Development of Nepali Character Database for Character Recognition based on Clustering

Aadesh Neupane
Department of Computer Science and Engineering
Kathmandu University
Dhulikhel, Kavre, Nepal

ABSTRACT
Character Recognition tasks requires large set of reliable dataset to apply recognition algorithms and generate efficient models out of them. In case of Nepali language, no such character dataset exists for character recognition research, at least in the public domain. Nepali language has 36 consonant characters, 12 vowels character and each vowel character can modify each consonant characters. In this regard, there can be total of 446 characters including Nepali numeric characters. So, manually creating dataset for Nepali characters requires tons of effort, cost and time. In this paper, an elegant way of creating Nepali character dataset using semi-supervised clustering approach is described which minimizes effort and time.

Also, optimization is done on existing segmentation algorithm [1] to segment Nepali characters for both handwritten and scanned Nepali text. Complex features are extracted from these segmented characters by applying Discrete Cosine Transform and Wavelet transform. Thus, these extracted features are used to create database of Nepali characters using phash and k-means cluster. Presently, the database contains 38,493 characters distributed among 52 different clusters.

General Terms
Pattern Recognition, Character Segmentation

Keywords
Nepali Character Segmentation, Nepali Character Database, Nepali Character Recognition, Nepali Character Clustering.

1. INTRODUCTION
Most of the people encounter various text scripts around the world each of them with unique features. Some are written from left to right and some from top to bottom. Among, them Nepali language is our interest as it is official language of Nepal. Nepali language is similar to Devanagari scripts like Bengali and Hindi which contains straight line on top of words. Characters in a single word have a connected headline, where as different words in a sentence have disconnected headline [1]. So, these connected headline make it complicated to segment word and get resulting individual characters. The techniques devised for other language doesn't work well in case of Nepali text [2] [7]. More over, segmentation techniques used in printed Devanagari script doesn't perform well in case of handwritten Nepali script [3]. So, to create dataset to include all these variations would take lots of time and resource. In addition, the resulting database created manually would not be efficient to be used as benchmarking dataset.

The objective of this paper is two-fold. One is to give a general algorithm for Nepali character segmentation for both printed and handwritten texts. Another is to use this general segmentation algorithms for clustering purpose. To meet these objectives, around 100 pages of both printed and handwritten Nepali text were collected. To the best of our information, no standard dataset is available - at least in public domain - for benchmark studies for Nepali script based document recognition. Unfortunately, research groups in Nepali script recognition have paid very little attention to the importance of standard dataset. In most of the cases [4] [5], they tested their algorithms on manually generated dataset or on unprofessionally collected samples. This might be a reason for very limited progress in Nepali script recognition as compared to other languages. Thus to overcome this shortcoming, this approach of automated Nepali character database creation would be ideal choice.

This paper describes segmentation algorithm, along with the challenges in section 2. Then it presents feature definition, extraction process, image hashing and clustering ensemble details in section 3, experimentation description in section 4 and conclusion in section 5.

2. THE SEGMENTAION
2.1 Nepali Character Set
Nepali language is similar to Devanagari scripts like Bengali and Hindi which contains straight line on top of words. Characters in a single word have a connected headline, where as different words in a sentence have disconnected headline and has no lower case and upper case identification. Nepali language possesses the basic character set with vowels and consonants, and that vowels act as modifiers when combined with consonants.

Examples:
Vowels
अ आ इ ई उ ऊ ए ऐ ओ औ

Consonants
क ख ग घ च छ ज झ ण प फ ब भ ञ

Modifiers
ा े ौ ि ी ू

Modifiers with Consonants
का ि क क कु कू के कै को कौ कँ

Nepali scripts has a horizontal line, which connects all the characters. This connector is called '*Shirorekha*' or headline. Based on the horizontal lines, Nepali scripts can be divided into three zones: top zone, core zone and bottom zone. The core and top zone are separated by the header line. Figure 1 shows the image of a word that contains four characters with both lower and upper modifiers.

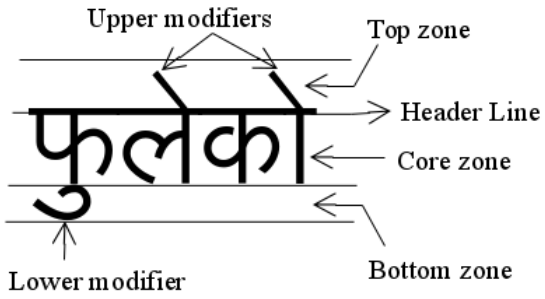
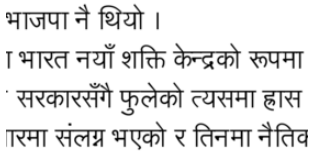
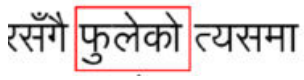


Figure 1 : Different zones of Nepali Word

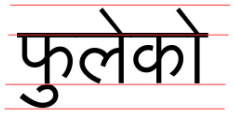
2.2 Illustration of Segmentation Process



Stage I) Multi lines documents are broken into individual lines based on the top zone and bottom zone of each line.



Stage II) After fragmenting into individual lines, each words are segmented from that line



Stage III) For each word, top zone, core zone and bottom zone is identified.

Stage IV) Top and bottom modifiers are removed. Also, header line is removed.

Stage V) Individual characters are extracted based on segmentation lines

Figure 2 : Illustration of Nepali Character Segmentation Process

From the above illustration, the hierarchy of algorithms for segmentation of Nepali characters from multiline documents can be noticed. In general, following hierarchy can be defined

1. Algorithm to split multiple lines text.
2. Algorithm to split physically disconnected words in a line.
3. Algorithm to remove top and bottom modifiers.
4. Algorithm to split each characters.

Considering the above in mind, implementation modules have been designed. All the algorithms defined in this section are applied to binarized image of Nepali text.

Algorithm 1 *Row Segment :* A document can have multiple or single line of text. These lines have a starting row and ending row boundary. The starting and ending row marker can be identified with the following algorithm

1. Calculate the sum of each row of the document and store it in a vector *RSUM*
2. Mark the starting row marker of particular line in the documents if after series of continuous zero in *RSUM* , non zero value in *RSUM* is found.
3. Mark the ending row marker of particular line in step 2 if after series of continuous non zeros, zero value in *RSUM* is found.
4. Go to step 2 until all *RSUM* is not processed.

Algorithm 2 *Column Segment :* After, lines are separated using *Row Segment* algorithm, different words in the line has be fragmented. Similar to *Row Segment*, The starting and ending marker can be identified for words with the following steps.

1. Calculate the sum of each column of the line and store it in a vector *CSUM* .
2. Mark the starting row marker of particular word in the line if after series of continuous zero in *CSUM,* a non zero value in *CSUM* is found.
3. Mark the ending row marker of particular word in step 2 if after series of continuous non zeros, zero value in *CSUM* is found.
4. Go to step 2 until all *CSUM* is not processed.

Algorithm 3 *Zone Segment :* As, already describe in section 2.1, Nepali words are divided into three zones. For segmenting the words into these zone , at first, header line need to be identified. Then, these zonal markers can be marked for words with the following steps.

1. Calculate the sum of each row of word and store it in *RSUMW*
2. Find the maximum value *MAX* in the vector *RSUMW* and its associated index as *MPOINT*
3. Starting from *MPOINT* traverse until half the value of *RSUMW* is greater than the current *RSUMW*. Mark the this ending row index *MPOINT1*
4. The headline starts for the word from index *MPOINT* and ends in *MPOINT1*. The header line width is *MPOINT* - *MPOINT1*.
5. Find the minimum vale *MIN* in the vector *RSUMW* and its associate index as *BPOINT*
6. Traverse *RSUMW* in the reverse order until the value of *RSUMW* is equal to *BPOINT*. If such values exists replace *BPOINT* with current index of *RSUMW*.
7. The zone above *MPOINT* is upper zone, the zone below the *BPOINT* is lower zone and the zone between *MPOINT1* and *BPOINT* is core zone.

Algorithm 4 *Character Segment :* Extraction of core zone is done using *Algorithm 3*, and upper and lower modifiers have been removed using *Algorithm 2*. Now, for segmenting these set of characters, identification of the markers which separate them from each other should be done. The starting and ending character markers can be identified using *Algorithm 1*. To segment fused Nepali characters, Multi-Factorial Analysis [9] and Collapsed Horizontal Projection [5] [8] is used.

1. Calculate the sum of each column of the line and store it in a vector *CSUMW* .
2. Mark the starting row marker of particular word in the line if after series of continuous zero in *CSUMW* , *as* a non zero value in *CSUMW* is found.
3. Mark the ending row marker of particular word in step 2 if after series of continuous non zeros, zero value in *CSUMW* is found.
4. Go to step 2 until all *CSUMW* is not processed.

2.3 Challenges

During Nepali character segmentation process as described above, most of the characters are properly segmented characters. So, some of the characters were segmented from conjunction resulting that consonant in half form [5]. If all the consonants in half form are included, the cluster size would grow and cluster accuracy would decrease. Thus, the half form consonants are removed after segmentation process.

3. FEATURES EXTRACTION AND CLUSTERING

In pattern recognition, a feature is an individual measurable heuristic property of a phenomenon being observed. Choosing discriminating and independent features is key to any pattern recognition algorithms being successful. Different types of numeric as well structural features are already being used in Devanagari Character recognition [4] [6] [10] [11]. Mostly these features are used for character recognition and will not perform well for clustering. So, new features set is devised for Nepali character clustering using below described algorithms.

3.1 Features Definition

After, character segmentation, features for each character image are extracted based on Discrete Cosine Transform (DCT) [12], Wavelet transform [15] and image Perceptual hashes (phash) [13] [14]. A brief definition for each feature extraction is given below.

Definition 1 : The discrete cosine transform of a list of *n* real numbers *s(x), x=0, ...,n-1*, is the list of length *n* given by:

$$S(u) = \sqrt{2/n}\, C(u) \sum_{x=0}^{n-1} s(x) \cos \frac{(2x+1)u\pi}{2n} \quad u = 0, .. n$$

$$where C(u) = 2^{-\frac{1}{\sqrt{2}}}$$
$$= 1 \quad otherwise$$

Definition 2 : The discrete wavelet transform (DWT) is an implementation of the wavelet transform using a discrete set of the wavelet scales and translations obeying some defined rules. In other words, this transform decomposes the signal into mutually orthogonal set of wavelets, which is the main difference from the continuous wavelet transform (CWT), or its implementation for the discrete time series sometimes called discrete-time continuous wavelet transform (DT-CWT).

The wavelet can be constructed from a scaling function which describes its scaling properties. The restriction that the scaling functions must be orthogonal to its discrete translations implies some mathematical conditions on them which are mentioned everywhere, e.g. the dilation equation transform

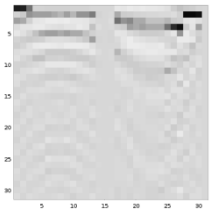
$$\theta(x) = \sum_{-inf}^{inf} a_k \theta(S_x - k)$$

where S is a scaling factor (usually chosen as 2). Moreover, the area between the function must be normalized and scaling function must be orthogonal to its integer translations.

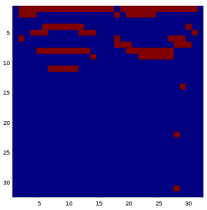
Definition 3 : Hash functions are an essential mathematical tool that are used to translate data of arbitrary size into a fixed sized output. There are many different kinds of these functions, each with their own characteristics and purpose. Perceptual hashes are another category of hashing functions that map source data into hashes while maintaining correlation. These types of functions allow us to make meaningful comparisons between hashes in order to indirectly measure the similarity between the source data.

3.2 Illustration of Features Extraction Process

Step I) Segmented Nepali Character (□)



Step II) Character Image after applying Discrete Cosine Transform (DCT)



Step III) Character Image after applying perceptual hashing

Figure 3 : Illustration of features extraction process

3.3 Features Extraction

The character images which were collected using the segmentation techniques are both handwritten and printed characters. So, using Discrete Cosine Transform, features were extracted of characters based on spectral sub-bands and with respect to the image's visual quality. The perceptual hashing algorithm used is combination of both DCT and 2D wavelet transform. Following are the steps to calculate features for each segmented character

1. Resize the segmented character to 32x32 pixels

2. Compute the Discrete Cosine Transform (DCT) of the image. The DCT separates the image into a collection of frequencies and scalars.
3. Compute 2D wavelet transform of the image. Compute standard deviation for each column and store it in a vector *V*.
4. Just keep the top-left 8x8 of the DCT. The top-left 8x8 represents the lowest frequencies in the image.
5. Compute the median value.
6. Compute the hash from the DCT. Set the 64 hash bits to 0 or 1 depending on whether each of the 64 DCT values is above or below the median value.
7. Append the hash value to the standard deviation vector *V*. Now, this vector *V* is feature set for each segmented character.

3.4 Clustering

Clustering is the task of grouping a set of objects in such a way that objects in the same group or cluster are more similar to each other that to those in other groups. Clustering algorithms is widely used to groups the similar objects in many fields including Data Mining, Natural Language Processing and Pattern Recognition. Different clustering algorithms are used to clusters image-set and among them k-mean is quite popular.[17] [18] . Different variants of K-means clustering algorithm is used to make 52 clusters of segmented images.

The features obtained after applying features extraction algorithms is saved as weka data file format(.arff) [16]. Weka tool is used for clustering of the images and creating resulting dataset. During clustering phase different algorithms were applied for clustering the characters. The results after applying clustering algorithm are also saved to a plain text weka format. Then a bash scripts finds the cluster label on each line and finds the associated image. For each cluster, a folder is maintained by the script and reading each line of the result file associated image is moved to the cluster folder. The distribution to each cluster folder varies on the input documents and frequencies of each character segment. As going through each cluster folder and checking the classification accuracy is not an efficient method for verification. So, to verify the accuracy of clustering algorithm, character numbers were counted is each cluster. It was found that folder named cluster4 had maximum numbers of characters with 3947 and 2721 after applying EM [20] and K-Means clustering respectively. So, cluster4 was selected for further evaluation . After analyzing the cluster4 images, it contained most of the character □. Further details comparison of different algorithms on cluster4 is show below.

Table 1. Cluster5 (□) metric

Algorithms	Accuracy
EM	0.9589
K-Means	0.9639

4. EXPERIMENT DESCRIPTION

For this task of creating Nepali Characters Dataset, 50 pages of Nepali text book was scanned. In addition, 30 pages of handwritten Nepali text were also scanned. These scanned text were fed to the reader module, which read the pages sequentially. In addition, characters were extracted using the segmentation techniques described in section 2. Then, the segmented image and its associate extracted features are saved in a plain text file. Also, the features obtained after applying features extraction algorithms were processed by weka and clustering was done.

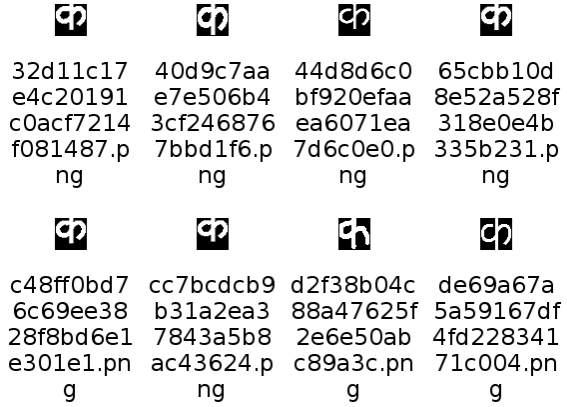


Figure 4 : Cluster4 contents samples

Moreover, the text file containing character image information is compared with result of clustering and grouped according the results of clustering. The final grouped text file is the required result as Nepali Character database. Each line in the text file has 1026 entries separated by tab. The first 1024 fields define the pixel position and the last two are addition *hasing* information. The individual characters can be created by reading each line, converting each line into 32x32 matrix and using image display functions. All the algorithms described in this paper is implemented in *octave* [19] using Intel 4th gen i7 processor and 8 GB RAM.

5. CONCLUSIONS

The method described in this paper is useful in creating Nepali Character dataset. It is more suitable when the dataset creation need to be less time consuming and effective. For creating complete Optical Character Recognition (OCR) system, this method is not very reliable as upper and lower modifiers are ignored in this method. But this method will help to build a generic Nepali Character dataset and this dataset can be used in OCR for training stage. Moreover, using this dataset exiting Nepali OCR can improve its accuracy. The major purpose of creation of cluster based Nepali character dataset is to use this dataset as a training resource for Nepali character recognition engine. Many forms of character recognition engine and models can be prepared using this datasets. This datasets is for those enthusiastic researchers who want to test and build robust Nepali character recognition engine. The development of character recognition system and models out of this dataset is left as future improvements.

6. ACKNOWLEDGMENTS

I would like to thank Mr. Bal Krishna Bal and Mr. Anjan Nepal for their valuable suggestion. Also, I would like to thank all the members of Inspiring Lab for encouraging and motivating me throughout this research.

7. REFERENCES

[1] Bal Krishna Bal and Prajwal Rupakheti, Research Report on the Nepali OCR, PANL10n Admin Reports, September 2009

[2] Eugene Borovikov, A survey of modern optical character recognition techniques (DRAFT), February 2004

[3] Vijay Kumar and Pankaj K Sengar, Segmentation of Printed Text in Devanagari Script and Gurmukhi Script, International Journal of Computer Applications, vol. 3, No. 8, pp 30–33, June 2010.

[4] Mitrakshi B. Patil ,and Vaibhav Narawade, Recognition of Handwritten Devnagari Characters through Segmentation and Artificial Neural Networks, Internation Journal of Engineering Research & Technology(IJERT), vol. 1, No. 6, August 2012.

[5] Veena Bansal, and R. M. K. Sinha, Segmentation of Touching and Fused Devanagari Characters, Indian Institute of Technology, Kanpur

[6] Ratnashil N Khobragade1, Dr. Nitin A. Koli and Mahendra S Makesar, A Survey of Recognition of Devnagari Script, International Journal of Computer Applications and Information Technology (IJCAIT), vol. 2, No. 1, January 2013.

[7] Richard G.Casey and Eric Lecolinet, A survey of Methods and Strategies in Character Segmentation, IEEE Transaction on PAMI, pp 690-706, 1996.

[8] Mudit Agrawal, Huanfeng Ma, and David Doermann, Generalization of Hindi OCR Using Adaptive Segmentation and Font Files, 2009.

[9] Sanjeev Maharjan, MPP Nepali OCR Report, PANL10n Admin Reports, July 2010 .

[10] Anilkumar N Holambe, Ravindra C Thool, Combining Multiple Feature Extraction Technique and Classifiers for Increasing Accuracy for Devanagari OCR, IJSCE, Vol. 3, No. 4, September 2013.

[11] Sheetal Dabra, Sunil Agrawal, and Rama Krishna Challa, A Novel Feature Set for Recognition of Similar Shaped Handwritten Hindi Characters Using Machine Learning, CCSEA 2011, Vol. 02, pp. 25-35, 2011.

[12] Andrew B. Watson, Image Compression Using the Discrete Cosine Transform, Mathematical Journal, Vol. 4, No. 1, pp. 81-88, 1994.

[13] Bian Yang,Fan Gu, and XiaMu Niu Image, Perceptual Hashing, IIH-MSP, pp. 167-172, December 2006.

[14] Christoph Zauner, Implementation and Benchmarking of Perceptual Image Hash Functions, Ph. D. Thesis, University of Sichere Informationsssteme, Hagenberg, July 2010.

[15] Lewis A.S and Knowles G, Image Compression using the 2-D wavelet transform, Image Processing, IEEE Transactions, Vol. 1, No. 2, pp. 244-250.

[16] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann ,and Ian H. Witten, The WEKA data mining software: an update, SIGKDD Explorations, Vol. 11, No. 1, pp. 10-18.

[17] Jacob Goldberger, Shiri Gordon, and Hayit Greenspan, Unsupervised Image-Set Clustering Using an Information Theoretic Framework, IEEE Transactions on Image Processing, Vol. 15, No. 2, pp. 449-458, February 2006.

[18] Venkat Rasagna, Anand Kumar, C.V. Jawahar, and R.Manmatha, Robust Recognition of Documents by Fusing Results of Word Clusters,

[19] John W. Eaton and David Bateman and Soren Hauberg, GNU Octave version 3.0.1 manual: a high-level interactive language for numerical computations, CreateSpace Independent Publishing Platform, 2009 .

[20] A. P. Dempster; N. M. Laird; D. B. Rubin, Maximum Likelihood from Incomplete data via the EM Algorithm, Journal of the Royal Statistical Society. Series B (Methodological), Vol. 39, No. 1, pp.1-38, 1997.