, and sorting checks by zip code [3]. This technology has significantly reduced turnaround time, resulting in significant economic benefits. OCR is also being used for data processing by government and non-governmental organizations that require processing of thousands of survey forms [4]. The process of computerizing large volumes of paper documents and books typically requires significant human effort and time. However, automation of this process through OCR can efficiently save both time and human resources [5].

Urdu belongs to the group of cursive scripts, which is characterized by separate or linked characters that form partial words called Ligatures. The commonly used fonts for printing Urdu are Naskh and Nastaleeq. The proposed methodology focuses on recognizing printed Urdu text using the Nastaleeq font. To achieve this, Histogram of Oriented Gradients (HOG) is used as a feature descriptor, which is known for its effectiveness in image segmentation and object detection using machine learning classification models. In OCR, these descriptors can also be applied to represent sub-word or character images.

The focus of this article is on recognizing printed Urdu text in Nastaleeq font offline and the obstacles involved. To improve the recognition accuracy, gradient features have been utilized to classify individual sub-words and characters. Additionally, the approach taken in this study is holistic, treating each ligature as a recognition unit. The paper also emphasizes the procedure for gathering training data, which includes ligature images and corresponding class IDs (labels) obtained from text line images in the UPTI [6] dataset, stored in a separate file.

1.1. Contributions to the Proposed work

The following are the main achievements of our suggested investigation.:

- A new method has been presented in this research study for automatic extraction of training and validation data from the UPTI dataset. By utilizing this technique, ligature images for training can be produced from the UPTI dataset, while also obtaining their corresponding class ID.
- A collection of characteristics that can enhance the accuracy of recognition has also been proposed in this study. This set of features is composed of HOG-based and statistical features.

The rest of the paper is structured in the following manner: Section 2 contains an extensive discussion of the key contributions made towards creating a recognition system for printed Urdu text in an image. Several techniques for recognizing printed Urdu text are explored in this section. The implementation methodology and pertinent information about our proposed system are outlined in Section 3. Finally, Section 4 presents the output results of our experiments and a discussion of their implications.

2. Literature Review

The Urdu language shares a script similar to that of Arabic, which means that OCR techniques developed for Arabic can also be utilized for Urdu. Therefore, this section discusses the OCR work done on both languages. Research conducted by Saeeda Naz et al. [7] recognized Urdu script through an implicit segmentation method when combined with a Multidimensional-LSTM Recurrent Network operating on UPTI dataset information. The developed system displayed a Nastaleeq Urdu font recognition precision rate of 98%. A printed Urdu Nastaleeq font text recognition methodology proposed by Israr Ud Din et al. [8] uses a sliding window approach to extract nine statistical features totaling 116 dimensions for each sub-word image. An accuracy rate of 92% was achieved by applying these features to the UPTI dataset when using a Hidden Markov model (HMM) for training.

An all-encompassing approach served as the basis for Urdu text recognition work conducted by Toflk et al. [9]. The text lines in the document get separated by using horizontal histogram projection before any text recognition process begins. A connected component algorithm segments each sub-word through its procedure. Feature descriptors of SIFT and SURF are measured on each separated sub-word output with segmentation. The system generates 1600 categories of sub-words which include multiple diacritic marks. The system matches features from input document sub-words against the 1600 category sub-words for identification. When feature matching leads to the highest score a designated sub-word receives its corresponding ID value. The ID becomes comparable to the sub-word file for identifying the matching sub-word. A training of 23204 sub-words/ligatures within the system achieved a 95% accuracy rate.

Israr Uddin et al. [10] have introduced a comprehensive strategy for recognizing Urdu language, specifically focusing on the Nastaleeq Urdu Script. To accomplish this task, the authors utilized the Discrete Wavelet Transformation technique to extract features from sub-words, which were then utilized to train a Hidden Markov Model as a classifier. The authors evaluated the system's performance using 2000 distinct and commonly used Urdu ligatures from the center for language engineering (CLE) dataset [11]. The findings indicate that the system achieved a recognition accuracy rate of 88.87% on 10,000 Urdu sub-words.

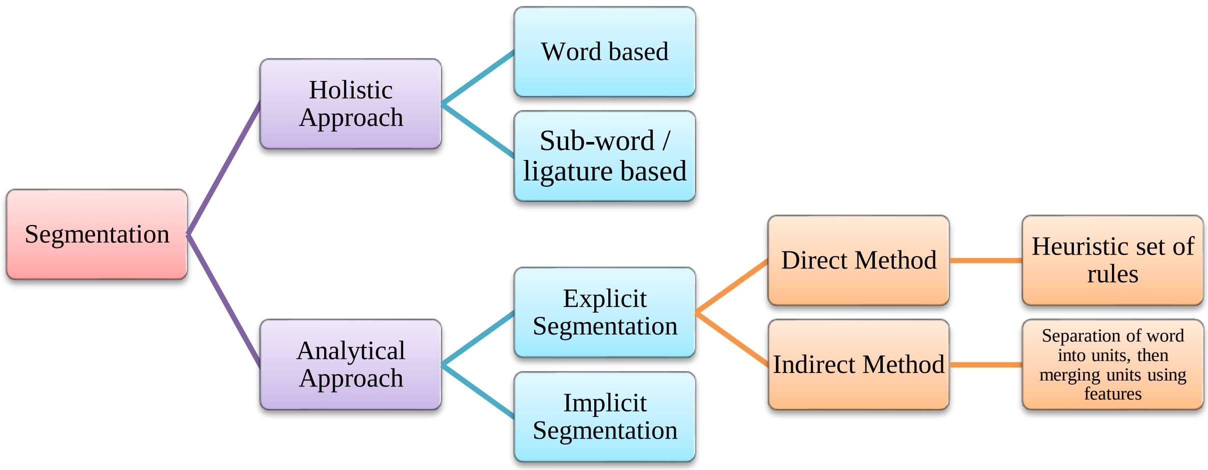


Figure 1: Categorization of OCR Techniques from the Segmentation Aspect

Pal & Sarkar [12] developed a technique to recognize isolated Urdu characters. The system segments the document image into text lines and then into individual characters. Character images are represented using water reservoir, topological, and contour features. The features include reservoir number, direction, flow level, position, and height, and topological components, loops, position relative to the character boundary, and loop-to-height ratio. The character contours are represented by projection profile features. A decision tree classification technique is used for character recognition.

Hussain et al. [13] developed an analytical approach for Urdu text recognition. They extracted primary and secondary ligatures from scanned document images and preprocessed them for noise removal. Characters were grouped into four classes, and character endpoints were computed using a local window sliding over every primary ligature's thinned image. They segmented 5,249 primary ligatures into 79,093 graphemes with 250 unique shapes. Low-frequency DCT features were computed using right-to-left sliding windows for all graphemes, and separate HMMs classifiers were trained for each grapheme class. For recognition, a query sub-word/ligature was split into primary and secondary ligatures, and the primary ligature was segmented into individual graphemes, which were recognized using trained HMM classifiers. The ligature was then generated by combining the recognized graphemes. The system achieved an 87.44% accuracy rate on 18,409 query ligatures.

In another work, Hassan et al. [14] employed BLSTM with CTC output layer to recognize Urdu text lines. Text line height is normalized to 30 pixels, and each column of a text-line image is fed to train the classification network. Results show accuracy rates of 94.85% and 86.43% for recognizing printed Urdu text lines with and without considering variations in character shapes, respectively. Ahmed et al. [26] used the same technique for recognizing cursive and isolated scripts. Hassan et al. achieved an accuracy of 96% on the UPTI dataset.

Table 1: Summary of Different Recognition Techniques

Study	Dataset	Recognition Techniques	Language Script	Results (accuracy)
Farhan M. A. Nashwan et al. [15]	Custom	DCT and center of gravity with Euclidean Distance score comparison	Arabic	84.8%
Ouled Jaafri Yamina et al. [16]	Custom dataset (30500 samples)	Set of 14 statistical features with SVC classifier	Arabic	95.03%
Hussein Osman et al. Error! Reference source not found.**]**	Watan-2004 APTI	ANN	Arabic	97.94%
Saad M. Darwish, Khaled O. Elzoghaly [18]	PATS-A01, APTI	14 statistical features from grey level Co-occurrence matrix with fuzzy KNN	Arabic	98.69%
Israr Uddin et al. [10]	CLE	DWT with HMM	Urdu	88.87%
Nazly Sabbour, Faisal Shafait [6]	UPTI	Shape context with KNN	Urdu, Arabic	86% (Arabic), 91% (Urdu)
Israr Ud Din et al. [19]	UPTI	116-dimensional Statistical features with HMM	Urdu	92%
Tofik et al. [9]	Custom	SIFT and Surf with Brute force Feature Matching	Urdu	95%
Mujtaba Husnain et al. Error! Reference source not found.	Custom	Statistical features and Raw pixels with CNN	Urdu	96.5%
Saeeda Naz et al. [7]	UPTI	MDLSTM (Analytical Approach)	Urdu	98%

Javed et al. [21] proposed an optical character recognition system using HMM classifier with 1282 high-frequency ligatures (HFLs). DCT-based features were used to represent each ligature image using sliding windows, and a separate class of models was trained for each ligature. A set of rules was used to associate recognized primary and secondary ligatures based on diacritic and dot position information. The system achieved a recognition rate of 92% on a dataset of 3655 ligatures, with errors mainly due to the system's inability to distinguish between ligatures that share the same primary main ligature body but differ only in diacritic and dot positions.

3. Proposed Methodology

The Following section describes the proposed methodology.

3.1. Dataset Description

The UPTI (Urdu Printed Text Images) dataset [22] is a freely available dataset that has been extensively employed for assessing various printed Urdu character recognition systems. Sabbour and Shafait [22] created this dataset in 2013, and it comprises 10,063 lines of printed Urdu text written in the Nastaleeq font, all of which were sourced from the Daily Jang newspaper [23]. The dataset is segmented into three sub-groups: images of printed lines, images of printed ligatures, and images with noise. Figure 3 displays some sample images from the dataset.

(a)
(b)
(c)

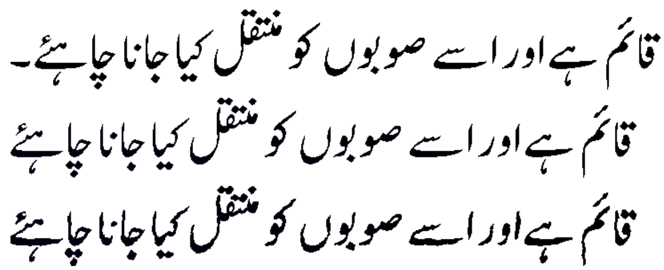


Figure 2: UPTI dataset printed text line Samples (a) Line text image – non-degraded (b) ligature-based image non-degraded (c) degraded/noisy ligature image.

If the dash that indicates the end of a sentence is taken away, the text image in a line is transformed into an image that is based on ligatures. To train the proposed system, distinct ligature training images are extracted from UPTI text line images by using the connected component labelling algorithm. The system's performance is then tested by using validation ligature images from the UPTI dataset that were not utilized in the training of the recognition model.

3.2. Methodology

This section provides a detailed explanation of the proposed recognition method. Each sub-word along with ligature serves as the core unit for recognition purposes within the developed approach. Holistic segmentation was selected over complex segmentation tasks because these approaches require extensive computation and pose significant identification challenges. Text lines have to be divided into sub-words/ligatures before an SVC classifier applies the recognition process for digitization of words through its output. The classification training process uses annotated ligature images drawn from text line images for extraction purposes. The training of the classifier depends on annotated ligature images. The trained classifier identifies predicted IDs for individual segmented ligatures contained in text-line images throughout the recognition phase. The predicted IDs are then matched with corresponding ligature text from a separate file containing all ligatures' texts and their IDs. The training process is explained in detail in the following section.

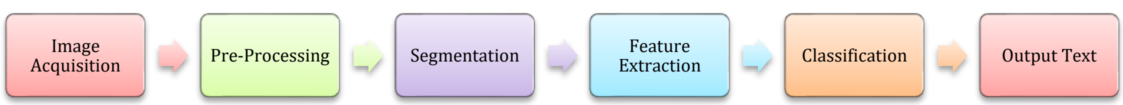
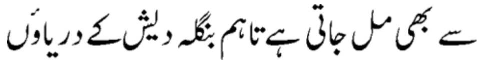


Figure 3: Ligature Recognition Process of Proposed OCR System

3.2.1. Preprocessing

The text lines provided in the dataset are in grayscale, but to make them more convenient for the designed classifier, they are converted to a binary form where each pixel is either 0 or 1. Figure 4 illustrates the process of converting the lines to binary.



a)

b)

Figure 4: a) Grayscale Image & b) Binarized Image

3.2.2. Ligature Segmentation

This section of the methodology focuses on the method of separating each ligature from a line of text. The connected component labelling algorithm is utilized to divide ligatures from images of text lines. This method of segmentation not only splits complete ligatures (combinations of primary and secondary ligatures) from the text line image but also separates primary and secondary ligatures from each other.

The process of extracting ligatures comes after binarization. This involves separating each image from a text line image and assigning a specific label to it, which is then replaced by a corresponding number during classification. The labels and their corresponding numbers are recorded in a CSV file. An illustration of this process can be seen in the figure below, which shows how a text line image is broken down into ligatures.

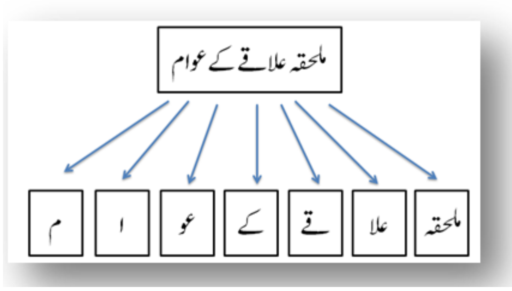
The output of the text split program	Unique Urdu Ligature	Ligature ID
	ا	1
	ملحقہ	5
	علا	6
	قے	7
	کے	8
	عو	9
	م	10

Figure 5: Ligature IDs of Unique Urdu Ligatures

Each Urdu script word requires one to several linked characters in order to form itself. The Urdu script characters follow predetermined rules while locking together into multiple ligature combinations. We can classify these ligatures into three types: complete, primary, and secondary. One complete ligature includes Urdu words with their diacritical marks and unwanted dots. With its diacritics and dots removed, the letter becomes a simple ligature structure. Secondary ligatures develop through dots alongside diacritics found in ligatures. The figure 6 under this section shows how these three ligature categories appear.

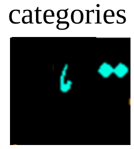
a)  b)  c) 

Figure 6: a) Complete Ligature, b) Primary Ligature and c) Secondary Ligature

The process of connected component labelling generates a primary and secondary ligature list from images of text lines with unique labels. The primary ligature list (PL_list) and secondary ligature list (SL_list) are produced. To create the complete ligature, the next step is to match the secondary ligatures with their corresponding primary ligatures. All complete ligatures are then saved in the complete ligature list (CL_list). The secondary ligatures are identified and separated from the combined list of primary and secondary ligatures extracted from each text line image during the segmentation process. The height of each ligature is calculated using its contour boundary, and stored in a separate list. After conducting experiments, it was determined that ligatures with a height less than 30% of the tallest ligature in the list are considered secondary and are stored in the SL_list. Starting indexes of diacritic marks in the SL_list are compared to the starting and end indexes of other primary ligatures in the combined list. If the starting index of a diacritic mark or secondary ligature falls between the starting and end indexes of a primary ligature in the PL_list, it is considered a part of that primary ligature and its label is replaced with the label of the associated primary ligature. All labels are stored in the ligature label list (LL_list).

Figure 7: Segmentation of text line using connected component labelling

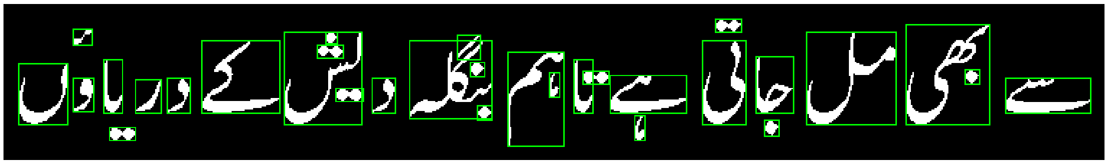


Figure 8: Identifying secondary ligatures from the ligatures list

Figure 9 illustrates how primary and secondary ligatures are linked. The significant role of excluding secondary ligatures from the complete list of all ligatures is emphasized in the association of ligatures. If the exclusion of secondary ligatures is not performed accurately, the segmentation process will be unable to separate complete ligatures from the text line images.

- **Algorithm 1: Algorithm for associating primary and secondary Ligatures**

Input: (*PL_list*, *SL_list*, *LL_list*)

Output: Updated connected components list having correct labels values for associated primary and secondary ligatures

```
DEFINE FUNCTION Ligature_Association(PL_list, SL_list, LL_list):
    FOR ( i ← 1  to length(PL_list))
        SET PL_st_idx ← PL_list [ i ][0]
        SET PL_en_idx ← PL_list[ i ][1]
        FOR ( j ← 1 to length(SL_list)):
            // Starting index of a diacritic mark in the diacritic list
            SET SL_st_idx ← SL_list[ j ][0]
            // Connected components label of a diacritic mark in the diacritic list
            SET SL_label ← SL_list[ j ][4]
            IF (SL_st_idx >= PL_st_idx AND SL_st_idx <= PL_en_idx):
                // Assigning primary ligature label to its associated secondary ligature
                // Connected components label of primary ligature
                SET PL_label ← PL_list[ i ][4]
                // updating labels
                SET LL_list[LL_list == SL_label] ← PL_label
            END IF
        END FOR
    END FOR
RETURN LL_list
```

Figure 9: Association of Primary and Secondary Ligatures

Algorithm 1 generates images of ligatures, which are subsequently labeled with their respective ground truth files for each text line derived from the UPTI dataset. The resulting output of the algorithm is presented below.:

Figure 10: UPTI text line segmentation using Connected component labelling

The ground truth for extracted ligatures is constructed using the following algorithm.

- **Algorithm 2: Text split algorithm**

1. Joiners ← ["ب","ت","ث","ج","ح","خ","س","ش","ص","ض", "ه","ي","ئ","ى","ط","ظ","ع","غ","ف","ق","ك","ل","م","ن"]
2. non_joiners ← ["إ","أ","ة","ء","ؤ","و","ز","ر","ذ","د","ا"]
3. **DEFINE FUNCTION** split(***Text_line***):
4. **FOR** word **IN** Text_line **do**
5. **SET** Text_ligature_list ← [ ] // initialize with empty list
6. **SET** Temp_Characters_list ← [ ]
7. **SET** complete_Characters_list ← [ ]
8. **FOR** char_id **and** char **IN** **enumerate**(word) **do** // taking each character from
   // single Arabic word to check
   // whether it is joiner or nonjoiner.
9. **FOR** joiner_character **IN** joiners **do** // first checking **char** using joiner list
10. **IF** char **EQUALS** joiner_character **do** **//if** a character is a joiner
11. **ADD** char **IN** Temp_Characters_list[ ] //keep adding characters
    //in list .
12. **END IF**
13. **END FOR**
14. **FOR** nonjoiner_character **IN** non_joiners **do** //checking **char** using nonjoiners
15. **IF** char **EQUALS** nonjoiner_character **do**
16. **ADD** char **IN** Temp_Characters_list [ ]
17. **SET** complete_Characters_list ← Temp_Characters_list
18. **SET** Temp_Characters_list ← [ ] //clearing the list to reuse
    // for the next word in the next iteration
19. **SET** complete_ligature ← NULL //variable to combine all
    // Characters in characters' list
    // to generate single
    // Ligature or sub-word
20. **FOR** each_character **IN** complete_Characters_list **do**

```
SET complete_ligature TO each_character + complete_ligature
END FOR
ADD complete_ligature IN Text_ligature_list [ ]
END IF
END FOR
// In the case we don't find any non-joiner, then all joiner characters
// stored in temp_characters_list will be output as a complete sub-
// word/ligature at the end of the word string length
IF char_id EQUALS length of (word) do        //if it is the last index of Arabic
//Word string
SET complete_Characters_list ← Temp_Characters_list
SET Temp_Characters_list ← [ ]
SET complete_ligature ← NULL
FOR each_character IN complete_Characters_list do
SET complete_ligature TO each_character + complete_ligature
END FOR
ADD complete_ligature IN Text_ligature_list [ ]
END IF
END FOR
END FOR
RETURN Text_ligature_list [ ]
```

The annotated ligatures are produced through algorithm-2. Table 2 shows the distribution of ligatures used in our experiments.

Table 2: Detail of Training and Validation Set

Sets	No. of ligatures
Training set	3,005
Validation set	92,315

The training set includes the standard and unique ligatures whereas the validation set is a rough set that contains duplicate ligatures with varying sizes which include noise of different levels.

3.2.3. Feature Extraction

A feature descriptor is an algorithm that generates a set of feature vectors from an image, which represent the most prominent features in the image. We utilized a feature set, which included the Histogram of Oriented Gradients (HoG) with 9 orientations and statistical features, to classify ligatures. The HoG descriptor captures the image's shape and structure by extracting gradients (changes in x and y direction) and orientations (magnitude and direction) of the features. The image is partitioned into smaller regions, and for each region, the gradients and orientation are computed. To create the Histogram of Oriented Gradients (HoG), a histogram is generated for each smaller region based on the gradients and orientations of the pixel values. The SVM is capable of handling a large number of classes by transforming the multi-class problem into multiple binary classification problems. To compute the HoG feature of a single sub-word/ligature image, the image is partitioned into several sub-portions, and gradients are computed for each block of 16x16.

The Gradient along the y-axis $\boldsymbol{G_y}$ of an Image I(x, y) is defined as the difference between the south pixels and north pixels of an image I(x, y).

$$\boldsymbol{G_y} = I(x, y+1) - I(x, y-1) \tag{1}$$

Similarly, a Gradient along the x axis $\boldsymbol{G_x}$of an Image I(x, y) is defined as the difference between the east pixels and west pixels of an image I(x, y).

$$\boldsymbol{G_x} = I(x+1, y) - I(x-1, y) \tag{2}$$

After computing the gradients of a ligature image along the x and y axis, its magnitude and direction are computed using equation 3.

$$G = \sqrt{(G_x)^2 + \left(G_y\right)^2} \tag{3}$$

$$\theta = arctan\left(\frac{G_y}{G_x}\right) \tag{4}$$

Computing Histogram of Gradients in 16×16 cells is done in the following steps.

- Each image of a ligature is divided into 16×16 cell blocks
- Along each 16×16 cell block HoG is calculated
- This gradient histogram is basically a 1D vector of 9 buckets (numbers) corresponding to angles ranging from 0 to 180 degrees (gap increments of 20 degrees).
- Values of these 256 cells (16X16) are binned and added into the 9 buckets of gradient histogram cumulatively.
- This process essentially reduces 256 values into 9 values for each cell block.

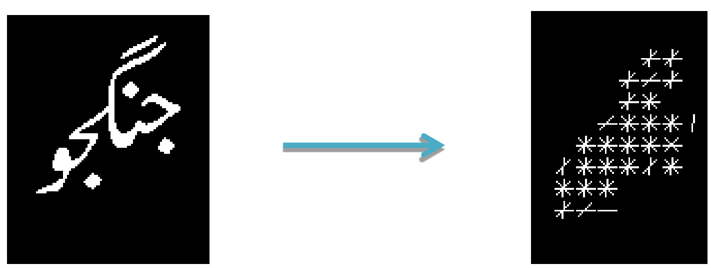


Figure 11: HOG features of the UPTI Urdu Ligature

The following statistical features are used in the system:

- Horizontal transition
- Vertical transition
- The ratio of black pixels over white pixels

To put it differently, the horizontal transition process entails examining of a sub-word image in the horizontal direction and tallying the number of instances where the pixel value changes from 1 to 0 or vice versa. Likewise, the vertical transition count method involves scanning the sub-word image from top to bottom and keeping track of the number of pixel value transitions.

Each sub-word image is divided into four parts. Top left portion, Top right portion, Bottom left portion, and Bottom right portion. The next 10 features are mentioned below.

- The Ratio of Black pixels over white pixels in the top left area of an image
- The Ratio of Black pixels over white pixels in the top right area of an image
- The Ratio of Black pixels over white pixels in the Bottom left area of an image

- The Ratio of Black pixels over white pixels in the Bottom right area of an image
- Number of Black pixels in Top left / Number of Black pixels Top Right
- Number of Black pixels in Bottom left / Number of Black pixels in Bottom Right
- Number of Black pixels in Top left / Number of Black pixels in Bottom left
- Number of Black pixels in Top right / Number of Black pixels in Bottom right
- Number of Black pixels in Top left / Number of Black pixels in Bottom right
- Number of Black pixels in Top right / Number of Black pixels in Bottom left

Holes: The number of holes presents within the sub-word image.

Our proposed system utilizes 15 features to classify sub-words or ligatures. These features are derived from the sub-word or ligature and passed on to the classifier to generate a class ID. Once the class ID is predicted, the corresponding sub-word/ligature text is selected from a separate file and added to a list to form a paragraph. Out of the 15 features, 14 are represented by a single integer or decimal value, while the HOG feature descriptor generates a feature map comprising 900 feature values distributed along 9 different orientations (100 features per orientation).

Table 3: Summary of features computed during feature extraction phase

Features	**Features Name**	**Feature size**
Gradient Features		
F1	HOG (9 orientations)	900
Gradient Features		
F2	Holes/loops	1
Statistical Features		
F3	Horizontal Transition	1
F4	Vertical Transition	1
F5	Black to White Ratio	1
F6	Black to White Ratio in Top left area.	1
F7	Black to White Ratio in Top right area.	1
F8	Black to White Ratio in the bottom left area.	1
F9	Black to White Ratio in the bottom right area.	1
F10	Black pixels in the Top left area / Black pixels in the Top right area	1
F11	Black pixels in the bottom left area / Black pixels in the bottom right area	1
F12	Black pixels in the top left area / Black pixels in the bottom left area	1
	Total Features	914

3.2.4. Training and Validation of Classifier

- Training of SVC classifier

The proposed system utilizes the Support Vector Classifier (SVC) as its machine learning classifier to predict the class ID for each segmented sub-word/ligature image based on a given set of features. The system is initially trained on annotated training data from the UPTI dataset, and during the recognition phase, the SVC classifier predicts the class ID for each segmented query image in a sequence similar to the sequence of words/sub-words in a paragraph. The predicted class ID for each sub-word is then stored in a list, which is used to select the corresponding sub-word/ligature text from a separate file.

The SVC classifier uses a hyperplane to separate data samples based on their feature points in n-dimensional feature space. To select the hyperplane that has the maximum distance from the nearest data points in either category, the concept of a kernel function K is introduced to transform non-linearly separable data in its input space into linearly separable data in a higher dimensional feature space. While selecting a hyperplane for linearly separable data is not challenging, most real-world problems deal with non-linearly separable data, making it more difficult to choose a hyperplane.

- Top of Form

$$K(x_i, x_j) = \emptyset(x_i) \,.\, \emptyset(x_j) \tag{5}$$

The right side of the equation $\emptyset(x_i) \,.\, \emptyset(x_j)$ represents the non-linear SVM function.

There are 4 kernel functions mostly used in SVM classification. In this work, we have used the Radial Basis function.

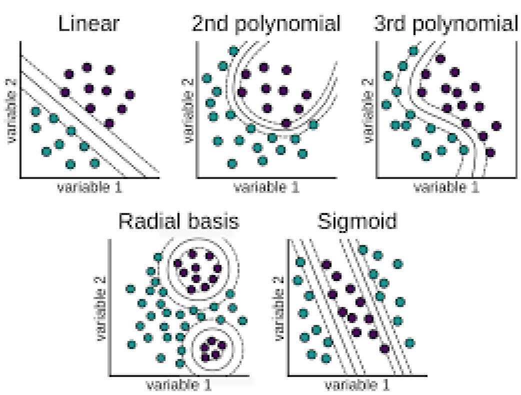


Figure 12: Different Kernel functions of the SVC Classification model.

Radial-based Function (RBF):

$$K(x_i, x_j) = exp\left(-\gamma \,\left|\left|x_i - x_j\right|\right|\right) + C \tag{6}$$

- Classifier Validation

The recognition task involves segmenting the printed Urdu text line images into individual ligatures images using the text line segmentation technique that was used during the training phase. This involves binarizing each text line image using the OTSU thresholding method and then extracting the ligatures from the image. Once the ligatures have been segmented, their features are used to recognize them and match them to their corresponding text. The training section provides more information about these features.

The dataset used for the validation purpose is more complex than the one used in training. Here are a few examples of the validation dataset.

Figure 13: Examples of Validation Set Extracted from UPTI

These features of each ligature image are passed to the SVC classifier that predicts their corresponding ligature ID. Using that predicted ligature ID the ligature text is selected from a separate ligature ID file that holds ligature text corresponding to their IDs.

4. Results and evaluation

This section presents the evaluation results of our proposed system on validation ligature images taken from the UPTI [6] dataset. The approaches mentioned in the proposed methodology section are evaluated on the validation set extracted from the UPTI dataset.

The proposed system is composed of two distinct classifiers. The first one is trained on 3005 unique Urdu ligature extracted from the UPTI dataset where each sub-word image has its own size (height and width) based on the ligature's dimensions. All statistical features are computed without resizing the ligature image. For the training process of calculating HoG features, each ligature image is resized to 100x100, and the Hog features are extracted along 9 orientations.

Figure 14: Ligature Images Samples Extracted from the UPTI Dataset.

4.1. Result Evaluation Metric

To evaluate the proposed system ligature recognition rate metric is used which is given below:

$$LRR = \frac{No\ \text{of}\ ligatures\ correctly\ classified}{Total\ Number\ of\ ligatures}$$

Along with each query image, we have placed the ground truth. Using the proposed text split algorithm explained in preprocessing section, the ground truth paragraph text is converted into individual sub-word texts and stored in a list. Then each predicted sub-word is matched to the sub-word/ligature present in the ground truth sub-words list. Whenever the segmented sub-word/ligature is not classified correctly, the matching score is not added to the score list. This score list is equal to the number of correctly classified sub-words and it is divided by the total number of sub-words to evaluate the accuracy of our recognition system.

The proposed system is evaluated on a validation set extracted from the UPTI dataset that comprises 92,000 images of printed Urdu ligatures belongs to 3,005 unique ligatures classes.

Table 4: Comparative Analysis of Various Studies.

Study	No. of Unique ligatures	LRR	Recognition of complete ligatures
Israr Uddin et al. [19]	2028	97.93%	No
Javed and Hussain [24]	1692	92.73%	No
Akram et al. [25]	1475	97.87%	No
Javed et al. [21]	1282	92.00%	No

Akram et al. ERROR! REFERENCE SOURCE NOT FOUND.	1475	87.15%	Yes
Israr Uddin et al. [10]	2017	88.87%	Yes
Proposed	3005	97.39%	Yes

5. Conclusion and Discussion

Our research work has presented a method for recognizing printed Arabic and Urdu Script without using segmentation. Our approach includes incorporating HOG feature descriptors, which have produced promising recognition results. We obtained our training data from the UPTI dataset, using 3005 unique ligature images of printed Urdu Script, and annotated the training and validation ligatures using the ground truth text files of the UPTI dataset. This research work has the following key findings, including HOG feature-based classification that outperforms other methods. The font size of the text in the image doesn't impact the recognition performance of the system when the system is trained using these HOG features. We also discovered that over-segmentation or under-segmentation can negatively impact recognition, and that appropriate preprocessing, such as thinning and noise removal, is crucial to avoid these issues. However, our proposed technique is limited to printed text and cannot handle handwritten text due to the complexity of ligature overlapping and variations in shape. Finally, future research directions may include exploring recognition in multi-font text, addressing text overlap during segmentation, and incorporating diacritics.

Declaration of Competing Interests: The Authors declare that they have no competing interest that could have been appeared to influence the work reported in this paper.

Acknowledgements: The research in this work has been supported by Center for Artificial Intelligence Research (CAIR), Department of Computer Science, Bahauddin Zakariya University, Multan, Pakistan. We also acknowledge the publishers of UPTI dataset which is used in this work.

References

[1] Singh, Amarjot, Ketan Bacchuwar, and Akshay Bhasin. "A survey of OCR applications." *International Journal of Machine Learning and Computing* 2, no. 3 (2012): 314.

[2] Khan, Naila Habib, and Awais Adnan. "Urdu optical character recognition systems: Present contributions and future directions." *IEEE Access* 6 (2018): 46019-46046.

[3] Agrawal, Prateek, Deepak Chaudhary, Vishu Madaan, Anatoliy Zabrovskiy, Radu Prodan, Dragi Kimovski, and Christian Timmerer. "Automated bank cheque verification using image processing and deep learning methods." *Multimedia Tools and Applications* 80 (2021): 5319-5350.

[4] Peng, Xujun, Huaigu Cao, Srirangaraj Setlur, Venu Govindaraju, and Prem Natarajan. "Multilingual OCR research and applications: an overview." In *Proceedings of the 4th International Workshop on Multilingual OCR*, pp. 1-8. 2013.

[5] Doucet, Antoine, Gabriella Kazai, Sebastian Colutto, and Günter Mühlberger. "Icdar 2013 competition on book structure extraction." In *2013 12th International Conference on Document Analysis and Recognition*, pp. 1438-1443. IEEE, 2013.

[6] Sabbour, Nazly, and Faisal Shafait. "A segmentation-free approach to Arabic and Urdu OCR." In *Document recognition and retrieval XX*, vol. 8658, pp. 215-226. SPIE, 2013.

[7] Naz, Saeeda, Arif Iqbal Umar, Riaz Ahmed, Muhammad Imran Razzak, Sheikh Faisal Rashid, and Faisal Shafait. "Urdu Nasta'liq text recognition using implicit segmentation based on multi-dimensional long short term memory neural networks." *SpringerPlus* 5 (2016): 1-16.

[8] Ud Din, Israr, Imran Siddiqi, Shehzad Khalid, and Tahir Azam. "Segmentation-free optical character recognition for printed Urdu text." *EURASIP Journal on Image and Video Processing* 2017 (2017): 1-18.

[9] Ali, Toflk, Tauseef Ahmad, and Mohd Imran. "UOCR: A ligature based approach for an Urdu OCR system." In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pp. 388-394. IEEE, 2016.

[10] Uddin, Israr, Imran Siddiqi, and Shehzad Khalid. "A holistic approach for recognition of complete Urdu ligatures using hidden Markov models." In *2017 International Conference on Frontiers of Information Technology (FIT)*, pp. 155-160. IEEE, 2017.

[11] CLE (2015) Urdu digest POS tagged corpus. (*http://www.cle.org.pk/software/localization.html)*

[12] Pal, U., and Anirban Sarkar. "Recognition of printed Urdu script." In *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.*, vol. 3, pp. 1183-1183. IEEE Computer Society, 2003.

[13] Hussain, Sarmad, Salman Ali, and Qurat ul Ain Akram. "Nastalique segmentation-based approach for Urdu OCR." *International Journal on Document Analysis and Recognition (IJDAR)* 18, no. 4 (2015): 357-374.

[14] Ul-Hasan, Adnan, Saad Bin Ahmed, Faisal Rashid, Faisal Shafait, and Thomas M. Breuel. "Offline printed Urdu Nastaleeq script recognition with bidirectional LSTM networks." In *2013 12th international conference on document analysis and recognition*, pp. 1061-1065. IEEE, 2013.

[15] Nashwan, Farhan MA, Mohsen AA Rashwan, Hassanin M. Al-Barhamtoshy, Sherif M. Abdou, and Abdullah M. Moussa. "A holistic technique for an Arabic OCR system." *Journal of Imaging* 4, no. 1 (2017): 6.

[16] Yamina, Ouled Jaafri, Mamouni El Mamoun, and Sadouni Kaddour. "Printed Arabic optical character recognition using support vector machine." In *2017 International Conference on Mathematics and Information Technology (ICMIT)*, pp. 134-140. IEEE, 2017.

[17] Osman, Hussein, Karim Zaghw, Mostafa Hazem, and Seifeldin Elsehely. "An efficient language-independent multi-font OCR for Arabic script." *arXiv preprint arXiv:2009.09115* (2020).

[18] Darwish, Saad Mohamed, and Khaled Osama Elzoghaly. "An enhanced offline printed Arabic OCR model based on bio-inspired fuzzy classifier." *IEEE Access* 8 (2020): 117770-117781.

[19] Khattak, Israr Uddin, Imran Siddiqi, Shehzad Khalid, and Chawki Djeddi. "Recognition of Urdu ligatures-a holistic approach." In *2015 13th International conference on document analysis and recognition (ICDAR)*, pp. 71-75. IEEE, 2015

[20] Husnain, Mujtaba, Malik Muhammad Saad Missen, Shahzad Mumtaz, Muhammad Zeeshan Jhanidr, Mickaël Coustaty, Muhammad Muzzamil Luqman, Jean-Marc Ogier, and Gyu Sang Choi. "Recognition of Urdu handwritten characters using convolutional neural network." *Applied Sciences* 9, no. 13 (2019): 2758.

[21] Javed, Sobia T., Sarmad Hussain, Ameera Maqbool, Samia Asloob, Sehrish Jamil, and Huma Moin. "Segmentation free nastalique urdu ocr." *World Academy of Science, Engineering and Technology* 46 (2010): 456-461.

[22] Sagheer, Malik Waqas, Chun Lei He, Nicola Nobile, and Ching Y. Suen. "Holistic Urdu handwritten word recognition using support vector machine." In *2010 20th international conference on pattern recognition*, pp. 1900-1903. IEEE, 2010.

[23] Jang News Paper *https://jang.com.pk/*

[24] Javed, Sobia Tariq, and Sarmad Hussain. "Segmentation based urdu nastalique ocr." In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications: 18th Iberoamerican Congress, CIARP 2013, Havana, Cuba, November 20-23, 2013, Proceedings, Part II 18*, pp. 41-49. Springer Berlin Heidelberg, 2013.

[25] Hussain, Sarmad, Aneeta Niazi, Umair Anjum, and Faheem Irfan. "Adapting Tesseract for complex scripts: an example for Urdu Nastalique." In *2014 11th IAPR International Workshop on Document Analysis Systems*, pp. 191-195. IEEE, 2014.

[26] Qurat-ul-Ain Akram, Sarmad Hussain, Farah Adeeba, Shafiq ur Rehman, and Mehreen Seed. " Framework for urdu nastalique optical character recognition", 2014.