High-Level Shape Representation in Printed Gujarati Characters

Mukesh M. Goswami[1] and Suman K. Mitra[2]
[1]*Faculty of Technology, Dharmsinh Desai University, Nadiad, Gujarat, India*
[2]*Dhirubhai Ambani Inst. of Information and Communication Tech., Gandhinagar, Gujarat, India*
mgoswami.it@live.com, suman_mitra@daiict.ac.in

Keywords: Pattern Recognition, Character Shape Representation, Shape Similarity, Character Recognition, Gujarati Characters.

Abstract: This paper presents extraction and identification of the high-level stroke (HLS) from printed Gujarati characters. The HLS feature describes a character as a sequence of predefined high-level strokes. Such a high-level shape representation enables approximate shape similarity computation between characters and can easily be extended to word-level. The shape similarity based character and word matching have extensive application in word-spotting based document image retrieval and character classification. Therefore, the proposed features were tested on printed Gujarati character database consisting of 12000 samples from 42 different symbol classes. The classification is performed using k-nearest neighbor with shape similarity measure. Also, a shape similarity based printed Gujarati word matching experiment is reported on a small word image database and the initial result are encouraging.

1 INTRODUCTION

India, being a multilingual country, has more than 22 officially listed languages written in 12 different scripts. Substantial work in character classification, OCR, and word-spotting is reported in the literature for dominating Indian scripts, like Devanagari, Bengali, Tamil, and Telugu. However, many scripts such as Gujarati still lakes attention of researchers. Despite years of efforts, the word-level accuracy of the OCR system for many Indian scripts have remained low as compared to western text mainly due to the large and complex character set including base characters, modifiers, and conjunct symbols (Kompalli et al., 2005). Therefore, the researchers are motivated to explore the recognition free approach for document image retrieval in many Indian scripts (Srihari et al., 2006; Hassan et al., 2009; Tarafdar et al., 2010; Jawahar et al., 2004). As a result, the shape similarity based character and word recognition for Indian scripts have gained considerable interest in recent time.

The majority of the recognition free systems depends on the shape-based features for the matching of characters and words (Doermann, 1998). Such a system demands features that are compact yet efficient in describing the high-level shape of characters. Also, it should be easy to compute the shape similarity between characters and words (Yang et al., 2008). Therefore, The current paper investigates a technique for compact and high-level shape representation of characters using the sequence of predefined high-level strokes (HLS). Such a sequential representation facilitates an efficient shape similarity matching between characters and words using dynamic programming based algorithm.

The rest of the paper is organized as follow: we start with a brief discussion on the related work in Section 2. Section 3 outlines the representation, extraction, and identification of high-level stroke. Section 4 discusses shape similarity computation between characters. Section 5 describes the experimental setup for classification of printed Gujarati characters using k-nearest neighbor and shape similarity measure as well as the shape similarity based word-matching experiments. Finally, the paper is concluded in Section 6.

2 RELATED WORK

Much work is found in the literature for the classification of characters from both North as well as South Indian script families. The character classification for prominent North Indian scripts like Devanagari, Bengali, Gurmukhi, and Oriya is discussed in (Chaudhuri and Pal, 1998; Chaudhuri et al., 2001; Lehal

Goswami, M. and Mitra, S.
High Level Shape Representation in Printed Gujarati Character.
DOI: 10.5220/0006191104180425
In *Proceedings of the 6th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2017)*, pages 418-425
ISBN: 978-989-758-222-6

and Singh, 2000), whereas the work on major Dravidian languages such as Tamil, Telugu, and Kannada is available in (Jawahar et al., 2003; Aparna and Ramakrishnan, 2002; Lakshmi and Patvardhan, 2002).

The work found for the printed Gujarati character classification is less compared to many other Indian scripts. Some of the early contributions include work by Antani and Agnihotri (Antani and Agnihotri, 1999), who used moment features with minimum Hamming distance and k-Nearest Neighbor (kNN) classifier to claim an accuracy of 67% on a small database of 800 samples. The most noticeable contributions include work by Dholakia *et al.* (Dholakia et al., 2007) that uses wavelet features with Neural Network and kNN classifiers. The overall accuracy claimed was of 96-97% on a database of 4173 symbols of 119 classes. Goswami *et al.* (Goswami et al., 2011) have used Self-Organizing Map(SOM) projection with the k-NN classifier and reported 84% accuracy on the moderately sized database. Hassan *et al.* (Hassan et al., 2014) have used Multiple Kernel Learning based Support Vector Machine (MKL-SVM) classifier with multiple features, like fringe distance map (FM), shape descriptor (SD), the histogram of gradients (HoG). The accuracy claimed was 97-98% on a database of 16000 symbols including modifiers, base characters, and conjuncts. Recently, Goswami and Mitra (Goswami and Mitra, 2015) have used low-level stroke features with the k-NN classifier for printed Gujarati character classification and claimed an accuracy of 95-98%.

Some initial work reported for word image retrieval on Devanagari, Bengali and Sanskrit script includes (Chaudhury et al., 2003; Srihari et al., 2006; Bhardwaj et al., 2008) which uses Geometric Graph, GSC, and Moment features, respectively. (Kumar et al., 2007) and (Meshesha and Jawahar, 2008) gives major contribution in document image retrieval for Telugu script. The experiment uses multiple features like Fourier descriptor, projection profiles, moments, etc., and the word images are compared using DTW (Rath and Manmatha, 2003). The experiments were carried out on a huge word image database extracted from 1800 pages of 7 machine-printed Telugu books. Other noticeable contribution includes (Hassan et al., 2009) that uses shape descriptor features with hierarchical locality sensitive indexing for word image retrieval from Devanagari, Bengali, and Malayalam scripts. (Tarafdar et al., 2010) have used a sequence of shape code with string edit distance for word image matching from Devanagari, Bengali, and Gurmukhi script. To the best of our knowledge, no work is reported in the literature for word image matching and retrieval for Gujarati script.

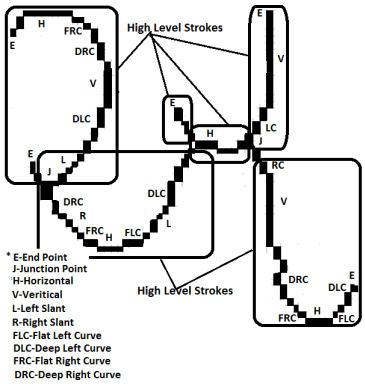


Figure 1: Formation of character as a set of high-level strokes where each high-level stroke is described as some sequence of shape primitives(i.e. lines, curves, and points).

The majority of the work reported until now uses transform domain, geometrical, or statistical features that give only a local view of the character shape and also generate a large feature vector. The global shape of the character can well be described using the native high-level strokes used to form a character. However, the decomposition of character into native stroke is not experimented much for the Indian script.

3 HIGH-LEVEL STROKE FEATURES

In the case of off-line text, the HLS can be described as a sequence of object pixel between two feature points in one pixel wide thinned character image(as shown in Figure 1). 48 major and minor HLS are identified that are visually non-redundant and sufficient to describe any middle zone symbols in Gujarati script (see Figure 2). Each HLS, in turn, is described as a sequence of shape primitives like lines, curves, junction points, and endpoints. Figure 3 shows the outline of the proposed method. The process begins by taking Low-Level Stroke (LLS) matrix computed using template matching algorithm proposed in (Goswami and Mitra, 2015). The HLS is extracted from the LLS matrix using a junction point based stroke scanning algorithm (discussed in Section 3.2). The extracted HLS are identified using linear chain Conditional Random Field (CRF) (Charles and McCallum, 2011) (discussed in Section 3.3).

3.1 Representation of High-Level Stroke

The HLS used to describe a character in Gujarati script are highly cursive and elongated (see Figure 2)

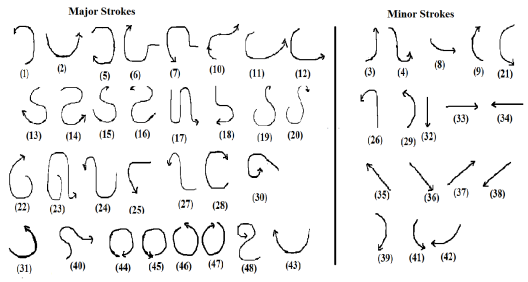


Figure 2: The set of major and minor high-level strokes present in Gujarati characters.

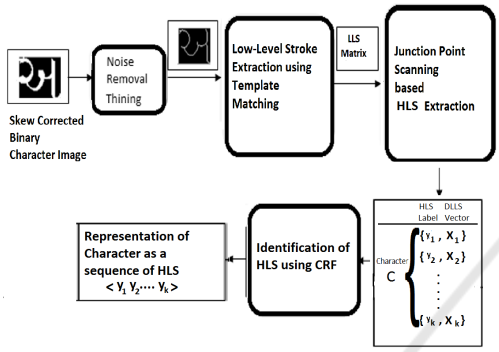


Figure 3: Outline of the high-level stroke extraction from printed Gujarati characters.

hence direct mathematical representation is infeasible. However, any complex HLS shape can be described as a sequence of shape primitives like points, lines, and curves(Figure 1). A template matching based algorithm was proposed in (Goswami and Mitra, 2015) to extract such shape primitives called low-level strokes (LLS). The algorithm takes MxN skew corrected, binarized, thinned character image as an input and generates a MxN matrix of LLS as output (as shown in Figure 4).

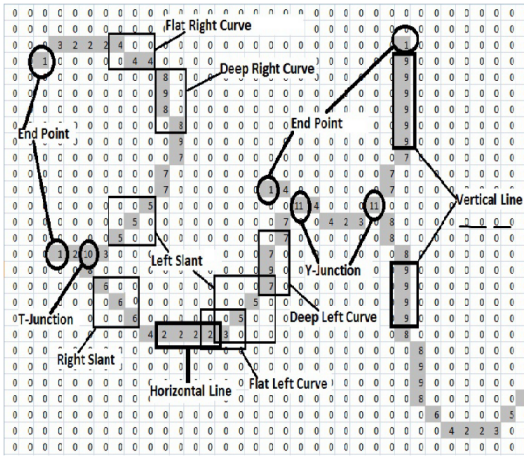


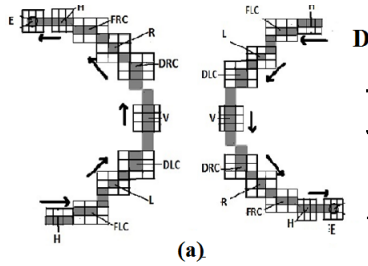
Figure 4: Output of the Low-Level Stroke extraction algorithm(Goswami and Mitra, 2015).

The LLS features can be used as elementary building blocks to represent a HLS (as shown in Figure 1). Apart from the sequence of LLS, it is also needed to know the direction information of LLS. For example, as shown in Figure 5(a), the sequence of LLS is same, but the direction is different resulting into two distinct HLS. Thus, the HLS can be defined as – *a sequence of Direction Encoded Low-Level Strokes (DLLS) between two feature points(i.e. junction point or end-points)*. Figure 5(b) shows 10 basic LLS combine with 8 direction code to obtained 18 different DLLS (Figure 5(c)).

3.2 Extraction of High-Level Stroke

A junction point based stroke extraction algorithm is proposed to obtain the direction and sequence information. The algorithm takes MxN matrix of LLS obtained from the input character image (shown in Figure 4) and extracts the high-level strokes present in the character. Each LLS in the sequence is replaced by corresponding DLLS depending on the direction information obtained while scanning. The necessary steps for the junction point based scanning algorithm are described as follow.

1. *STEP 1:* If a character has more than one stroke it has at least one junction point. During the first step, the middle region of LLS matrix is scanned in left to right order to obtain the list of junction points present in the matrix(Figure 6(a)).
2. *STEP 2:* The 3x3 neighborhood of each junction point in the list is scanned in clockwise order to obtain the starting point of each HLS originating from the given junction point. The clockwise scan resembles left to right and top to bottom writing order of Gujarati script. The touching junction points are handled by recursively invoking the scanning algorithm for each junction in the 3x3 neighborhood of current junction point.
3. *STEP 3:* A contour tracing algorithm is used to extract the HLS starting from the start point (obtained in step 2) till either an end point or a junction point is not reached. Each LLS in the contour has exactly two neighbors, already visited previous point and next unvisited point. The direction of LLS is obtained by finding the relative position of next point w.r.t. the current LLS. Thus, LLS is combined with direction information to obtain DLLS used to generate the output sequence. The HLS so extracted are deleted from the LLS matrix to avoid duplication.
4. *STEP 4:* Repeat Step 2 and 3 for each junction point in the list obtained in Step 1



Direction embedded Low-Level Strokes (DLLS)

Sr.No.	Low Level Stroke	Directional Code	Final Encoding
1	End Point	-NA-	1
2	Horizontal Line	(4, 5 or 6)	(2 and 10)
		(1,2 or 8)	
3	Left Flat Curve	(1,2 or 3)	(3 and 11)
4	Left Slant	(5,6 or 7)	(5 and 13)
5	Left Deep Curve		(7 and 15)
6	Right Flat Curve	(3,4 or 5)	(4 and 12)
7	Right Slant	(7,8 or 1)	(6 and 14)
8	Right Deep Curve		(8 and 16)
9	Vertical Line	(6,7 or 8)	(9 and 17)
		(2,3 or 4)	
10	Junction Point	NA	18

(c)

Figure 5: (a) Different HLS with same sequence of LLS but different direction (b) Direction encoding of LLS (c) Set of Directional Embedded Low-Level Strokes (DLLS).

5. *STEP 5:* Many characters in the Gujarati script are formed using single high-level strokes only, hence does not have any junction point. In such scenario, the starting point is obtained by finding the first endpoint in the top-left quadrant of the LLS matrix using zigzag scan. If the endpoint does not exist in the top-left quadrant then top-right, bottom-left, and bottom-right quadrants are scanned respectively, to obtained the starting endpoint. If the character contains neither a junction point nor an endpoint (for example symbol "zeros"), then the first LLS obtained in zigzag order from top-left quadrant is selected as starting point(see Figure 6(b)). Once the starting point is obtained the contour tracing algorithm discussed in step 3 is used to extract the HLS.

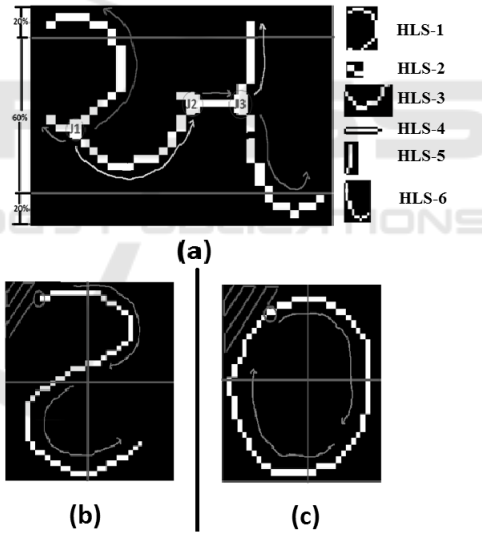


Figure 6: High-Level Stroke Extraction using (a) Junction Point (b) Endpoint or (c) First LLS in zigzag order from top-left corner.

3.3 Identification of High-Level Stroke(HLS)

The stroke extraction algorithm discussed in the previous section represent every character sample having k HLS as a set of k order pairs $(y_1,X_1),(y_2,X_2)$ $\ldots,(y_k,X_k)$ where y_i gives i^{th} HLS label and X_i gives corresponding DLLS vector. The identification of HLS refers to the problem of predicting the sequence of values $y_1,y_2,\ldots,y_k$ given the sequence of DLLS vectors $X_1,X_2,\ldots,X_k$ where each $X_i = \langle x_{i1},x_{i2},\ldots,x_{im}\rangle$. The stroke label $y_i \in \{1,2,\ldots,48\}$ and elements of DLLS vector $x_{ij} \in \{1,2,\ldots,18\}$.

Since each HLS y_i in the character depends on corresponding DLLS vector X_i as well as other y's present in the character, the Conditional Random Fields (CRF) (Charles and McCallum, 2011), an undirected probabilistic graphical model, is employed next to identify the HLS from the DLLS vector. The selection of CRF is justified in this context because it not only captures the dependency between HLS y_i and DLLS vector X_i but also consider the dependency between the current HLS y_i with other HLS present in the character. In the simplest case, known as linear

chain CRF, the current HLS y_i depends on DLLS vector X_i and the previous HLS y_{i-1}. The conditional probability of the set of HLS $Y = \{y_1, y_2, \ldots, y_k\}$ given the set of corresponding DLLS vectors $X = \{X_1, X_2, \ldots, X_k\}$ can be computed using Equation 1.

$$P(Y|X) = \frac{1}{Z(X)} \prod_{i=1}^{n} \exp\left(\sum_j \lambda_j f_j(y_i, y_{i-1}, X_i)\right) \quad (1)$$

where $Z(X)$ gives normalizing factor, f_j gives feature function and λ_j gives parameters of CRF, which needs to be learned from labeled training database using standard gradient optimization algorithm like L-BFGS. The HLS y_i in each character sample are labeled manually using stroke labeling tool to generated a labeled stroke database required for training the CRF model.

4 SHAPE SIMILARITY COMPUTATION

CRF model discussed in the previous section takes as an input the sequence of DLLS vectors $\{X_1, X_2, \ldots, X_k\}$ and labels corresponding HLS $\{y_1, y_2, \ldots, y_k\}$ for each character. Thus, a character, after HLS identification, is represented as a sequence of HLS $C =< y_1, y_2, \ldots, y_k >$. Therefore, the approximate shape similarity between the characters can be obtained by finding the similarity between HLS sequences for the given characters. Since it is needed to compare the entire HLS sequence (end to end alignment) to match the characters, the global sequence alignment techniques proposed by (Needleman and Wunsch, 1970) can be used to find the regions of similarity between two HLS sequences.

The algorithm, shown in Figure 7, gives the length of the maximum matching sub-sequence between two sequences. However, unlike traditional Longest Common Subsequence (LCS) algorithm which does not assign a penalty to mismatch, the NW algorithm assigns a penalty of -1 to mismatch as well as the gap. The similarity score is obtained by dividing the length of maximum matching subsequence by the minimum of the length of two sequences. The highest value of the similarity score is 1 if both the sequences are same and close to 0 if they are dissimilar. The set of all high-level strokes can be divided into two groups based on their importance in describing a character class, namely major and minor strokes (see Figure 2). Therefore, the indicator function, I, in the original algorithm is replaced by a customized similarity score matrix, $S_{48\times48}$, which gives more weight to major strokes than minor strokes while computing the similarity between the HLS sequences.

5 EXPERIMENTS AND RESULTS

Two different experiments are reported in the following Section, namely printed character classification and word matching, to show the effectiveness of shape similarity measure computed using HLS representation.

5.1 Printed Gujarati Character Classification

The experiment for printed Gujarati character classification is performed using k-NN classifier with shape similarity measure as a distance function to demonstrate the applicability of shape similarity measure discussed above. The optimum value of k in the k-NN depends on the distribution of samples in database (Murphy, 2012). However, for any given database, not all the classes have same distribution (i.e. the samples of some class are denser than others). Therefore, a single value of k may not give an optimum result for all the classes. The current experiment uses a simple heuristics to handle this issue. It will first find all the neighbors within a tight radius of the unknown sample (i.e. 80% similarity region in this case) and predict the class label using majority voting. However, if no data sample is found within 80% similarity region (i.e. the class has a sparse distribution of samples), then the 1^{st} nearest neighbor is used to predict the class label. The three-fold cross validation technique is used to make results more authentic. The average test accuracy over all three runs is used as the primary performance measure.

The database used in the experiment consist of 12000 samples of 42 middle zone character symbols from Gujarati script. The samples are collected from three different sources namely, machine printed books (BOOKDB), newspapers(NEWSDB), and laser printed documents (LASERDB) to ensure the varieties in terms of font type, style, size, ink thickness, etc.

The results of the experiment are shown in Table 1. The average test accuracy obtained on the combined database is **94.97%**. Table 2 shows the comparison of the results obtained with existing work. It is evident that the results obtained are 2-3% lower than the best-reported work (Hassan et al., 2014) in the literature. The drop in the accuracy is mainly due to the compactness of features since the size of the feature vector in (Hassan et al., 2014) is almost

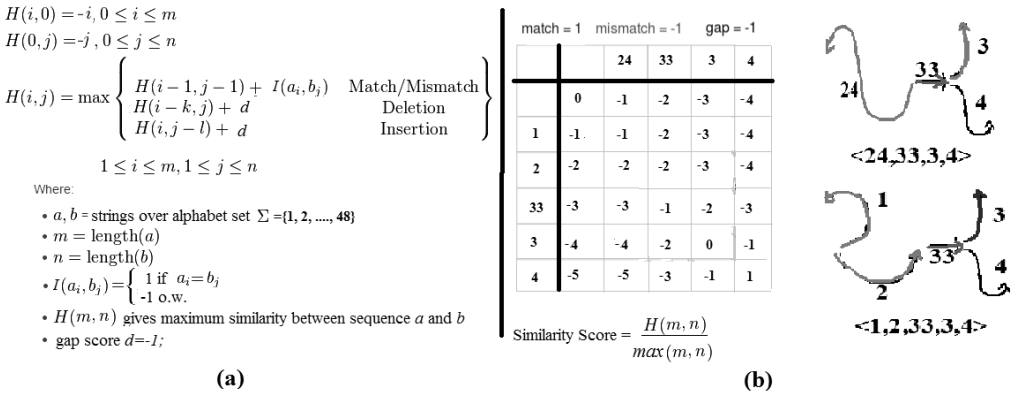


Figure 7: Compute shape similarity between characters using approximate string matching algorithm.

Table 1: Average test accuracy on BOOKDB, LASERDB, and NEWSDB using k-NN classifier with shape similarity measure.

Database	1st Run	2nd Run	3rd Run	Average
BOOKDB	98.43%	98.77%	98.50%	98.57%
LASERDB	94.73%	94.67%	90.64%	93.35%
NEWSDB	93.51%	94.10%	89.08%	92.23%
ALL COMBINED	97.72%	95.07%	92.12%	94.97%

Table 2: Comparison of methods for classification of printed Gujarati characters.

Author	Features/Classifier	Accuracy
Hassan *et al.* (2014)	HoG, Fringe Map and Shape Descriptor/ MKL-SVM	97-98%
Dholakia *et al.* (2009)	Daubechies D4 Wavelet Feature/ GRNN	97.59%
Goswami and Mitra (2015)	Histogram of Low-Level Stroke/ K-NN	95.35%
Proposed Method	**High-Level Strokes/ K-NN with Shape Similarity measure**	**94.97%**
Goswami *et al.* (2011)	SOM projection k-NN	84%
Antani and Agnihotri (1999)	Moment features K-NN with Minimum Hamming Distance	67%

100 times the size in the proposed approach. Unlike, the features used in other character classification methods, the HLS provides a high-level view of the character shape and enables only approximate matching between character rather than exact. Therefore, similar looking characters, as shown in Figure 8, are very often misclassified. Since the HLS features are extracted from thinned character image, the structural noise added by thinning algorithm also affects the performance. Therefore, the accuracy reported on BOOKDB is higher(98.43%) than LASERDB(94.73%) and NEWSDB(93.51%) because the newspaper and the laser printed symbols tend to have higher structural noise introduced by thinning as compared to BOOKDB (Suthar et al., 2014).

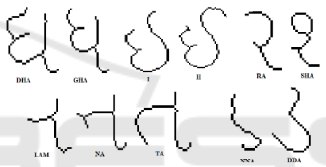


Figure 8: Characters with similar shape that are most commonly misclassified.

In summary, the HLS feature provides a compact representation of the high-level shape of the character. The feature compactness allows an efficient comparison between character shape. However, they provide only inexact matching hence not suitable for character classification and OCR application. Such, approximate shape matching is desirable for word-spotting based experiments where the objective is to match a query word image with all morphological variants. Moreover, also the sequential representation enables fast shape similarity computation using dynamic programming based sequence matching algorithms. Thus, the features could be useful in shape similarity based word-matching application.

5.2 Printed Gujarati Word Matching

The idea of shape similarity computation using HLS can easily be extended at the word-level. As shown in Figure 9, the skew-corrected, binarized, and thinned word-image is first segmented into character symbols using connected component analysis. The HLS are extracted from each symbols using stroke extraction algorithm (Section 3.2) and identified using CRF

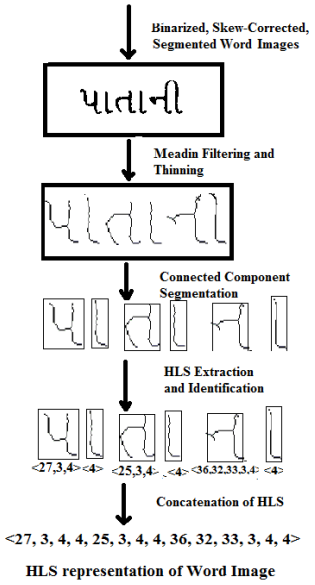


Figure 9: High-level stroke representation of Word Image.

(Section 3.3). The HLS sequences of all symbols are concatenated to generate a single HLS sequence representing the word-image. Finally, the shape similarity between word-images is computed by finding the global alignment score between the HLS sequence representing the word-images (Section 4).

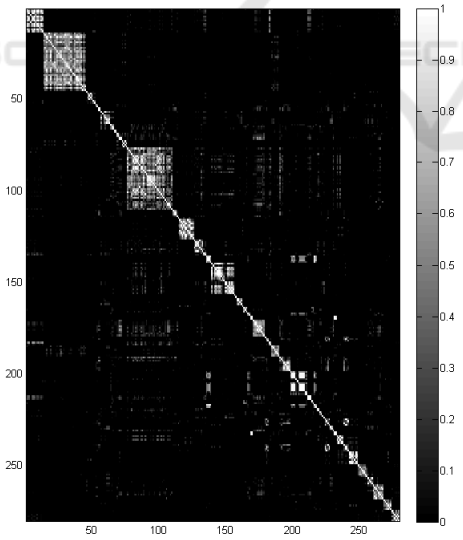


Figure 10: Shape Similarity matrix computed using HLS representation of 280 Gujarati word images of 48 different word groups.

The effectiveness of shape similarity measure for word-image was verified on a small word-group database consist of 280 word-images of 48 different word-groups. Each word-image in the database was represented as HLS sequence and shape similarity between each pair of word-images was computed using shape similarity measure discussed in Section 4. The results were represented using a similarity matrix of size 280×280 (shown visually in Figure 10). It is evident from Figure 10 that the HLS based shape similarity score between the pair of word-images in the same group is higher the one in different groups.

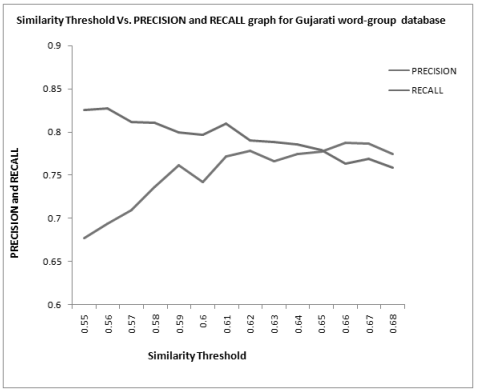


Figure 11: Similarity Threshold vs Precision and Recall graph for Gujarati word-group database.

In the word image retrieval experiment, a random query image is selected from each word group, and all matching word images are retrieved from the database based on the shape similarity threshold. The value of precision and recall were computed based on the number of relevant and retrieved images for each query and averaged over all queries. Figure 11 shows the value of precision and recall versus similarity threshold graph. The optimum values of precision and recall were **77.61%** and **80.91%**, respectively with similarity threshold equal to 0.62.

6 CONCLUSION AND FUTURE WORK

The paper discusses representation, extraction, and identification of high-level strokes from printed Gujarati characters. The salient characteristics of HLS features are compactness, high-level shape description, easy to compute shape similarity, and extendability at word-level. The features are tested on the moderately sized symbol level database of printed Gujarati characters with the font, size, style, and ink thickness variations. The experiments were also performed for shape-similarity based word-matching on a small Gujarati word group database, and the initial results are encouraging. In future, more extensive experiments can be carried out on large word image

database. Also, the HLS features can be combined with other features based on the detailed analysis of error and misclassification to improve the retrieval results.

REFERENCES

Antani, S. and Agnihotri, L. (1999). Gujarati character recognition. In *Proc. of the 5th Int. Conf. on Document Analysisand Recognition (ICDAR'99)*, pages 418–421.

Aparna, K. and Ramakrishnan, A. (2002). A complete tamil optical character recognition system. In Lopresti, D., Hu, J., and Kashi, R., editors, *Document Analysis Systems V*, pages 53–57. Springer Berlin / Heidelberg.

Bhardwaj, A., Damien, J., and Govindaraju, V. (2008). Script independent word spotting in multilingual documents. In *Proc. of 2nd Int. Workshop on Cross Lingual Information Access*, pages 48–54.

Charles, S. and McCallum, A. (2011). Introduction to conditional random fields. *Foundation and Trends in Machine Learning*, 4(4):267–373.

Chaudhuri, B. and Pal, U. (1998). A complete printed bangla ocr system. *Pattern Recognition*, 31(5):531–549.

Chaudhuri, B., Pal, U., and Mitra, M. (2001). Automatic recognition of printed oriya script. In *Proc. of the 6th Int. Conf. on Document Analysis and Recognition (ICDAR'01)*, pages 795–799. IEE.

Chaudhury, S., Sethi, G., Vyas, A., and Harit, G. (2003). Devising interactive access techniques for indian language document images. In *Proc. of the Int. Conf. on Document Analysis and Recognition (ICDAR)*, pages 885–889.

Dholakia, J., Yajnik, A., and Negi, A. (2007). Wavelet feature based confusion character sets for gujarati script. In *Proc. of the Int. Conf. on Computational Intelligence and Multimedia Applications*, pages 366–370.

Doermann, D. (1998). The indexing and retrieval of document images: A survey. *Computer Vision and Image Understanding*, 70(3):287–298.

Goswami, M. and Mitra, S. K. (2015). Classification of printed gujarati characters using low-level stroke features. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 15(4):25:1–26.

Goswami, M., Prajapati, H., and Dabhi, V. (2011). Classification of printed gujarati characters using som based k-nearest neighbor classifier. In *Proc. of the Int. Conf. on Image Information Processing*, pages 1–5. IEEE.

Hassan, E., Chaudhury, S., and Gopal, M. (2009). Shape descriptor based document image indexing and symbol recognition. In *Proc. of the 10th Int. Conf. on Document Analysis and Recognition (ICDAR'09)*, pages 206–210.

Hassan, E., Chaudhury, S., and Gopal, M. (2014). Feature combination for binary pattern classification. *International Journal of Document Analysis and Recognition (IJDAR)*, 17(4):375–392.

Jawahar, C., Kumar, P., and Kiran, S. (2003). A bilingual ocr for hindi-telugu documents and its applications. In *Proc. of the 7th Int. Conf. on Document Analysis and Recognition (ICDAR'03)*, pages 408–412.

Jawahar, C. V., Balasubramanian, A., and M., M. (2004). Word-level access to document image datasets. In *Proceedings of the workshop on computer vision, graphics and image processing*.

Kompalli, S., Setlur, S., and Govindaraju, V. (2005). Challenges in ocr of devanagari documents. In *Proc. of the 8th Int. Conf. on Document Analysis and Recognition (ICDAR'05)*, pages 1–5. IEEE.

Kumar, A., Jawahar, C., and Manmatha, R. (2007). Efficient search in document image collections. In Yagi, Y., editor, *ACCV:LNCS*, volume 1 of *4843*, pages 586–595. Springer-Verlag Berlin / Heidelberg.

Lakshmi, C. and Patvardhan, C. (2002). A multi-font ocr system for printed telugu text. In *Proc. of the Langauge Engineering Conference*, pages 7–17.

Lehal, G. and Singh, C. (2000). A gurmukhi script recognition system. In *Proc. of the 15th Int. Conf. on Pattern Recognition (ICPR'00)*, pages 557–560.

Meshesha, M. and Jawahar, C. (2008). Matching of word image for content-based retrieval from printed document images. *International Journal of Document Analysis and Recognition (IJDAR)*, 11(1):29–38.

Murphy, K. (2012). *Machine Learning: A Probabilistic Perspective*. The MIT Press, Cambridge, Massachusetts London, England.

Needleman, S. B. and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453.

Rath, T. and Manmatha, R. (2003). Word image matching using dynamic time wrapping. In *Proc. of the Int. Conf. on Computer Vision and Pattern Recognition (ICVRP)*, volume 2, pages 521–527.

Srihari, S., Srinivasan, H., Huang, C., and Shetty, S. (2006). Spotting words in latin, devanagari and arabic scripts. *Vivek*, 16(3):2–9.

Suthar, S., Goswami, M., and Thakkar, A. (2014). Empirical study of thinning algorithms on printed gujarati characters and handwritten numerals. In Meenakshi, N., editor, *Proc. of the 2nd Int. Conf. on Emerging Research in Computing, Information, Communication, and Applications (ERCICA'14)*, volume 2, pages 104–110. ELSEVIER.

Tarafdar, A., Mondal, R., Pal, S., Pal, U., and Kimura, F. (2010). Shape code based word-image matching for retrieval of indian multi-lingual documents. In *Proc. of the Int. Conf. on Pattern Recognition (ICPR)*, pages 1989–1992.

Yang, M., Kpalma, K., and Ronsin, J. (2008). A survey of shape feature extraction techniques. In Yin, P., editor, *Pattern Recognition*, pages 43–90. IN-TECH.