ON-LINE HANDWRITTEN DEVANAGARI CHARACTER RECOGNITION USING FUZZY DIRECTIONAL FEATURES

Keywords: Devanagari script, online Handwritten Character Recognition (HCR), Fuzzy Directional Code Features

Abstract: This paper describes a new feature set for use in the recognition of on-line handwritten Devanagari script based on Fuzzy Directional Features. Experiments are conducted for the automatic recognition of isolated handwritten character primitives (sub-character units). Initially we describe the proposed feature set, called the Fuzzy Directional Features (FDF) and then show how these features can be effectively utilized for writer independent character recognition. Experimental results show that FDF set perform well for writer independent data set at stroke level recognition. The main contribution of this paper is the introduction of a novel feature set and establish experimentally its ability in recognition of handwritten Devanagari script.

1 INTRODUCTION

Interest in on-line handwritten script recognition has been active for a long time. In the case of Indian languages, research work is active especially for Devanagari (Joshi et al., 2005; Namboodiri and Jain, 2004), Bangala (Parui et al., 2008; Bhattacharya et al., 2007), Telugu (Babu et al., 2007) and Tamil (Sundaram and Ramakrishnan, 2007; Bharath and Madhvanath, 2007) to name a few. In English script, the mostly widely researched, barring a few alphabets, all the alphabets can be written in a single stroke. But most of the Indian languages have characters which are made up of two or more strokes which makes it necessary to analyze a set of strokes to identify the entire character. We identified, through visual inspection of the script, a basis like set of 44 strokes[1] called *primitives* which are sufficient to represent all the characters in Devanagari script. The set of primitives used to write the complete Devanagari character set are shown in Figure 1. Note that these set of primitives can be used to write all the possible alphabets in Devanagari by concatenating one or more of the primitives. In a very loose sense we can call this set of primitives a basis[2] which span the Devanagari character set. In this paper we use these primitives as the units for recognition taking parallel from the phoneset used in speech recognition literature. In an unconstrained hand written script these primitives exhibit large variability in shape, direction and order of writing. A sample set of primitives collected from different writers is shown in Figure 2 to show the variability in the way primitives are written. Variations within the primitives even for the same writer exist and as seen in Figure 2 the variation due to different writers is even larger; making the task of recognizing these primitives difficult.

The main challenge in on-line handwritten character recognition in Indian language is the large size of the character set, variation in writing style (when the same stroke is written by different writers or the same writer at different times) and the similarity between different characters in the script. In this paper, we propose the use of a new feature set called Fuzzy Directional Features (FDF) for the recognition of the primitives (which are also strokes); this is the main contribution of this paper. The rest of the paper is organized as follows. We introduce the Fuzzy Directional Features set in Section 2. Experimental results

[1] Usually the segment of pen motion from the pen-down to the pen-up position is a loose definition of a stroke

[2] in a loose sense; taking hints from vector algebra

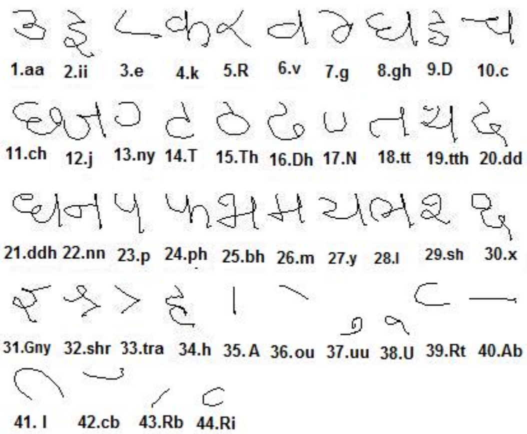


Figure 1: Primitive that can be used to write the complete alphabet set in Devanagari.

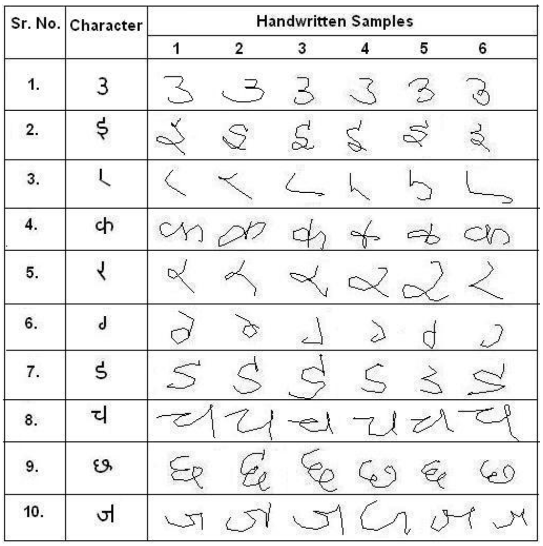
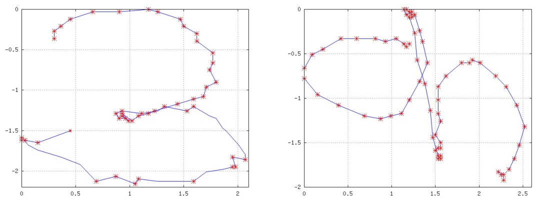


Figure 2: Variability in writing primitives.

are outlined in Section 3, and conclusions are drawn in Section 4.

2 FEATURE EXTRACTION

Several temporal features have been used for script recognition in general(Menier et al., 1994; Garcia Salicetti et al., 2001; Schenk and Rigoll, 2008; Connell and Jain, 2001) and for on-line Devanagari script recognition in particular. We propose a simple yet effective feature set called Fuzzy Directional Features set[3]. The detailed procedure for obtaining these Fuzzy Directional Features is given below. We first describe the usual directional feature set that is used in literature and then describe the *fuzzy* aspect that is imposed on the directional features to derive Fuzzy Directional Features.



(a) (b)

Figure 3: Sample strokes

Let an on-line handwritten stroke be represented by a variable number of 2D points which are in a time sequence. For example an on-line stroke of length n would be represented as

$$\{(x_{t_1},y_{t_1}),(x_{t_2},y_{t_2}),\cdots,(x_{t_n},y_{t_n})\}$$

where, t denotes the time and $t_1 < t_2 < \cdots < t_n$. Equivalently we can represent the on-line character (see Figure 3) as

$$\{(x_1,y_1),(x_2,y_2),\cdots,(x_n,y_n)\}$$

by dropping the variable t without any loss of information. The number of points denoted by n vary depending on the size of the character and also the speed with which the character is written. Most script digitizing devices (popularly called electronic pen) sample the script uniformly in time, generally at 100 Hz. For this reason, the number of sampling points is large when the writing speed is slow which is especially true at curvatures (see Figure 3); we exploit these curvature points in extracting directional features and FDF.

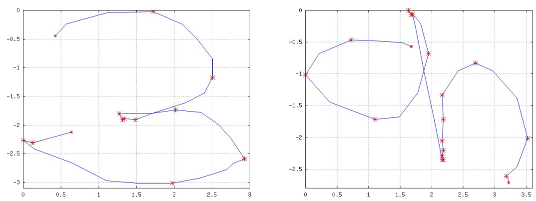
We first identify the curvature points (called critical points) from the smoothed (we use discrete wavelet transform) handwriting data. The sequence $(x_i,y_i)_{i=0}^{n}$ represents the handwritten data of a stroke. We treat the sequence x and y individually and calculate the critical points for each of these time sequences separately. For the sequence x, we calculate the first difference

$$x_i' = sgn(x_i - x_{i+1})$$

where

$$\begin{aligned} sgn(\tau) &= +1 \quad \text{if} \quad \tau > 0, \\ sgn(\tau) &= -1 \quad \text{if} \quad \tau < 0 \\ sgn(\tau) &= 0 \quad \text{if} \quad \tau = 0 \end{aligned}$$

[3]Note that (Mukherji and Rege, 2008) talks of fuzzy feature set for Devanagari script albeit for offline script



(a) (b)

Figure 4: Identified curvature points

We use x' to compute the critical point. The point i is a critical point if

$$x'_i - x'_{i+1} \neq 0$$

Similarly we calculate the critical points for the sequence y. The final list of critical points is the union of all the points marked as critical points by both the sequence x and the sequence y. Figure 4 shows the identified critical points of the strokes in Figure 3. It must be noted that the position and number of curvature points computed for different samples of the same primitive vary.

Let k be the number of curvature points (denoted by $c_1, c_2, \cdots c_k$) extracted from a stroke of length n; usually $k << n$. The k critical points form the basis for extraction of the directional features and FDF.

2.1 Directional Features (DF)

We first compute the angle between the two curvature points, say c_l and c_m, as

$$\theta_{lm} = \tan^{-1}\left(\frac{y_l - y_m}{x_l - x_m}\right)$$

where (x_l, y_l) and (x_m, y_m) are the coordinates corresponding to the curvature point c_l and c_m respectively. In general m and l are in time sequence next to each other. Now the directional feature set is computed by first computing the direction between the curvature points, namely d_{lm} corresponding to the angle θ_{lm} (computed using the Algorithm 1; note that d_{lm} can take a value between 1 and 8) and is the direction between the curvature point c_l and c_m. Note that k curvature points produce a vector of size $k-1$ namely, $d_{i,i+1}$ for $i = 1, 2, \cdots, k-1$.

Trimming of curvature points is carried out on the obtained k direction sequence by removing all *spurious* curvature point. A curvature point is said to be spurious if a set of three curvature points results in the same direction. For the sake of discussion lets assume that there are no spurious curvature points. The directional feature of the stroke is represented as the histogram of the directions obtained for a stroke. This is best explained by an example. If there are 6 curvature points identified for a stroke, and the corresponding 5 directions obtained (using Algorithm 1) are say, 1 3 4 3 7, then the directional vector of the stroke is computed as

$$\mathcal{D} = [1\ 0\ 2\ 1\ 0\ 0\ 1\ 0]$$

which is a vector of size 8.

Algorithm 1 Angle between two curvature point conversion into direction

int deg2dir(double θ)
int dir = -1;
if ($\theta > -\pi/8$ & $\theta < \pi/8$) **then**
  dir = 1;
end if
if ($deg >= \pi/8$ & $\theta < 3\pi/8$) **then**
  dir = 2;
end if
if ($\theta >= 3pi/8$ & $\theta < 5pi/8$) **then**
  dir = 3;
end if
if ($\theta >= 5\pi/8$ & $\theta < 7\pi/8$) **then**
  dir = 4;
end if
if (($\theta >= 7\pi/8$ & $\theta < 9\pi/8$) || ($\theta >= -9\pi/8$ & $\theta < -7\pi/8$)) **then**
  dir = 5;
end if
if ($\theta >= -7\pi/8$ & $\theta < -5\pi/8$) **then**
  dir = 6;
end if
if ($\theta >= -5\pi/8$ & $\theta < -3\pi/8$) **then**
  dir = 7;
end if
if ($\theta > -3\pi/8$ & $\theta < -\pi/8$) **then**
  dir = 8;
end if
return(dir);

2.2 Fuzzy Directional Features (FDF)

The FDF set is computed using θ_{lm} as is done as shown for computing directional features. We use Algorithm 3 assisted by triangular membership function described in Algorithm 2 to compute the FDF. Note that every θ_{lm} (represented by θ in Figure 5 is the angle the blue dotted line makes with the 0^o axis) has two directions (say $d^1_{lm} = 1$, $d^2_{lm} = 2$, note that the line in dotted blue in Figure 5 lies in both the triangles represented by direction 1 and direction 2) associated with it having m^1_{lm}, m^2_{lm} membership values respectively (represented by the green and the red dot

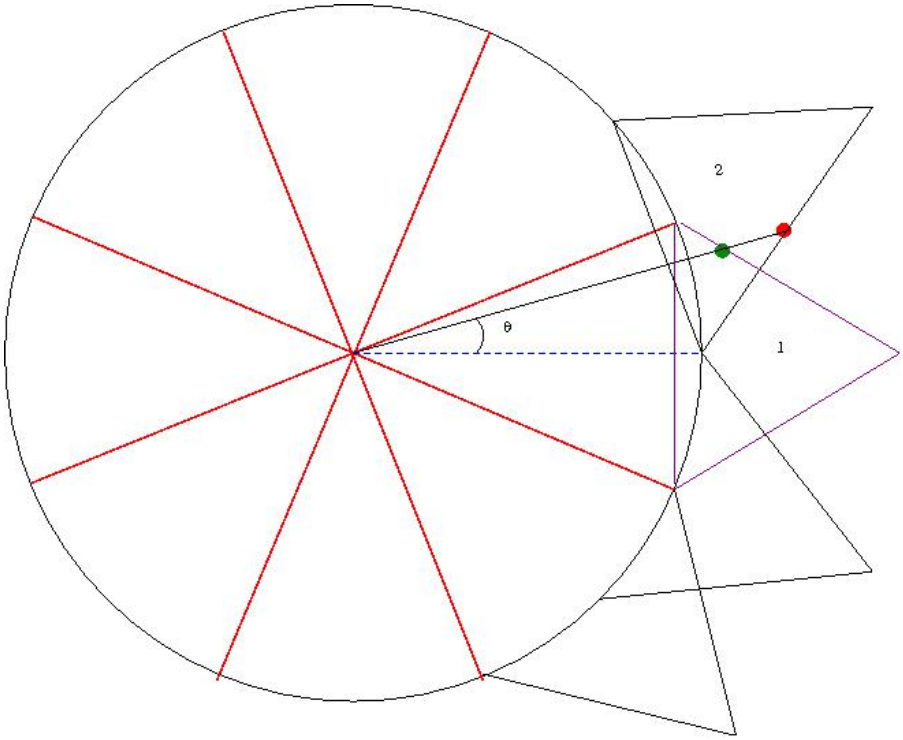


Figure 5: θ contributing to two directions (1, 2) with corresponding membership values (green and red dot).

respectively in Figure 5). Also note (a) $m^1_{lm} + m^2_{lm} = 1$ and (b) d^1_{lm}, d^2_{lm} are adjacent directions, for example if $d^1_{lm} = 5$ then d^2_{lm} could be either 4 or 6.

Algorithm 2 Triangular Fuzzy Membership Function.

fuzzy_membership(θ_c, θ)
$m = 1.0 - \frac{(|(\theta_c - \theta)|)}{(\pi/4)}$;
return(m)

It should be noted that the sum of the membership functions of a particular row (see Figure 6) is always 1. Given an on-line character, we extract the FDF shown in Figure 6. Then we calculate the mean FDF by averaging across the columns, so as to form a vector of dimension 8. The mean is calculated as follows; for each direction (1 to 8), collect all the membership values and divide by the number of occurrences of the membership values in that direction. For example, for Figure 6, the mean for direction 1 is calculated as $f_1 = \frac{(m^1_{23} + m^1_{34})}{2}$. In all our experiments we have used this mean FDF

$$\mathcal{F} = [f_1, f_2, \cdots f_8] \tag{1}$$

to represent a stroke. Clearly, the *fuzzy* aspect comes into picture due to the membership function which associates the angle between two curvature points with two directions (instead of one as in Directional Feature extraction). We show experimentally that this improves the performance of recognition.

3 EXPERIMENTAL ANALYSIS

For experimental analysis, we collected handwritten data from 10 persons, each of whom wrote all the primitives of Devanagari text using Mobile e-Notes Taker[4]. The mobile e-note taker is a portable pen based handwriting capture device which allows user to write on a normal paper using the electronic pen to capture handwritten text. This raw stroke data is smoothed using Discrete Wavelet Transform (DWT) decomposition[5] to remove noise in terms of small undulation due to the sensitiveness of the sensors on the electronic pen. For each stroke we extracted the directional feature and the FDF set as described in Section 2.

For training[6], we calculated $\mathcal{F}$ as in (1) for all

[4]http://www.hitech-in.com/mobile_e-note_taker.htm

[5]We do not dwell on this since this is well covered in pattern recognition literature.

[6]in this section we describe only the process adopted for

$(\theta \downarrow)(d \rightarrow)$	1	2	3	4	5	6	7	8
θ_{12}			m_{12}^1	m_{12}^2				
θ_{23}	m_{23}^1	m_{23}^2						
θ_{34}	m_{34}^2							m_{34}^1
$\vdots$								
θ_{lm}							m_{lm}^2	m_{lm}^1
$\vdots$								
θ_{k-1k}					m_{k-1k}^2	m_{k-1k}^1		
$\mathcal{F}$	$f_1 = \frac{(m_{23}^1+m_{34}^1)}{2}$	$f_2 = m_{23}^2$	$f_3 = m_{12}^1$	$f_4 = m_{12}^2$	$f_5 = m_{k-1k}^2$	$f_6 = m_{k-1k}^1$	$f_7 = m_{lm}^2$	$f_8 = \frac{(m_{34}^1+m_{lm}^1)}{2}$

Figure 6: Fuzzy Directional Features.

strokes corresponding to the same primitive and computed the average to model the primitive. If there are β strokes corresponding to a primitive, then we have β $\mathcal{F}$'s, say, $\mathcal{F}_1, \mathcal{F}_2, \cdots, \mathcal{F}_\beta$ then the average (μ)

$$\mu = \frac{1}{\beta}\sum_{i=1}^{\beta}\mathcal{F}_i \qquad (2)$$

So a primitive was represented by μ which is a vector of length 8 by taking the average over all the occurrences of the primitive in the training set

We used 5 user data for training and the other 5 for the purpose of testing the performance of the FDF set. We initially hand tagged each stroke in the collected data using the 44 primitives that we selected (see Figure 1). All the experimental results are based on this data set (from 10 different writers). For testing purpose, we took a stroke (t) to be recognized, we first extracted FDF (using Algorithm 3) and computed the mean FDF using (1), say F_t.

In the first instance, we compared F_t with the μ of the FDF model of all the 44 reference strokes using the usual Euclidean distance measure. For this we computed for the test stroke t, its distance from all the primitives, namely,

$$D_{\mu_i} = ||F_t - \mu_i||^2$$

for $i = 1, \cdots 44$ and arranged them in the increasing order of magnitude (best match first). The results for this are shown in Table 1 for both the train data and the test data for $\alpha = 1, 2, 5$. Note that the values in Table 1 are computed as follow. For $N = \alpha$, the test stroke t is recognized as the primitive l if l occurs at least at the α^{th} position from the best match (this is generally called the N-best in literature).

The results for both directional features and FDF are shown in Table 1. It should be noted that the accuracies are writer independent and for stroke level recognition. As expected the recognition accuracies are poor (very similar to the phoneme recognition by a speech engine) for $\alpha = 1$ and improves with increasing α for both the types of features. However, observe that there is a consistent improvement in recognition accuracies due to use of Fuzzy Directional Features compared to the directional features, as much as 8% for the test data set for $\alpha = 1$.

FDF; however the same process was adopted for directional feature

4 CONCLUSIONS

In this paper we have introduces a new on-line script feature set, called the Fuzzy Directional Features. We have evaluated the performance of the novel feature set by presenting the recognition accuracies for writer independent stroke level data set and comparing with the traditionally used directional feature set. It is well known, both in speech and script recognition literature that stroke (phoneme in case of speech) recognition is always poor. As in speech we plan use (a) Viterbi traceback to enhance alphabet (multiple stroke) recognition and/or (b) cluster strokes using spatio-temporal information to form alphabets and then use the cluster of strokes to recognize them. This we believe will lead to better accuracies of writer independent script recognition. We also plan to use second order statistics (as against just the mean that has been used in this paper) to evaluate the performance of FDF; this however requires large amount of data.

REFERENCES

Babu, V., Prasanth, L., Sharma, R., Rao, G., and Bharath, A. (2007). HMM-based online handwriting recognition system for Telugu symbols. In *ICDAR07*, pages 63–67.

Table 1: Recognition accuracies for train and test data set for (a) FDF and (b) DF using measures μ

Features	Data	$\alpha = 1$	$\alpha = 2$	$\alpha = 5$
FDF	Train	63.0% (139/220)	87.9% (193/220)	93.3%(205/220)
	Test	37.0% (82/220)	54.6% (120/220)	78.1% (172/220)
DF	Train	57.27% (126/220)	75.45% (166/220)	89.54%(197/220)
	Test	29.09 % (64/220)	43.63 % (96/220)	70.45 % (155/220)

Bharath, A. and Madhvanath, S. (2007). Hidden markov models for online handwritten Tamil word recognition. In *ICDAR07*, pages 506–510.

Bhattacharya, U., Gupta, B., and Parui, S. (2007). Direction code based features for recognition of online handwritten characters of Bangla. In *ICDAR07*, pages 58–62.

Connell, S. and Jain, A. (2001). Template-based online character recognition. *Pattern Recognition*, 34(1):1–14.

Garcia Salicetti, S., Dorizzi, B., Gallinari, P., and Wimmer, Z. (2001). Maximum mutual information training for an online neural predictive handwritten word recognition system. *IJDAR*, 4(1):56–68.

Joshi, N., Sita, G., Ramakrishnan, A., Deepu, V., and Madhvanath, S. (2005). Machine recognition of online handwritten Devanagari characters. In *ICDAR05*, pages II: 1156–1160.

Menier, G., Lorette, G., and Gentric, P. (1994). A genetic algorithm for on-line cursive handwriting recognition. In *ICPR*, pages B:522–525.

Mukherji, P. and Rege, P. P. (2008). Fuzzy stroke analysis of devnagari handwritten characters. *W. Trans. on Comp.*, 7(5):351–362.

Namboodiri, A. and Jain, A. (2004). Online handwritten script recognition. *PAMI*, 26(1):124–130.

Parui, S., Guin, K., Bhattacharya, U., and Chaudhuri, B. (2008). Online handwritten Bangla character recognition using HMM. In *ICPR08*, pages 1–4.

Schenk, J. and Rigoll, G. (2008). Neural net vector quantizers for discrete hmm-based on-line handwritten whiteboard-note recognition. In *ICPR*, pages 1–4.

Sundaram, S. and Ramakrishnan, A. (2007). A novel hierarchical classification scheme for online Tamil character recognition. In *ICDAR07*, pages 1218–1222.

Algorithm 3 Computing Fuzzy Directional Features.

```
deg2fuzzydir(double θ)
i=1; d[i] = -1; m[i] = -1
if (θ > −π/8 & θ < π/8) then
  d[i] = 1; m[i] = fuzzy_membership(0,θ);
end if
if (θ >= 0 & θ < 2π/8) then
  d[i] = 2; m[i] = fuzzy_membership(2π/8,θ);
  i++;
end if
if (θ >= π/8 & θ < 3π/8) then
  d[i] = 3; m[i] = fuzzy_membership(3π/8,θ);
  i++;
end if
if (θ >= 2π/8 & θ < 4π/8) then
  d[i] = 4; m[i] = fuzzy_membership(4π/8,θ);
  i++;
end if
if ((θ >= 3π/8 & θ < 5π/8)) then
  d[i] = 5; m[i] = fuzzy_membership(5π/8,θ);
  i++;
end if
if ((θ >= −5π/8 & θ < −3π/8)) then
  d[i] = 5; m[i] = fuzzy_membership(−3π/8,θ);
  i++;
end if
if (θ >= −4π/8 & θ < −2π/8) then
  d[i] = 6; m[i] = fuzzy_membership(−2π/8,θ);
  i++;
end if
if (θ >= −3π/8 & θ < −π/8) then
  d[i] = 7; m[i] = fuzzy_membership(−π/8,θ);
  i++;
end if
if (θ > −2π/4 & θ < 0) then
  d[i] = 8; m[i] = fuzzy_membership(0,θ);
  i++;
end if
return(d[i], m[i]);
```