2013 12th International Conference on Document Analysis and Recognition

Search space reduction for holistic ligature recognition in Urdu Nastalique script

Akram El-Korashy
Department of Computer Science and Engineering
German University in Cairo
Email: akram.elkorashy@gmail.com

Faisal Shafait
School of Computer Science and Software Engineering
The University of Western Australia
Email: faisal.shafait@uwa.edu.au

***Abstract*—This paper addresses the problem of holistic recognition of printed ligatures in Nastalique writing style of the Urdu language. The main difficulty of the recognition process lies in the large number of classes/ligatures (17,000 different possible ligatures in our Urdu text data). This large number of classes not only limits the efficiency (run-time) of the recognition algorithms, but also makes it difficult to use state-of-the-art classifiers – like Random Forests – that can only handle up to a few hundred classes. Nearest neighbor classifiers scale up well to tackle such large-scale classification problems, however their poor run-time efficiency poses a major obstacle. In this paper, we investigate two strategies for improving the efficiency (reducing the search space) of nearest neighbor based classification of Urdu ligatures. The first approach uses spectral hashing to resort to approximate nearest neighbor classification. The second approach is based on the idea of hierarchical classification to partition the search space based on the number of characters in a ligature. Experiments using spectral hashing show that the search space of nearest neighbor comparison can be reduced by about 50% without a loss in recognition accuracy. Further experiments demonstrate that the Random Forest classifier can be reliably used as the first stage classifier to distinguish one-character ligatures from multiple-character ligatures in a hierarchical classification scheme. We hope that the ideas presented in this paper would build the foundations for practical large-scale ligature classification systems not only for Nastalique, but also for other Urdu and Arabic scripts.**

Keywords. Urdu Nastalique, Character recognition, Nearest Neighbor classification

I. Introduction

Nastalique script is the most widespread writing style for Urdu language. It is used to write most of the Urdu books, magazines, and newspapers. Urdu alphabet uses an extended version of the Arabic alphabet. The difficulty of OCR in Urdu (which is mainly written in Nastalique script) as compared to Arabic (which is mainly written in Naskh script) lies in two main aspects. The first one is the difficulty of ligature (also called parts-of-words) segmentation. The second aspect is the bigger number of valid *ligatures* of the Urdu language compared to Arabic languages. Holistic recognition of ligatures aims at solving a classification problem with a large number of classes (each valid ligature representing one class) in a segmentation-free manner.

Most of the recent methods for tackling this problem focus on one of the two main approaches. One approach is to focus on developing an accurate system that works on only a chosen subset of either random or most-used ligatures. This allows the use of different classifiers like Hidden Marov Models as was done by Javed et al. [1]. The second approach is to work on a larger numbers of classes and to use Nearest Neighbor classification followed by more advanced classifiers (like Neural Networks) after detecting a class of the base shape (skeleton) of the ligature as was done in [2]. In this approach, the step of nearest neighbor classification is the bottleneck of performance. Many comparison operations have to be done for each test ligature – one with each of the training set ligature. Even for a small number of training samples per class leads to a large-sized training set. We use a synthetic dataset containing 80,000 training samples representing 17,000 classes in total. The choice of using a synthetic dataset was made due to the possibility of doing controlled experimentation [4].

The main goal of this paper is to discuss two approaches that can be applied for recognition of Urdu ligatures in combination with nearest neighbor classification and to show their effect on performance of the recognition process in terms of efficiency and accuracy. Efficiency is measured in terms of the average number of comparisons done between a single test sample and the training examples. The accuracy is measured based on recognition results of randomly degraded (unclean) test sets of ligatures, independent from the possible errors due to inaccuracy in text-line segmentation. The datasets are adopted from those used to originally train the Arabic and Urdu NabOCR [9] system.

This paper first introduces briefly why segmentation-free recognition is used in most of the Urdu OCR research in Section II. Then, it shows how the use of spectral hashing can reduce the number of comparisons needed for recognizing a given ligature in Section III. Finally, the application of hierarchical classification is investigated to separate the recognition of frequently occurring ligatures from that of other ligatures (Section IV), which can improve both efficiency and accuracy on real Urdu text. The paper is concluding with some ideas for future research direction in Section V.

II. Segmentation-free recognition

Urdu language alphabet has 38 symbols (characters). It is predominantly written in Nastalique script which is cursive in nature, i.e. several characters join together to form a single shape, called a ligature, based on orthographic rules of the script. A ligature by itself can consist of one or more characters. It can be a meaningful word separately, only a part of a meaningful word, or it can be meaningless in Urdu [3]. In this paper, we are mainly concerned with *valid* ligatures,

1520-5363/13 $26.00 © 2013 IEEE
DOI 10.1109/ICDAR.2013.228

i.e. only those ligatures that constitute parts of meaningful words. Decomposing a ligature image back into its constituent characters is a challenging problem due to the complexity and variability of the shape of a joined sequence of characters. This complexity can be attributed to the properties of cursiveness, context sensitivity, non-monotonicity of the direction of joining, and the arbitrariness of the baseline orientation of ligatures in Nastalique script.

A word in Urdu may consist of one or more ligatures (see Figure 1). A ligature can be easily separated from another adjacent ligature by analyzing the connected components in the image and identifying them as separate groups (after doing necessary heuristics that eliminate diacritics or identify the dots) [3].

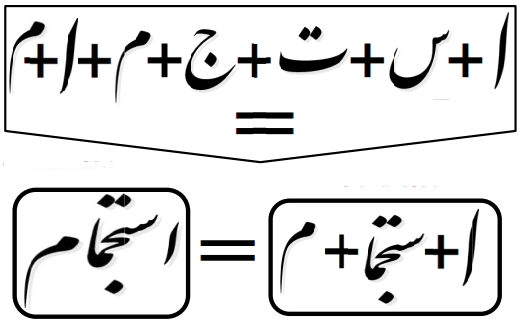

Fig. 1: Individual characters are joined together according to rules for every consecutive pair of characters in order to form groups of characters called ligatures. A word consists of one or more ligatures written next to each other. Holistic recognition uses ligature-level shapes (*lower-right box*) to perform the recognition of Urdu text because it is easy to decompose a word into its ligatures. The difficulty of segmentation into individual characters using image processing techniques is avoided.

III. Approximate Nearest Neighbor by Spectral hashing

Hashing-based search works by trying to find a mapping function of the features vector of a certain data sample into a binary code. The binary codes that are produced by this function on a certain dataset ideally have the property that the Hamming distance between the binary codes of any two data samples is a good relative representative of a distance measure (Euclidean distance, Manhattan distance, ...) between these two features vector (for more detail, please refer to [5] [6]).

A. Experiment settings and Methodology

One-nearest-neighbor (1NN) classifier has been used to recognize a ligature represented by a features vector. In 1NN classification, a test data point is compared against each of the samples in the training data set, and assigned the label of the training sample that has the least distance from it. Spectral hashing is used to achieve a reduction on the number of comparisons to be performed on testing a new data point.

Features vector: We used **shape context** [7] [8] as the basic feature extraction scheme. This technique extracts features from an image by approximating the shape of the ligature into a contour of points, then by calculating histograms from the contour that represent the distance and orientation values with respect to each point on the contour and calculating the sum of these histograms. A modified version of the shape context features developed by Sabbour and Shafait [9] was employed. The modification involves dividing the image into four quadrants and computing shape context feature from each quadrant independently. This technique helps in keeping spatial information to a certain extent about the relative distribution of contour points. Contour extraction is done using a technique proposed by Hassan et al. [8]. Contour points are defined as the transition points (horizontally or vertically) in the binary image. The contour of the ligature is used to calculate the features vector. The features vector is calculated as four shape context histograms representing the four quadrants of the ligature image. The following step are followed:

- The ligature contour image is divided into four regions (four quarters).
- For each point on the contour shape, a shape context histogram is calculated taking into consideration only the points that lie in the same region as the point for which the histogram is calculated. This histogram represents the distribution of the points in the region in terms of the distance to the inspected point, and the orientation (angle) with respect to the inspected point (polar coordinates with respect to the inspected point).

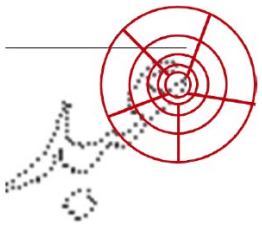

Fig. 2: A histogram is calculated for every point. It represents the number of points that lie in each bin of distance and orientation ranges. In this example, the point considered is taken from the upper right quadrant of the image. The histogram for a point using this configuration represents the counts of points in the different 20 bins (5 orientation bins $\times$ 4 distance bins) [9].

  The histogram uses logarithmically quantized levels of distance, in combination with division into equal sectors around the inspected point (representing the different orientation levels) to divide the region into location bins with respect to the inspected point (Figure 2).
- For each of the four regions, the sums of corresponding shape context histograms of the points lying in the region are calculated to get a histogram for each region.
- The ligature can then be described by the concatenation of the histograms of the four regions.

Training data: The training data set used for this experiment contains the whole set of ligatures in the Urdu language. The size of the training dataset was about 80,000 samples. Figure 3 shows an example of the different samples used to represent one class of ligatures in the training dataset. Each ligature is assigned a degradation level [12] at random. It contains for each possible ligature (class) between 4 and 50 samples, depending on the frequency of occurrence of each ligature in real Urdu texts. The number of samples is determined in proportion to the number of occurrences in text ground-truth data for scanned Urdu books.

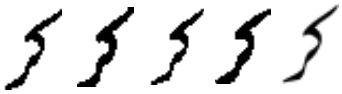

Fig. 3: An example of the different degradation levels that represent different samples from the training dataset of the same ligature shape (same ligature class).

Hashing parameters: The performance of all hashing techniques used for approximate nearest neighbor strongly depends on the number of bits (*nbits*) and the number of tables (*ntables*). Using more bits to represent the binary code (which is used as the hash key) leads to a greedier data reduction (smaller number of results on querying the hash structure) and less recall. Using more tables leads to less reduction and more recall. The number of tolerance bits, *ntolerance* is the maximum number of bits of the binary code that are allowed to be different between the test data point and any of the training samples returned in the result list. In our experiment, different combinations of *nbits*, *ntables*, and *ntolerance* were used to train on the full training dataset described above. The values of *nbits* $\in \{1, ..., 16\}$, values of *ntables* $\in \{1, ..., 16\}$, and values of *ntolerance* $\in \{1, 2, 3\}$ were used to form the combinations.

B. Experiment Results

1NN classification performed on the same training and test data sets without any approximation (testing by comparing a test vector against the whole training dataset) results in an accuracy of about 81.6%. The **percentage of comparisons**, described in the results of the experiments, represents the average number of matching data points returned upon querying the hash structure. This number is the complementary of reduction, i.e., reduction = 100% - (percentage of comparisons). The test dataset consists of approximately 17,000 test data points, one for each ligature class. Each one is a random degradation of one of the ligature classes. Figure 4 shows the effect of changing the number of tables on a certain setting of *nbits* = *7* and *ntolerance* = *2*. Note that this experiment does not reflect the true performance of the ligature based recognition scheme, since in real Urdu text, the frequency distribution of ligatures is highly skewed. However, we chose to pick up this experimental setting to be able to investigate the discriminating capabilities of the feature vector in combination with spectral hashing for describing 17,000 different shapes.

Figure 6 shows the effect of changing the number of bits that represent the binary code on the accuracy of recognition and on the percentage of training samples returned on average

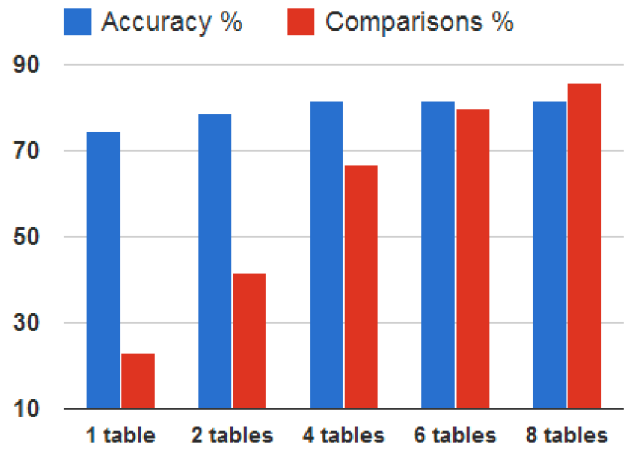

Fig. 4: The effect of the number of tables used for Spectral hashing on the accuracy and the percentage of training examples with which comparisons are done. The accuracy of recognition increases with increasing the number of tables and the average number of comparisons that have to be done to recognize a test data point also increases which means less reduction.

on each query. This configuration has the fixed values (*ntables* = *1, ntolerance* = *2*). Figure 5 shows three examples of misclassified samples that happened due to the increased number of bits of the spectral hashing classifier.

Fig. 5: Three pairs of ligatures representing misclassification errors due to the increased number of bits (from 3 to 8 bits). The top row represents the actual test data points which were correctly classified using a 3-bit spectral hashing structure. The bottom row shows the corresponding results of the 8-bit classifier representing misclassification errors due to increasing the number of bits. For the leftmost error, a three-character ligature was misclassified as the two-character ligature shown below it. In the middle, the correct ligature consists of three characters, and the misclassification is four-character long. On the right, both ligatures are three-character ligatures.

Best reduction-accuracy compromise: There were some significant results that achieve high accuracy and high reduction. They are shown in table I which lists for each accuracy level the best reduction achieved and the corresponding spectral hashing parameters used to achieve it.

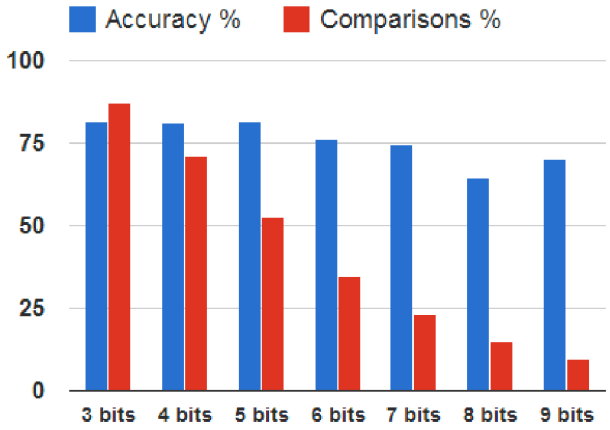

Fig. 6: The effect of number of bits, used to represent the binary code, on the accuracy and the percentage of training examples returned, on average, on each query. The accuracy of recognition, in general, decreases with increasing the number of bits and the average number of comparisons that have to be done to recognize a test data point also decreases which means better reduction.

Accuracy	Comparisons in knn	*(nbits, ntables, ntolerance)*
81.5%	**37538 (47.2%)**	**7, 9, 1**
81%	31553 (39.7%)	7, 7, 1
80.5%	23975 (30.1%)	8, 9, 1
79.5%	29736 (26.1%)	7, 4, 1
78%	**18737 (23.6%)**	**8, 7, 1**
76%	15392(19.4%)	7, 3, 1

TABLE I: Spectral hashing – Best Results achieved: high accuracy and high reduction on the number of comparisons. This table shows, for each of the accuracy levels, the best reduction achieved.

IV. Hierarchical classification using a Random Forest

An experiment was carried out to study the possibility of applying the concept of hierarchical classification using supervised learning given the ground-truth information about the number of characters a training data point contains. This information can be used in order to partition the search space (for KNN classifier) which can allow using different classifiers that suite each group of ligatures independently by training on datasets with ligatures that consist of the same number of characters. Random Forest classifier [10] was chosen as the first-stage classifier due to its good performance and built-in feature selection [11].

A. Experiment settings

Different ways of partitioning the ligatures in the training dataset based on the number of characters were evaluated. Better reduction will be achieved if there are more classes representing the number of characters. However, this will compromise the accuracy of recognition because the accuracy of the Random Forest to distinguish between a four-character ligature and a five-character ligature for example would be very low due to the high similarity between the shapes. The experiments carried out involved various possible partitions that cluster together ligatures consisting of a higher number of characters into one class.

Features vector: The features vector used included the description provided by the shape context method. Some other features have been added to the features vector. Some of them describe the size and locations of dots by analyzing the size and locations of smaller connected components in the ligature shape and associating them to locations on a 3×3 grid. There are also some size features of the ligature image like width, height, and aspect ratio that have been used.

Training dataset: The training dataset used was a reduced dataset which contained only 25% of the number of samples that represent each ligature class in the previous experiment. This 25% was taken at random from the randomly degraded samples of each ligature category. The size of the training data set used is about 20,000 samples. The test dataset consisted of about 18,000 samples representing all the ligature classes with more weight for the one-character ligatures. All the samples are randomly degraded as well.

Random Forest classifier settings: The number of decision trees used to build the Random Forest is 101 trees. Building the decision tree was done using 70% of the features at each node split. The ratio of the training samples used to train each decision tree (bagging percentage) was 70%. The other 30% is used to test the decision tree in order to calculate measure for the test error of the whole Random Forest.

B. Experiment results

Labeling the ligatures into four different labels representing the number of characters (1 to 3, and 4 or more characters) lead to an accuracy of only 78.7%. Table II shows the frequency of misclassified pairs of classes for this particular partitioning. Further subdividing the ligature classes into 5, 6, and 7 classes resulted in a significant drop in accuracy (45.4%, 20.7%, and 16.7% respectively). The reason for the substantial decrease in accuracy of distinguishing the number of characters in large ligature is due to the very similar structure of ligatures containing four or more characters. Thus, grouping all of these classes together in one class representing ligatures consisting of three or more ligatures would lead to a higher accuracy. This can still be useful in achieving reduction because one-character and two-character ligatures are the two most frequent classes of ligatures and their frequency in the Urdu language is much higher than ligatures consisting of a higher numbers of characters.

Labeling the ligatures into three different labels representing the number of characters (one, two, and three-or-more characters) lead to an accuracy of 96.4%. This result is quite significant especially if we take a look at the confusion matrix of this classifier (see Table III) which shows that there is a high confidence (98%) in the result of a ligature being a one-character ligature. This confidence can be useful in improving the recognition accuracy of one-character ligatures. It can also inherently improve the efficiency of the recognition on real Urdu text data.

Actual label/Result	1	2	3	4+	Recall
1	**1139**	91	0	3	**92.4%**
2	297	**86**	20	238	**13.4%**
3	590	3	**171**	2659	**4.9%**
4+	41	0	5	**13213**	**99.6%**
%true positives	**55.1%**	**47.8%**	**87.2%**	**82.0%**	-

TABLE II: Confusion matrix of the Random Forest classifier using four categories representing the one-character, two-character, three-character, and four-or-more-character ligatures. Overall accuracy is 78.7%.

Actual label/Result	1	2	3+	Recall
1	**1131**	88	14	**91.9%**
2	16	**94**	531	**17.2%**
3+	7	2	**16627**	**99.9%**
%true positives	**98%**	**51%**	**96.8%**	-

TABLE III: Confusion matrix of the Random Forest classifier using three categories representing the one-character, two-character and three-or-more-character ligatures. High confidence is observed when the classifier result is class 1.

Improved efficiency is expected because one-character ligatures can be recognized with an accuracy of 98.74% by using a 1NN classifier on a training dataset of about 1000 samples. This means that less than 2% of the comparisons (1000 vs. 80,000) will be done by using the 1NN classifier of one-character ligatures compared to the 1NN classifier of the whole set of ligatures. Interesting statistical results about the Urdu language show that almost 30% of the ligatures in a large text corpus of Urdu language are 1-character ligatures. So, upon recognition on real Urdu text data, significant performance improvement can be achieved. Figure 7 shows one of the test data points that was misclassified. This specific misclassification error is one that seems to be mainly caused by the weakness of the features in distinguishing the shapes that are not very complex. Trying to distinguish the kind of misclassification errors in Figure 7 (even by using a specialized classifier that would be consulted if the Random Forest outputs class 1) might potentially increase the confidence of the Random Forest on the one-character class.

Fig. 7: An example of a misclassification error where the three-character ligature shown in the shape (inside the box) is classified by the Random Forest as a one-character ligature.

V. Conclusion and Outlook

This paper discussed two aspects of the segmentation-free recognition process of Nastalique text for Urdu language.

The first aspect is to improve the efficiency of the recognition process of individual ligatures by performing approximate nearest neighbor classification instead of 1NN on the whole training dataset of Urdu ligatures. Several experiments using the technique of Spectral Hashing were carried out. Results are measured in terms of the ratio of comparisons (as a percentage of the full training dataset size) and the accuracy of recognition. Zero loss in accuracy (accuracy of 81.5% which is the same as that achieved by 1NN) was achieved when spectral hashing was used to reduce the number of comparisons performed by the 1NN classifier down to 47.2% of the comparisons done when the whole training dataset was involved.

The second aspect that was targeted by the experiments is to build a new structure of classification that can help achieve a reduction on the number of comparisons and also try to improve the accuracy of certain subsets of the classes. The property that was used to build a hierarchy of classifiers is the number of characters in the ligature because it is easy to have ground-truth information that label the ligature with the number of characters it is composed of. Several partitioning values were used and experimented. The Random Forest classifier was used to identify the number of characters in a ligature. One partitioning that achieved high accuracy was to differentiate between one-character, two-character and three-or-more-character ligatures. A true positive rate of 98% was achieved for identifying single character ligatures.

This work has demonstrated two useful strategies for search space reduction in holistic ligature recognition. Our future work will focus on an integrated approach that not only combines these two ideas into a single framework, but also explores other strategies for further partitioning larger ligatures, for instance using base ligature shape only.

References

[1] S. T. Javed, S. Hussain, A. Maqbool, S. Asloob, S. Jamil and H. Moin, *Segmentation free Nastalique Urdu OCR*. World Academy of Science, Engineering and Technology 46, 2010.

[2] S.A. Husain, *A multi-tier holistic approach for Urdu Nastaliq recognition*. Proc. IEEE Multi-Topic Conference, (INMIC), 2002.

[3] M.J. Rizvi, *Development of algorithms and computational grammar for Urdu*. Tech. Rep., Pakistan Institute of Engineering and Applied Sciences, Nilore, Pakistan, 2007.

[4] J. V. Frasch, A. Lodwich, F. Shafait, T. M. Breuel, *A Bayes-True Data Generator for Evaluation of Supervised and Unsupervised Learning Methods*. Pattern Recognition Letters, 32(11), 1523-1531, 2011.

[5] S. Marukatat and W. Sinthupinyo, *Improved Spectral Hashing*. Proc. Pacific-Asia Conf. on Knowledge Discovery and Data Mining (160-170), 2011.

[6] Y. Weiss, A. Torralba, and R. Fergus, *Spectral Hashing*. Proc. Advances in Neural Information Processing Systems, 2008.

[7] S. Belongie, J. Malik, and J. Puzicha, *Shape matching and object recognition using shape contexts*. IEEE Trans. on Pattern Analysis and Machine Intelligence, 24(4), 509-522, 2002.

[8] E. Hassan, S. Chaudhury, and M. Gopal, *Shape descriptor based document image indexing and symbol recognition*. Proc. Int. Conf. on Document Analysis and Recognition, 206-210, 2009.

[9] N. Sabbour, and F. Shafait, *A segmentation free approach to Arabic and Urdu OCR*. Proc. Document Recognition and Retrieval XX, 2013.

[10] L. Breiman, *Random Forests*. Machine Learning, 45(1), 5-32, 2001.

[11] M. Reif, F. Shafait, A. Dengel, *Meta-Learning for Evolutionary Parameter Optimization of Classifiers*. Machine Learning, 87(3), 357-380, 2012.

[12] H. S. Baird, *Document Image Defect Models* in H. S. Baird, H. Bunke, and K. Yamamoto (Eds.), Structured Document Image Analysis, Springer-Verlag: New York, 1992.