Devanagari Isolated Character Recognition by using Statistical features

(Foreground Pixels Distribution, Zone Density and Background Directional Distribution feature and SVM Classifier)

Mahesh Jangid
Department of Computer Science & Engineering
Dr. B R Ambedkar National Institute of Technology
Jalandhar, India

***Abstract*— In this paper, we present a methodology for off-line Isolated handwritten Devanagari character recognition. The proposed methodology relies on a three feature extraction techniques. The first technique is based on recursive subdivisions of the character image so that the resulting sub-images at each iteration have balanced (approximately equal) numbers of foreground pixels, as far as this is possible. Second technique is based on the zone density of the pixel and third is based on the directional distribution of neighboring background pixels to foreground pixels. The 314 sized feature vector is form from the three feature extraction techniques for a handwritten Devanagari character. The dataset (12240 samples) of handwritten Devanagari Character, have been prepared by writing the different – 2 people who belongs to different age group and obtained the 94.89 % recognition accuracy.**

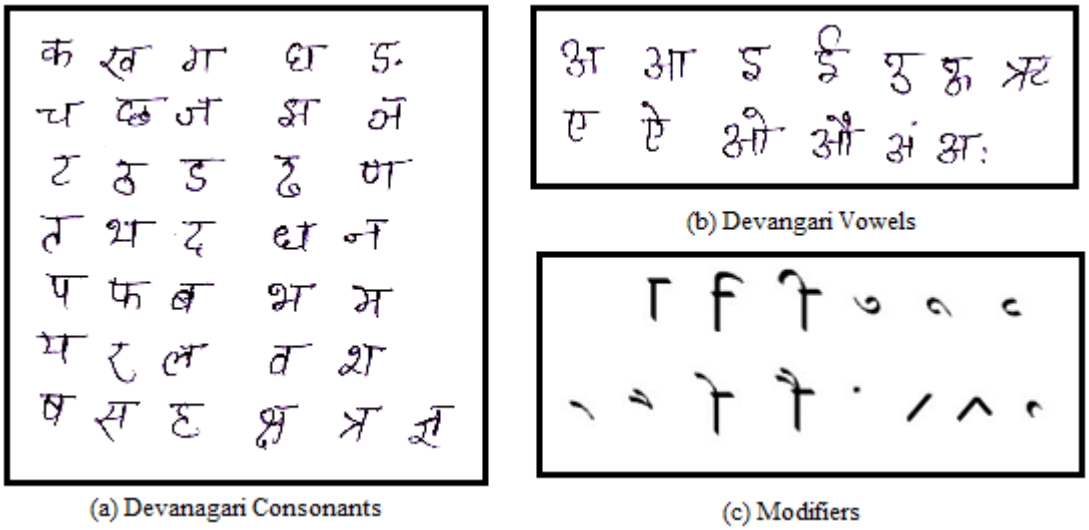
Keywords- Davanagari Character Recognition, Forground pixel, Zone density, Background directional distribution, Support Vector Machine

I. INTRODUCTION

Machine simulation of human reading has become a topic of serious research since the introduction of digital computers. The main reason for such an effort was not only the challenges in simulating human reading but also the possibility of efficient applications in which the data present on paper documents has to be transferred into machine-readable format. Automatic recognition of printed and handwritten information present on documents like cheques, envelopes, forms, and other manuscripts has a variety of practical and commercial applications in banks, post offices, libraries, and publishing houses. Optical Character Recognition (OCR) is a field of research in pattern recognition, artificial intelligence and machine vision. OCR is a mechanism to convert machine printed or handwritten document file into editable text format. This field is broadly divided into two parts, Online and offline character recognition. Off-line Character recognition further divided into two parts, machine printed and handwritten character recognition. In handwritten Character Recognition, there are lots of problems as compare to machine printed document because of the different peoples have different writing styles, the size of pen-tip and some people have skewness in their writing. All this challenges make the researches to solve the problems.

India is a multi-lingual multi-script country and there are twenty two languages. Eleven scripts are used to write these languages and Devnagari Script is an oldest one that is used to write many languages such as Hindi, Nepali, Marathi, Sindhi and Sanskrit where Hindi is the third most popular language in the world and it is the national language of the India [1]. 300 million people use the Devnagari Script for documentation in central and northern parts of India [2].

The script has a complex composition of its constituent symbols. Devanagari script (Hindi) has 13 vowels and 36 consonants shown in the figure 1. They are called basic characters. Vowels can be written as independent letters, or by using a variety of diacritical marks which are written above, below, before or after the consonant they belong to. When vowels are written in this way they are known as modifiers and the characters so formed are called conjuncts. Sometimes two or more consonants can combine and take new shapes. These new shape clusters are known as compound characters. All the characters have a horizontal line at the upper part, known as Shirorekha or headline. No English character has such characteristic and so it can be taken as a distinguishable feature to extract English from these scripts. In continuous handwriting, from left to right direction, the Shirorekha of one character joins with the Shirorekha of the previous or next of the same word. In this fashion, multiple characters and modified shapes in a word appear as a single connected component joined through the common Shirorekha. All the characters and modified shapes in a word. Also in Devanagari there are vowels, consonants, vowel modifiers and component characters, numerals. Moreover, there are many similar shaped characters. All these variations make the handwritten character recognition, a challenging problem.



(a) Devanagari Consonants

(b) Devangari Vowels

(c) Modifiers

Figure 1: Devanagari Isolated Handwritten Characters, Modifiers

II. RELATED WORK

In literature survey, there are many researchers have done work towards the off-line handwritten Devanagari character recognition. The first research work report on handwritten Devanagari characters was published in 1977. After that researchers continuously doing work on the recognition and try to solve the problem associated with handwritten Devanagari characters. Features used by Sharma et al. [3] for handwritten Devanagari characters are obtained from the directional chain code information of the contour points of the characters. The bounding box of a character is segmented into blocks and a CH (Chain code histogram) is computed in each of the blocks. Based on the CH, they have used 64-D features for recognition. Sharma et al. proposed a quadratic classifier-based scheme for the recognition of handwritten characters and obtained 80.36 % accuracy with the 11,270 dataset size. The work reported in [4] discusses the use of regular expressions (RE) in handwritten Devanagari character recognition, where a hand-written character is converted into an encoded string based on chain-code features. Then, RE of stored templates is matched with it. Rejected samples are then sent to a MED (minimum edit distance) classifier for recognition. On the 5000 samples, this work has been done and 82 % accuracy has been reported. The distortions in handwritten Devanagari characters are removed in [5] using a thickening process followed by thinning and pruning operations. The features are represented using normalized vector distances for each character. The Shirorekha and spine in a handwritten character are detected using a differential-distance-based technique.50000 samples is used and 89.12 % accuracy has obtained. The recognition of handwritten characters in [6] is based on the modified exponential membership function fitted to the fuzzy sets derived from the features of the characters. A Reuse Policy that provides guidance from the past policies is also utilized in the paper to improve the speed of the learning process and obtained 90.65 % accuracy. For the recognition of handwritten Devanagari non compound characters, shadow features, and CH features are computed in [7]. Two MLPs and a minimum edit distance (MED) method are used for classification of handwritten Devanagari non compound characters in [61]. In the first stage of classification, characters with distinct shapes are classified using two MLPs. Shadow features are used for one MLP and CH features are used for the other MLP for classification. In the second stage of classification, confused characters having similar shapes are classified using a MED method. This method makes use of corners detected in a character image using modified Harris corner detection technique. Kumar [8] compared performances of five feature-extraction methods on handwritten characters. The various features covered are Kirsch directional edges, distance transform, chain code, gradient and directional distance distribution. From the experimentations, it is found that Kirsch directional edges are least performing and gradient is best performing with SVM classifiers. With multilayer perceptron (MLP), the performance of gradient and directional distance distribution is almost same. The chain-code-based feature is better as compared to Kirsch directional edges and distance transform. A new feature is also proposed in the paper, where the gradient direction is quantized into four-directional levels and each gradient map is divided into 4 × 4 regions. This is combined with total distances in four directions and neighborhood pixels weight. The features used by Pal et al. [9] for handwritten characters are mainly based on directional information obtained from the arc tangent of the gradient and Gaussian filter. A modified quadratic classifier is applied on the features of handwritten characters for recognition. Elastic matching (EM) technique based on an Eigen deformation (ED) for recognition of handwritten Devanagari characters is proposed in [10]. In [11], two classifiers are combined to get higher accuracy of character recognition with the gradient features. Combined use of SVM and MQDF is applied for the same. Many approaches have been proposed toward

handwritten Devanagari numeral, character, and word recognition in the past decade [12]. A comparative study of Devanagari hand-written character recognition using 12 different classifiers and four sets of features is presented. Feature sets used in the classifiers are computed based on curvature and gradient information obtained from binary as well as gray-scale images.

.

III. PROPOSED SYSTEM

A. Dataset prepartion & preprocessing

There is no standard dataset is available for Devanagari handwritten characters. So dataset is prepared by writing 34 different people whose belongs to different age groups. In this work we only used Devanagari consonants. 10 samples of each Devanagari consonant have been written by each people means each people have written 360 (10*36) Devanagari characters in A4 size sheet. After that this sheet is scanned and saves as jpeg image (grayscale image). The following steps are used to create dataset. Entire work has been done in Matlab.

- Adjust image intensity values of the image using imadjust () function of Matlab.
- Convert the image into binary image by choosing Ostu's method.
- Remove from a binary image all connected components (objects) that have fewer than 30 pixels.
- Segment the image linewise by finding the black pixel in row.
- After linewise segmentation, each line segmented vertically.

B. Feature extraction techniques

- *Foreground pixel distribution*

Suppose that im(x, y) is a handwritten Devanagari character image in which the foreground pixels are denoted by 1's and background pixels are denoted by 0's. Feature extraction algorithm sub-divided the character image recursively. At granularity level 0 the image divided into four parts and gives a division point (DP) (x_0, y_0). The following algorithm shows that how x_0 is calculated and likewise y_0.

Algorithm:

Step 1: input im (x_{max}, y_{max}) where x_{max} and y_{max} be the width and the height of the character image
Step 2: Let v_0 [x_{max}] be the vertical projection of image (fig 2.b)
Step 3: Create v_1 [2*x_{max}] array by inserting a '0' before each element of v_0 (fig 2.c)
Step 4: Find x_q in v_1 that minimizes the difference between the sum of the left partition [1, x_q] and the right partition [x_q, 2*x_{max}] or left partition should be greater than right if not able to equally divide.
Step 5: $x_{0=}x_q/2$;
Step 6: if x_q mod 2 = 0
Two sub-images [(1, 1), (x_0, y_{max}) and (x_0, 1), (x_{max}, y_{max})]
Else
Two sub-images are [(1, 1), (x_0, y_{max}) and (x_0+1, 1), (x_{max},y_{max})]

Figure 2 shows the vertical division of handwritten Devanagari character image where the x_q =10 and x_0 =5 and x_q mod 2 is 0 than the co-ordinates of two sub-images are [(1,1),(5,10)] and [(5,1),(10,10)].

The number of sub-images, at the specified granularity level (L) will be $4^{(L+1)}$. Let L=0 then the number of sub-images are four and when the L=1 it will be 16. The number of DP (division point) equals to 4^L (figure 3). At level L, the co-ordinates (x_i, y_j) of all DPs are stored as features. So for every L a $2*4^L$-dimensional feature vector is extracted.

All feature vectors are scaled to (0, 1), by the help of normalized dimension value in our case it is 90. All the co-ordinates of feature vector are divided by 90.

$f" = f/90$ (1)

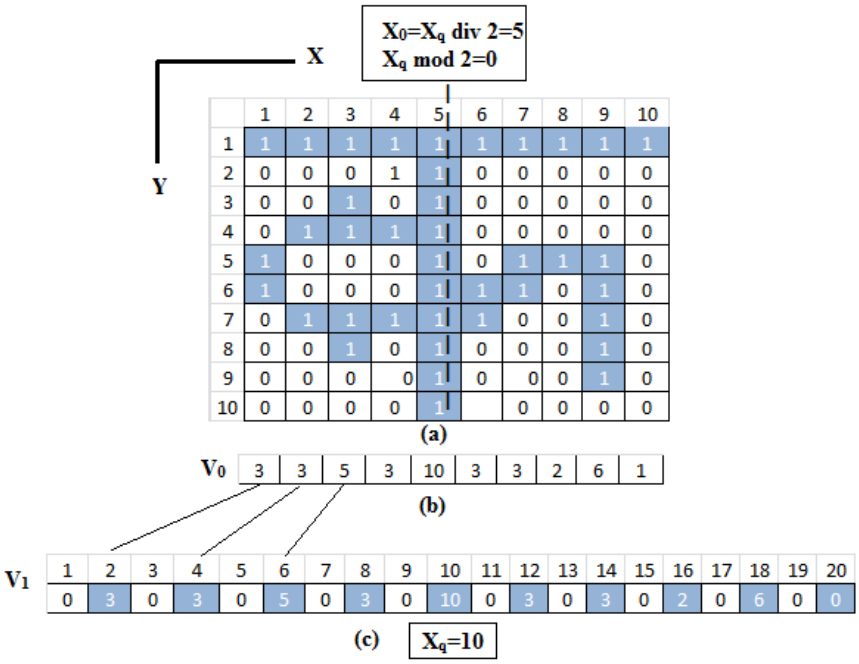


Figure 2(a) Vertical division of an image array (x_{max}=10, y_{max}=10) (b) vertical projection of image (c) v_1 created from v_0 to calculate x_q

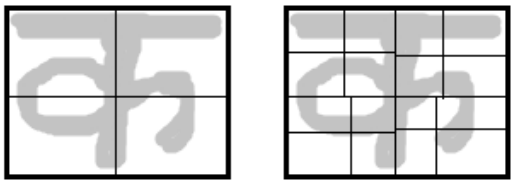


Figure 3: Devanagari Handwritten character (KA) segmentation at Level 0, 1 shown in corresponding (A) (B)

- *Zone Density feature*

We have created 16 (4*4) zones of our 32*32 sized sample. By dividing the number of foreground pixels in each zone by total number of pixels in each zone i.e. 64 we obtained the density of each zone. Thus we obtained 16 zoning density features. (Fig.4)

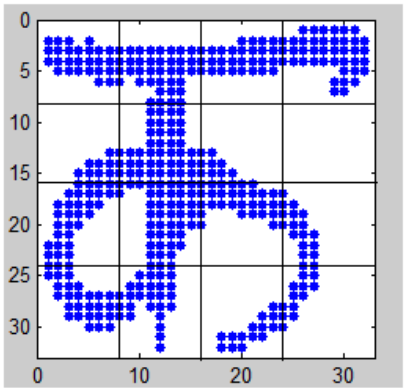
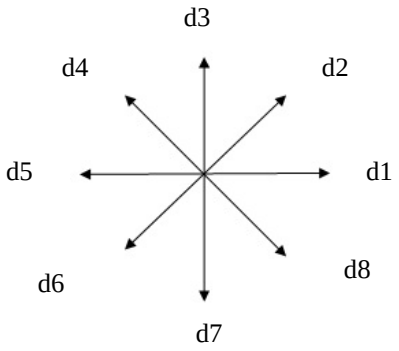


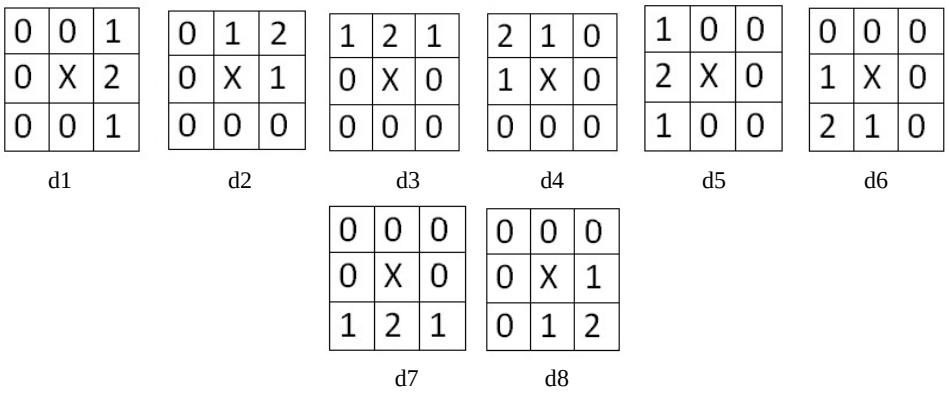
Figure 4: shows the 16 zone of the Devanagari character (KA)

- *Background Directional Distribution feature*

For these features we have considered the directional distribution of neighboring background pixels to foreground pixels. We computed 8 directional distribution features. To calculate directional distribution values of background pixels for each foreground pixel, we have used following masks for each directional values (Fig.5). The pixel at center 'X' is foreground pixel under consideration to calculate directional distribution values of background.



(a)



(b)

Fig.5. (a) 8 directions used to compute directional distribution, (b) Masks used to compute directional distribution in different directions.

To compute directional distribution value for foreground pixel 'X' in direction d1, for example, the corresponding mask values of neighboring background pixels will be added. Similarly we obtained all directional distribution values for each foreground pixel. Then, we summed up all similar directional distribution values for all pixels in each zone, described earlier in zoning density features description. Thus we finally computed 8 directional distribution feature values for each zone.

We combined three types of features extracted from foreground pixel distribution, zoning density and directional distribution. SVM classifier uses these features for classification.

IV. SVM CLASSIFIER

Support Vector Machine is supervised Machine Learning technique. The existence of SVM is shown in figure 6. It is primarily a two class classifier. Width of the margin between the classes is the optimization criterion, i.e. the empty area around the decision boundary defined by the distance to the nearest training pattern. These patterns called support vectors, finally define the classification function.

All the experiments are done on LIBSVM 3.0.1[15] which is multiclass SVM and select RBF (Redial Basis Function) kernel. A feature vector set $fv(x_i)$ i=1…m, where m is the total number of character in training set and a class set $cs(y_j)$ j=1…m , $cs(y_j) \epsilon$ { 0 1 ….9} which defines the class of the training set, fed to Multi Class SVM.

LIBSVM implements the "one against one" approach (Knerr et al .., 1990) [13] for multi-class classification. Some early works of applying this strategy to SVM include, for example, Kressel (1998) [14]. If k is the number of classes, then k (k-1)/2 classifiers are constructed and each one trains data from two classes. For training data from the i^{th} and j^{th} classes, we solve the following two class classification problem:

In classification we use a voting strategy: each binary classification is considered to be a voting where votes can be cast for all data points x - in the end a point is designated to be in a class with the maximum number of votes.

$$\begin{aligned} &\min_{w^{ij},\, b^{ij},\, \xi^{ij}} \quad \tfrac{1}{2}(w^{ij})^T w^{ij} + c\textstyle\sum_t (\xi^{ij})_t, \\ &\text{subject to } (w^{ij})^T \phi(x_t) + b^{ij} \geq 1 - \xi_t^{ij}, \\ &\quad \text{if } x_t \text{ in the } i^{th} \text{ class} \\ &\quad (w^{ij})^T \phi(x_t) + b^{ij} < -1 + \xi_t^{ij}, \\ &\quad \text{if } x_t \text{ in the } j^{th} \text{ class}, \\ &\quad \xi_t^{ij} \geq 0. \end{aligned} \qquad (2)$$

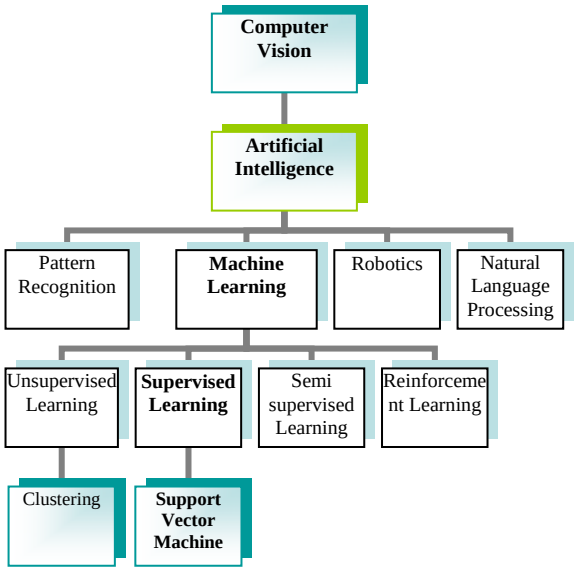


Figure 6

In case that two classes have identical votes, though it may not be a good strategy, now we simply choose the class appearing first in the array of storing class names.

LIBSVM is used with Radial Basis Function (RBF) kernel, a popular, general-purpose yet powerful kernel, denoted as

$$K(x_i, x_j) \equiv \exp(-\gamma \|x_i - x_j\|^2) \qquad (3)$$

Now a search is applied to find the value of γ which is parameter of RBF as like find the value of c that is cost parameter of SVM using cross-validation. The value of both variance parameters are selected in the range of (0, 1] for gamma γ and (0, 1000] for cost (c) and examines the recognition rate.

V. EXPERIMENTS AND RESULTS

As we have discussed in about section we have three types of feature to recognition the handwritten Devanagari character. 12240 samples have been used for the experiments. Those are written by the 34 different people. In orders to classify the handwritten Devanagari character and evaluate the performance of the technique; we have carried out the experiment by setting various parameter examples $L_{best,}$ gamma, and cost parameter. All experiments was performed on a Intel® core 2 duo CPU T6400 @ 2GHz with 3 GB RAM under 32 bit windows 7 Ultimate operating system.

In the estimation of the first feature which is foreground pixel distribution, we have to estimate the granularity level at which we will estimate the feature. So for estimating the best granularity level we have done some experiments separately and find the granularity level 3 gives the best result. So to estimate the feature, the L_{best} is 3 and the normalized the image by 90. The size of feature vector is 170 ($2*4^L - 2*4^0+2*4^1+2*4^2+2*4^3$). The second zonal density feature is obtained by dividing the normalized (32*32) image in 16 zones and evaluates the density of each zone. So it gives 16 features. The third Background Directional Distribution feature gives 128 features. Now the final feature vector is 314. 10 fold cross validation is applied for recognition accuracy. We experiment with different -2 values of the gamma (γ) shown in table 1 and obtained 94.89 % recognition rate at the value of gamma (γ) =0.4 and cost (c) = 500. Figure 7 shows the results in the graph form

.

S. No	10 –folds, Dataset size = 12240, Cost (c)=500,	
	Gamma (γ)	Recognition Accuracy
1	0.1	85.72 %
2	0.2	86.83 %
3	0.3	89.84 %
4	**0.4**	**94.89 %**
5	0.5	93.89 %
6	0.6	90.89 %
7	0.7	88.86 %
8	0.8	87.86 %
9	0.9	87.82 %
10	1	86.81 %

.

Table 1: shows the recognition accuracy

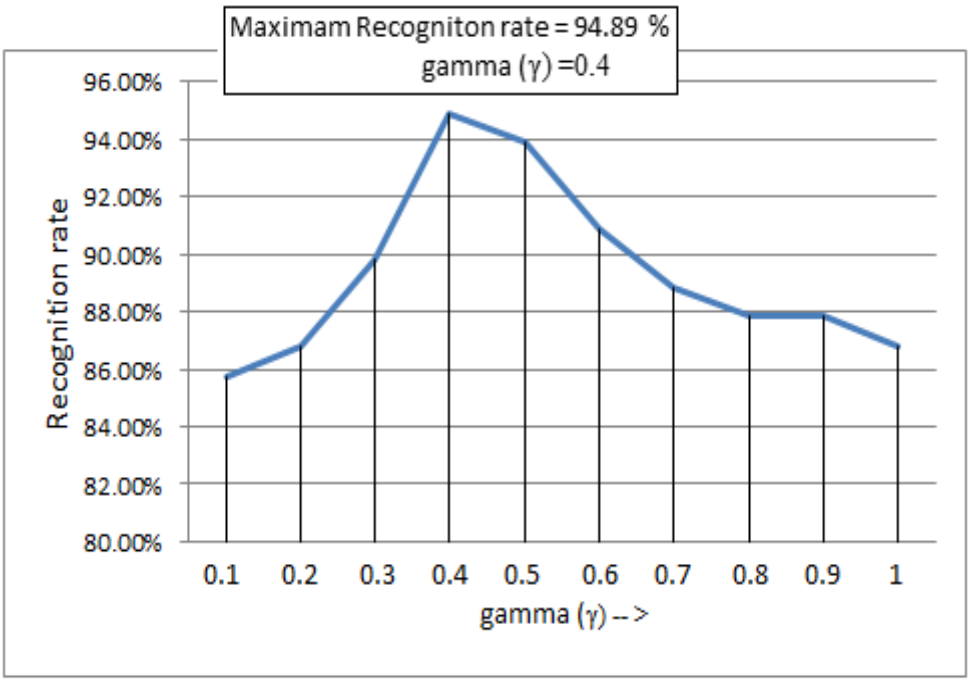


Figure 7: shows the recognition accuracy in graph form

REFERENCES

[1] U.Pal and B B Choudhuri, "Indian script character recognition: A survey" Pattern Recognition ,Vol 37,pp 1887-1899,2004

[2] R M K Sinha, " A journey from Indian scripts processing to Indian language processing ", IEEE Ann. Hist. Computer, vol 31, no 1, pp 831, 2009

[3] N. Sharma, U. Pal, F. Kimura, and S. Pal, "Recognition of offline hand-written Devnagari characters using quadratic classifier," in Proc. Indian Conf. Comput. Vis. Graph. Image Process., 2006, pp. 805–816.

[4] P. S. Deshpande, L. Malik, and S. Arora, "Fine classification & recognition of hand written Devnagari characters with regular expressions & minimum edit distance method," J. Comput., vol. 3, no. 5, pp. 11–17,2008

[5] S. Arora, D. Bhatcharjee, M. Nasipuri, and L. Malik, "A two stage classification approach for handwritten Devanagari characters," in Proc.Int. Conf. Comput. Intell. Multimedia Appl., 2007, pp. 399–403.

[6] M. Hanmandlu, O. V. R.Murthy, and V. K.Madasu, "FuzzyModel based recognition of handwritten Hindi characters," in Proc. Int. Conf. Digital Image Comput. Tech. Appl., 2007, pp. 454–461

[7] S. Arora, D. Bhattacharjee, M. Nasipuri, D. K. Basu, and M. Kundu,"Recognition of non-compound handwritten Devnagari characters using a combination ofMLP and minimum edit distance," Int. J. Comput. Sci.Security, vol. 4, no. 1, pp. 1–14, 2010.

[8] S. Kumar, "Performance comparison of features on Devanagari hand-printed dataset," Int. J. Recent Trends, vol. 1, no. 2, pp. 33–37, 2009.

[9] U. Pal,N. Sharma, T.Wakabayashi, and F.Kimura, "Off-line handwritten character recognition of Devnagari script," in Proc. 9th Conf. Document Anal. Recognit., 2007, pp. 496–500.

[10] V. Mane and L. Ragha, "Handwritten character recognition using elastic matching and PCA," in Proc. Int. Conf. Adv. Comput., Commun. Control, 2009, pp. 410–415.

[11] U. Pal, S. Chanda, T. Wakabayashi, and F. Kimura, "Accuracy improvement of Devnagari character recognition combining SVM and MQDF,"in Proc. 11th Int. Conf. Frontiers Handwrit. Recognit., 2008, pp. 367–372.

[12] U. Pal, T. Wakabayashi, and F. Kimura, "Comparative study of Devanagari handwritten character recognition using different features and classifiers," in Proc. 10th Conf. Document Anal. Recognit., 2009, pp. 1111–1115

[13] S. Knerr, L. Personnaz, and G. Dreyfus. Single-layer learning revisited: a stepwise procedure for building and training a neural network. In J. Fogelman, editor, Neu-rocomputing: Algorithms, Architectures and Applications. Springer-Verlag, 1990.

[14] U. H.-G. Kressel. Pairwise classication and support vector machines. In B. Scholkopf, C. J. C. Burges, and A. J. Smola, editors, Advances in Kernel Methods { Support Vector Learning, pages 255{268, Cambridge, MA, 1998. MIT Press.

[15] http://www.csie.ntu.edu.tw/~cjlin/libsvm

AUTHORS PROFILE

Mahesh Jangid is a student (M.Tech.) in computer science & engineering, department of Dr. B R Ambedkar National Institute of Technology. He has completed his B.E. degree in 2007 from Rajasthan University. He has the 2 year teaching experience from JECRC Jaipur and Many research papers have been published in international journals and conference. His research area is image processing, optical character recognition, pattern recognition.