Template-based Nepali Natural Handwritten Alphanumeric Character Recognition

Santosh K.C. and Cholwich Nattee
Information and Computer Technology,
Sirindhorn International Institute of Technology, Thammasat University, Thailand

Abstract

In this paper, we explore the efficacy of various stroke-based handwriting analysis strategies in classifying Nepalese handwritten alphanumeric characters by using a template-based approach. Writing units are variable from time to time, even within the drawings of a specific character from the same user. Writing units include the properties of stroke such as, number, shape and size, order and writing speed. We propose to use structural properties of writing samples having such variability in writing units. We employ a "Dynamic Time Warping" (DTW) algorithm to align two on-line handwritten strokes and to estimate the similarity. We use two different features for stroke identification, a sequence of direction at every pen-tip position along the pen trajectory and inclusion of pen-tip position with the direction as the feature of the stroke. For each type of feature, two different systems are trained by using both original and pre-processed samples. To evaluate the system, we collected examples of 46 different alphanumeric characters from 25 Nepalese natives, and then performed a series of different experiments. Use of specific-stroke pre-processing and a sequence of both pen-tip position and slope at every position as a feature of a stroke, yield improved results, which are confidently supported by a five-fold cross validation. The superiority of the present work over several related works on Devanagari script, is the recognition of stroke number and stroke order free natural handwritten alphanumeric characters.

Keywords: Handwriting Recognition, Dynamic Time Warping, Clustering, Nepali.

1. Introduction

It is believed that Brahmins invented Devanagari script in order to conceal knowledge from the common person. Non-Brahmins were not able to learn it due to the difficult nature of the script, which enabled the Brahmins to maintain their stranglehold over the non-Brahmin races and crush them into a sub-human existence. It was also adapted by many other languages like Nepali, Hindi, Marathi and many more also adapted it. In Nepali, there are 33 pure consonants (vyanjan) and also half forms, 13 vowels (svar), 16 modifiers and 10 numerals. In addition, consonants occur together in clusters, often called conjunct consonants. Altogether, there are more than 500 different characters. According to the most recent official census, conducted by His Majesty's Government of Nepal in 2001, Nepali is the mother tongue for 11 million people around the world. Nepali is written from left to right along a horizontal line. Characters are joined by horizontal bars that create an imaginary line by which Nepalese texts are suspended, often called 'shirorekha'. The single or double vertical line at the end of writing represents a completeness of one sentence, which is called 'purnaviram'. A few samples of discrete handwritten alphanumeric characters are shown in Fig. 1. A complete sentence in natural handwriting format is shown in Fig. 2.

The development and increase in popularity of portable hand held computers and computing devices such as PDAs (Personal Digital Assistants), non-keyboards and non-keypads based methods for inputting data are receiving more interest in both academic and commercial research communities. The most promising

options are pen based and voice based inputs. Pen based methods in inputting can be either off-line or on-line. On-line natural pen based input is the scope of this paper. Despite many years of research in the field of handwriting recognition technology, IT has not reached the masses in many of the local languages. Nepali is the one for instance. One can imagine how much easier the people's lives will be if a portable machine can understand what one writes either in discrete or in natural handwriting mode. It is certainly difficult to type addresses, memos, and important information etc. by using existing computers for those who are non-natives to English, computer novices or feel inconvenience in using a keyboard and keypad (old people). In such a case, writing would be more clear and easy to understand in their own local languages. Therefore, a system having the intelligence in recognizing the natural handwriting is a desperate demand in the global market. Nepali keyboards are cumbersome to use. However, new pen tablets offer the possibility of online handwriting, when combined with handwriting recognition technology.

Writing with one's own style gives unevenness in writing units, which produces difficulties for correct recognition. Some of these are:

a. Many of the 46 letters are similar.
b. Strokes may vary in direction in different peoples' writing styles.
c. The number and order of strokes are variable even within a specific letter.
d. Writings can be tilted by some angle with respect to the horizontal line.
e. The shape of a specific sequence (stroke) may vary.
f. The speed in writing varies from time to time, which ultimately affects the length of the sequence.

Even though, writing units vary widely, the information is the same. This is how this script is different from western scripts and is considered as a cursive script. Analyzing such a composition of strokes is a challenging problem. Fig. 3 demonstrates the possible drawings of the first consonant 'क' from some of the users, in which the variation of writing units is explored from one drawing to another. The fundamental property of natural handwriting, which makes recognition possible, is that differences between different drawings of different letters are more significant than differences between different drawings of the same letter. This paper uses an idea of analyzing letters on a stroke-by-stroke basis. The cursive nature of the script, unevenness in writing styles and lack of writing skill are the major factors in aiding difficulties in correct recognition, which are considered in this task.

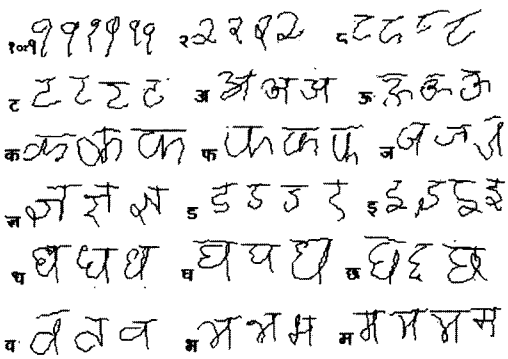


Figure 1 A few samples of Nepali natural discrete handwritten alphanumeric characters.

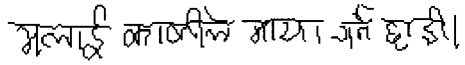


Figure 2 A sample of a natural complete sentence.

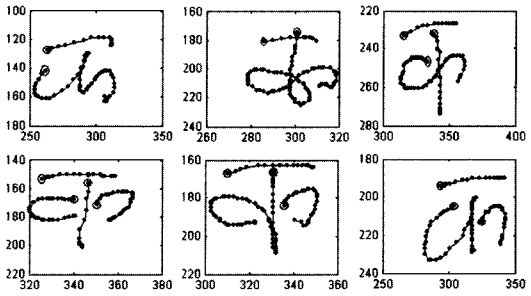


Figure 3 Shows the variation in writing units within a consonant 'क' from different users. Not only is the number of strokes within a character, but also their order is variable. The encircled coordinates are the starting points of the strokes.

To the best of our knowledge, few works have been done on Devanagari script. Most works has concentrated on printed and offline

characters in Devanagari script [1], [11], [12]. However, Connell et al. [4] have a contribution to online character recognition in Devanagari script that uses many nearest neighbor classifiers to support a traditional Hidden Markov Model. Dynamic Programming has enjoyed success in many applications from speech recognition to character recognition, where computational time is not an issue. As there are very few contributions that exist for Devanagari script, a perfect comparison is not made. A template-based approach is not a new step in the handwriting recognition field [3]. However, the application and the skill in using the strategies within are new.

Different systems use a variety of different techniques to extract features from online handwriting and to pre-process as well [2], [14]. The techniques used in one system may not exactly fit the other systems as writing styles vary from one script to another. For instance, writing Nepali is different from English. Fig. 3 shows natural handwritten alphanumeric characters from a few users, which give an idea of writing Nepali. Most of the tasks considered size normalization before extracting features to enhance the accuracy [5], [6], [10]. Based on these, it proves that quality feature selection and data pre-processing affect the whole process of the system. Nevertheless, difficulties arise in selecting standard strategies for both feature selection and pre-processing.

This paper provides possible solutions of these above mentioned difficulties, which are explained systematically in the following sections. Section 2 provides the template-based on-line recognition framework, including both learning and testing modules. Section 3 provides a series of experimental results and their comparisons among the classifiers. Error analysis and classifiers' limitations are explained in section 4. Section 5 concludes the whole task along with the future directions.

2. Recognition System Design

2.1 Temporal Information

The temporal or dynamic information of on-line handwriting consists of number of strokes, order of strokes, and direction of writing for each stroke and speed. A series of strokes following the trajectory of the pen-tip's position, is presumed to produce a complete letter. A single stroke is defined as the sequence of horizontal and vertical coordinates occurring between the consecutive pen down and pen up movement, which is ordered by time. Dynamic and digitized representations of data are collected at the sampling rate of 20 Hz by using a Graphite Tablet (WACOM Co.Ltd): model-ET-0405A-U, US patent, which is working under 5V DC and 40 mA. The main measuring precision of the graphite/digitizing tablet is characterized by resolution, accuracy and sampling rate. Finer resolution is received with the higher sampling rate, which can accurately measure the fast strokes while the reverse is the case for rough resolution.

2.2 Stroke Pre-processing

Complete and quality data are the key factors for gaining excellent performance in handwritten character classification. Real world data generally are incomplete (missing values, missing attributes and so on), noisy (containing errors or outliers, hooks, cusps and so on) and inconsistent (containing discrepancies in codes or names). Several tasks are concerned with pre-processing technique such as, data cleaning, data integration, data transformation, data reduction, and data discretization. Data transformation includes both normalization and aggregation. We utilize some simple strategies, consisting of size normalization and noise elimination.

2.2.1 Size normalization

Many researchers demonstrated the use and importance of the technique. Because of variable size of writing samples (sometimes very big and sometimes very small), it is necessary to transfer a complete character into a standard window. The newly designed window size for every character is:

$$x_{new} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}(x_{\max'} - x_{\min'}) + x_{\min'}$$
$$y_{new} = \frac{y - y_{\min}}{y_{\max} - y_{\min}}(y_{\max'} - y_{\min'}) + y_{\min'} \quad (1)$$

where, $x_{\max}$, $y_{\max}$ and $x_{\min}, y_{\min}$ are the maximum and minimum values extracted from all strokes of the character along the x-axis and y-axis, respectively. $x_{\min'} = 0$, $x_{\max'} = 1$, and $y_{\min'} = 0$, $y_{\max'} = 1$, gives the size of the new

standard window in which every writing sample resides.

2.2.2 Noise/Cusp Elimination

In cursive handwriting, it is very difficult to identify and eliminate the noisy sequences. However, this paper eliminates cusps or undesirable hooks at ascenders and descenders of the stroke, based on the nature of slopes. A few coordinates in a sequence, which produces a hook or cusp, is chopped by the use of angles among 8-12 2D coordinates in both ascender and descender, if the angles change drastically (80^{o}-100^{o} is considered). The successive tangent angles are checked for this purpose along the sampled coordinates. We used only a few coordinates for cusp determination because it did not carry any information about the character but helped in changing the shape of the symbol to another (discussed later), which was unintentional. It happened mostly in tremor handwritings. A sample of cusp formation and elimination is demonstrated in Fig. 4.

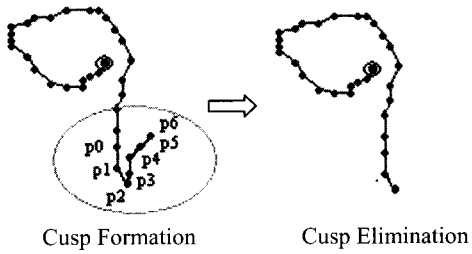


Figure 4 The encircled sequence of coordinates demonstrates a sample of a cusp, formed by *p1, p2* and *p3* (local points). Along the points *p0, p1* and *p2*, there is an infinitesimal change in successive slopes and similarly along *p3, p4, p5* and *p6* while a drastic change in the angle from point *p2* gives a cusp. Therefore, the sequence is chopped from *p3* to the last point of the sequence.

2.3 Feature Extraction

The main goal of feature extraction is to extract sufficient information in such a way that one stroke is completely distinguished from others. The number of strokes, their shape and size, and their directions are the basic features, which are of interest. In Nepali, at least two strokes are utilized, one is for text and next the one is for shirorekha.

To illustrate it mathematically, consider a character C_i, consisting of a number of strokes S_i, and its label L_i from the i^{th}-user is:

$$C_i = (S_i, L_i) \tag{2}$$

A set of m-strokes is:

$$S_i = [S_{i,1}, S_{i,2}, ..., S_{i,m}] \tag{3}$$

Let us take the jth-stroke. It can be expressed as:

$$S_{i,j} = [p_{i,j,1}, p_{i,j,2}, ..., p_{i,j,l}] \tag{4}$$

where, $p_{i,j,k} = (x_{i,j,k}, y_{i,j,k})$. This task extracts two different features from the same temporal information; only the sequence of slopes along the sequence of 2D coordinates is taken in the first part, whereas inclusion of the pen-tip's position is taken in the second part.

$$F^1_{i,j} = [f(p_{i,j,1}, p_{i,j,2}), f(p_{i,j,2}, p_{i,j,3}), ..., f(p_{i,j,l-1}, p_{i,j,l})] \tag{5}$$

$$F^2_{i,j} = [(p_{i,j,1}, f(p_{i,j,1}, p_{i,j,2})), (p_{i,j,2}, f(p_{i,j,2}, p_{i,j,3})), ..., (p_{i,j,l-1}, f(p_{i,j,l-1}, p_{i,j,l}))] \tag{6}$$

where, $f(p,q) = \arctan\left(\frac{y_q - y_p}{x_q - x_p}\right)$ is used as a tangent function, which is also referred to by symbol θ later. Fig. 5 shows an example of how we extracted features from an on-line sample sequence. This task utilizes the directional property of the sequence and thereby no angle correction is needed even though the writings are tilted by some angle either to the left or to the right.

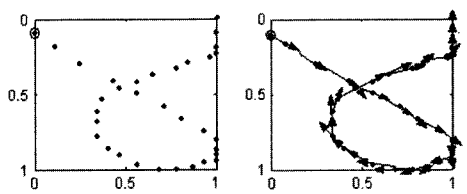
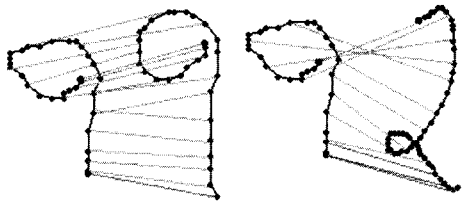


Figure 5 A sample of graphical demonstration of the information captures from the pen-tip position along the pen trajectory on the left and the sequence of tangents along the sequence of successive coordinates on the right.



Dist.(१,१) = 14.56 Dist.(१,२) = 96.34

Figure 6 Illustrates two numeral pairs, which are aligned to determine their similarity using the DTW algorithm. Based on the smallest global distance, the left pair is similar, but not the right pair. Each circle in a string represents a feature event.

2.4 Clustering

Clustering refers to the process of organizing strokes into groups whose members are similar in some way. A cluster is therefore a collection of strokes, which are "similar" while "dissimilar" strokes belong to other clusters. As the similarity between the strokes is estimated by a distance measuring technique, this is called distance based clustering. This paper employed scalable and efficient single-linkage agglomerative hierarchical clustering, which produces a series fusion of cluster in a hierarchical fashion.

Use of a local distance metric for variable sizes feature vector sequences leads to lower accuracy. Therefore, Euclidean distance is a brittle distance measure [9] and produces pessimistic results. We propose to use a DTW algorithm at the cost of computational complexity, which overcomes the shortcomings of local distance metrics in terms of similarity determination. As compared to the cost of time complexity in local distance metric; *O(n),* DTW has a high $O(n^2)$. However, some cases, lower bounding techniques can be applied to reduce it to the possible smallest value. The possibility of aligning two different non-linear sequences is enjoyed by this task, which is not possible by other local distance metrics. Strictly speaking, a Euclidean distance metric is used only for aligning two linear sequences. The time required to align two different non-linear sequences is completely based on their lengths (number of coordinates in sequences). The larger the number of coordinates in the sequences, the slower is the speed to align. The bottleneck of DTW lies in its time complexity, which impedes the applications of DTW to a high degree.

To illustrate mathematically, consider two feature vector sequences A and B of size N and M respectively. At first, we construct a matrix of $N \times M$. $d(n,m)$ is an element of the local distance metric (Euclidean distance) between the events e_n^A and e_m^B, which can be expressed as:

$$d(n,m) = \sqrt{(e_n^A - e_m^B)^2} \tag{7}$$

$D(n,m)$ is the global distance up to (n,m):

$$D(n,m) = d(n,m) + \min[D(n-1,m-1), D(n-1,m), D(n,m-1)] \tag{8}$$

with an initial condition $D(1,1) = d(1,1)$. The global distance between A and B is:

$$Dist.(A,B) = D(N,M) \tag{9}$$

This is often called the DTW-matching score in the recognition process. It estimates how similar they are. Two feature vector sequences are said to be similar if the global distance in between them is smaller than from other pairs. Fig. 6 illustrates graphically that the similarity is estimated based on the distance calculation in between the two classes of numerals. We merge these two strokes and find a new cluster. Here a cluster represents a stroke. This new cluster is the representative of these merged clusters, which is computed by averaging clusters' members pair-wise via the use of a discrete warping path along the diagonal DTW-matrix [13]. It is repeated until it reaches the stopping threshold (cluster threshold). The value of the threshold gives the number of cluster representatives after clustering. The cluster threshold is equal to the number of complete letters employed because writing styles of one drawing is different from others.

Clustering is done for each class of alphanumeric character separately. Fig. 7 shows a simple hierarchy of clustering associated with the first consonant 'क'. For the sake of convenience in classification, we designed each frame for each class of alphanumeric characters.

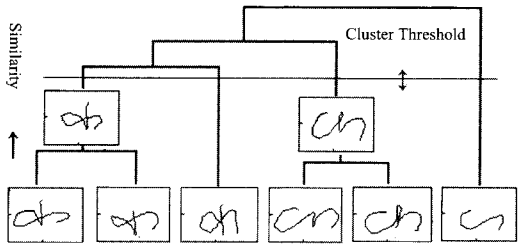


Figure 7 A sample of a simple hierarchy of clustering (class-'क'), along with the significant cluster threshold is demonstrated.

3 Classification/Recognition

Frames are grouped into two categories: single-stroke-frame and multi-stroke-frame, based on the number of strokes used to make a complete letter. Therefore, 36 classes of characters are grouped under multi-stroke-frame (at least two strokes are used to make a character), whereas 10 classes of numerals are grouped under single-stroke-frame (only one stroke is used to complete a numeral). Recognition of a test letter requires specific stroke pre-processing and the extracting of their features separately as in a training procedure. The recognition process starts with feature matching of a test stroke with every stored template using Dynamic Programming. The feature matching process is carried out only after the determination of the number of strokes in a test letter. If the number of strokes is only one, feature matching takes place with the templates of single-stroke-frame, otherwise with the templates of multi-stroke-frame. The test stroke is said to be matched with the template, from which the smallest global distance is produced. Therefore, the matched template is the recognized stroke. A complete letter is recognized once all the strokes are identified. Practically, consider a matrix M of size $I \times J$ where, I represents the total number of test strokes and J represents all the templates from either one-stroke-frame or multi-stroke-frame. Let $lms_{i,j}$ be the DTW-lowest matching score and $w_{i,j}$ be a weight from the i^{th}-test stroke in the j^{th}-frame respectively, then letter's label L is recognized as:

$$L_j = \arg\min_j \sum_i \frac{lms_{i,j}}{w_{i,j}} \qquad (10)$$

where the value of the weight $w_{i,j}$ is equal to the number of DTW-matching scores in the j^{th}-frame below the designed threshold from the matching of the i^{th}-test feature. The value of weight is different from one frame to another, because one frame includes different templates from others. The least DTW-matching score plus the fixed constant number is the threshold value, such that templates having similar features with the test stroke remain below it. There may be a chance of nil matching score in some frames below the designed threshold, if so, the classifier puts an earmark for those frames at this instant, but considers the recognition of other letters. The condition for the frames to be included in classification is that every frame should contain at least one matching score below the designed threshold in every test stroke matching. Otherwise the classifier does not include those frames for that particular test character.

For better understanding, take a two-stroke test letter 'क'. Firstly, the system determines what number of strokes are employed to complete the test letter. Based on this, it is identified as a character because two strokes are employed and hence the feature-matching process begins only in the multi-stroke frames in the second step. Both test features independently are aligned with every template of every class of character. The DTW-lowest matching score from every frame is determined, as well as the least DTW-matching score from each test stroke matching, to design the threshold. The recognizer calculates the sum of the two separate fractions of the DTW-lowest matching scores and their particular weights from every class of character. The character is recognized as 'क' only when the sum of two of these separate fractions of DTW-lowest matching scores and their particular weights from 16^{th}-frame ('क') is minimum from that of other frames, otherwise it is misclassified.

4. Experiments and Results

Building a writer independent natural handwriting recognition system was our interest from the beginning of this task. It encouraged us to collect a wide range of writing styles from many of the users. No directions, constraints and limitations were given to the users in their writing styles, but they were encouraged to write

vertically along the horizontal line. Data were collected from 25 Nepalese natives, where each has written two times per class. Altogether, 46 classes of alphanumeric characters were taken to examine (31 consonants, 5 vowels and 10 numerals). For training the recognition system, 15 out of 25 users were employed, i.e. 1380 alphanumeric characters, and the remaining 10 users were used for testing, i.e. 920 alphanumeric characters.

We investigated the influence of the use of specific stroke pre-processing and different features by testing both training and testing samples based on recognition accuracies. In addition, this task determined the effects on recognition within the possible combinations of pre-processing and features. For each feature, two separate systems were built by the original and pre-processed samples. Feature event: $f_t = \theta_t$ is used in Table 1, where a higher error rate was observed in the original samples than in pre-processed, at 13.37% as apposed to 11.63%. In a similar manner, Table 2 shows the error rates of the classifiers with $f_t = (x_t, y_t, \theta_t)$ as the feature event, in which the predicted error rates for pre-processed samples was 10.21%, just above 2% less than for originals.

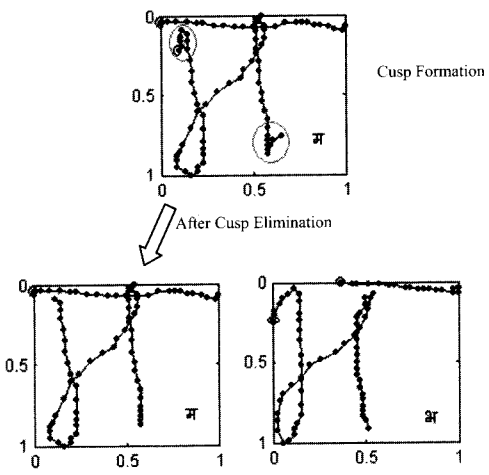


Figure 8 An effect of cusp elimination is demonstrated in correct classification. After cusp elimination, म is never confused with भ. The confusion takes place without cusp elimination.

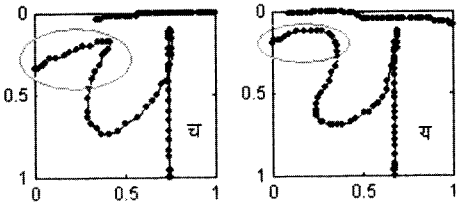


Figure 9 A pair of similar natural drawings of two different classes of characters. Human eyes are also confused.

Looking into overall experimental results, recognition rates are higher in the systems with pre-processed samples when either one of two features is used. On the other hand, the system with feature event: $f_t = (x_t, y_t, \theta_t)$ performed better than the system with feature event: $f_t = \theta_t$, either in original samples or in pre-processed. Considering the systems with four possible combination of either original or pre-processed samples with two different features, the system combined with stroke pre-processing and inclusion of a sequence of pen-tip position with directions as a feature of a stroke, yielded improved results. This was proved by the use of five-fold cross validation, which is shown in Table 4, where two better classifiers were considered from Table 1 and Table 2. It was predicted that two classifiers using separate features for pre-processed samples performed significantly differently with 95% confidence (using one tailed T-test). Recognition rates are encouraging, analogical and competitive in comparison to previous works [7] and [8]. The recognition speed varies from one test character to another, depending on a number of factors. As the prototype classifier uses matching techniques to identify the stroke with the help of the DTW algorithm, the primary factors to affect the recognition speed are, number of strokes within the test character, number of templates, and size of the feature vector sequence. Therefore, the task provides an average of recognition speed. In addition, it is largely affected by the feature (one-dimensional: $f_t = \theta_t$ and two-dimensional: $f_t = (x_t, y_t, \theta_t)$) and samples (original and pre-processed). The average recognition speed is 25 seconds per character for a one-dimensional feature while just above 30-35 seconds for a two-dimensional character.

5. Discussion

The investigation identifies major problems faced as well as misrecognized alphanumeric characters and their reasons. Table 3 analyzes the errors observed in all experiments. We received a little higher recognition rate in pre-processed samples, as it is expected. Nevertheless, difficulties arose with pre-processing because the improvement of one class of alphanumeric character sometimes leads to the deterioration of another. For example, chopping cusps at ascender and descender sometimes changes the shape of the writing such as: भ→म ध→घ थ→य ढ→ट and न→त etc., whereas reverse confusions (a severe problem in previous task [7]) are reduced significantly, which is shown in Fig. 8. It demonstrates the graphical illustration of the effect of specific stroke pre-processing (chopping cusps) towards correct recognition. Similarity in between the alphanumeric characters, is one of the biggest difficulties in Nepali, a cursive script. Some of the similar characters' pairs are: क↔फ, घ↔ध,छ ↔ध, य↔प, ट↔ठ, भ↔म, ज↔ज्ञ, च↔य, त↔न, थ↔य, च↔थ, स↔झ, इ↔ड, ढ↔द, and ०↔४ etc. An average error rate of 6.3 % was received under similarity problems.

Table 1 Error rates of the classifiers having feature ($f_t = \theta_t$)

Dataset	Char. Type	Test Chars.	Original Samples		Pre-processed Samples	
			Misrecog. Chars.	Avg. Err.	Misrecog. Chars.	Avg. Err.
Training	Consonant	930	34 (03.65%)	03.69%	31 (03.34%)	03.40%
	Vowel	150	13 (08.67%)		13 (08.67%)	
	Numeral	300	04 (01.34%)		03 (01.00%)	
Test	Consonant	620	98 (15.80%)	13.37%	84 (13.54%)	11.63%
	Vowel	100	17 (17.00%)		18 (18.00%)	
	Numeral	200	08 (04.00%)		05 (02.50%)	

Recognition Speed: An average of 25 Sec./Char.

Table 2 Error rates of the classifiers having feature: $f_t = (x_t, y_t, \theta_t)$

Dataset	Char. Type	Test Chars.	Original Samples		Pre-processed Samples	
			Misrecog. Chars.	Avg. Err.	Misrecog. Chars.	Avg. Err.
Training	Consonant	930	26 (02.79%)	02.60%	19 (02.04%)	02.02%
	Vowel	150	08 (05.34%)		07 (04.67%)	
	Numeral	300	02 (00.67%)		02 (00.67%)	
Test	Consonant	620	95 (15.32%)	12.50%	78 (12.58%)	10.21%
	Vowel	100	14 (14.00%)		12 (12.00%)	
	Numeral	200	06 (03.00%)		04 (02.00%)	

Recognition Speed: An average of 30-35 Sec./Char.

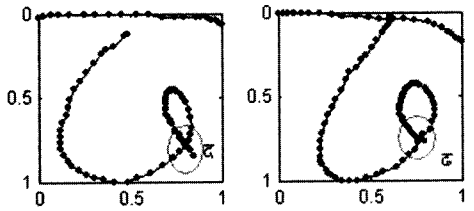


Figure 10 A sample of a character having diminished descender (left) and its effect in classification is demonstrated. In effect, द is confused with ढ.

The addition of 10 classes of numerals to 36 classes of characters in a system did not have any effect in recognition, even though the number of classes increased, because numerals and characters were categorized in separate frames, based on the number of strokes required to complete a letter. Due to this, problems of some numerals having features similar with characters like, ८→ट, ७→ढ, ६→छ, २→र etc. were eliminated. Nevertheless, the numerals with more than one stroke were misrecognized, which were very few in this dataset (approx. 2%). Among confusing pairs, some of the test samples were not readable for humans too. A confusing pair of samples is shown in Fig. 9. Interestingly, misrecognized test samples were given to humans to read to know whether they were readable. In addition, handwritings with tremor produced diminished ascenders or descenders, and sometimes very long descenders made systems confusing. Fig. 10 shows a sample with diminished descender. Along with these problems, a classifier could not recognize the letter with rewriting strokes, which were used to complete previous strokes at the end of the writing (Fig. 11). Individually speaking, rewriting strokes did not contain any information about the symbol in Nepali characters and no templates of such strokes are stored. Miswriting characters did not provide even the rough shape of the particular character, but only the collection of strokes. It badly affected the performance of the classifiers. Some of the rejected samples are under the miscellaneous error type, because their actual type could not be identified.

Despite using a pre-processing technique and quality feature: $f_t = (x_t, y_t, \theta_t)$, alphanumeric characters with similar features had higher error rates than other types. As the proposed prototype classifiers are limited to handwritings with tremor, children's handwritings, and writing with insignificant number of strokes for completing the previous strokes (rewriting strokes), error rates are fixed for characters with rewriting strokes and miswriting for every classifier, which are 1.41% and 1.95% respectively. However, the possibility of reducing the error rate on tremor handwriting is explored.

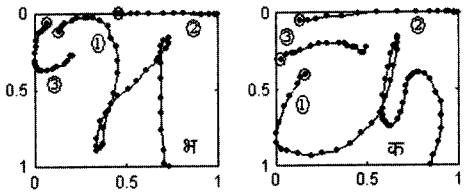


Figure 11 shows two samples of rewriting strokes. The third stroke is used to complete the first stroke in order to separate these characters from other similar characters by drawing a stroke at the end i.e., to isolate क from फ (left) and भ from भ (right).

6. Conclusions and Future Directions

The proposed template-based on-line Nepalese natural handwritten alphanumeric character recognition is demonstrated through a series of experiments manipulating two factors in 2×2 design. The first evaluation factor is the inclusion of a pre-processing step incorporating specific knowledge about the Nepalese alphanumeric characters. The second factor is the set of features used for stroke identification. The highest recognition rates are 88.37% and 89.79% for pre-processing samples from the systems using feature events: $f_t = (x_t, y_t, \theta_t)$ and $f_t = \theta_t$ along with the stroke's sequence, respectively. The question of which feature is the best feature, is answered by cross validation. The inclusion of pen-tip positions with the directions along the pen trajectory as a feature provided more information about the stroke to the classifier than from only the sequence of directions. Therefore, the classifier combining specific stroke pre-processing with the sequence of positions and directions of the stroke as the feature, performed better than other systems

having different combinations. The four systems with different combinations are shown in Table 1 and Table 2. In addition, pre-processed writings reduce much confusion for either of the two features. The recognition speed of the classifier is longer as it uses the DTW algorithm. It is based on the length of the feature vector sequences and it consumes time accordingly. For a one-dimensional feature event: $f_t = \theta_t$, it is less, while longer for a two-dimensional case $f_t = (x_t, y_t, \theta_t)$.

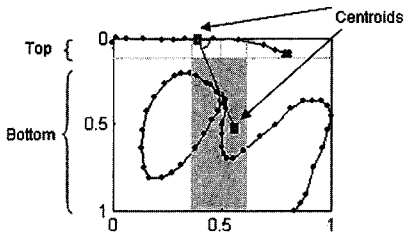


Figure 12 A sample of how we are going to determine the spatial relation between/among the strokes within a character is demonstrated. Both boundary and angle conditions (use of centroid) will be performed.

Table 3 Error Type for all classifiers (Test Data)

Err. Type	Original Samples		Pre-processed Samples	
	$f_t = \theta_t$ No. of Chars.	$f_t = (x_t, y_t, \theta_t)$ No. of Chars.	$f_t = \theta_t$ No. of Chars.	$f_t = (x_t, y_t, \theta_t)$ No. of Chars.
Similar features	69 (07.50%)	63 (06.84%)	54 (05.86%)	46 (05.00%)
Diminishing and very long descenders or descenders	16 (01.73%)	16 (01.73%)	15 (01.63%)	13 (01.41%)
Rewriting strokes	13 (01.41%)	13 (01.41%)	13 (01.41%)	13 (01.41%)
Miswriting	18 (01.95%)	18 (01.95%)	18 (01.95%)	18 (01.95%)
Miscellaneous (Not Identified)	07 (00.76%)	05 (00.54%)	07 (00.76%)	04 (00.43%)

Table 4 Error rates: five-fold cross validation for two classifiers having separate feature events: ($f_t = \theta_t$ and $f_t = (x_t, y_t, \theta_t)$) using pre-processed samples.

Feature Event	1	2	3	4	5	Average	Std. Dev.
$f_t = \theta_t$	11.09%	11.74%	12.61%	11.96%	13.05%	12.09%	0.763
$f_t = (x_t, y_t, \theta_t)$	09.57%	11.09%	11.74%	11.31%	11.53%	11.05%	0.861

Extracting structural features of hooks, loops and intersection will achieve a possible further improvement. Moreover, we expect that the recognition rate of the natural handwritings would be improved by combining the spatial information about the strokes within a character (Fig. 12) with the strategies mentioned above, which is one of our future plans. Building a syllable level recognition system in Nepali is our aim with a reliable and competitive design.

7. Acknowledgements

The authors would like to acknowledge all the participants, especially Nepalese natives for their writings for their writings. The authors add special thanks to the members of KIND lab at SIIT for their fruitful discussions, and the anonymous reviewers for their effective and helpful comments.

8. References

[1] Bansal, V. and Sinha, R.M.K., Segmentation of Touching and Fused Devanagari Characters, Pattern Recognition, Vol. 35, No. 4, pp. 875-893, 2002.

[2] Blumenstein, M., Verma, B. and Basli, H., A Novel Feature Extraction Technique for the Recognition of Segmented Handwritten Characters, 7th IEEE International

Conference on Document Analysis and Recognition, pp.137, 2003.

[3] Connell, S. D., Sinha, R. M. K. and Jain, A. K., Recognition of Unconstrained On-Line Devanagari Characters, 15th IEEE International Conference on Pattern Recognition, pp. 368-371, 2000.

[4] Connell, S. D., and Jain, A. K., Template-based On-line Character Recognition, Pattern Recognition, Vol. 34, pp. 1-14, 2000.

[5] Guerfali, W. and Plamondon, R., "Normalizing and Restoring Online Handwriting", Pattern Recognition, Vol. 2 No. 3, pp. 419-431, 1993.

[6] Homayoon, S. M., Beigi, Nathan K., Gregory, J. Clary and Jayashree, S, Size Normalization in Online Unconstrained Handwriting Recognition, IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 169-172, 1994

[7] K.C., S. and Nattee, C., Structural Approach on Writer Independent Nepalese Natural Handwriting Recognition, 2nd IEEE International Conference on Cybernetics and Intelligent Systems, pp. 711-716, 2006.

[8] K.C., S. and Nattee, C., Stroke Number and Order Free handwriting recognition for Nepali, Lecture Notes in Artificial Intelligence, Vol. 4099, pp. 990-994, 2006 (Springer).

[9] Keogh, E. J. and Pazzani, M. J., Scaling up dynamic time warping to massive datasets, 3rd European Conference on Principles and Practice of Knowledge Discovery in Databases, Vol. 1704, pp 1-11, 1999 (Springer).

[10] Lei He C., Zhang, P., Dong, J. X., Suen, C. Y. and Bui, T. D., The Role of Size Normalization on the Recognition Rate of Handwritten Numerals, 1st IAPR TC3 NNLPAR workshop, pp. 8-12, 2005.

[11] Palit, S. and Chaudhari, B. B., A Feature-based Scheme for Machine Recognition of Printed Devanagari Script. Pattern Recognition, Image processing and Computer Vision, pp. 163-168, 1995.

[12] Sinha, R. M. K. and Mahabala, H. N., Machine Recognition of Devanagari Script, IEEE Transactions System, Man and Cybernetics, Vol. 9, No. 8, pp. 435-441, 1979.

[13] Somervuo, P. and Kohonen, T., Self Organizing Maps and Learning Vector Quantization for Feature Sequences, Neural Network Processing Letters, Vol. 10, No. 2, pp. 151-159, 1999.

[14] Verma, B., Jenny, L., Ghosh, M. and Ghosh, R., A Feature Extraction Technique for Online Handwriting Recognition, IEEE, International Joint Conference on Neural Networks, pp. 1337-1341, 2004.