J. Indian Inst. Sci., May–June 2000, **80**, 215–224.

Handwritten script identification using possibilistic approach for cluster analysis

D. GHOSH AND A. P. SHIVAPRASAD
Department of Electrical Communication Engineering, Indian Institute of Science, Bangalore, India 560 012

Abstract

One very important field of application of pattern recognition is character recognition. But, until now, almost all character-recognition systems make an implicit assumption that the script and/or the language of the document to be processed is known. This may not be true in an international environment. So, script identification plays an important role in automatic processing of document images. In this paper, we present a new script identification algorithm that can identify the same from the characters in a handwritten document. The proposed scheme is based on training via clustering and classification using possibilistic membership grades of a character to all the script classes. This enables the recognizer to work with ambiguous and uncertain cases. The recognizer may reject a pattern when it can not make any clear decision ensuring high reliability. Simulation results confirm the effectiveness of the proposed script-identification technique.

Keywords: Character recognition, script identification algorithm, handwritten script, clustering and classification.

1. Introduction

1.1. *Optical character recognition*

One very important field of application of pattern recognition is character recognition. Character recognition is better known as *optical character recognition* (OCR) since it deals with optically processed characters. Character recognition is an interesting problem in the present-day world because of its many applications.[1] Several techniques for the recognition of both machine- and hand-printed characters had been reported in the past.[1–6] Presently, recognition rates for machine-printed characters can reach over 99%, and so may be considered a solved problem.[5] But, in the case of hand-printed characters, it is yet to achieve the level of reliability required for practical applications. The inherent difficulty in handwritten character recognition is due to large variations among characters of the same class and similarities among characters belonging to different classes. Despite extensive research in this field for over four decades, there is a large gap between human- and machine-reading capabilities for hand-printed characters. This has imposed a challenge to computer scientists and hence there is still a significant amount of ongoing research in the field of handwritten character recognition.

1.2. *Script recognition*

Until now, character-recognition systems used script-specific methodologies. Almost all existing works on OCR make an implicit assumption that the script and/or the language of the document to be processed is known. But, in an international environment, where documents written in many scripts may be presented to an OCR system for processing, prior knowledge of

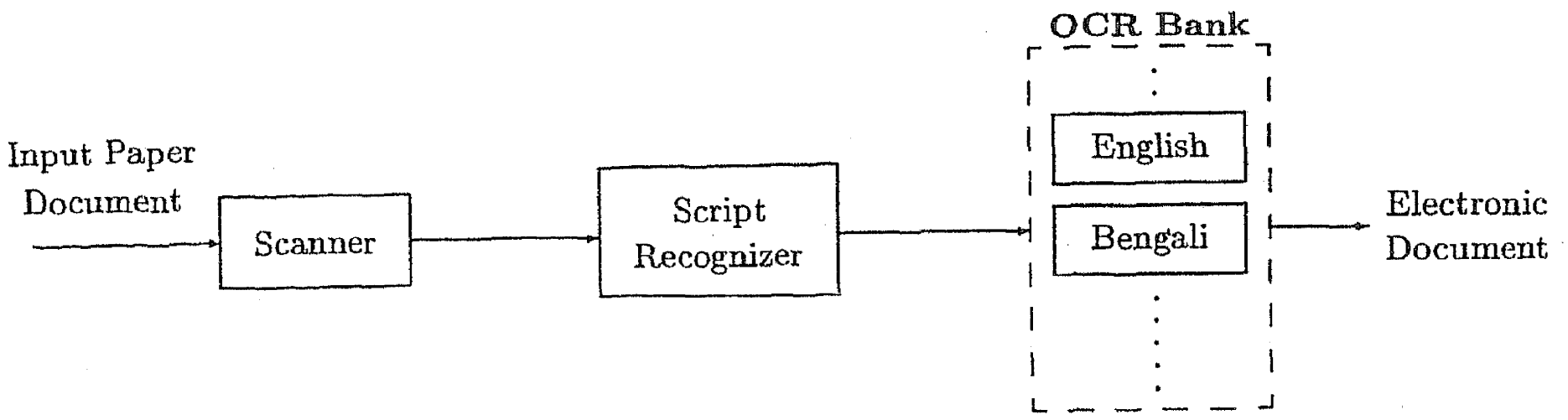


FIG. 1. Character recognition in a multi-script environment.

the script used in the document is essential for choosing an appropriate OCR algorithm (Fig. 1). This plays an important role in routing of facsimile documents as they arrive so that appropriate secondary analysis, which may include OCR, is selected for each document. This may also be used for reading postal codes that may be written in any script and then sorting out the letters appropriately through secondary analysis. Some other applications, such as indexing or translation, depends on identifying the language used in a document. For scripts used by only one language, script identification is a necessary first step for identifying the language of a document. Discrimination between languages using the same script (alphabets and numerals) requires a second level of processing which identifies common usage of the alphabets/numerals. So, in an international environment, script identification plays an important and key part for automatic processing of document images.

Generally, one can deduce a document's script and language from its country of origin, or by examining the document. So, till the early years of this decade, scientists were satisfied with manual recognition of script/language. But, as the world is getting increasingly interconnected, there is a real need for automatic script identification. With increasing demand for document processing in our day-to-day life, people frequently face situations where the volume and variety of scripts/languages make such manual identification unworkable. Also, in a country like India, where a large number of scripts and languages are in use, the country of origin is not at all a useful information to decide the script/language of a document. This has motivated the research on script/language recognition in the field of OCR. Researchers of the OCR community have recently concentrated on developing OCR systems that can identify the script of a document prior to recognition of characters in the document. Such OCR systems are versatile in the sense that they can process documents written in any of the scripts (or languages) within its knowledge. However, research in script identification is in its early stage and hence, not much literature is available in this field.

2. A brief survey of script recognition

Most of the researches on script identification aim at extracting some characteristic features from an input document that distinguish one class of script from others. In general, script identification is based on stroke structure and connections and the fundamentally different writing styles associated with different character sets. So, some initial efforts to identify scripts used in preparing documents were based on connected component analysis. Wood *et al.*[7] offered an algorithm for determining the script of machine-generated documents. The algorithm is based

on a combination of Hough transform approach and standard morphological filtering operations of erosion and dilation. The algorithm given by Hochberg *et al.*[8,9] develops a set of representative symbols (templates) for each script by clustering textual symbols obtained from a set of training documents. 'Textual symbols' include discrete characters in scripts such as Cyrillic, as well as adjoined characters, character fragments, and whole words in connected scripts such as Arabic. Their algorithm discovers frequent characters or word shapes in each script from a set of training documents and develops a set of representative symbols (templates) for each script by clustering these textual symbols. To identify a new document's script, the system compares a subset of symbols from the document to each script's templates, screening out rare or unreliable templates, and choosing the script whose templates provide the best match. A texture-based approach is proposed.[10–12] Since different scripts often have distinctive visual appearances, a block of text in each script may be regarded as a distinct texture pattern. Any standard texture-classification algorithm, hence, may be employed to perform script recognition. A unified syntactic approach to script recognition is presented in Malaviya *et al.*[13] The fuzzy pattern description language FOHDEL is used to store fuzzy feature values in the form of fuzzy rules. These fuzzy rules aid in decision making during classification. Spitz[4] has developed a scheme for automatic language identification on the basis of character density or optical density distribution. Spitz's work has been extended in Deng *et al.*[15] to cope with complex degraded document images and with more languages.

In all the script-identification techniques described above, except the algorithm proposed by Hochberg *et al.*, the document as a whole is treated as the input pattern and the algorithms attempt to extract some distinguishing features from it. The algorithm proposed by Hochberg *et al.* uses cluster-analysis methodology. But the algorithm must include a large number of documents in the training set. Also, these documents must be large in size and need to be used as a whole during the training process. This is required to guarantee the formation of templates for all possible textual symbols in a script. Moreover, the algorithm has no provision for discriminating one script from another that have some textual symbols common between them. This has motivated the development of our new cluster analysis-based script identification scheme.

3. Proposed script identification scheme

3.1. *Training algorithm*

In our script recognition technique, the representative patterns for each class of script is generated from samples belonging to all the different classes of characters used in that particular script, rather than from samples of text documents. The sample patterns are mathematically represented in the form of vectors, and so also the prototype (representative) patterns. The vectors are formed via extraction and selection of some characteristic feature values from the given pattern.[16]

The training patterns (samples) are supplied with known class labels, i.e. both the script class and the character class are known for each training pattern. The recognition system may be trained to develop some classification rules using these training samples. But, samples belonging to the same class of script and character may exhibit different structures because of the variations in the writing styles from one individual to another. Hence, it is necessary to

variations in the writing styles from one individual to another. Hence, it is necessary to identify all the unknown different structures present in the set of samples belonging to a particular class of script and character. This is accomplished through *clustering*. In general, any standard clustering algorithm may be used for partitioning of a data set. Some of these algorithms are available elsewhere.[17–23]

Suppose, there are a total of $\mathcal{K}$ script classes and the number of character classes in the qth script be $\mathcal{K}^{(q)}$. Then, the set of training patterns may be represented as

$$\chi_{trg} = \chi^{(1)} \cup \chi^{(2)} \cup \cdots \cup \chi^{(\mathcal{K})} = \bigcup_{q=1}^{\mathcal{K}} \chi^{(q)} \tag{1-a}$$

where $\chi^{(q)}$ is the set of training patterns belonging to the script class q, given as

$$\chi^{(q)} = \chi^{(q,1)} \cup \chi^{(q,2)} \cup \cdots \cup \chi^{(q,\mathcal{K}(q))} = \bigcup_{\varrho=1}^{\mathcal{K}(q)} \chi^{(q,\varrho)}. \tag{1-b}$$

The set $\chi^{(q,\varrho)}$ represents the set of all training samples corresponding to the ϱ^{th} character class of the qth script. Each of these sets $\chi^{(q,\varrho)}$, $q = 1, 2,\ldots, \mathcal{K}$, $\varrho = 1, 2,\ldots, \mathcal{K}^{(q)}$ is subjected to clustering independent of the other sets. However, the number of optimum clusters that may be formed from one such set is generally not known. Any standard cluster validity functional found in the literature[17, 21, 24, 25] may be used to find the appropriate number of valid clusters that can be formed from a given set of samples and the set is accordingly partitioned into that many clusters. Suppose, the training set $\chi^{(q,\varrho)}$ is partitioned into $\chi_1^{(q,\varrho)}$, $\chi_2^{(q,\varrho)}$,..., $\chi_{C_{opt,q\varrho}}^{(q,\varrho)}$, where $C_{opt,q\varrho}$ is the optimum number of clusters that may be formed from the set $\chi^{(q,\varrho)}$. The centroids of these clusters, viz., $\mathbf{y}_1^{(q,\varrho)}$, $\mathbf{y}_2^{(q,\varrho)}$,..., $\mathbf{y}_{C_{opt,q\varrho}}^{(q,\varrho)}$, form the set of prototype patterns for the ϱth character class in the qth script. Consequently, the set of all the clusters that are generated from the training sets of all the different classes of characters in the qth script, i.e., $\chi_j^{(q,\varrho)}$, $\varrho = 1, 2,\ldots, \mathcal{K}^{(q)}$, $j = 1, 2,\ldots, C_{opt,q_\varrho}$ forms the complete set of clusters for the qth script class and the centroids of all these clusters are representatives (prototypes) of that script class.

3.2. *Classification rules*

The classification of an input document, in our proposed scheme, is based on the evaluation of the possibilities of belongingness of one or more characters, contained in the document, in different script classes. A measure for the possibility of belongingness of a sample in a cluster is the possibilistic membership grade defined in Krishnapuram and Keller.[23] The possibilistic membership grade of a character sample x_n in the jth cluster of the ϱth character class in qth script, i.e. the cluster $\chi_j^{(q,\varrho)}$, is given as

$$\mu_{j,(q\varrho)}(x_n) = \left[1 + \left(\frac{d^2\left(x_n, y_j^{(q,\varrho)}\right)}{\eta_j^{(q,\varrho)}}\right)^{\frac{1}{m_j^{(q,\varrho)}-1}}\right]^{-1} \tag{2}$$

The term $m_j^{(q,\varrho)}$ is the *degree of fuzziness* associated with the cluster $\chi_j^{(q,\varrho)}$. The parameter $\eta_j^{(q,\varrho)}$ is the squared distance from $y_j^{(q,\varrho)}$ at which the membership grade of x_n in cluster $\chi_j^{(q,\varrho)}$ is half, and hence, referred to as the *3dB parameter*. The values of the degrees of fuzziness and the 3dB parameters for all the clusters are suitably chosen during training so as to maximize the number of correctly classified training samples.

Starting from the first character in the input document, and until a clear decision (classification) can be made, the possibilities of belongingness of each character in different script classes are calculated. For the nth character in the document, say x_n, the first step of the classification algorithm involves the evaluation of $\mu_{j,(q,\varrho)}(x_n)$, $q = 1, 2,\ldots, \mathcal{K}$, $\varrho = 1, 2,\ldots, \mathcal{K}^{(q)}$, $j = 1, 2,\ldots, C_{opt,q\varrho}$. Subsequently, a class-membership grade for x_n to each of the $\mathcal{K}$ script classes is assigned. The cluster representing the qth script class is the union of all the clusters obtained for all the character classes in the qth script, i.e.

$$\chi^{(q)} = \bigcup_{\varrho=1}^{\mathcal{K}^{(q)}} \bigcup_{j=1}^{C_{opt,q\varrho}} \chi_j^{(q,\varrho)} \tag{3}$$

In fuzzy set theory, the membership of an element in the set of union of two or more sets is generally taken as the maximum of the memberships of the element in all the component sets.[26, 27] Therefore, the membership grade of x_n to the qth script class is defined as

$$\nu_q(x_n) = \max_{\substack{j=1,2,\ldots,C_{opt,q\varrho} \\ \varrho=1,2,\ldots,\mathcal{K}^{(q)}}} \mu_{j,(q\varrho)}(x_n) \tag{4}$$

This gives the value of confidence of belongingness of the character x_n to class q. The average value of confidence of belongingness of x_n to class q, over all the n characters used so far, is given as

$$\nu_{q,avg} = \frac{1}{n}\sum_{t=1}^{n} \nu_q(x_t). \tag{5}$$

The possibilistic class label-vector that is now assigned to the input document is given as

$$\mathbf{p} = [\nu_{1,avg}, \nu_{2,avg},\ldots,\nu_{\mathcal{K},avg}]. \tag{6}$$

The task of final decision making with regard to script recognition from the label-vector $\mathbf{p}$ is accomplished by defuzzification of the label-vector. One simple defuzzification rule is to classify the input as belonging to the script class to which it has the maximum value of confidence. But, such a defuzzification rule essentially makes our classification system a hard classifier only and makes it incapable of handling any uncertainty and/or ambiguity. In order to make full use of the information available in the label-vector $\mathbf{p}$ and take final decision accordingly, we propose to include some criteria to reject uncertain and ambiguous cases.

Two types of rejections may occur: (1) Membership and (2) Ambiguity rejection.[28] Membership rejection occurs when $\nu_{q,avg}$ for all $q = 1, 2,\ldots, \mathcal{K}$ is very small. This is the case when

the input is highly degraded making it difficult to classify. Ambiguity rejection occurs when at least one class membership (except the highest one) is very close to the highest one. Such a situation occurs when two or more script classes share some similar (or exactly same) characters. Accordingly, we set the criteria for rejecting an input as follows:

1. Reject an input document if the class membership of the input in at least one of the script class does not exceed (or at least equal to) a threshold T_{memb}.
2. Reject an input document if the ratio of the second maximum class membership of the input across all the classes to the first maximum is not below (or at most equal to) a threshold T_{ambg}.

So, in the next step, the first and the second maximum values of class memberships over all the classes are determined, respectively, as

$$\nu_{\max 1} = \max_{q=1,2,\ldots,\mathcal{K}} \nu_{q,avg} \tag{7-a}$$

$$\nu_{\max 2} = \max_{q=1,2,\ldots,\mathcal{K}} \nu_{q,avg}, \quad \nu_{q,avg} \neq \nu_{\max 1} \tag{7-b}$$

Now, if $\nu_{\max 1} \geq T_{memb}$ and $(\nu_{\max 2}/\nu_{\max 1}) \leq T_{ambg}$, the script used in the input document is classified as belonging to the class corresponding to $\nu_{\max 1}$. Else, using the next character in the document, the above steps are repeated to modify the possibilistic class label-vector $\mathbf{p}$ and then check if classification is possible. If the script of the input document cannot be determined even after all the characters in the document are exhausted, the document is rejected by the classifier.

4. Experimental results

In our experiment, we have used numeric characters in four different scripts: Arabic, Hindi, Bengali and Kannada. That means, for each script class there are ten character classes corresponding to the ten numerals 0 to 9. We train our recognition system with 300 samples for each of the character classes. The character samples are generated artificially using our proposed technique for generation of artificial handwritten characters.[29] Each character is in binary format and is within a rectangular frame of size 32×24. The training algorithm described earlier is used to form the set of clusters corresponding to each class of character, and consequently clusters corresponding to each script class.

Table I
Recognition result for documents in different scripts of size four characters

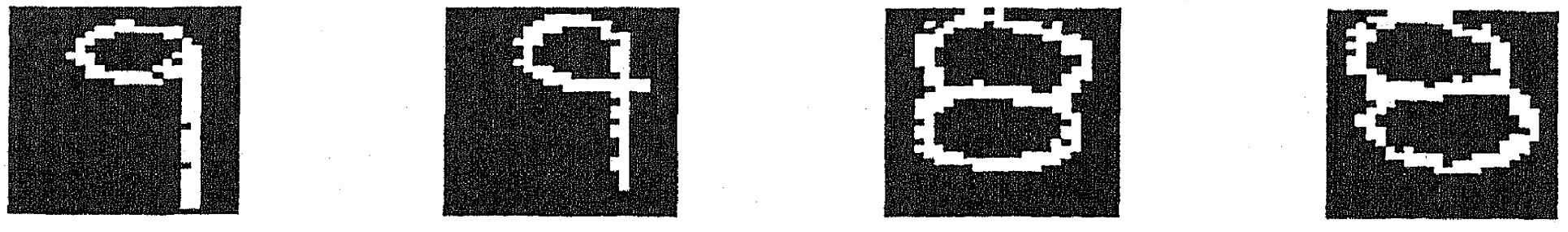
Actual script class	No. of test documents assigned to script class					Acc rate	Err rate	Rej rate
	Arabic	Hindi	Bengali	Kannada	Rejection			
Arabic	47	0	0	0	3	94.0	0.0	6.0
Hindi	0	44	0	1	5	88.0	2.0	10.0
Bengali	0	0	48	0	2	96.0	0.0	4.0
Kannada	0	1	0	48	1	96.0	2.0	2.0
					Total	93.5	1.0	5.5

Table II
Recognition result for documents in different scripts of size six characters

Actual script class	No. of test documents assigned to script class					Acc rate	Err rate	Rej rate
	Arabic	Hindi	Bengali	Kannada	Rejection			
Arabic	49	0	0	0	1	98.0	0.0	2.0
Hindi	0	47	0	1	2	94.0	2.0	4.0
Bengali	0	0	49	0	1	98.0	0.0	2.0
Kannada	0	0	0	48	2	96.0	0.0	4.0
					Total	96.5	0.5	3.0

For testing, we have used small size documents as input that comprise a small set of disconnected characters written in any of the four different scripts, viz., Arabic, Hindi, Bengali and Kannada, but with no mixing of characters belonging to different scripts. The size of each document (i.e. the number of characters in a document) is taken as four. The characters in each of the documents are generated artificially, following our technique for artificial character generation. The format of each character is the same as that of the training patterns, i.e. they are in binary format and are within a rectangular frame of size 32×24. The number of test documents used per script is fifty. The characters in each document are generated at random. The string of characters, thus generated, is fed to the recognizer. The script that may have been used in the input document is identified using our proposed script-identification scheme. The result obtained in our experiment is shown in Table I. In a second set of experiment, the size of the documents is increased from four to six. The number of test samples used per script is the same, i.e 50 samples per script. The recognition result for this is given in Table II.

In the present problem, there exists a large amount of overlapping between clusters. This is due to the similarities among some characters belonging to different script classes and presence of some common characters in different script classes. For example, the Arabic numeral 'nine' and the Bengali numeral 'seven' looks similar [Figs 2a and 2b] while the Arabic 'eight' and the Bengali 'four' are the same [Figs 2c and 2d]. So, there exists the possibility of handling ambiguous cases. A reliable classifier must prefer to reject ambiguous and uncertain patterns, rather than taking wrong decisions. The results shown here demonstrate the reliability of our system. This we conclude from the fact that the rejection rate of our classifier is higher than the error rate. However, the compromise between the rejection rate and the error rate can be controlled by varying the two rejection thresholds, T_{memb} and T_{ambg}. Higher value of T_{memb} and lower value of T_{ambg} increases the rejection rate, while the error rate is reduced. This implies,



(a) (b) (c) (d)

FIG. 2a. Handwritten Arabic 'nine', b. Handwritten Bengali 'seven', c. Handwritten Arabic 'eight', d. Handwritten Bengali 'four'.

higher T_{memb} and lower T_{ambg} is preferable in designing a reliable recognition system. But, for very high T_{memb} and very low T_{ambg}, the recognizer will reject a pattern even for slight amount of ambiguity and/or uncertainty. So, most of the patterns will be rejected by the classifier and hence may not be useful for automating the recognition task. Therefore, these threshold values are to be chosen judiciously. In this study, we have taken the values of these two thresholds as 0.3 and 0.7, respectively.

On the other hand, recognition rate may be improved by increasing the size of the document, i.e. by increasing the number of characters in the given document. This is because of the fact that all the characters in a set are ambiguous is rarely encountered. When one character in the document fails to provide sufficient information due to ambiguity and uncertainty, some other character in the set can accomplish the task efficiently. This we observe by increasing the document size from four to six characters. However, it is to be noted that when some characters belonging to two or more script classes are similar (or exactly same), it is always possible that an input document will be the combination of only all such characters. So, it is theoretically impossible to achieve 100% accuracy. For example, whatever may be the size of the document, it is possible to have all the characters in it (Arabic 'eight' or Bengali 'four'), and so cannot be classified. But, the probability of occurrence of such combinations diminishes as the string length is increased. So, we can expect better performance as the document size is increased.

5. Conclusion

In this paper, an efficient script-identification scheme is introduced. The technique can be used for recognizing the script in which a document is written. So, it finds extensive application in international scenario where it may be required to handle documents written in many different scripts. In our proposed technique for script identification, we have used possibilistic approach to classification to make the recognizer work with ambiguous and uncertain inputs. Possibilistic membership function measures the typicality and thus the recognizer can compute the extent of belongingness of a character in the document to a particular script class. For each character in the document, starting from the first, the recognizer assigns some degree of confidence as to the possibility of it belonging to a particular script class. Classification is based on the aggregation of successive confidence values. The recognizer may reject a pattern when it cannot make any clear decision. Simulation results show that the proposed script-identification technique has a very high performance.

The main advantage of the proposed scheme is that it can be used for a small size document that contains only a few characters, whereas all the earlier proposed script-identification methods can only be used on relatively large documents. One more interesting advantage of our script-identification system, over all the existing script-identification techniques, is that it does not require a separate hardware or software for character recognition. For every character in the document, the recognizer computes the membership grades of the input to all the clusters. We can make use of these membership grades not only to assign confidence values as to the belongingness to the script classes, but also to find the possibilities of the input to belong to different character classes. The main disadvantage of the scheme is that the method requires extensive computation as compared to other methods. Also, the recognizer needs to have a

large memory to store information related to all characters in all the different scripts it is supposed to handle. However, the scheme proves to be more reliable than any other script-identification technique.

In this work, we have assumed that the characters in the document are all discrete and are not connected to each other. But, in real-world situation, we generally encounter cursive unconstrained characters. The characters in the document will, in general, be connected to each other and so a reliable technique for segmenting them into discrete components is required. One more potential problem for future investigation is script identification in a document where the character set may be mixed with some characters belonging to a different script. For example, mixing of Roman alphabets in Tamil or Telugu text is very common. For such a document, we expect our recognizer to output Tamil (or Telugu), rather than Roman. But, our proposed scheme will output Roman if the first character in the document is Roman. Hence, our algorithm demands suitable modification so that the recognizer does not take any wrong decision.

References

1. GOVINDAN, V. K. AND SHIVAPRASAD, A. P. Character recognition—A review, *Pattern recognition*, 1990, **23**, 671–683.
2. SUEN, C. Y., BERTHOLD, M. AND MORI, S. Automatic recognition of handprinted characters – The state of the art, *Proc. IEEE*, 1980, **68**, 469–487.
3. MORI, S., SUEN, C. Y. AND YAMAMOTO, K. Historical review of OCR research and development, *Proc. IEEE*, 1992, **80**, 1029–1058.
4. SUEN, C. Y., NADAL, C., LEGAULT, R., MAI, T. A. AND LAM, L. Computer recognition of unconstrained handwritten numerals, *Proc. IEEE*, 1992, **80**, 1162–1180.
5. SRIHARI, S. N. AND LAM, S. W. Character recognition, Technical Report, Center of Excellence for Document Analysis and Recognition (CEDAR), no. CEDAR-TR-95-1.
6. REVOW, M., WILLIAMS, C. K. I. AND HINTON, G. E. Using generative models for handwritten digit recognition, *IEEE Trans.*, 1996, **PAMI-18**, 592–606.
7. WOOD, S. L., YAO, X., KRISHNAMURTHI, K. AND DANG, L. Language identification for printed text independent of segmentation, *Proc. Int. Conf. On Image Processing (ICIP'95)*, October 23–26, 1995, Vol. III, pp. 428–431.
8. HOCHBERG, J., KERNS, L., KELLY, P. AND THOMAS, T. Automatic script identification from images using cluster-based templates, *Proc. Third Int. Conf. on Document Analysis and Recognition (ICDAR'95)*, August 14–16, 1995, vol. 1, pp. 378–381.
9. HOCHBERG, J., KELLY, P., THOMAS, T. AND KERNS, L. Automatic script identification from document images using cluster-based templates, *IEEE Trans.*, 1997, **PAMI-19**, 176–181.
10. PEAKE, G. S. AND TAN, T. N. Script and language identification from document images, *Proc. Workshop on Document Image Analysis (DIA'97)*, 20 June, 1997, pp. 10–17.
11. TAN, T. N. Written language recognition based on texture analysis, *Proc. Int. Conf. on Image Processing (ICIP'96)*, 16–19 September, 1996, Vol. 2, pp. 185–188.

12. TAN, T. N. Rotation invariant texture features and their use in automatic script identification, *IEEE Trans.*, 1998, **PAMI-20**, 751–756.

13. MALAVIYA, A., LEJA, C. AND PETERS, L. Multi-script handwriting recognition with FOHDEL, *Biennial Conf. North American Fuzzy Information Processing Society (NAFIPS)*, 19–22 June, 1996, pp. 147–151.

14. SPITZ, A. L. Determination of the script and language content of document images, *IEEE Trans.*, 1997, **PAMI-19**, 235–245.

15. DING, J., LAM, L. AND SUEN, C. Y. Classification of oriental and European scripts by using characteristic features, *Proc. Fourth Int. Conf. on Document Analysis and Recognition (ICDAR'97)*, August 18–20, 1997, Vol. 2, pp. 1023–1027.

16. TRIER, O. D., JAIN, A. K. AND TAXT, T. Feature extraction methods for character recognition—A survey, *Pattern recognition*, 1996, **29**, 641–662.

17. DUDA, R. O. AND HART, P. E. *Pattern classification and scene analysis*, Wiley, 1973.

18. TOU, J. T. AND GONZALEZ, R. C. Pattern recognition principles, Addison-Wesley, 1974.

19. FU, K. S. (ed) *Digital pattern recognition*, Springer-Verlag, 1980.

20. BEZDEK, J. C., EHRLICH, R. AND FULL, W. FCM: The fuzzy c-means clustering algorithm, *Computers Geosci.*, 1984, **10**, 191–203.

21. JAIN, A. K. AND DUBES, R. C. *Algorithms for clustering data*, Prentice-Hall, 1988.

22. SCHALKOFF, R. J. Pattern recognition: Statistical, structural and neural approaches, Wiley, 1992.

23. KRISHNAPURAM, R. AND KELLER, J. M. A possibilistic approach to clustering, *IEEE Trans. Fuzzy Systems*, 1993, **1**, 98–110.

24. BEZDEK, J. C. *Pattern recognition with fuzzy objective function algorithms*, Plenum, 1981.

25. PAL, N. R. AND BEZDEK, J. C. On cluster validity for the fuzzy *c*-means model, *IEEE Trans. Fuzzy Systems*, 1995, **3**, 370–379.

26. KLIR, G. J. AND FOLGER, T. A. *Uncertainty and information*, Prentice Hall, 1988.

27. ZIMMERMANN, H. J. *Fuzzy set theory and its applications*, Kluwer, 1991.

28. FRELICOT, C. Multi prototype-based fuzzy classification and reject options, *Proc. Fifth IEEE Int. Conf. on Fuzzy Systems (FUZZ-IEEE'96)*, September 8–11, 1990, Vol. III, pp. 2026–2031.

29. GHOSH, D. AND SHIVAPRASAD, A. P. An analytic approach for generation of artificial hand-printed character database from given generative models, *Pattern Recognition*, 1999, **32**, 907–920.