Chapter 4

A Structural Approach on a Template-based Handwritten Character Recognition along with the Additional Use of Strokes' Spatial Information

4.1 System Design: Spatial Relation between/among the Strokes based Clustering - A Modern Approach to Writer Independent Character Recognition.

In this task, a new scheme is purposed in building a prototype classifier for classifying Nepali natural handwritten characters in order to overcome the shortcomings in the previous classifiers. It uses a novel idea for analyzing a character based on both the number of strokes used and the strokes' spatial relation within a character for Nepali. Handling spatial information about the strokes is not the new technique in handwriting recognition field (Murakatat, et al., 2004; Bouteruche et al., 2005), however, this task provides different strategy in collecting spatial information and different aspects in designing a prototype recognition engine.

The novel idea different from previous related tasks goes like this.

Firstly, the classifier determined the number of strokes used to complete a character and then separated based on this into different groups. It means those characters having identical number of strokes are collected in one group whereas the dissimilar are in other groups.

Secondly, captured strokes are separated and locations from each sequence to another within a character are determined.

Thirdly, strokes' representatives from a number of similar strokes are determined using DTW based on the distance calculation. Only those strokes of a specific class of character of the specific group are sent to clustering block instead of using all strokes of every character (previous case). Use of both number of strokes and their locations in clustering enjoys the advantages of not merging two different strokes from different groups. However, it has a chance of having different features in a same space of a same character from different users, but is negligible.

Fourthly, strokes' representatives after merging are stored as the templates from every class of character. This ends the training strategy.

Finally, a character is tested stroke-by-stroke basis. A simple template matching procedure is carried out in stroke identification as in the previous classifier. However, the way of matching

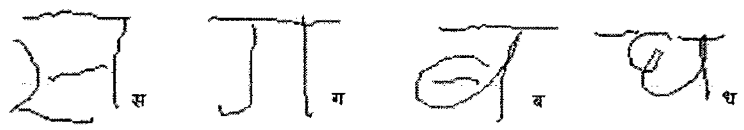


Figure 4.1 Four samples having multiple straight sequences

the test stroke with templates is different. Each stroke is aligned with those templates, which have both identical spatial properties and number of strokes used to complete a character as that of a test character. Hence, the basic idea to recognize a complete test character is to identify all the components (strokes) according to their order. Creating spaces for the templates and matching are the common approach; however, this task explores different aspects in templates' placement and matching for easier classification.

4.1.1 Spatial Relation between/among the Strokes Based Clustering

Spatial information has been a great effort in stroke identification along with the use of feature in case of Nepali characters. Once all strokes are identified, character is classified. Spatial information gives both location and the size of the string from one another. Every character has one 'shirorekha' in which text is suspended. Some times two (Fig. 4.1), which depends on the users' style in writing and appear in some characters only. Having at least one 'shirorekha' in every character has been a unique feature sequence in Nepali in comparison to other scripts. Looking into the users' natural handwriting, the 'shirorekha' may not always be horizontal (it may be either a line with negative or positive slope) and always not on the top exactly (sometimes top-left, sometimes top-right and sometimes it may intersect with the text), what it should be. Not only this, it may also be a small curve sometimes, which acts as a 'shirorekha' but not the text. As locations of both 'shirorekha' and texts vary widely, it is difficult to build a general rule for the determination of spatial relation among them. In general, location of the 'shirorekha' is assumed on the top portion (not specific like, top, top-right and top-left) and then the text(s) in reference to 'shirorekha' is/are determined.

Sequences are provided with six regions: top-left, top, top-right, bottom-left, bottom and bottom-right. In order to determine the location of the stroke, two main criterions are carried out, which are; boundary and angle criterions. Boundary condition uses both maximum and minimum values of both horizontal and vertical axis, while center of gravity of the strings are used in case of angle condition to be checked. Fundamentally, every character has at least two kinds of sequences, one is straight sequence and another is curve. Naturally, a straight sequence resembles a 'shirorekha'. In order to identify a sequence either a straight or a curve sequence, the following conditions are checked.

$$St-CS = \frac{d(\mathbf{p_1},\mathbf{p_l})}{\sum_1^{l-1} d(\mathbf{p_i},\mathbf{p_{i+1}})} \tag{4.1}$$

where, $d(\mathbf{p_1},\mathbf{p_2}) = \sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$, gives the distance between two coordinates $\mathbf{p_1}$ and $\mathbf{p_2}$.

- *For Straight sequence*: $0.8 \le St-CS \le 1$

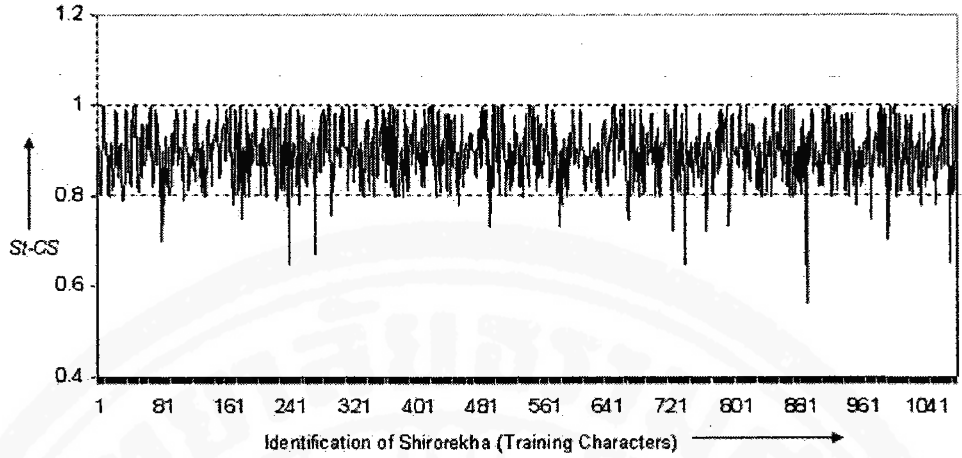


Figure 4.2 Threshold range $(St - Cs)$ determination for shirorekha

- *For Curve sequence*: $0 \leq St - CS < 0.8$

How did we determine the range of $St - CS$ for straight sequence ('shirorekha') is shown in the Fig 4.2. It demonstrates that approximately 98% of 'shirorekha' from all training characters are lying within the range mentioned above. Finding straight sequence does not mean that the 'shirorekha' is determined. For the two-stroke characters, it is easy to separate the straight and curve sequence. But, in case of characters having more than two strokes, there may be chance of multiple sequences containing the feature of straight sequence, which are shown in Fig. 4.1. In such a case, we need to determine the 'shirorekha' from many straight sequences because not every straight sequence is the 'shirorekha'. It makes sense that a straight sequence can be a text sometimes. The only difference is, where does it lie? Here, we design extra conditions, which are,

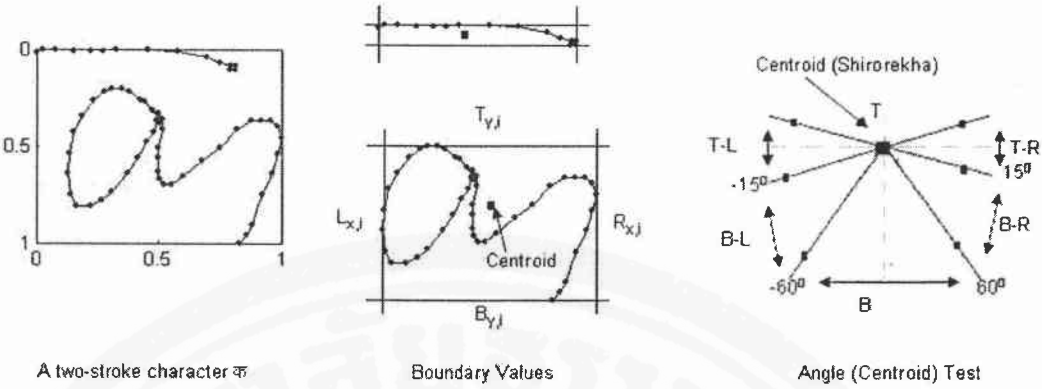
a. Find direction from initial to the end coordinate.
   $Dir.(Angle) = \arctan(\frac{y_l - y_1}{x_l - x_1}) \times (\frac{180^\circ}{pi})$

b. Find width of the sequence along x-axis.
   $Width = x_{max} - x_{min}$

c. Find positional relation among the straight lines. (a concrete boundary condition)

The straight sequence, which has small change in direction either to positive or to negative from initial to final coordinate, has larger width along the x-axis and has positioned on the top in reference to other straight sequences is the 'shirorekha'.

After separating the 'shirorekha' from other curve texts, their positions (based on their size) are determined by using both boundary and angle conditions. Fig. 4.3. shows the basic boundary and angle conditions to test the strokes locations. Assuming that the 'shirorekha' is on the top portion, location(s) of text(s) is/are estimated. Finally, the location of the 'shirorekha' along with its size is confirmed, once the location(s) of the text(s) is/are determined. A real example of two strokes character क is shown in Fig. 4.4. In addition, the character with two 'shirorekha' is also possible to determine the location in the provided regions.



Index: B: Bottom, B-L: Bottom Left, B-R: Bottom Right, T: Top, T-L: Top Left and T-R: Top Right

Figure 4.3 Spatial relation between two strokes of the character क

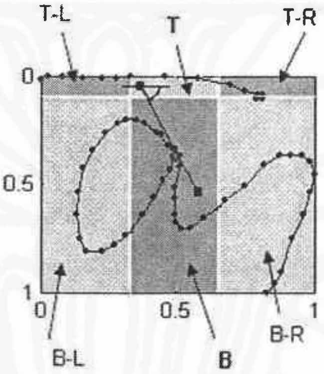


Figure 4.4 A real example of how spatial relation is determined by taking a character क

In such a case, the first 'shirorekha' is the reference for another according to the order of strokes. Three regions: Top-right, top and top-left are provided for 'shirorekha'.

After separating strokes based on the number and the location, a modern approach in clustering is explored. Broadly, a clustering is a technique for collecting items into specific groups, which are similar in some way. Items of one group are dissimilar with another items belonging to another groups. This technique helps in two aspects. Firstly, it reduces the number of similar items into one (representative), i.e. compact system. Secondly, it increases the speed. Clustering consists of two steps. The first step is to organize a number of characters into different groups based on the number of strokes used in completing a character, which is shown in Fig. ??. It is assumed that the characters having identical number of strokes have almost same way of writing even from different users. The number of strokes used is varied from user to user even within a specific character. However, it is known that at least two strokes are used in completing a character from any of the users. In the second step, a number of similar strokes are agglomerating pair wise within a group. We used single-linkage agglomerative hierarchical clustering. Based on the spatial relation between/among the strokes within a character, agglomerating has taken place. It means a number of strokes which are at specific location (bottom, bottom-right, bottom-left, top, top-right and top left) are feed to the clustering block. How can similarity between two sequences be measured? The answer

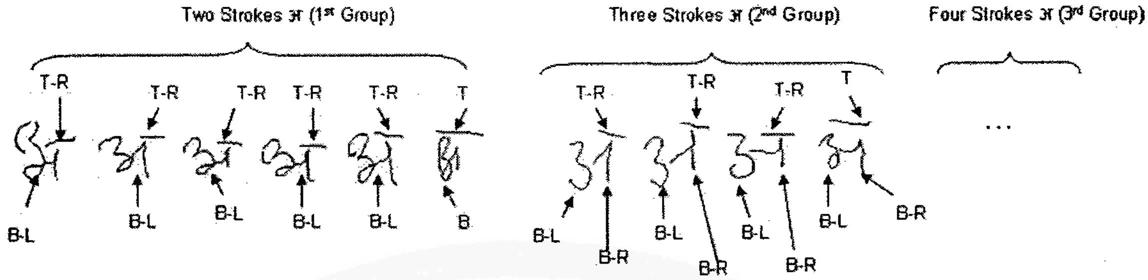


Figure 4.5: Grouping of characters of a specific class अ based on the number of strokes used

is 'distance measurement technique' to estimate the the similarity as in the previous chapter.

4.1.2 Templates' Management

It is important to note that a frame is created for a class of character. More over, there are variable numbers of groups inside each frame because it entirely depends on the number of strokes in completing a character. Fig. 4.6 shows two samples of hierarchical clustering for two strokes अ, where the strokes are clustered based on their locations. In other words, strokes are clustered based on the spatial relation between/among them (texts at bottom-left and bottom are clustered separately and similarly shirorekha on top-right and top). Only six users are employed for the demonstration. It is important to note that the threshold is designed only for text strokes but not the shirorekha. For shirorekha, the number of templates is equal to the number of different locations, but it is variable for texts. More specifically, only those strokes,which are at bottom-left position, are clustered in one group and those strokes on top-right are clustered separately in another group. However, the frame allocated for अ contains every cluster representative from every group in different spaces. For instance, the strokes' representatives from two strokes characters, three strokes characters and four strokes characters are stored in first group, second group and third group respectively. Fig. 4.7 shows a sample of template management strategy by taking few groups. A sample of how we created spaces for those strokes' representatives in each group is demonstrated. A space is used for storing a template. Spaces for both the text and the shirorekha are designed based on where they are located. This is the situation for all classes of characters happened in storing strokes' essence representatives. As the merging of similar strokes has taken place based on the location, it does not have a chance of merging three kinds of shirorekha (i.e. top, top-right and top-left), store even in one group even though they have identical feature. It is because of different locations, as location of shirorekha has a significant place in recognition.

One importance of preserving the locations of shirorekha is illustrated here in some confusion pairs of characters: (ध, घ), (भ, म), (थ, य) etc.

4.1.3 Weight Determination

This section deals with the number of strokes and their spatial relation in reference to one another with the use of probability. In other words, how many strokes are at specific location

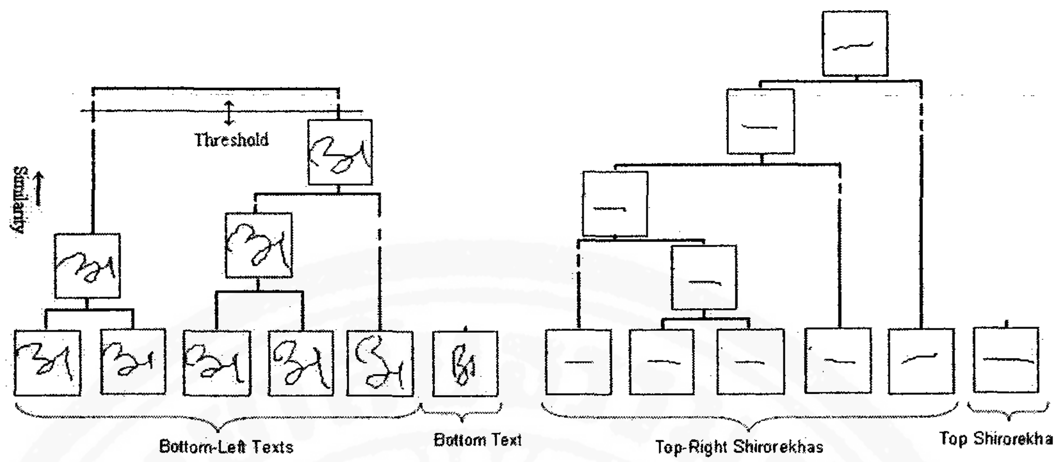


Figure 4.6 Clustering technique for two-stroke अ

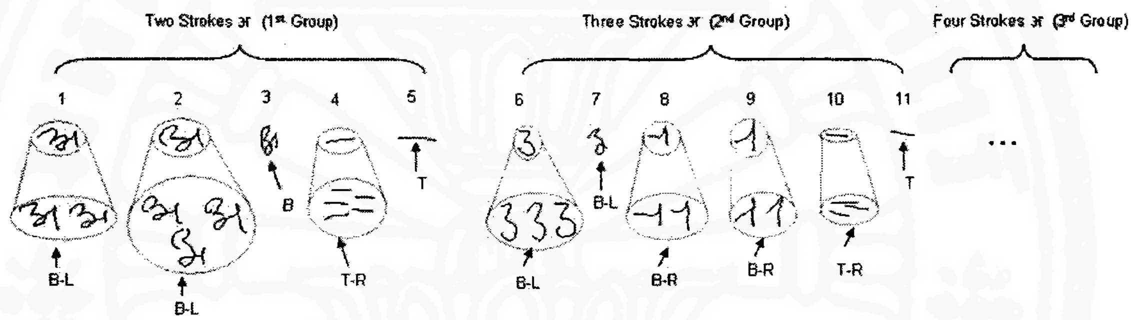


Figure 4.7 Template management for a class of character अ having many groups

in a specific group is the aim to find out in this section. Specifically, it deals with every stroke's representative after clustering in every group.

For instance, let us take a look into Fig. 4.5 for better understanding. Consider the first group having two strokes अ only.

- *For shirorekha*:

  Probability of being on top = No. of shirorekha on top / No. of users = 1/6 = 16.7%
  Probability of being on top-right = 5/6 = 83.4%
  Probability of being on top-left = 0/6 = 0%

- *For Text*:

  Probability of being at bottom = 1/6 = 16.7%
  Probability of being at bottom-right = 0/6 = 0%
  Probability of being at bottom-left = 5/6 = 83.4%

This is the way of calculating weight of each template of every group inside every class of character. Further more, it provides us an idea about how people write and it is important to know the writing pattern of every class of character from many users. These probabilities are considered in template matching in order to find the test strokes in classification block.

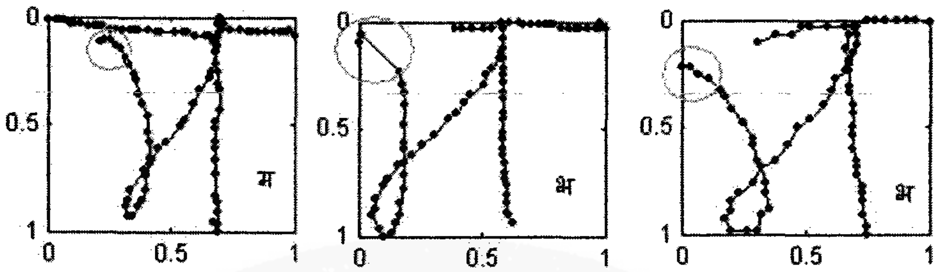


Figure 4.8: Correct recognition of confusion character म (भ) by handling spatial information of the strokes

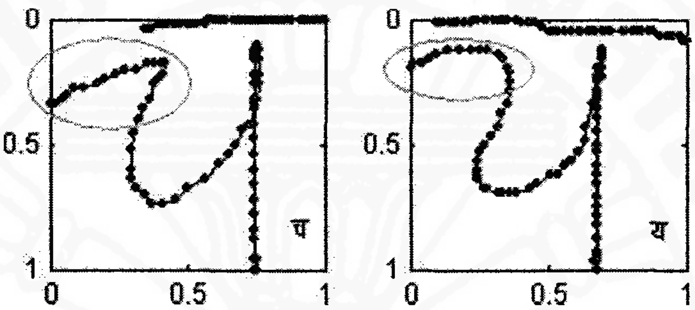


Figure 4.9 Confusion between य↔च

4.1.4 Classification/Recognition

A character is recognized using stroke-by-stroke matching phenomenon. It means any number of strokes used in test character is matched with the templates independently. The stroke is said to be similar with the template from which the lowest distance is produced.

As we have an idea about template management for all class of character in the previous section, we are now able to understand the algorithm used in recognizing the test character. This simple algorithm gives a complete idea of how test strokes are matched with the templates.

a. Determine the stroke' spatial relation.

b. Each stroke is matched only with specific templates. The templates, which are to be matched with the test strokes, are determined based on the number of strokes used in a test character and their spatial relation. For instance, for two strokes test character, matching is involved with only those templates of those groups, which are created from two strokes character. More specifically, shirorekha is matched with only the templates of shirorekha of those groups, where two strokes characters are used for making templates and texts are also matched in the similar manner. Every matching produces a matching score. Mathematically, the distance matrix from n set of test

Table 4.1 Experimental Results

Dataset	Char. Type (classes) (Test chars.)	Misrecog. Chars.	Rejections	Avg. Err.
Training (15 Users)	Consonants (31) (930)	9	2	1%
	Vowels (5) (150)	1	1	
Test (10 Users)	Consonants (31) (620)	29	5	5%
	Vowels (5) (100)	4	3	

No. of Users: 25

Training Samples: Pre-processed

Feature: Position and direction at every point along the pen trajectory ($f_t = (x_t, y_t, \theta_t)$)

Accuracy: 95%

Average Speed: 12 Seconds/Character

Total Bad Data: 1.12%

strokes matching is,

$$\mathbf{D} = \begin{pmatrix} \mathbf{D}_1 \\ \mathbf{D}_2 \\ \dots \\ \mathbf{D}_n \end{pmatrix}$$

where, $\mathbf{D_i} = [\mathbf{F_i^1}, \mathbf{F_i^2}, \dots, \mathbf{F_i^k}]$. $\mathbf{F_i^k}$ contains some matching scores from specific group of k-th frame when i-th test stroke is matched. We have 36 frames for 36 classes of character. Even though we have fixed strokes' templates in every frame, the numbers of matching scores are from only the specific group, i.e. $\mathbf{F_i^k} = [D_{i,1}^k, D_{i,2}^k, \dots, D_{i,m}^k]$. $D_{i,m}^k$ is the matching score from m template whenever i-th test stroke is employed. $D_{i,m}^k = \infty$, if m template is not used in matching.

c. Each matching score is divided by the weight of the template based on location. Then, the weighted matching score is,

$$D_{i,m}^{k(w)} = D_{i,m}^k / weight(k,m) \tag{4.2}$$

d. Determine the threshold by taking all matching scores from all frames. Threshold for every i-th test stroke is,

$$Thshld_i = min(\mathbf{D_i}) + C \tag{4.3}$$

where, C is the fixed constant.

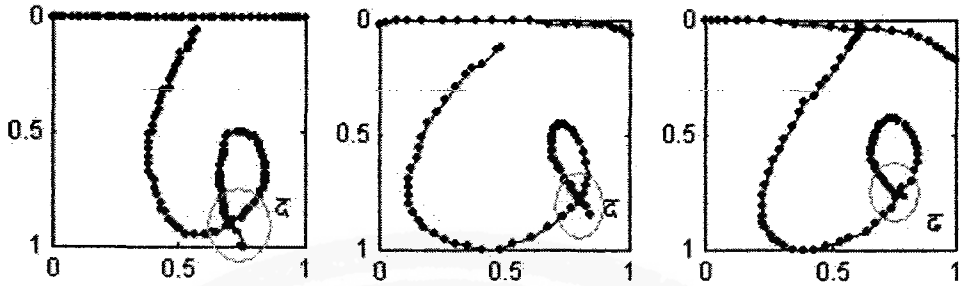


Figure 4.10 Two confusions pairs: द→ढ

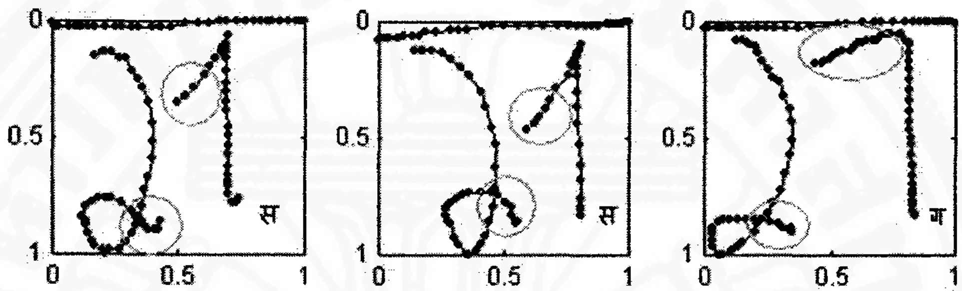


Figure 4.11 Two confusions pairs: स→ग

e. Count the number of matching scores below the threshold in every frame. This number determines how many similar strokes are available as templates. Mathematically, for every i-th test stroke in k-th frame, weight can be determined as,

$$W_i^k = \sum_{j=1}^{m} C(D_{i,j}^{k(w)}, Thshld_i) \tag{4.4}$$

where,

$$c(x,y) = \begin{cases} 1 & \text{if } x < y \\ 0 & \text{otherwise} \end{cases}$$

f. Find the lowest matching score from every matching and from every frame if matching takes place. It should be kept in mind that the lowest matching score is produced only from the most similar template. Mathematically,

$$lms_i^k = min(\mathbf{F_i^k}) \tag{4.5}$$

g. Finally, the character's label is classified as,

$$L = argmin_k \sum_{i=1}^{n} \frac{lms_i^k}{W_i^k} \tag{4.6}$$

The template matching is straight forward when spatial information is not considered. It refers to previous works (K.C. et al., 2006).

Table 4.2 Error Analysis (Test Data)

Error Type	Chars. (Err.)
Feature Similarity	19 (02.63%)
Diminished and/or very long Ascender and Descender	7 (00.97%)
Re-writing	8 (01.12%)
Mis-writing	8 (01.12%)

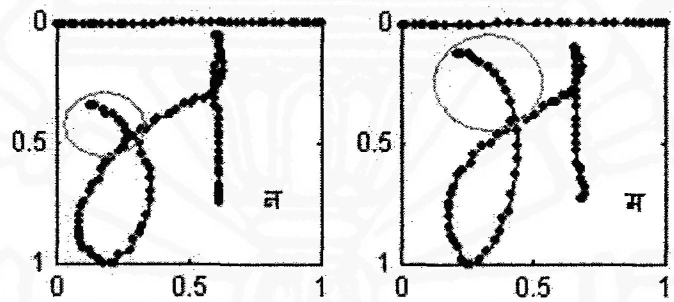


Figure 4.12 Misclassified example: न→म

4.2 Experiment VI

4.2.1 Dataset

The database was composed of 1800 characters from 36 classes of characters, where 25 Nepalese natives were used. Each writer had given a chance of writing two times per class of character. 15 writers were employed for training the system and remaining 10 writers were for testing. As, no directions, constraints and instructions were given to the users, the database was set up of completely from natural handwritings.

4.2.2 Results

Table 4.1 reveals the experimental results for both training and test datasets. Characters were tested one to one basis. Testing of same training characters on to the system confirmed that how good it was trained. Only 1% error rate was received from those training characters. This error rate confirmed that the classifier was trained with 99% intelligence. In the similar manner, 5% error rate was obtained from both consonants and vowels testing. An average recognition speed is 12 seconds per character. The recognition speed of the classifier is significantly reduced due to the technique used in template management, where spatial information about the strokes along with the number of strokes used in completing a character is considered and matching algorithm.

Table 4.3 Class-wise Experimental Results (Test Data)

Class	Recog. Char.s	Confusion	Rejection	Class	Recog. Char.s	Confusion	Rejection
Consonant							
क	17	2 (फ)	1	ख	20	0	0
ग	18	2 (स)	0	घ	20	0	0
च	19	1 (य)	0	छ	20	0	0
ज	20	0	0	झ	13	5(स)	2
ट	20	0	0	ठ	20	0	0
ड	20	0	0	ढ	20	0	0
त	18	1(व), 1(न)	0	थ	18	1(य), 1(च)	0
द	16	2(ढ), 1(ट)	1	ध	19	1 (घ)	0
न	18	1(म), 1(त)	0	प	20	0	0
फ	19	1(क)	0	ब	19	0	1
भ	19	1(म)	0	म	19	1(भ)	0
य	18	1(थ), 1(प)	0	र	20	0	0
ल	19	1(त)	0	व	19	1(त)	0
स	18	1(झ), 1(ग)	0	श	20	0	0
ह	20	0	0	क्ष	20	0	0
ज्ञ	20	0	0				
Vowel							
अ	18	1(भ)	1	ए	18	1(ग)	1
उ	19	0	1	ऊ	20	0	0
इ	18	2(ड)	0				

4.3 Discussions

One of the biggest problems, character similarity among many classes is significantly reduced by the help of spatial relation between/among the strokes within every class of character. Some of the similar characters' pairs like, (भ, म), (थ, य) and (ध, घ) etc. are classified correctly by using the strokes' spatial relation in addition to the small curve (dissimilar feature in pairs). It means the positions of the shirorekha of these pairs are at different locations. An example of improvement in accuracy of the classifier is shown in Fig. 4.8, where भ is not confused with another similar character म due to the distinguishing spatial information of 'shirorekha' even when the curve is not clear in the text. But, sometimes, the location of shirorekha is same for both characters, which can be found in Fig. 3.10. In such a case, the idea used in pre-processing: cusp elimination at both ascender and descender of a sequence, has advantages in classifying these similar characters. That is to say, this is helpful in case shirorekha in both the class of characters are at same location. Fig. 3.11 demonstrates the correct classification of similar characters by chopping at ascender. Similarly, Fig. 4.9 and Fig. 4.10 provide an idea how the confusion takes place. Not surprisingly, some of the characters pairs, having a lot of dissimilarity feature in printed format are also confused to one another. For example, Fig. 4.11, Fig. 4.12, Fig. 4.13 and 4.14 are the some of the examples to demonstrate.

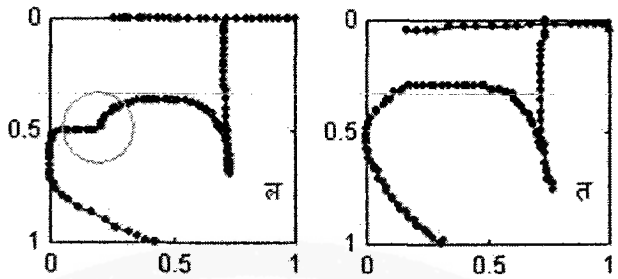


Figure 4.13 Misclassified example: ल→ त, which are looking different in printed format

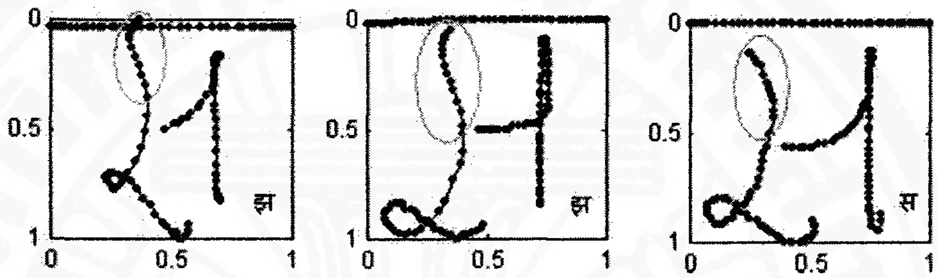


Figure 4.14: Confusion of झ with स even though a lot of dissimilar feature in between them in printed format (झ→स)

Writing a stroke at the end to complete the previous strokes is often called re-writing stroke. These strokes are helpful in giving perfect/complete graphical (off-line) representation but the information carried by only a re-writing stroke is nothing. This is still a problem in this classifier too. Complete information of the specific stroke will be exist if we combine the previous strokes with re-writing stroke. However, it is difficult for the characters, which have large number of strokes i.e., how to know which one is the re-writing stroke and which stroke is to be connected to the re-writing stroke. We only know the re-writing strokes after having 2D plot. Since re-writing strokes are not counted as the complete strokes, we did not have templates. Fig. 3.12 shows two samples of characters having re-writing strokes, which are misclassified. One of the ideas used in pre-processing: noisy sequence elimination, sometimes deletes the re-writing strokes, from which it has a chance of recognizing in testing. Due to this effect, number of misclassified characters with re-writing strokes is reduced.

With the help of statistical information about the strokes, some of the characters which are misclassified before due to their diminished and or very long ascender and descender feature are now classified.

Mis-writing character is another problem for the classifier, which does not give any shape of the character having unnecessary number of strokes.

Having both the accuracy of the classifier (Table 4.1) and the number of misclassified characters (Table 4.2) along with their types in hand, it is concluded that the reliability of the classifier is more than 95%.

In comparison to the classifier, where only the structural properties of the stroke are considered, it has higher intelligence with better reliability. However, the classifier is also limited to tremor handwriting, children handwriting and writing with re-writing strokes.