IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received June 19, 2019, accepted August 4, 2019, date of publication August 19, 2019, date of current version September 9, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2936363

A Scale and Rotation Invariant Urdu Nastalique Ligature Recognition Using Cascade Forward Backpropagation Neural Network

KHAWAJA UBAID UR REHMAN AND YASER DAANIAL KHAN
Department of Computer Science, University of Management and Technology, Lahore 54770, Pakistan
Corresponding author: Khawaja Ubaid Ur Rehman (ubaid.rehman@umt.edu.pk)

ABSTRACT In the emerging age of technologies, machines are becoming more and more skilled and capable just like humans. Despite the fact that machines do not have their own intelligence, but still due to advancement in Artificial Intelligence (AI), machines are rapidly advancing. The area of Pattern Recognition (PR) deals with bringing enhancements to identify obscure patterns corresponding to specific classes. Optical Character Recognition (OCR) is a subfield of PR which deals with the recognition of characters. A great work has been done for Japanese, Hindi, Arabic and Chinese scripts, but only a diminutive work has been done for Urdu script. The Urdu language is highly cursive and is written in different calligraphic styles like Naskh, Nastalique, Kofi, Devani and Riqa. The Nastalique font is very calligraphic with aesthetic beauty. The ligature segmentation of Urdu Nastalique is also more difficult as compared to other languages. Urdu Nastalique has some characteristics like stacking of ligatures and cursiveness which makes its ligature segmentation a difficult task. Cursiveness means ligatures are joined together to form a new shape. It contains connected ligatures which makes it more complicated as compared to other languages. The ligature recognition of Urdu text by an OCR is a strenuous task due to variants of scaling, rotation, orientation and font style. In this study, a scale and rotation invariant classifier for Urdu Nastalique OCR is proposed. A combination of scale and location invariant moments is used for feature extraction and the classification is performed using Cascade Forward Backpropagation Neural Network. The model is validated through independent dataset testing and 5-fold cross-validation which gave 96.474% and 96.922% accuracy. The results depict the adaptability of the proposed model due to its high accuracy for recognition of Urdu Nastalique Ligature.

INDEX TERMS Deep neural network (DNN), optical character recognition (OCR), scale invariant classifier, rotation invariant classifier.

I. INTRODUCTION

Urdu is the official dialect of Pakistan. This language is spoken in many different nations around the globe. The word ''Urdu'' is originated from the Turkish word ''order'' meaning camp or army [1]. The Urdu language is a combination of different languages. Sometimes Urdu is linked with Hindi. Urdu and Hindi share the same background. In Bangladesh, Urdu is used as a medium of communication and is referred to as Behari. Urdu has the influence of Arabic, Persian and Turkish languages. It includes words of many other languages which enhance its appeal as poetic language.

The associate editor coordinating the review of this article and approving it for publication was Naveed Akhtar.

The Urdu language is predominantly cursive as it contains connected ligatures which makes ligature segmentation more complicated as compared to other languages. Urdu can be written in different scripts. Some of the Urdu scripts are Naskh, Nastalique, Kofi, Thuluth, Devani, and Riqa [2]. Commonly used Urdu script fonts are Nastalique and Naskh which are illustrated in Fig. 1 and Fig. 2 respectively for comparison.

Among Nastalique and Naskh fonts, Nastalique font style is more complex due to the joining pattern of characters. The nastalique textual style is a blend of two unmistakable textual styles Naskh and Taleeq. It was firstly made by eminent author Mir Ali Tabrezi in the 14th century [2]. Nastalique was widely used during the Mughal Empires who were actually

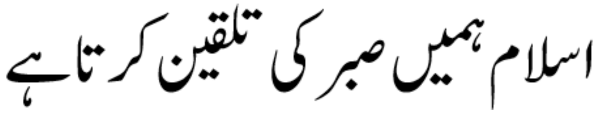

FIGURE 1. Nastalique font style.

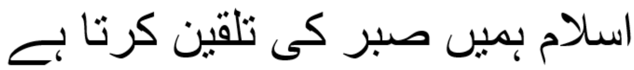

FIGURE 2. Naskh font style.

Persians. In 1980, Mirza Ahmed Jameel created a computerized Nastalique font based on 20,000 Monotype ligatures which were used in computers for printing [2]. Mirza Ahmed Jameel named it as Noori Nastalique. This was supposed as a rebellion in the Urdu printing industry.

A. PROBLEMS OF URDU NASTALIQUE

In comparison with other languages like English, the optical character recognition of printed text in Urdu poses several challenges. Although the calligraphic style of this font enhances its beauty, it also renders it to be a very cursive script. Segmentation of individual ligatures in such cursive languages is very intricate. The context-sensitive nature of Nastalique writing style has a variable pitch and vertical ascent, whereas writing style of English language has a fixed vertical ascent as shown in Fig. 3 (b). Nastalique is written slantwise from top right to bottom left, while numerals are written from left to right. All the ligatures of Nastalique font style are slanted at some angle. Due to the context-sensitive nature, it forces letters to form different shapes. Line and character segmentation of Nastalique writing style consume less space in contrast with Naskh font style. There is an overlapping present of کر with ہر, پنے with شخص and کہہ with ں as shown in Fig. 3 (a). The other complex issues of Nastalique writing style are intra-ligature overlapping, diagonalization and intra ligature merging are shown in Fig. 4.

B. COMPOSITION OF LIGATURES

The recognition of baseline for Nastalique content is a yet unsolved issue because each word is composed of more than one Urdu alphabet. As illustrated in Fig. 4(a) each word is composed of more than one alphabet.

C. OVERLAPPING NATURE OF URDU NASTALIQUE SCRIPT

The overlapping nature of Urdu Nastalique script leads to various ambiguities. The connected/overlapped nature of Urdu Nastalique script makes its segmentation tough as compared to Naskh script as illustrated in Fig. 4(b), 4(c) respectively. Vertical overlapping exists between ligatures although they do not touch each other. There are two categories of overlapping (1) Inter-ligature overlapping and (2) Intra-ligature overlapping [31], [43]. Inter-ligature overlapping is the overlapping present between two different ligatures as shown in Fig. 4(e). On the contrary, intra-ligature overlapping is the overlapping present within a ligature as shown in Fig. 4(d).

D. TOUCHING OF CHARACTERS WITH BASELINE

There are few ligatures in Urdu Nastalique script which touch with baseline. In the case of ligatures like حا, خا, چا, جا the starting character touches with the baseline which causes false loops. This is a challenge in Urdu Nastalique OCR. False loops as shown in Fig. 4(f) makes it convoluted to distinguish them from real loops.

E. COMPLETELY FILLED LOOP

In Urdu Nastalique there are some characters like Fay, Wao Qaf and Mim whose starting shape is filled as shown

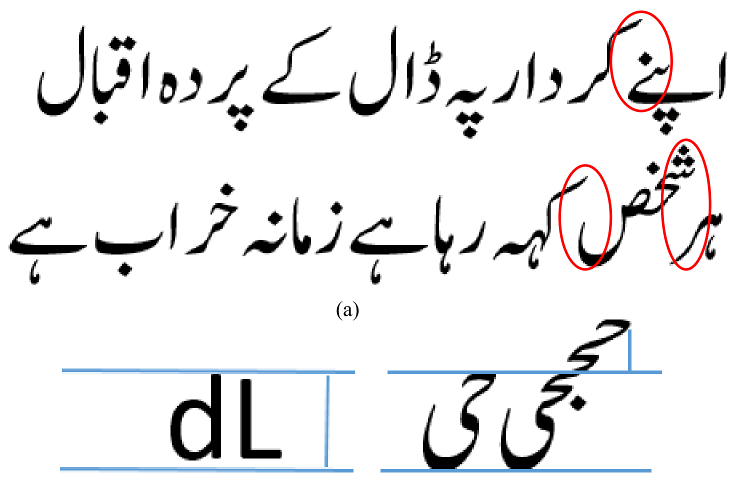

FIGURE 3. (a) Overlapping of Nastalique Ligatures. (b) The fixed vertical ascent of English characters and variable ascent and pitch of Nastalique ligatures.

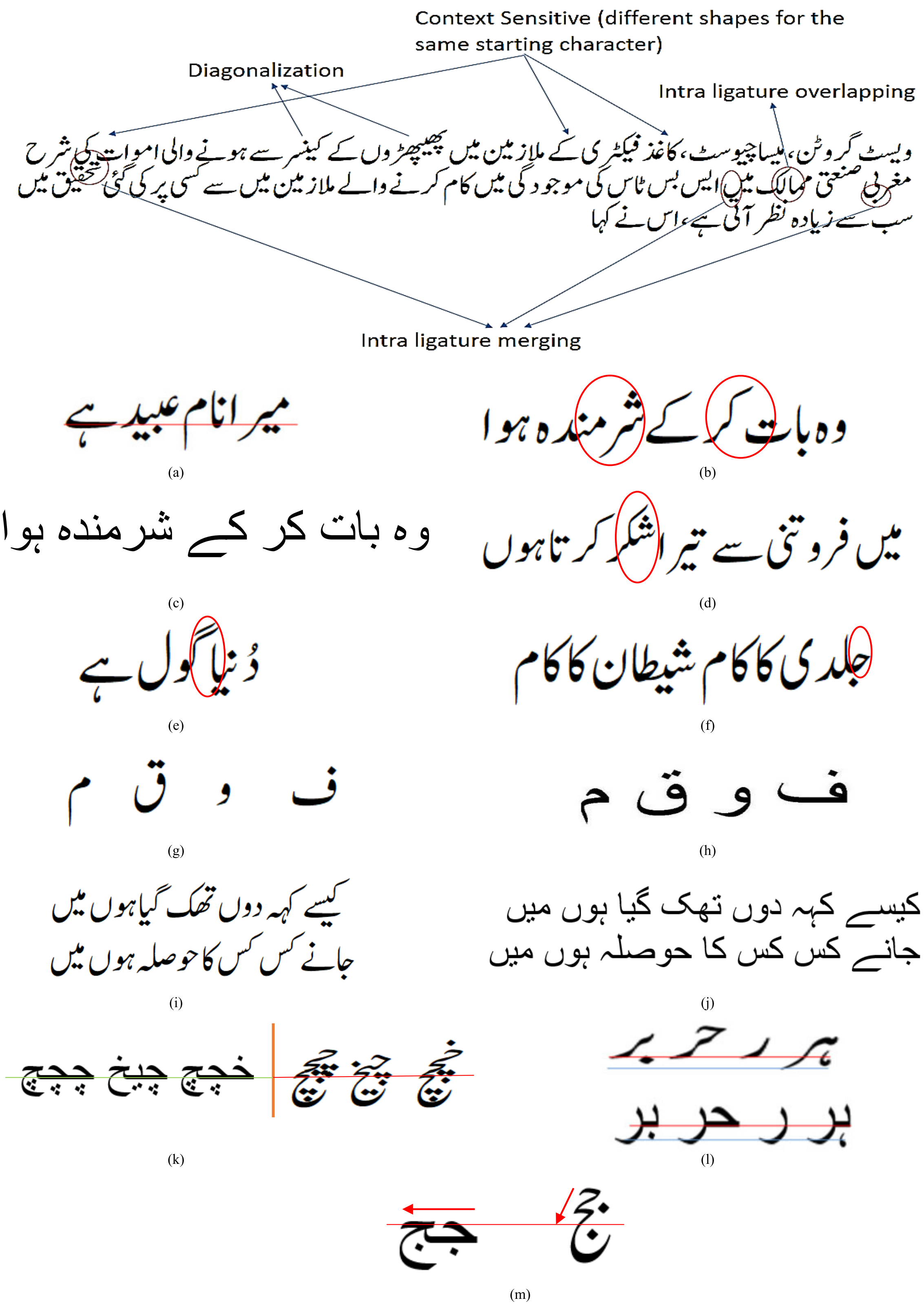

FIGURE 4. Complexities of Nastalique writing style. (a) Each word is composed of more than two alphabets. (b) Overlapping nature of Urdu Nastalique script. (c) Non-overlapping nature of Urdu Naskh script. (d) Intra-ligature overlapping. (e) Inter-ligature overlapping. (f) Jeem is shown in red colour touch with the baseline leading to false loop. (g) Urdu Nastalique ligatures Fay, Wao, Qaf and Mim, which cause filled loop. (h) Urdu Naskh ligatures Fay, Wao, Qaf and Mim whose starting character does not cause filled loops. (i) Urdu Nastalique script requires less space. (j) Urdu Naskh script requires more space. (k) Urdu Nastalique and Naskh dots placement. (l) Challenges in the baseline of Urdu scripts are shown with a blue line. Redline is the baseline of Urdu Nastalique and Naskh script [44]. (m) Diagonal nature of Urdu Nastalique script.

in Fig. 4(g). In Naskh, these shapes are open as shown in Fig. 4(h). The ligature recognition of Urdu Nastalique becomes intricate which causes ligatures to become identical with other ligatures due to filled loops [43].

F. SPACING

In English sentences, there are spaces after a word or a character. Just like in English language spaces may occur in the Urdu language after a word or a ligature. These spaces differ in size. Among Urdu scripts, Urdu Nastalique requires less space whereas Naskh requires more space as illustrated in Fig. 4(i) and Fig. 4(j) respectively.

As Naskh script occupies more space, therefore, Nastalique is used as a source of the medium in the documentation, magazines, newspaper etc. Also, line and ligature segmentation of this script consume less space.

G. DIACRITICS AND DOTS PLACEMENT

Some other issues of Urdu script are diacritics. The accuracy and performance of Urdu OCR applications are significantly affected by diacritics and dots placement. The dots above and below the baseline affect the processing. In Naskh script single dot, two dots and three dots are along a horizontal baseline as shown in green colour in Fig. 4(k) whereas in Urdu Nastalique single dot, two dots and three dots are along a virtual baseline as shown in red colour in Fig. 4(k).

H. BASELINE DETECTION

The baseline detection in Urdu OCR applications is one of the crucial tasks. Among Urdu scripts, the baseline detection of Naskh script is reasonably easy in comparison to Urdu Nastalique script. The baseline plays a pivotal role in writing a script and it is used in skew detection. The baseline of Naskh script is along a horizontal line, whereas the baseline of Urdu Nastalique is along a virtual line as shown in Fig. 4(l) [44].

I. DIAGONAL NATURE OF URDU NASTALIQUE

Another main issue of Urdu Nastalique is diagonal nature of Nastalique. Urdu Nastalique is written slantwise from top right to the left titled at some angle and the angle along with ascent is variable depending upon the ligatures [2], [45]. Due to the diagonal nature of Nastalique script, it makes ligature segmentation even more difficult as compared to Naskh.

J. IMPORTANCE OF THE PROBLEM

OCR systems for Chinese, Japanese, English and Arabic languages have been comprehensively developed and are now commercially available whereas quite less work has accomplished for Urdu OCR systems [42]. Owing to the importance of Urdu language and the need for integrating Urdu documents into several crucial applications, there is an inherent need of Urdu Optical Character Recognition systems. Significant improvements have been made in developing OCR software throughout the years. This is due to huge advances in machine learning algorithms. The goal of character segmentation is to emulate the human perusing capacity, with the human exactness yet at a much higher speed. The objective execution is in any event five characters every second with a 99.9% acknowledgement rate. The OCR is an essential tool for different applications, for example, report computerization, check verification, data entry applications, advanced scanning, perusing machine for outwardly impaired, and an extensive assortment of numerous other banking and business applications. The OCR is a functioning territory of research and its significance has entrenched standard compatibility in terms of Digital Image Processing, Pattern Recognition, Artificial Intelligence, Database Systems, Natural Language Processing, Human-machine interaction, and communications. All these software applications can be performed adequately if the characters from text images are categorized and perceived explicitly [41].

K. HOW DO RESEARCHERS DEAL WITH THE ISSUE

The author in [16] trained on a feed-forward back propagation neural network. The model trained on a pre-characterized group of ligatures from Urdu Noori Nasta'liq script. In spite of possessing great accuracy, however when it was tested on 200 character ligatures, the system lacks performance on some unidentified ligatures. For the essential printed disconnected/single characters or letters, Multi-Layer Perceptron (MLP) was used [3]. This classifier involves three layers, i.e. the input layer (150 neurons), the hidden layer (250 neurons), and the output layer (16 neurons). Besides, the OCR works on a contribution to the type of a binary image of size 10x15 pixels or 150 neurons. The 250 neurons at the hidden layer are settled on an experimentation premise, whereas the total number of neurons in the output layer are ascribed to the 16-bit Unicode. The constraint of the system is that it can't perceive joined, associated or compound characters of Urdu content. The segmented characters are utilized to prepare a three-layered multilayer feed forward neural system (FFNN) for the classification and recognition in [46], by utilizing an Ariel text style of size 36. In any case, however the system actually does not recognize diacritics. We can presume that it is done to diminish the error rate as the diacritics characters are responsible for huge recurrence of errors at the closure character of a word. The word recognition and classification are done, in a comparative work [46]. The mentioned segmented Urdu printed characters were given as input to feed forward neural network classifier without lexicon. The authors guarantee a 93.4% accuracy, however, don't give any strong verification of strategies for proof. In addition, they presume the input text is diacritics free and of the fixed font size.

The FFNN was utilized for the classification of extracted features from 53 classes [47]. In their strategy, the preeminent training performance of FFNN occurs after 757 epochs. The impediment of the system is that it doesn't utilize the majority of the character set of Urdu. The authors propose a methodology for isolated Urdu characters, without the utilization of any statistical model. This work depends on what the authors call as a soft converter, which perceives a character from the database and its width, height and "X" value (count of the black pixels). The authors assure that recognition rates are up to 100% and 97.3% for the hard matching and soft matching, respectively [48].

Another system, that passes the extracted features as an input for training to Classification and Regression Tree (CART). In the CART, the decision the tree node has to take is on the presence and absence of a specific component. In spite of the fact that the strategy performs well on separated characters and numerals, it endures on account of compound characters, and different sizes and textual styles [17].

CONTRIBUTION

The proposed line segmentation algorithm meticulously extracts each line. Further, the algorithm also accurately extracts each ligature. Subsequently, the proposed model assiduously recognizes each ligature, diacritic and numerals. The work done before for such cursive scripts seems to be limited to scale invariant or font invariant classifier, but this study has also contributed to rotation invariant classifier for Urdu Nastalique font. Urdu Nastalique is a cursive language and by joining different characters it forms a new shape so it contains hundreds of classes which leads to the complex design of output layer. The training of such a large data set with a soaring number of classes will exceed the number of resources. Therefore, to simplify the problem, this study has proposed a hierarchical approach to split the data set and to reduce the training time and optimize the accuracy of the neural network. The proposed ligature based recognition systems use Cascade Forward Back-propagation Neural Network yielding an overall accuracy of 96.474%. Last, but not the least various experiments were also performed for calculating the accuracy of the model. The benchmarks used for calculating accuracy were Sensitivity, Specificity, ACC and MCC which are illustrated using Confusion Matrix, ROC.

II. RELATED WORK

Work of several researchers' exhibit remarkable benefaction for cursive scripts like Arabic, Farsi and Urdu, etc. There subsist two methodologies for OCR systems first one is segmentation based and another one is segmentation free. There are also some methodologies which work on isolated characters [3]–[6] and other works on intact ligature [7]–[11]. The proposed methodologies which work on isolated characters work much better than words or ligatures. The segmentation of comprehensive Urdu text is itself a tough task due to several challenges discussed earlier.

The baseline detection of Urdu Naskh is relatively easy whereas the baseline detection of Urdu Nastalique is tough. Numerous approaches have been developed to search the baseline of Urdu Nastalique like Horizontal Projection Profile [12], but due to diacritics, the accuracy has decreased. There are different approaches to segmentation. One is line segmentation which divides a paragraph into lines and another one is ligature segmentation which divides a complete word into ligatures. The accuracy to classify characters using ligature segmentation for the Urdu Nastalique font is 99% [13]. But this method is costly in terms of finding the optimal character from a line.

Shah [14] built up a ligature based OCR for Nastalique Urdu. Features were extricated from the isolated characters and Hidden Markov Model was used for training purpose. Around 1499 plus ligatures were trained and the proposed model reported accuracy about 92%. The flaw in this proposed methodology was the use of shape reliance for classification of ligatures which reduced it to be a scale variant system. Due to this, every ligature has to be trained with high precision which is a monotonous task.

For the Mongolian language, a multi-font recognition technique was suggested by Peng et al. [15]. The suggested technique used an analytical method that utilized the properties of projection profiles and linked its elements for determining segmentation points. The authors claim that the accuracy is 96.9 on real-world data with multiple fonts, but the major reason for recognition errors is due to segmentation. On the contrary, a segmentation free approach is independent of segmentation. In this approach, interlinked components are selected for feature extraction. Multi-tier holistic methodology for Urdu Nastalique recognition [16] and printed Urdu Nastalique recognition using water reservoir principal [17] are based on a holistic approach. The limitation of the segmentation free methodology is that they are affected by the variation of scaling and orientation. The summary of some notable contributions in the field of OCR is given in Table. 1.

An OCR involves a number of steps which are Image Attainment, Preliminary-Processing, Segmentation, Feature Extraction, Classification and Post Processing. In this study, a methodology is proposed to classify Urdu text written in Nastalique font. The Urdu language consists of 39 basic characters and 10 numerals as given in Table 2.

III. MATERIALS AND METHODS

The feature-based approach is devised to predict for classification of ligatures. Initially, images are acquired as input and preprocessing is performed. Further, an image is segmented into different lines and each line is subsequently split into ligatures. After line segmentation and ligature segmentation, features are extracted for each unit. Consequently, features and labels are clamped to a deep neural network for training. Eventually, when the classifier is sufficiently trained it is used to rigorously test the accuracy of the model. The class of a ligature is predicted on the basis of trained neural network results. The linear flow diagrams of the training and testing computational model are shown in Fig. 5.

A. DATA SET COLLECTION

The performance of most of the classification models is dependent on the data used for training. The data must be reliable, robust and meticulously labelled. In order to form such data, a systematical approach has been designed and moreover, it should be verifiable. The data set was collected from a well-known source namely Center for Language

TABLE 1. Summary of notable studies on OCR in different languages.

Authors	Language	Methods	Classification	Dataset	Accuracy
Ghods and Kabir [52]	Farsi	Structural (direction, angles, number of strokes)	ID3	TMU database 4000	94-99%
Ghods et.al [19]	Farsi	Structural	HMM with lexicon reduction	1000 (1200 groups)	Top1: 85.2% Top10: 96.7%
Ghods et.al. [32]	Farsi	Fusion of Statistical and Structural features	HMM	1000 Online ligature samples	Testing: 87.5% Training: 92.9%
AlKhateeb et al. [50]	Arabic	Sliding window and structural	HMM and re-ranking	26459 and 32492 IFN/ENIT database	83.55%
Xiang et al. [49]	Arabic	Densities of foreground, concavity and arc length	HMM	26459	97.99%
Awaida and Khorsheed [51]	Arabic	Run-length encoding (RLE)	HMM	APTI database 94, 418	96.65%
Acharya et.al [53]	Devanagari	Structural	CNN	DHCD 92000	98.47%
Ul-Hasan et.al [54]	English-German	Not given	LSTM	90,000 text line images in variety of fonts	98.186%
Simistira et.al [55]	Greek	Not given	LSTM	Polyton-DB 15,689 text lines	85.32-94.49%
Choudhary [18]	English	Binarization features	NN	1300	85.62%
Zhao [56]	Chinese	a sequence of feature vectors is extracted from the feature maps	LSTM	CMDD	96.8%
Obaidullah [57]	Persian, Bangla,Oriya, Kannada	Directional stroke identification (DSI) Structural and visual appearance (SVA) SVA + DSI	MLP	PBOK 707 text pages	93.77 95.54 98.60
N. Sabbour & F. Shafait [59]	Urdu and Arabic script	Structural features	NABOCR	10,000 UPTI database	UPTI Urdu:91% Arabic: 86%
Hassan, T et.al [61]	Bengali	statistical, structural moment-based features, global transform based features	KNN classifier	CMATER DB 3.1.1	96.7%
Sanjrani et. al [62]	Sindhi	Structural features	KNN, SVM	Own created database of Sindhi numerals	63%
Kaur et.al [63]	Gurmukhi	Hierarchical centroid method	SVM using nu-SVC type and RBF kernel	Own created database of Gurmukhi having 29 different font styles	97.87%
Khan et.al [64]	Sindhi	Zoning feature extractor	KNN and NN	Created own 4488 classes and 102 samples for each letter	KNN: 70.05% NN: 72%
Ahmed et. al [65]	English	MSER text detection method	Deep MDLSTM	EASTR-42K	94.1%
Agnihotri et. al [66]	Punjabi	Local and Global features	ANN	Different Punjabi fonts data set size not mentioned	87%

Engineering available at http://www.cle.org.pk. The corpus available on the website used for training and testing purposes contains Urdu Nastalique data having 591 document images, 13,712 number of lines, 386,648 total ligatures and 6,452 number of unique ligatures. The mentioned corpus contains Noori Nastalique font style having font size 14.

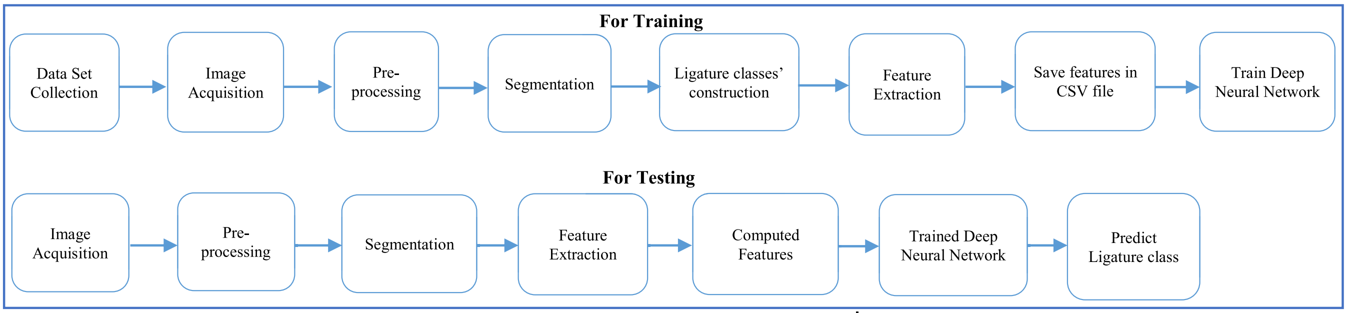

FIGURE 5. Linear flow diagram of Training and Testing Computational model.

TABLE 2. The character set of Urdu script.

خ	ح	چ	ج	ث	ٹ	ت	پ	ب	ا
ص	ش	س	ژ	ز	ڑ	ر	ذ	ڈ	د
ل	گ	ک	ق	ف	غ	ع	ظ	ط	ض
	ے	ی	ء	ھ	ہ	و	ں	ن	م
۰	۱	۲	۳	۴	۵	۶	۷	۸	۹

The Noori Nastalique font has been used because it is more cursive. Due to the context-sensitive nature of Nastalique, it is very challenging to perform segmentation of such script.

B. IMAGE ACQUISITION

In this study, the corpus is obtained from the Center of Language Engineering (CLE) website.

C. PRE-PROCESSING

Pre-processing is considered one of the very important steps in pattern recognition. The transformation of the data set into a specific format is necessary to reduce factor that may impair or disturb the segmentation process. It mainly involves binarization, noise removal, smoothing, thinning or de-skewing of the images. During the pre-processing different procedures were applied on images such as median filtering, image thresholding and thinning. Thinning is a process of converting the ligature into glyph like structure. This glyph like structure is very crucial as it forms the basis of ligature recognition.

D. SEGMENTATION

Segmentation is a process in which an image is divided into paragraphs, paragraphs are split into lines, the lines are split into words and words are further split into characters. Other than this it also involves separating borders from images and texts from images etc. There are two levels of segmentation.

1) LINE SEGMENTATION

To distinguish between textual and background areas the complement of the image is formed. The area of the image where there are black pixels are transformed into white pixels and similarly, white pixels are transformed into black pixels. Subsequently, this complemented image is padded with zeros for computing border pixels. The padding size is a vector of non-negative integers which specify the amount of the padding to add and the dimension along which to add. The mean values of the padded image are also computed. The "mean" function of MATLAB returns the mean values of the elements along with their specified dimension. After computation of the mean values, the areas where the lines or background are present are also identified. If there are ones, then it is a line, otherwise, it is a background. The difference between the rising and falling edges (shown in Fig. 7) are computed to predict the starting row and the ending row of each line.

The top beginning row of each line is computed by finding the indices greater than zero and similarly the bottom row of each line is computed by finding the indices which are less than zero. The next line is found where the first white pixel is located. Starting from the top right row to the bottom left row all the lines are separated, and then again the complement of segmented lines are computed and finally, all the segmented lines are saved in line segmentation folder. The plot of Fig. 6 using this method is shown in Fig. 7. The segmented lines of Fig. 6 using this line segmentation algorithm are shown in Fig. 9 (a), (b), (c), (d), (e), (f), (g), (h) and (i) respectively.

Horizontal projection is the mean of the pixel values in each row of the input value. The plot in Fig. 7 shows that there are 10 peaks present so, the number of lines in the image (shown in Fig. 6) is ten.

2) LIGATURE SEGMENTATION

Ligature segmentation is the next step after line segmentation. If the segmented files are placed in a folder from which they

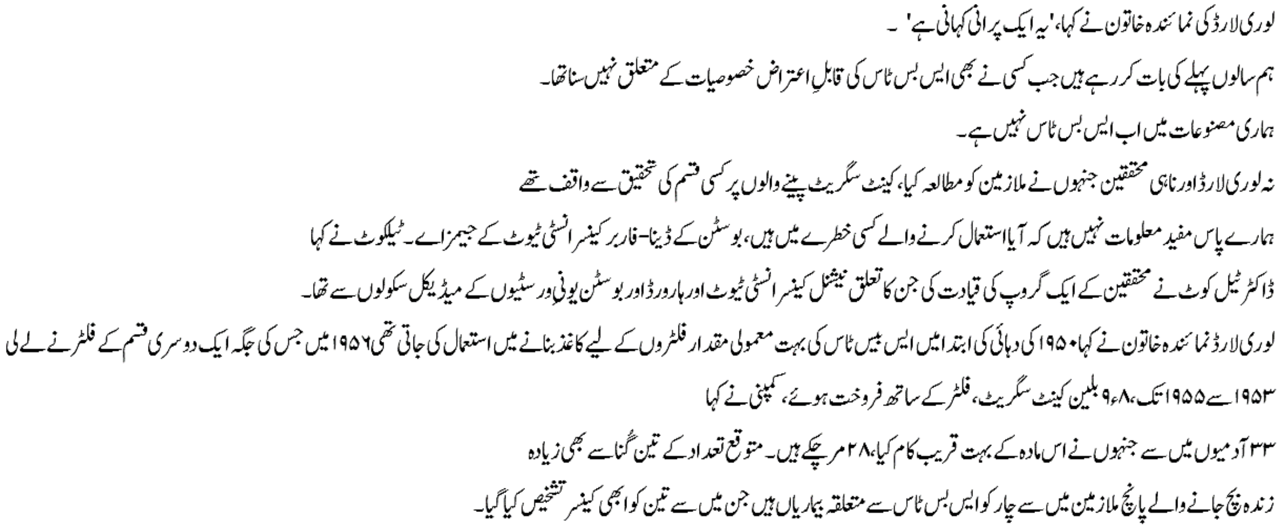

FIGURE 6. Urdu Nastalique image.

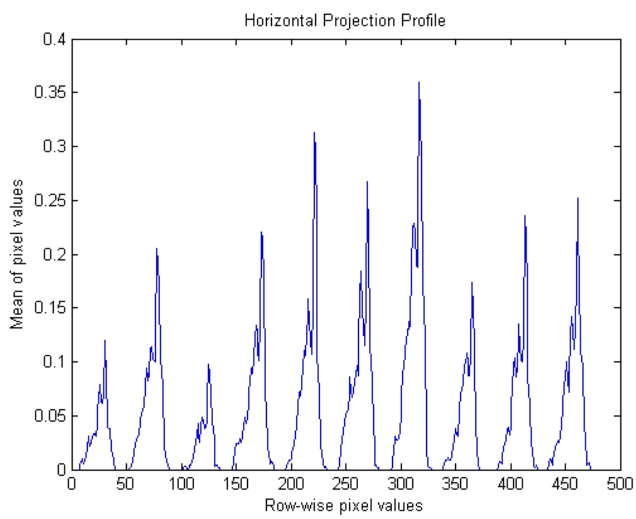

FIGURE 7. Plot using Horizontal Projection Profile showing there are 10 lines present.

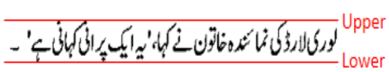

FIGURE 8. Upper and Lower ones.

are later retrieved, then there is a possibility that they are not retrieved in the order as they are placed in the text. Intuitively, this method will create problems, especially for reading Urdu text. This problem is solved by applying sorting on the file names in the ascending order. The image which contains some of the noisy edges in the mask is removed by using the morphological operations with a disk-shaped structuring element. Furthermore, the complement of the image is computed, i.e. all the white pixels are converted to black pixels and vice versa.

The ''mean'' function of MATLAB returns the mean values of the elements along with the specified dimension. After computation of the mean values where the ligatures or background are present are identified. If there are ones, then it is a ligature otherwise it is a background. The difference between the rising and falling edges are computed to predict the starting column and the ending column of each ligature. The top beginning column of each ligature is computed by finding the indices which are equal to 1 and similarly the bottom column of each ligature is computed by finding the indices which are equal to -1. For extracting ligatures from right to left direction, i.e. in Urdu style, the loop is executed in reverse order. Subsequently, the morphological operation ''thin'' is applied to the image 0.1 times. The connected components in the binary image are computed in the 8-connected surrounding neighbourhood. The ligatures which belong to the connected components are found and are extracted line wise in a ''Ligature Segmentation Line-wise'' folder. The line wise segmented ligatures of Fig. 10 are shown in Fig. 11. In this way ligatures of all other lines are segmented. The next step is to merge all the line wise segmented ligatures in the same folder. The files which are read from different folders are not in order so, all the pathnames and filenames are sorted in ascending order. Another issue present in the segmented ligatures is that it contains some extra spaces after ligature segmentation as hown in Fig. 11. In order to overcome this issue again, the connected components are computed in the 8-connected surrounding neighbourhood as shown in Fig. 13. The position of row and column is tracked by using MATLAB ''find'' function and then from the bottom row to extreme row & from bottom column to extreme column all ligatures are obtained. The complement of the image is calculated and at-last all the images are saved in the ligature segmentation final folder.

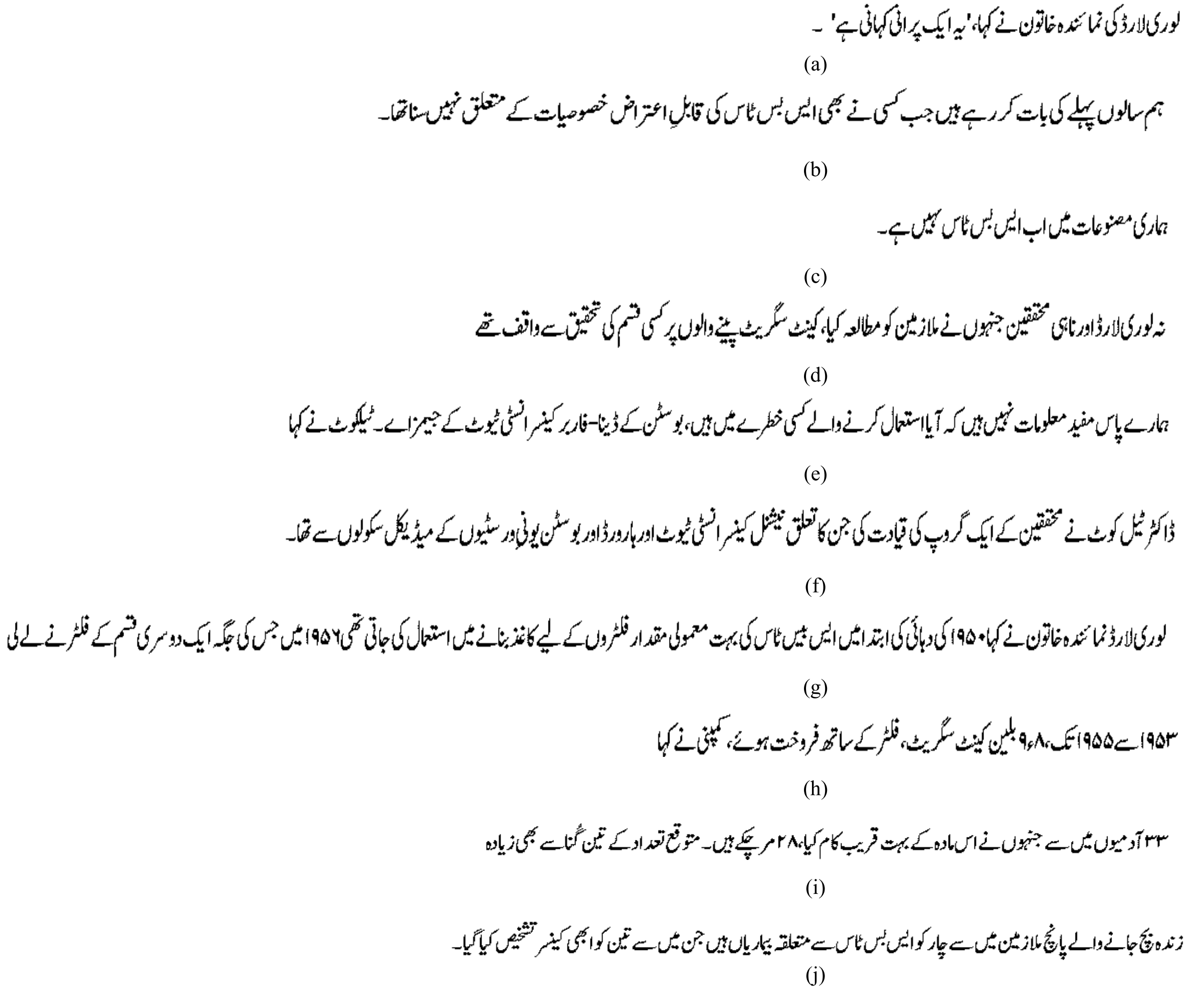

FIGURE 9. (a). 1st Line Segmentation. (b). 2nd Line Segmentation. (c). 3rd Line Segmentation. (d). 4th Line Segmentation. (e). 5th Line Segmentation. (f). 6th Line Segmentation. (g). 7th Line Segmentation. (h). 8th Line Segmentation. (i). 9th Line Segmentation. (j). 10th Line Segmentation.

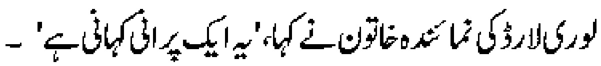

FIGURE 10. Line Segmentation of 1st line.

FIGURE 11. Ligature Segmentation Line-wise of 1st line.

FIGURE 12. Vertical regions of separation between ligatures and words.

Vertical projection is the mean of the pixel values in each column of the input value. The plot in Fig. 14 shows that there is a total of six peaks present, so the number of ligatures are six. As the number of ligatures in the word لوری and لارڈ are six. This plot also shows that there are six ligatures present.

E. LIGATURE CLASSES

The next step after ligature segmentation is the construction of ligature classes. Diacritics are removed from the main body of each ligature by extracting largest connected component as shown in Fig. 15. The ligature having the same shape irrespective of diacritics are placed in the same folder. For instance, با, ٹا, تا, پا, یا and ثا have the same shape as shown in Fig. 15.

FIGURE 13. Segmented ligatures after diacritics removal.

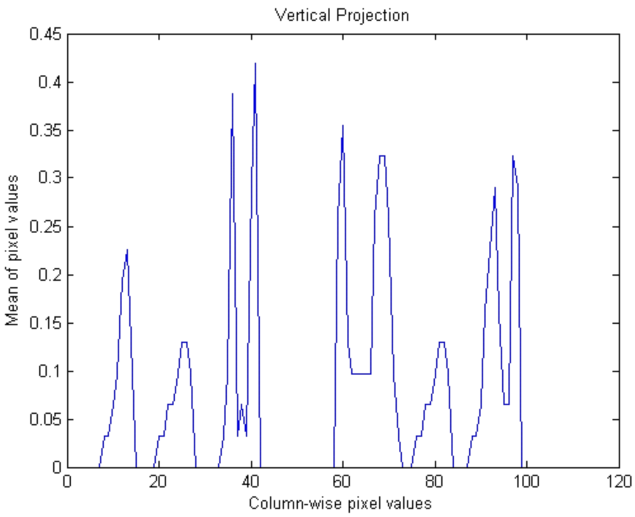

FIGURE 14. Plot of Vertical Projection using Mean shows there are six ligatures present.

FIGURE 15. The same shape of ba, taa, tay, paa, ya and na.

Depending upon the ligature shape, all the ligatures are placed in a folder one by one. A ligature having the same shape is placed in the same folder and a folder is given a class name. In this way, about 2,496 unique ligature classes were constructed. This corpus contains high-frequency ligatures. Each ligature class contains 15 samples and there are about 37,440 ligatures samples constructed. Although the corpus obtained from CLE website contains 6,452 unique ligature classes, however in the proposed work 2,496 frequently used unique ligatures classes have been used.

There are in excess of 26,000 remarkable ligatures in Urdu and a research on the occurrences of these ligatures specify that the majority of these ligatures occur very rarely. More than 99% of whole Urdu corpus can be comprised by utilizing only 10% (around 2600) of all Urdu ligatures [60].

F. FEATURE EXTRACTION

After construction of ligature classes, the next step is the extraction of crucial information from each ligature instrumental in distinguishing one ligature from all other ligatures. This step is called feature extraction. This step plays a pivotal role in the majority of the optical ligature recognition frameworks. It is generally classified into two categories, structural and statistical features. In this study, features are extracted on the basis of statistical features. Statistical features are numerical measures of geometrical regions within an image, for example, moments, zoning and crossing. The image moments [36] describes a ligature and differentiate it from other ligatures after segmentation. The image moments are the weighted moments of the image pixels. For the image moments, firstly the complement is computed and then the features are extracted. After calculation of features, data is saved in Comma Separated Values (CSV) files along with its associated annotation. The features which are extracted from each image includes raw [38], [39], central [38], [39], scale-invariant, Hu [38], [39] and Zernike moments up to order of 3. These features are briefly explained below:

1) RAW MOMENTS

Raw moments are the moments about the origin. For a 2D continuous function $T(\mathfrak{g}, \mathfrak{h})$ the raw moment of order $r+s$ are as follow:

$$R_{rs} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mathfrak{g}^r \mathfrak{h}^s T\left(\mathfrak{g}, \mathfrak{h}\right) d\mathfrak{g}\, d\mathfrak{h} \tag{1}$$

where, r, s = 0, 1, 2, 3 . . .

For a binary image of scale $A \times B$ with pixel intensities $T\left(\mathfrak{g}, \mathfrak{h}\right)$. The raw moment R_{rs} are calculated as:

$$R_{rs} = \sum_{\mathfrak{g}=0}^{A-1}\sum_{\mathfrak{h}=0}^{B-1} \mathfrak{g}^r \mathfrak{h}^s T\left(\mathfrak{g}, \mathfrak{h}\right) \tag{2}$$

where, r, s = 0, 1, 2, 3 . . .

A and B are the number of rows and columns of an image respectively, $T\left(\mathfrak{g}, \mathfrak{h}\right)$ is the intensity of the image at point $(\mathfrak{g}, \mathfrak{h})$.

2) CENTRAL MOMENTS

Central moments are the moments about the mean. For a 2D continuous function $T(\mathfrak{g}, \mathfrak{h})$ the central moment of order $r+s$ is as follows:

$$\mu_{rs} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \left(\mathfrak{g}-\bar{\mathfrak{g}}\right)^r \left(\mathfrak{h}-\bar{\mathfrak{h}}\right)^s T\left(\mathfrak{g}, \mathfrak{h}\right) d\mathfrak{g}\, d\mathfrak{h} \tag{3}$$

where, $\bar{\mathfrak{g}} = \frac{R_{10}}{R_{00}}$, $\bar{\mathfrak{h}} = \frac{R_{01}}{R_{00}}$ are the centroids

For a binary image of scale $A \times B$ with pixel intensities $T\left(\mathfrak{g}, \mathfrak{h}\right)$ the central moment μ_{rs} is calculated by:

$$\mu_{rs} = \sum_{\mathfrak{g}=0}^{A-1}\sum_{\mathfrak{h}=0}^{B-1} \left(\mathfrak{g}-\bar{\mathfrak{g}}\right)^r \left(\mathfrak{h}-\bar{\mathfrak{h}}\right)^s T\left(\mathfrak{g}, \mathfrak{h}\right) \tag{4}$$

Central moments of an image up to the order of 3 are simplified as follows:

$$
\begin{aligned}
\mu_{00} &= R_{00} \\
\mu_{01} &= 0 \\
\mu_{10} &= 0 \\
\mu_{11} &= R_{11} - \bar{g}R_{01} = R_{11} - \bar{h}R_{10} \\
\mu_{20} &= R_{20} - \bar{g}R_{10} \\
\mu_{02} &= R_{02} - \bar{h}R_{01} \\
\mu_{21} &= R_{21} - 2\bar{g}R_{11} - \bar{h}R_{20} + 2\bar{g}^2R_{01} \\
\mu_{12} &= R_{12} - 2\bar{h}R_{11} - \bar{g}R_{02} + 2\bar{h}^2R_{10} \\
\mu_{30} &= R_{30} - 3\bar{g}R_{20} + 2\bar{g}^2R_{10} \\
\mu_{03} &= R_{03} - 3\bar{h}R_{02} + 2\bar{h}^2R_{01}
\end{aligned} \tag{5}
$$

3) SCALE INVARIANT MOMENTS

Scale-invariant moments are derived from the central moments. These moments are computed by using the 0^{th} central moment i.e. μ_{00} as quotient. These moments are translation invariant [20].

$$\eta_{ij} = \frac{\mu_{ij}}{\mu_{00}^{\left(1+\frac{i+j}{2}\right)}} \tag{6}$$

where $i + j \geq 2$

The scale-invariant moments, of the order up to 3, are yielded using following simplified equations:

$$
\begin{aligned}
\eta_{11} &= \frac{\mu_{11}}{\mu_{00}^2} \\
\eta_{12} &= \frac{\mu_{12}}{\mu_{00}^{5/2}} \\
\eta_{02} &= \frac{\mu_{02}}{\mu_{00}^2} \\
\eta_{20} &= \frac{\mu_{20}}{\mu_{00}^2} \\
\eta_{03} &= \frac{\mu_{03}}{\mu_{00}^{5/2}} \\
\eta_{30} &= \frac{\mu_{30}}{\mu_{00}^{5/2}}
\end{aligned} \tag{7}
$$

4) HU MOMENTS

Moment invariants have become popular amid the most recent years. They were presented for pattern recognition problems by Hu [21], who utilized the consequences of the hypothesis of mathematical invariants [22] & [23] and determined his well-known invariants to elucidate problems in 2-D. Hu moments are rotation invariant. They are constructed using scale-invariant moments. Hu moments up to the order of 3 are as follows:

$$
\begin{aligned}
I_1 &= \eta_{20} + \eta_{02} \\
I_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
I_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2
\end{aligned} \tag{8}
$$

5) ZERNIKE MOMENTS

a: CONVERSION OF CARTESIAN COORDINATES TO POLAR COORDINATES

Zernike moments are computed for a circular geometric structure since they exhibit rotation invariance. A rectangular two-dimensional image has to be converted into a circular disk-like canvas for this purpose. Cartesian coordinates of an image are transformed into Polar coordinates for this purpose by finding r and θ as shown in Fig. 16. To do the conversion, scan across the output image and treat $\mathcal{X}$ and $\mathcal{Y}$ as if they were r and θ then use them as r and θ to look up the corresponding pixel in the input image using the following formulas:

$$
\begin{aligned}
r &= \sqrt{\mathcal{X}^2 + \mathcal{Y}^2} \\
\theta &= \tan^{-1}\frac{\mathcal{Y}}{\mathcal{X}} \\
cos\theta &= \frac{x}{r} \Rightarrow x = rcos\theta \\
sin\theta &= \frac{y}{r} \Rightarrow y = rsin\theta
\end{aligned} \tag{9}
$$

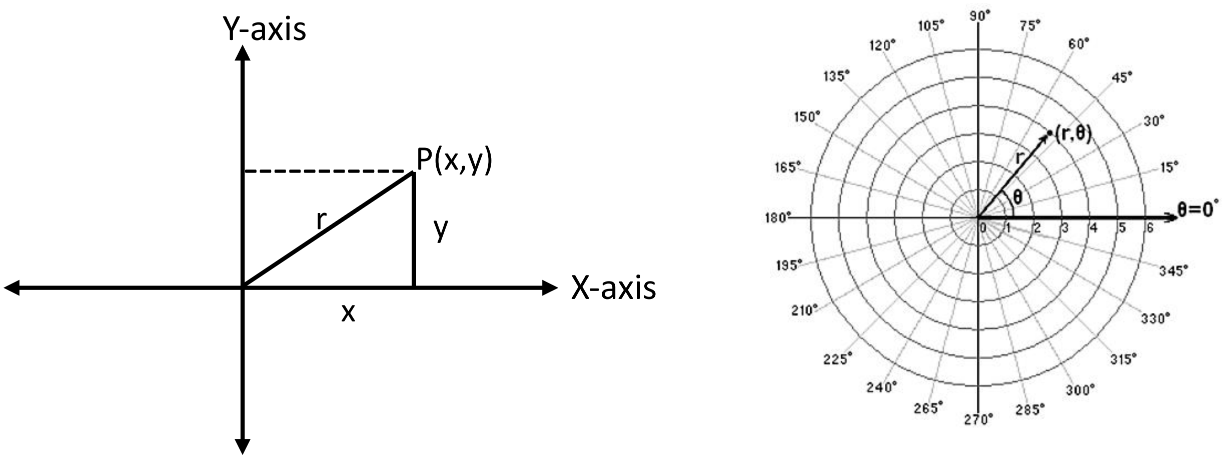

FIGURE 16. The transformation from Cartesian to Polar coordinates.

In this way, all the Cartesian coordinates are converted into Polar coordinates. While performing an image transformation and manipulation techniques, it is frequently essential to apply some kind of interpolation or filtering technique in order to attain a good quality image. To construct an image having polar coordinates bilinear interpolation technique was used.

Bilinear Interpolation is a technique in which we find unknown pixel by using the nearest neighbour pixels [24].

Using bilinear interpolation circular disk was constructed. The ligatures after diacritics removal are converted to circular disk. Similarly, all other ligatures are converted to circular disk. After the conversion of an image from rectangular to circular disk, the Zernike moments of all the ligatures are calculated up to order of 3. The concept of Zernike moments was firstly instigated by Teague [25] for image analysis. The reason for selecting the Zernike Moment was rotation invariance property. If we rotate an image it will not alter the magnitude of Zernike moments. It requires lower computational precision, on the contrary, regular moments requires high computational precision.

The calculation of Zernike moments for an image comprises of computation of radial polynomials, complex Zernike functions and complex Zernike moments respectively [26]–[29]. The method for calculating Zernike moments for an image starts with the computation of Zernike radial polynomials [27]. The Radial polynomial $Rad_{e,f}$ is described as

$$Rad_{e,f} = \sum_{w=0}^{(e-|f|)/2} (-1)^w \frac{(e-w)!}{w!(\frac{e+|f|}{2}-w)!(\frac{e-|f|}{2}-w)!} \rho^{e-2w} \tag{10}$$

here, e is any positive integer which represents the order of the radial polynomial, f is a positive or negative integer that fulfills bounds $e - |f| =$ even, $|f| \leq e$ represent's the repetition of azimuthal angle and ρ is the length of the vector from the origin to (i, j) [26] & [30]. The complex Zernike functions are defined within the unit circle which is composed as:

$$X_{e,f}(\rho, \theta) = Rad_{e,f}(\rho)\, e^{jm\theta}, \quad |\rho| \leq 1 \tag{11}$$

The complex Zernike polynomials meet the requirements of the orthogonal property [26]–[29].

$$\int_0^{2\pi}\int_0^1 X^*_{e,f}(\rho,\theta)\, X_{s,t}(\rho,\theta)\, \rho d\rho d\theta = \begin{cases} \frac{\pi}{e+1} e=s, & f=t \\ 0, & else \end{cases} \tag{12}$$

here $*$ represents the complex conjugate. The orthogonal property neither suggests redundancy nor overlapping of data amid the moments with dissimilar orders and repetitions. This characteristic empowers every moment to be significantly distinct and autonomous from the data in an image. The complex Zernike moments of order e with repetition f are described as [28].

$$Z_{e,f} = \frac{e+1}{\pi}\int_0^{2\pi}\int_0^1 X^*_{e,f}(\rho,\theta)\, X_{s,t}(\rho,\theta)\, \rho d\rho d\theta \tag{13}$$

Furthermore, the coordinates of the image must be transformed from Cartesian to Polar coordinates as shown in Fig. 16. The pixels which are situated outside the circular disk are excluded from the calculation. The discrete form of Zernike moments for an input image of size $A \times A$ is given as [30]:

$$\begin{aligned} Z_{e,f} &= \frac{e+1}{\lambda_A}\sum_{g=0}^{A-1}\sum_{h=0}^{A-1} T(g,h) X^*_{e,f}(g,h) \\ Z_{e,f} &= \frac{e+1}{\lambda_A}\sum_{g=0}^{A-1}\sum_{h=0}^{A-1} T(g,h) Rad_{e,f}(\rho_{gh}) e^{-jm\theta_{gh}} \end{aligned} \tag{14}$$

where $0 \leq \rho_{gh} \leq 1$ and λ_A is a Normalization factor.

For the normalization factor, it is compulsory that a number of pixels must be situated inside the circular disk. The converted distance ρ_{gh} and the phase θ_{gh} at the pixel of (g, h) are expressed by the following formulas:

$$\begin{aligned} \rho_{gh} &= \frac{\sqrt{(2g-A+1)^2 + (2h-A+1)^2}}{A} \\ \theta_{gh} &= \tan^{-1}\left(\frac{A-1-2h}{2g-A+1}\right) \end{aligned} \tag{15}$$

Here g and h shows columns and rows of an image respectively. At last, complex Zernike absolute is computed.

G. CLASSIFIER

In this study, cascade forward back propagation neural network [3], [37] & [40] was used for training. There are different variants of back propagation algorithm such as gradient descent algorithm with adaptive learning rate, Fletch Reeves update conjugate gradient, Polak-Ribere update conjugate gradient, Powell-Beale restarts conjugate gradient and scaled conjugate gradient algorithm, etc. Scaled conjugate gradient backpropagation algorithm was used as a training function because this algorithm gives better results. In addition, the mean square error was used as a performance function [33].

1) OUTPUT LABELS

Marking of output labels of such a large data set will take a lot of time, therefore, the output labels are not marked manually. There are 2,496 ligature classes and each class contain about 15 samples which makes 37,440 total ligature samples. The marking of 37,440 output labels using an algorithm will save a lot of time. So, in order to save time output labels were marked using the pseudocode given in Fig. 17.

2) ARCHITECTURE OF NEURAL NETWORK

A neural network comprises of the Input layer, Hidden layer and an Output layer. A deep neural network can have

```
procedure MarkOutputLabels(TotalInputFeatures,Labels,Classes,M,N,Data,AllData)
    i ← 1
    for each integer i in Classes do
        Labels[i] ← 0
    end for
    j ← 1
    for each integer j in Classes do
        LigatureClasses[j] ← Sort all ligature class names in alphabetically order
    end for
    i ← 1
    k ← 1
    for each integer in i in M do
        j ← 1
        for each integer in j in N do
            Img ← Read Image of Ligature Class
            Data[k] ← Compute All Features of Img upto order of 3
            Labels[i] ← 1
            p ← 1
            for each integer in p in TotalInputFeatures do
                AllData[p] ← Data[p]
            end for
            q ← 1
            for each integer in q in Classes do
                AllData[p] ← Labels[q]
                p ← p + 1
            end for
            k ← k + 1
        end for
        Labels[i] ← 0
    end for
end procedure
```

FIGURE 17. Pseudocode of Marking output labels.

N number of layers. The neural network, used in this study for the training of features, contains two hidden layers where each layer has 100 neurons because of more than one hidden layer it is a deep neural network. The transfer function used for the first hidden layer was hyperbolic tangent sigmoid transfer and similarly, the transfer function used for the second hidden layer was 'purelin'. The architectural view of cascade forward back propagation neural network in MATLAB is shown in Fig. 18. For the training of deep neural network 31 features and output, labels were given as input to the deep neural network. The features which were calculated for each image includes raw moments, central, scale-invariant, Hu and Zernike moments up to order of 3. As the default number of epochs is 1000 in MATLAB so, the number of epochs is increased to 2,50000 to reach the performance goal and the minimum step size is increased from $1e^{-6}$ to $1e^{-100}$. We have trained our deep neural network on high frequently used ligature corpus of CLE.

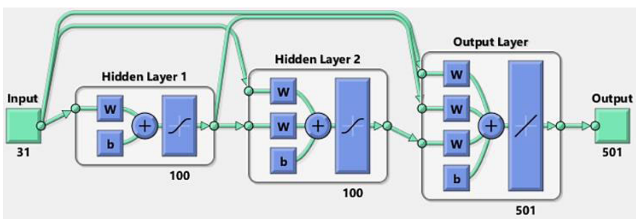

FIGURE 18. Architectural view of Cascade Forward Back-propagation Neural Network in MATLAB. It shows there are 31 input features, two hidden layers each having 100 neurons and there are 501 output classes.

H. HIERARCHICAL APPROACH

The total dataset contains 2,496 unique ligature classes which are split into five different sets in order to form a hierarchical model as shown in Fig. 20. From set 1 to 4 each set comprises of 501 ligature classes. The 501$^{\text{th}}$ ligature class contains all other set ligature classes' samples. Each set from 1 to 4 contains 500 classes having total 7500 samples forming the

```
procedure LigatureRecognition(TotalLigatures,NumberofFeatures)
    i ← 1
    for each integer i in TotalLigatures do
        Img ← Read Ligature Image
        Data[k] ← Compute All Features of Img upto order of 3
    end for
    Store all the features in a CSV file
    NumberofLigatureClasses ← Total number of records in a CSV file
    Load all Trained Neural Networks
    Load all Ligature Class Names
    Load CSV file containing ligature features
    for each integer i in NumberofFeatures do
        Instances ← Read each feature vector from CSV file
        Features ← Compute Transpose of Instances
        PredictedLabels ← Pass Features to the 1st Trained Network
        labels ← Compute Transpose of PredictedLabels
        Index ← Find maximum index of labels
        if Index <= 500 then
            Predict Ligature Class
        else
            PredictedLabels ← Pass Features to the 2nd Trained Network
            labels ← Compute Transpose of PredictedLabels
            Index ← Find maximum index of labels
            if Index <= 500 then
                Predict Ligature Class
            else
                PredictedLabels ← Pass Features to the 3rd Trained Network
                labels ← Compute Transpose of PredictedLabels
                Index ← Find maximum index of labels
                if Index <= 500 then
                    Predict Ligature Class
                else
                    PredictedLabels ← Pass Features to the 4th Trained Network
                    labels ← Compute Transpose of PredictedLabels
                    Index ← Find maximum index of labels
                    if Index <= 500 then
                        Predict Ligature Class
                    else
                        PredictedLabels ← Pass Features to the 5th Trained Network
                        labels ← Compute Transpose of PredictedLabels
                        Index ← Find maximum index of labels
                        if Index <= 496 then
                            Predict Ligature Class
                        end if
                    end if
                end if
            end if
        end if
    end for
end procedure
```

FIGURE 19. Pseudocode of ligature recognition.

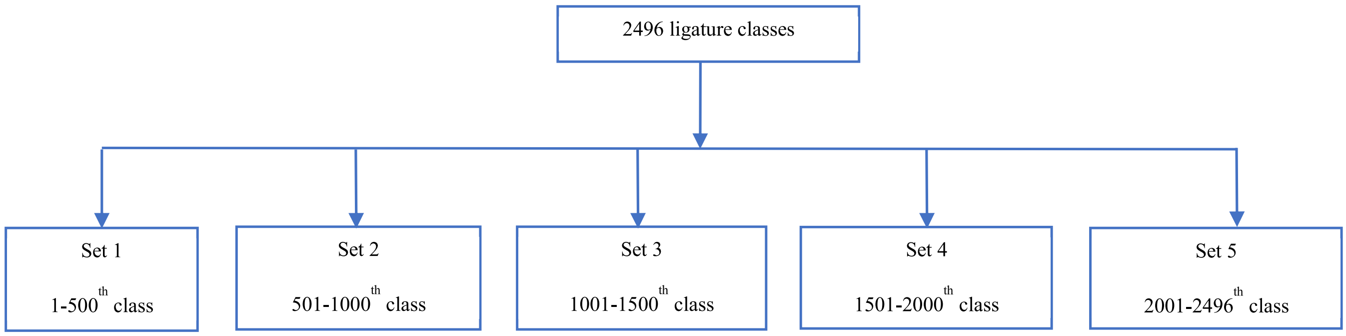

FIGURE 20. Ligature classes splitting using hierarchical approach.

positive dataset while 4 samples from each of the remaining class are used as a negative data set. Adding more than 4 samples considerably increases the training overheads. Furthermore, a cascading approach has been implemented. A 501th class is defined in each unit of the cascaded model. This 'other class' acts as a cascading switch over. In case the input is recognized as belonging to the other class it is switched over to the cascaded subsequent unit of the entire network. There is no need for cascading for the last unit because no more classes are left behind. That is why no 'other class' is defined for the last set. The pseudocode for ligature recognition using cascade forward back propagation neural network is given in the previous section.

1) LIGATURE RECOGNITION

Once the neural network is trained the model is ready for recognition of any ligature/diacritic. The computed features and the trained neural networks are passed to MATLAB "SIM" function. The MATLAB "SIM" function returns the predicted indexes of ligatures/diacritic. After this, the maximum index of the predicted label is computed. If the predicted ligature class is within 1 to 500 classes, then it will predict the class otherwise it will predict the other class i.e. 501th class. If it predicts the other class, then it will cascade and forward to the next trained neural network as shown in Fig. 21. All the predicted ligature classes are saved in CSV file. For each set neural network was trained using the already discussed neural network architecture. There were five trained neural networks. To predict the ligature class using the trained neural network the pseudocode given in Fig. 19 was adopted.

IV. EXPERIMENTS AND RESULTS

The objective evaluation of a newly developed prediction model is always carried out to depict its quality of prediction for targeted classes. Various measures and methods are employed by researchers to evaluate a prediction model. However, the most employed measure is accuracy. Herein, different experiments were carried out to find the accuracy of the proposed model which was based on scale and rotation invariant classifier for optical character recognition of the Urdu Nastalique font. The same CLE corpus was used in order to evaluate the accuracy of the proposed model. Several benchmarks such as the Confusion Matrix, ROC, Sensitivity, Specificity, Precision, ACC and MCC were adopted to calculate the accuracy of the proposed model. These benchmarks are described below.

A. CONFUSION MATRIX

The confusion matrix illustrates the overall accuracy of the classifier, along with its ability to truly predict positive and negative samples. This eminent benchmark describes the accuracy of the predicted data against the actual data. In case of binary classification, this matrix is usually comprised of four components i.e. (i) True Positive (TP), which represents here the number of positive ligatures identified by the classifier as positive, (ii) False Negative (FN), which depicts number of positive ligatures, identified by the classifier as negative, (iii) True Negative (TN), which, herein, defines number of negative, ligatures identified correctly by the classifier as negative, and lastly, (iv) False Positive (FP), which are positive ligatures identified by the classifier as negative. Based on these four components, the accuracy to identify any ligature or diacritic of all networks is given in Table 3.

The below table shows in Network 1 there are total 15,484 training samples out of which number of TP are 5529, number of FN is 597, number of TN is 9195 and number of FP are 163. Similarly, the number of TP, FN, TN and FP of all other networks are given in Table 3.

TABLE 3. Confusion Matrix results.

Network	Total Training Data Set	TP	FN	TN	FP
1	15,484	5529	597	9195	163
2	13,484	5387	438	7438	221
3	11,484	5412	277	5532	263
4	9,484	5418	70	3769	227
5	7,440	7272	0	34	134

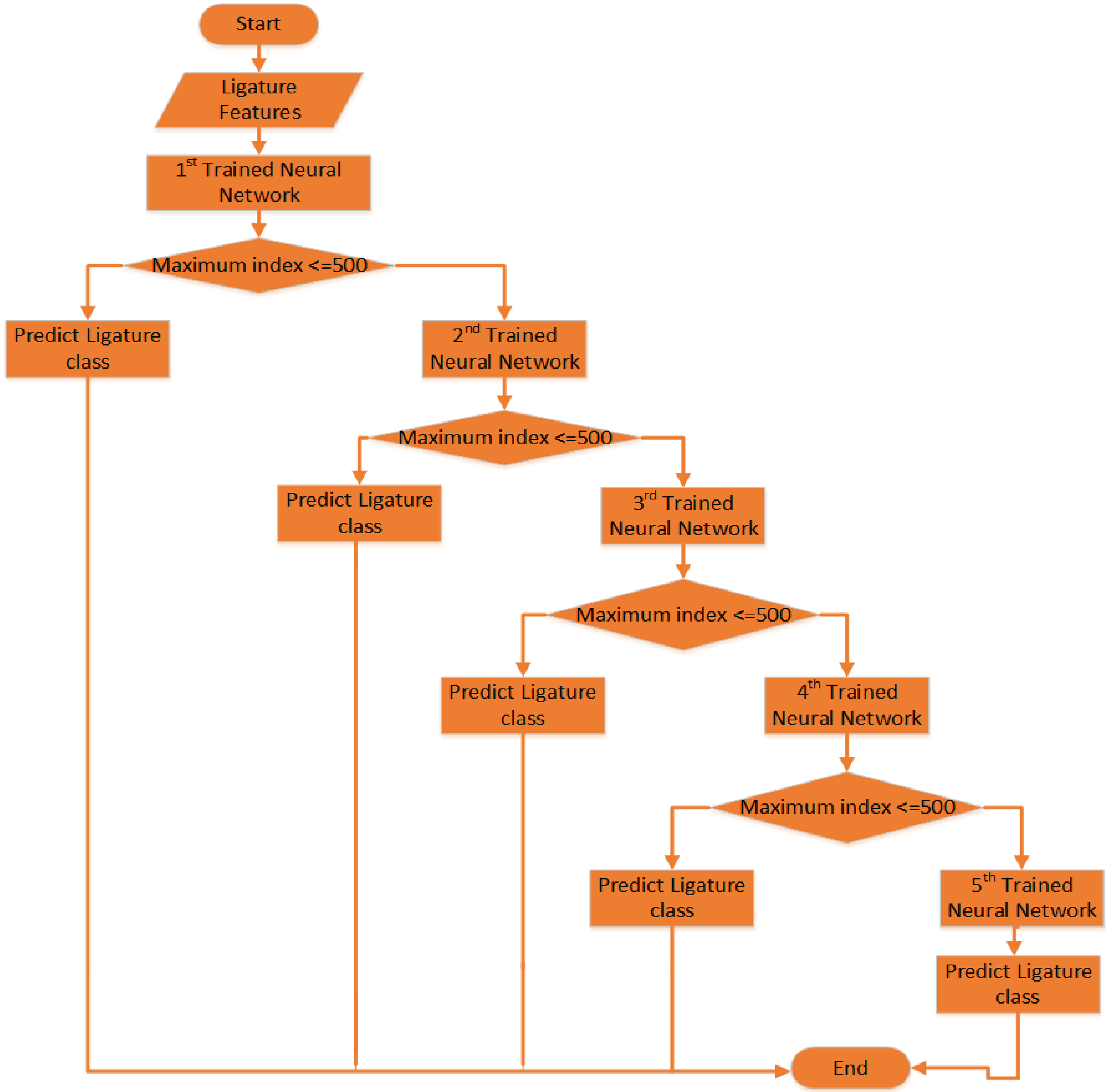

FIGURE 21. **Flowchart of ligature recognition using Cascade Forward Back-propagation Neural Network.**

B. RECEIVING OPERATING CHARACTERISTICS (ROC)

ROC graph depicts the area under the curve, the higher the area above the tangent line drawn at the angle of 45-degree demonstrates higher the accuracy and closer the area to the tangent line implies lesser accuracy. If the area under the curve is minimum, it implies an actual line exactly map on the tangent line and it shows the predictive model has failed to perform the required prediction. This graph is a trade-off between the True Positive Rate (TPR), also known as sensitivity of the predictive model, and the False Positive Rate (FPR), which is also known as the specificity of the predictive model. As the sensitivity increases, specificity decreases and vice versa. If the graph touches the upper left corner, then the predictive model is excellent and the value of TPR is 100%. It is apparent from Fig. 22 (a), (b), (c), (d) and (e) that the area under the curve of all classes touches the upper left corner which means the trained networks have high accuracy. The result of the ROC of all the networks is shown in the following Fig. 22 (a), (b), (c), (d) and (e) respectively.

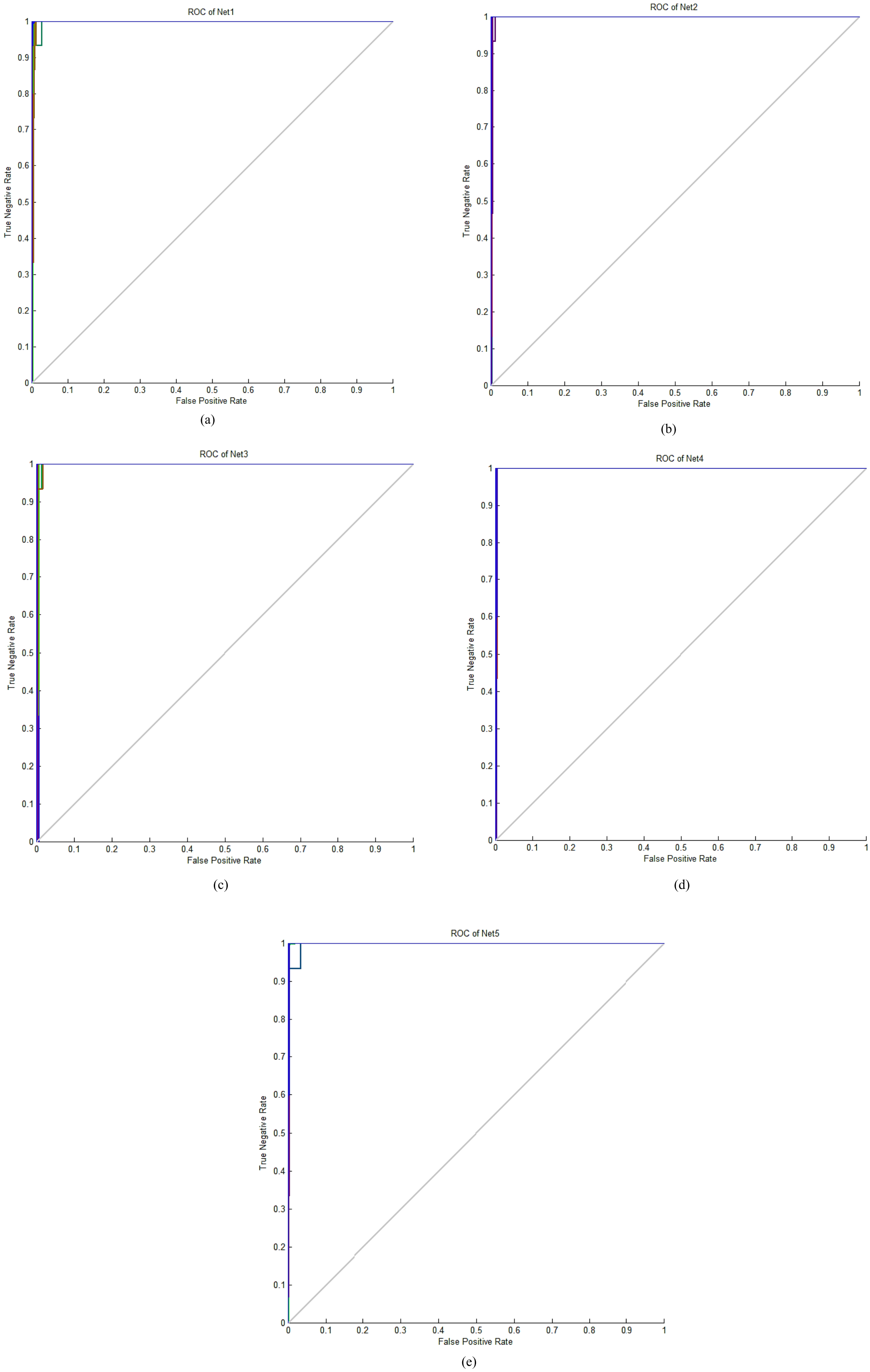

FIGURE 22. **(a). ROC of Network 1. (b). ROC of Network 2. (c). ROC of Network 3. (d). ROC of Network 4. (e). ROC of Network 5.**

C. SENSITIVITY

The ability of the predictive model to detect a ligature class can be determined through sensitivity calculation. Sensitivity shows the number of true positives i.e. number of correctly identified ligatures. High sensitivity is very important where we need to identify ligatures due to multi shapes, different position of characters and context sensitivity. The root cause of context sensitivity is filled or false loops and ligature overlapping. The sensitivity (S_n) of any predictive model can be calculated as

$$S_n = \frac{Number\ of\ true\ positves}{Number\ of\ true\ positives + Number\ of\ false\ negatives}$$

$$S_n = \frac{TP}{TP + FN} \tag{16}$$

where TP represents the number of true positives and FN represents the number of false negatives.

After calculation, the average sensitivity obtained for the proposed model is 95.318 % (Table 4). Sensitivity analysis shows the predictive model is appreciably accurate.

TABLE 4. The sensitivity of all networks.

Network	Sensitivity
1	90.254%
2	92.481%
3	95.131%
4	98.724%
5	100.000%
Average Sensitivity	95.318%

D. SPECIFICITY

The specificity (S_p) of a predictive model is calculated as

$$S_p = \frac{Number\ of\ true\ negatives}{Number\ of\ true\ negatives + Number\ of\ false\ positives}$$

$$S_p = \frac{TN}{TN + FP} \tag{17}$$

High specificity test, will correctly rule out how many ligatures does the model predict which actually not the class of ligature and the model also predicted that it is not the class of that ligature. For the proposed predictive model, the average specificity obtained is 81.078%, as given in Table 5 which shows the predictive model is appreciably accurate.

TABLE 5. The specificity of all networks.

Network	Specificity
1	98.258%
2	97.114%
3	95.461%
4	94.319%
5	20.238%
Average Specificity	81.078%

E. PRECISION

Precision tells the probability, given a positive test result how many ligature samples are positive. Precision also tells how many positively identified ligatures were relevant. Precision is also known as Positive Predictive Value (PPV), which is calculated as

$$Precision = \frac{Number\ of\ true\ positives}{Number\ of\ true\ positives + Number\ of\ false\ positives}$$

$$Precison = \frac{TP}{TP + FP} \tag{18}$$

The average precision obtained for the prediction model is 96.546%, as given in Table 6, which is depicting the overall ability of the predictive model to identify positive samples, accurately.

TABLE 6. The precision of all networks.

Network	Precision
1	97.136%
2	96.059%
3	95.365%
4	95.978%
5	98.190%
Average Precision	96.546%

F. ACCURACY (ACC)

In order to calculate the accuracy of the predictive model the ACC metric is used which is calculated as

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{19}$$

The average ACC obtained for all networks is 96.113% as given in Table 7. ACC value shows the predictive model is highly accurate.

TABLE 7. The accuracy of all networks.

Network	ACC
1	95.091%
2	95.112%
3	95.298%
4	96.868%
5	98.199%
Average ACC	96.113%

G. MATTHEWS CORRELATION COEFFICIENT (MCC)

To calculate the accuracy of the predictive model the MCC metric is used which is given by the following formula:

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP+FN)(TN + FP)(TN+FN)}} \tag{20}$$

After putting all the values in the above MCC formula the average MCC obtained is 0.817 as given in Table 8. MCC value shows the predictive model is stable.

TABLE 8. MCC of all networks.

Network	MCC
1	0.898
2	0.900
3	0.906
4	0.936
5	0.446
Average MCC	0.817

H. INDEPENDENT DATASET TESTING

The classification power of the trained neural network can be evaluated through various testing methods. Various techniques and measures are adopted by researchers to evaluate the overall predictive model. To evaluate the accuracy of the trained deep neural network through independent dataset testing, the data set is divided into two segments, training and testing dataset. The linear flow diagram of the testing computational model is shown in Fig. 5.

In independent testing, Urdu Nastalique images were tested having different ligatures, using which the predictive model was not trained before. The corpus taken from the CLE website contained 591 document images out of these 278 images containing 37,440 ligatures samples were separated for testing. These 37,440 ligature samples contain 2,496 unique high-frequency ligature classes. The results of independent testing are given in Table 9.

TABLE 9. Independent testing results.

Total no of ligatures	37,440
Correctly identified ligatures	36,120
Incorrectly identified ligatures	1320
Accuracy	96.474%
Error	3.526%
Specificity	81.078%
Sensitivity	95.318%
ACC	96.113%
Precision	96.546%
MCC	0.817

I. k-FOLD CROSS-VALIDATION

k-fold cross-validation is a method to evaluate the predictive model by splitting the dataset into *k* disjoint folds. The model is trained and tested *k* times. The validation of the dataset is arbitrarily distributed into *k* folds of uniform size subsamples. From these *k* subsamples, a single subsample is selected for the validation of the testing model and the remnant *k-1* are used for the training of the model. This method is repeated *k* times with each of the *k* subsamples used only once in the validation data. The results of *k* folds are averaged to produce a single estimate. Fig. 23 shows that blue circles are used for testing data and purple circles are used for training data.

As there are 2,496 classes and each class contains 15 samples, all the classes are also divided into different sets. In 1st fold cross validation 1st, 6th and 11th samples are used for testing from all the sets and the remnant ligature samples are used for training. The features of these training samples are computed as discussed previously. These features and labels are given as input to the deep neural network for training. After this, the features of the testing images are computed and these features are given as input to the trained deep neural network for class prediction. The output classes of all the sets are saved in excel file. These classes predicted by the deep neural network are validated by matching the segmented image ligature class with the original image ligature class. The accuracy of all the five networks is averaged to compute the 1st fold cross-validation accuracy. Similarly, in 2nd fold cross validation 2nd, 7th and 12th samples are used for testing from all the sets and the remaining ligature samples are used for training. In 3rd fold cross validation 3rd, 8th and 13th samples are used for testing from all the sets and the remaining ligature samples are used for training and so on till 5th fold cross-validation. The average accuracy of all 5 folds is computed by taking the mean of the accuracy of all 5-folds. Each set from set 1 to set 4 contains 500 classes. For testing three samples are separated from each class from set 1 to set 5. Set 5 contains only 496 classes. So, the total number of ligature samples tested are 37, 440. The results of 5-fold cross validation are given in Table 9. Since there are 15 positive samples for a single class, therefore, the natural choice for validation is 5 fold cross-validation. In 5-fold cross-validation, 5 disjoint partitions are formed randomly for training while the rest of the samples are used for testing purposes. It is not possible to partition data into 10 disjoint equal sized partitions hence rendering 10-fold cross validation flawed. The below-calculated accuracy, i.e. 96.922% as given in Table 10 of 5-folds is nearly equal to independent testing accuracy, i.e. 96.474% as given in Table 9. This cross-validates the proposed methodology accuracy and it shows it is highly accurate. Moreover, for cross-validating other results of independent testing like specificity, sensitivity, precision, MCC and ACC, confusion matrixes are constructed for *k1*, *k2*, *k3*, *k4* and *k5* respectively. The summary of the benchmark (5-fold cross-validation) results are given in Table 11.

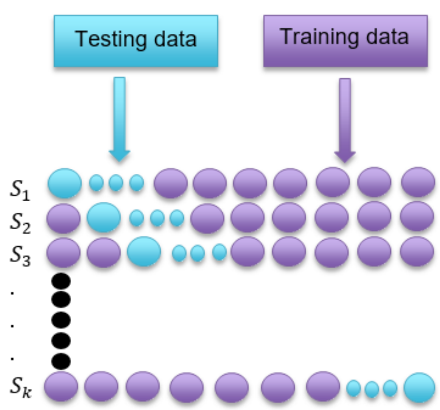

FIGURE 23. K-fold cross validation.

TABLE 10. 5-fold cross-validation results.

	Network 1	Network 2	Network 3	Network 4	Network 5	Average
k1	93.820%	97.312%	96.817%	98.166%	96.767%	96.576%
k2	96.042%	97.083%	97.313%	98.652%	98.998%	97.617%
k3	95.000%	96.809%	98.632%	97.572%	97.333%	97.069%
k4	94.792%	94.792%	97.750%	99.375%	98.111%	96.964%
k5	92.542%	94.375%	97.292%	99.375%	98.333%	96.383%
5 fold Average						**96.922%**

TABLE 11. Summary of benchmarks (5-folds).

	k1	k2	k3	k4	k5	Average
Sensitivity	96.352%	95.190%	96.070%	96.259%	96.021%	95.978%
Specificity	79.018%	82.153%	80.441%	81.329%	82.567%	81.101%
Precision	95.905%	96.635%	96.115%	96.789%	94.870%	96.062%
ACC	96.271%	96.258%	96.185%	96.101%	95.839%	96.130%
MCC	0.809	0.818	0.829	0.816	0.803	0.815

Study	Features	Classifier	Dataset for Nastalique Font	Accuracy
Shabbir et. al [34]	Horizontal projection, vertical projection, upper profile and lower profile	Hidden Markov Model	2,017 ligatures	**92%**
Khattak et. al [35]	Projection, Concavity, Curvature	Hidden Markov Model	2, 208 ligatures	**97.93%**
Proposed Model	Raw moments, Central moments, Scale-invariant moments, Hu moments and Zernike moments (up to an order of 3)	Cascade Forward Back Propagation Neural Network	2,496 ligatures	**96.474%**

FIGURE 24. Comparison of Existing and Proposed Technique for Urdu Nastalique Font on CLE corpus.

Table 11 analysis shows that the average sensitivities, specificities, precisions, ACC's and MCC's of 5-folds are nearly equal to independent testing results as given in Table 9. This shows that the proposed model is highly accurate.

J. COMPARISON OF PROPOSED AND EXISTING TECHNIQUES

The proposed predictive model was compared with already existing predictors and the accuracy comparison of the proposed model with other existing notable predictors as shown in Fig. 24. It is observed by different experiments that the proposed method is noticeably accurate than the previous techniques because our features deeply recognize each ligature and diacritic. The accuracy reported by Khattak et al. [35] is appreciably more, but this accuracy is achieved on 2,208 ligatures only and it is font invariant whereas the proposed model on the same CLE corpus reports better accuracy on 2,496 ligatures and is also scale-invariant and rotation-invariant.

V. CONCLUSION AND FUTURE WORK

The ligature recognition of Urdu Nastalique is more difficult as compared to other languages. Urdu Nastalique has some characteristics like stacking of ligatures and cursiveness which makes ligature segmentation a difficult task. Thus, the ligature recognition of Urdu OCR is a strenuous task due to variants of scaling, rotation, orientation, textual style, the least number of samples available for training. Herein, several benchmark methods and metrics such as Confusion Matrix, Specificity, Sensitivity, Mathews Correlation Coefficient (MCC), Receiving Operating Characteristics (ROC), Accuracy (ACC) and Precision were computed to objectively evaluate the proposed predictive model. These benchmark results show the predictive model is highly accurate. By comparing independent testing and 5-folds cross-validation results it is concluded that the proposed model is highly accurate as compared to the already existing predictors.

This study can be improved further by doing semantic and contextual analysis of such cursive script. By doing the semantic analysis we can predict a ligature most appropriate class and we can check the meaning of that ligature in the dictionary. Furthermore, we can textually characterize a text and can infer knowledge from it by doing contextual analysis.

VI. CONFLICT OF INTEREST

There are no conflicts of interest.

REFERENCES

[1] H. Sharma and D. V. Sharma, "State of the art in Nastaleeq script recognition," *Int. J. Comput. Trends Technol.*, vol. 39, no. 1, pp. 40–46, 2016.

[2] S. T. Javed, S. Hussain, A. Maqbool, S. Asloob, S. Jamil, and H. Moin, "Segmentation free nastalique Urdu OCR," *World Acad. Sci., Eng. Technol.*, vol. 46, no. 10, pp. 456–461, 2010.

[3] I. Shamsher, Z. Ahmad, J. K. Orakzai, and A. Adnan, "OCR for printed Urdu script using feed forward neural network," *World Acad. Sci., Eng. Technol.*, vol. 23, pp. 172–175, Aug. 2007.

[4] N. Tabassam, S. A. H. S. Naqvi, H. ur Rehman, and F. Anoshia, "Optical character recognition system for Urdu (Naskh font) using pattern matching technique," *Int. J. Image Process.*, vol. 3, no. 3, p. 92, 2009.

[5] N. B. Amor and N. E. B. Amara, "Multifont Arabic character recognition using Hough transform and hidden Markov models," in *Proc. 4th Int. Symp. Image Signal Process. Anal. (ISPA)*, Sep. 2005, pp. 285–288.

[6] K. Khan, R. Ullah, N. A. Khan, and K. Naveed, "Urdu character recognition using principal component analysis," *Int. J. Comput. Appl.*, vol. 60, no. 11, pp. 1–4, 2012.

[7] Z. Ahmad, J. K. Orakzai, I. Shamsher, and A. Adnan, "Urdu Nastaleeq optical character recognition," *World Acad. Sci., Eng. Technol.*, vol. 26, pp. 249–252, Dec. 2007.

[8] S. T. Javed and S. Hussain, "Segmentation based Urdu Nastalique OCR," in *Proc. Iberoamer. Congr. Pattern Recognit.* Berlin, Germany: Springer, Nov. 2013, pp. 41–49.

[9] M. W. Sagheer, C. L. He, N. Nobile, and C. Y. Suen, "Holistic Urdu handwritten word recognition using support vector machine," in *Proc. 20th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2010, pp. 1900–1903.

[10] S. Sardar and A. Wahab, "Optical character recognition system for Urdu," in *Proc. Int. Conf. Inf. Emerg. Technol. (ICIET)*, Jun. 2010, pp. 1–5.

[11] A. Muaz, "Urdu optical character recognition system," M.S. thesis, Dept. Comput. Sci., Nat. Univ. Comput. Emerg. Sci. Lahore, Lahore, Pakistan, 2010.

[12] S. T. Javed and S. Hussain, "Improving Nastalique specific pre-recognition process for Urdu OCR," in *Proc. IEEE 13th Int. Multitopic Conf. (INMIC)*, Dec. 2009, pp. 1–6.

[13] G. S. Lehal, "Ligature segmentation for Urdu OCR," in *Proc. 12th Int. Conf. Document Anal. Recognit. (ICDAR)*, Aug. 2013, pp. 1130–1134.

[14] Z. A. Shah, "Ligature based optical character recognition of Urdu-Nastaleeq font," in *Proc. Int. Multi Topic Conf. Abstracts (INMIC)*, Dec. 2002, p. 25.

[15] L. Peng, C. Liu, X. Ding, J. Jin, Y. Wu, H. Wang, and Y. Bao, "Multi-font printed Mongolian document recognition system," *Int. J. Document Anal. Recognit.*, vol. 13, no. 2, pp. 93–106, 2010.

[16] S. A. Husain, "A multi-tier holistic approach for Urdu Nastaliq recognition," in *Proc. Int. Multi Topic Conf. Abstracts (INMIC)*, Dec. 2002, p. 84.

[17] U. Pal and A. Sarkar, "Recognition of printed Urdu script," in *Proc. 7th Int. Conf. Document Anal. Recognit.*, Aug. 2003, pp. 1183–1187.

[18] A. Choudhary, R. Rishi, and S. Ahlawat, "Off-line handwritten character recognition using features extracted from binarization technique," *AASRI Procedia*, vol. 4, pp. 306–312, 2013.

[19] V. Ghods, E. Kabir, and F. Razzazi, "Decision fusion of horizontal and vertical trajectories for recognition of online Farsi subwords," *Eng. Appl. Artif. Intell.*, vol. 26, no. 1, pp. 544–550, Jan. 2013.

[20] Y. Li, "Reforming the theory of invariant moments for pattern recognition," *Pattern Recognit.*, vol. 25, no. 7, pp. 723–730, Jul. 1992.

[21] M.-K. Hu, "Visual pattern recognition by moment invariants," *IRE Trans. Inf. Theory*, vol. 8, no. 2, pp. 179–187, Feb. 1962.

[22] J. Flusser, "On the independence of rotation moment invariants," *Pattern Recognit.*, vol. 33, no. 9, pp. 1405–1410, Sep. 2000.

[23] J. Flusser and T. Suk, "Rotation moment invariants for recognition of symmetric objects," *IEEE Trans. Image Process.*, vol. 15, no. 12, pp. 3784–3790, Dec. 2006.

[24] K. T. Gribbon and D. G. Bailey, "A novel approach to real-time bilinear interpolation," in *Proc. 2nd IEEE Int. Workshop Electron. Design, Test Appl. (DELTA)*, Jan. 2004, pp. 126–131.

[25] M. R. Teague, "Image analysis via the general theory of moments," *J. Opt. Soc. Amer.*, vol. 70, no. 8, pp. 920–930, 1980.

[26] S.-K. Hwang and W.-Y. Kim, "A novel approach to the fast computation of Zernike moments," *Pattern Recognit.*, vol. 39, no. 11, pp. 2065–2076, 2006.

[27] W. Wang, J. E. Mottershead, and C. Mares, "Mode-shape recognition and finite element model updating using the Zernike moment descriptor," *Mech. Syst. Signal Process.*, vol. 23, no. 7, pp. 2088–2112, Oct. 2009.

[28] S. Li, M.-C. Lee, and C.-M. Pun, "Complex Zernike moments features for shape-based image retrieval," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 39, no. 1, pp. 227–237, Jan. 2009.

[29] X. Li and A. Song, "A new edge detection method using Gaussian–Zernike moment operator," In *Proc. 2nd Int. Asia Conf. Informat. Control, Autom. Robot. (CAR)*, vol. 1, Mar. 2010, pp. 276–279.

[30] A. Tahmasbi, F. Saki, and S. B. Shokouhi, "An effective breast mass diagnosis system using Zernike moments," in *Proc. 17th Iranian Conf. Biomed. Eng. (ICBME)*, Nov. 2010, pp. 1–4.

[31] M. I. Razzak, F. Anwar, S. A. Husain, A. Belaid, and M. Sher, "HMM and fuzzy logic: A hybrid approach for online Urdu script-based languages' character recognition," *Knowl.-Based Syst.*, vol. 23, no. 8, pp. 914–923, Dec. 2010.

[32] V. Ghods, E. Kabir, and F. Razzazi, "Effect of delayed strokes on the recognition of online Farsi handwriting," *Pattern Recognit. Lett.*, vol. 34, no. 5, pp. 486–491, Apr. 2013.

[33] S. Goyal and G. K. Goyal, "Cascade and feedforward backpropagation artificial intelligence models for prediction of sensory quality of instant coffee flavoured sterilized drink," *Can. J. Artif. Intell., Mach. Learn. Pattern Recognit.*, vol. 2, no. 6, pp. 78–82, 2011.

[34] S. Shabbir and I. Siddiqi, "Optical character recognition system for Urdu words in Nastaliq font," *Int. J. Adv. Comput. Sci. Appl.*, vol. 7, no. 5, pp. 567–576, May 2016.

[35] I. U. Khattak, I. Siddiqi, S. Khalid, and C. Djeddi, "Recognition of Urdu ligatures—A holistic approach," in *Proc. 13th Int. Conf. Document Anal. Recognit. (ICDAR)*, Aug. 2015, pp. 71–75.

[36] Y. D. Khan, S. A. Khan, F. Ahmad, and S. Islam, "Iris recognition using image moments and k-means algorithm," *Sci. World J.*, vol. 2014, Apr. 2014, Art. no. 723595.

[37] Y. D. Khan, F. Ahmad, and M. W. Anwar, "A neuro-cognitive approach for iris recognition using back propagation," *World Appl. Sci. J.*, vol. 16, no. 5, pp. 678–685, 2012.

[38] A. H. Butt, S. A. Khan, H. Jamil, N. Rasool, and Y. D. Khan, "A prediction model for membrane proteins using moments based features," *BioMed Res. Int.*, vol. 2016, Jan. 2016, Art. no. 8370132.

[39] Y. D. Khan, N. Rasool, W. Hussain, S. A. Khan, and K. C. Chou, "iPhosT-PseAAC: Identify phosphothreonine sites by incorporating sequence statistical moments into PseAAC," *Anal. Biochem.*, vol. 550, pp. 109–116, Jun. 2018.

[40] Y. D. Khan, F. Ahmed, and S. A. Khan, "Situation recognition using image moments and recurrent neural networks," *Neural Comput. Appl.*, vol. 24, nos. 7–8, pp. 1519–1529, 2014.

[41] S. Naz, K. Hayat, M. I. Razzak, M. W. Anwar, S. A. Madani, and S. U. Khan, "The optical character recognition of Urdu-like cursive scripts," *Pattern Recognit.*, vol. 47, no. 3, pp. 1229–1248, Mar. 2014.

[42] S. Naz, A. I. Umar, S. H. Shirazi, and M. M. Ajmal, "The optical character recognition for cursive script using HMM: A review," *Res. J. Appl. Sci., Eng. Technol.*, vol. 8, no. 19, pp. 2016–2025, 2014.

[43] D. A. Satti and K. Saleem, "Complexities and implementation challenges in offline Urdu Nastaliq OCR," in *Proc. Conf. Lang. Technol.*, Nov. 2012, pp. 85–91.

[44] M. I. Razzak, M. Sher, and S. A. Hussain, "Locally baseline detection for online Arabic script based languages character recognition," *Int. J. Phys. Sci.*, vol. 5, no. 7, pp. 955–959, 2010.

[45] A. Husain, A. Sajjad, and F. Anwar, "Online urdu character recognition system," in *Proc. MVA*, May 2007, pp. 98–101.

[46] Z. Ahmad, J. K. Orakzai, and I. Shamsher, "Urdu compound character recognition using feed forward neural networks," in *Proc. 2nd IEEE Int. Conf. Comput. Sci. Inf. Technol.*, Aug. 2009, pp. 457–462.

[47] W. Slany, "Recognition of segmented Arabic/Urdu characters using pixel values as their features," in *Proc. 15th Int. Conf. Comput. Inf. Technol. (ICCIT)*, 2012, pp. 507–512.

[48] J. Tariq, U. Nauman, and M. U. Naru, "Softconverter: A novel approach to construct OCR for printed Urdu isolated characters," in *Proc. 2nd Int. Conf. Comput. Eng. Technol.*, vol. 3, Apr. 2010, p. V3-495.

[49] D. Xiang, H. Liu, X. Chen, Y. Cheng, and H. Yao, "Recognition of off-line Arabic handwriting using Hidden Markov model toolkit," in *Proc. 11th Int. Symp. Distrib. Comput. Appl. Bus., Eng. Sci.*, Oct. 2012, pp. 409–412.

[50] J. H. AlKhateeb, J. Ren, J. Jiang, and H. Al-Muhtaseb, "Offline handwritten Arabic cursive text recognition using Hidden Markov Models and re-ranking," *Pattern Recognit. Lett.*, vol. 32, no. 8, pp. 1081–1088, Jun. 2011.

[51] S. M. Awaida and M. S. Khorsheed, "Developing discrete density Hidden Markov Models for Arabic printed text recognition," in *Proc. IEEE Int. Conf. Comput. Intell. Cybern. (CyberneticsCom)*, Jul. 2012, pp. 35–39.

[52] V. Ghods and E. Kabir, "Feature extraction for online Farsi characters," in *Proc. 12th Int. Conf. Frontiers Handwriting Recognit.*, Nov. 2010, pp. 477–482.
[53] S. Acharya, A. K. Pant, and P. K. Gyawali, "Deep learning based large scale handwritten Devanagari character recognition," in *Proc. 9th Int. Conf. Softw., Knowl., Inf. Manage. Appl. (SKIMA)*, Dec. 2015, pp. 1–6.
[54] A. Ul-Hasan, M. Z. Afzal, F. Shafait, M. Liwicki, and T. M. Breuel, "A sequence learning approach for multiple script identification," in *Proc. 13th Int. Conf. Document Anal. Recognit. (ICDAR)*, Aug. 2015, pp. 1046–1050.
[55] F. Simistira, A. Ul-Hassan, V. Papavassiliou, B. Gatos, V. Katsouros, and M. Liwicki, "Recognition of historical Greek polytonic scripts using LSTM networks," in *Proc. 13th Int. Conf. Document Anal. Recognit. (ICDAR)*, Aug. 2015, pp. 766–770.
[56] Y. Zhao, W. Xue, and Q. Li, "A multi-scale CRNN model for Chinese papery medical document recognition," in *Proc. IEEE 4th Int. Conf. Multimedia Big Data (BigMM)*, Sep. 2018, pp. 1–5.
[57] S. M. Obaidullah, C. Halder, K. C. Santosh, N. Das, and K. Roy, "PHDIndic_11: Page-level handwritten document image dataset of 11 official Indic scripts for script identification," *Multimedia Tools Appl.*, vol. 77, no. 2, pp. 1643–1678, 2018.
[58] S. B. Ahmed, S. Naz, M. I. Razzak, S. F. Rashid, M. Z. Afzal, and T. M. Breuel, "Evaluation of cursive and non-cursive scripts using recurrent neural networks," *Neural Comput. Appl.*, vol. 27, no. 3, pp. 603–613, 2016.
[59] N. Sabbour and F. Shafait, "A segmentation-free approach to Arabic and Urdu OCR," *Proc. SPIE*, vol. 8658, Feb. 2013, Art. no. 86580N.
[60] G. S. Lehal, "Choice of recognizable units for URDU OCR," in *Proc. Workshop Document Anal. Recognit.*, Dec. 2012, pp. 79–85.
[61] T. Hassan and H. A. Khan, "Handwritten Bangla numeral recognition using Local Binary Pattern," in *Proc. Int. Conf. Elect. Eng. Inf. Commun. Technol. (ICEEICT)*, May 2015, pp. 1–4.
[62] A. A. Sanjrani, J. Baber, M. Bakhtyar, W. Noor, and M. Khalid, "Handwritten optical character recognition system for Sindhi numerals," in *Proc. Int. Conf. Comput. Electron. Elect. Eng. (ICE Cube)*, Apr. 2016, pp. 262–267.
[63] S. Kaur and R. Bhatia, "Gurmukhi printed character recognition using hierarchical centroid method and SVM," *Int. J. Comput. Appl.*, vol. 149, no. 3, pp. 24–27, 2016.
[64] S. Khan, H. Ali, Z. Ullah, N. Minallah, S. Maqsood, and A. Hafeez, "KNN and ANN-based recognition of handwritten Pashto letters using zoning features," 2019, *arXiv:1904.03391*. [Online]. Available: https://arxiv.org/abs/1904.03391
[65] S. B. Ahmed, S. Naz, M. I. Razzak, and R. B. Yusof, "A novel dataset for English-Arabic scene text recognition (EASTR)-42K and its evaluation using invariant feature extraction on detected extremal regions," *IEEE Access*, vol. 7, pp. 19801–19820, 2019.
[66] G. Agnihotri, "An approach of neural networks to apprehending of Gurmukhi character style," to be published.

KHAWAJA UBAID UR REHMAN received the B.S. degree in computer science and the M.S. degree in software engineering from the University of Management and Technology (UMT), in 2015 and 2018, respectively. His major areas of interests include pattern recognition, image processing, and computer vision.

YASER DAANIAL KHAN is currently an Associate Professor with the University of Management and Technology (UMT). His major areas of interests include pattern recognition, image processing, computer vision, and bioinformatics. He is leading the research group on "pattern recognition and bioinformatics" at UMT. He has published articles in well-reputed journals and conferences.