Assessing and Analyzing Tesseract Based Nepali Script OCR

Sudan Prajapati[1], Aman Maharjan[2], Shashidhar Ram Joshi[3], Bikash Balami[4]

[1]Department of Computer Science, Deerwalk Institute of Technology, Kathmandu, Nepal
sudan.prajapati@deerwalk.edu.np

[2]Central Department of Computer Science and Information Technology, Tribhuvan University, Kirtipur, Nepal
aman.maharjan@gmail.com

[3]IOE, Pulchowk Campus
srsrjoshi@ioe.edu.np

[4]Central Department of Computer Science and Information Technology, Tribhuvan University, Kirtipur, Nepal
bikuji@gmail.com

Abstract: Character recognition is commonly referred to as Optical Character Recognition as it deals with the recognition of optically processed characters. With the advent of digital optical scanners, a lot of paper- based books, textbooks, magazines, articles, and documents are being transformed into an electronic version that can be manipulated by a computer. OCR is an instance of off-line character recognition, where the system recognizes the fixed static shape of the character. This paper focuses on character recognition of printed text in Nepali script. This work analyzes the efficiency of Nepalese OCR based on Tesseract engine. The benchmark of this investigation and analysis is to create the dataset of the 69 different fonts with the 2,484 samples of consonants data of Nepali script. The overall accuracy of 96% was obtained in the training phase and 69% in the testing phase.

Keywords: Optical Character Recognition, Nepali Script, Tesseract, Nepali Font, Character Recognition

1. Introduction

The recognition of the character by machines has been a research topic for decades. Before the age of digitized computers, not much research was done in the field of character

recognition. In early research works, printed character recognition generally used template matching; low-level image processing techniques were used on the binary image to extract feature vectors, which were then fed to statistical classifiers.

A mechanical or electronic translation of images of handwritten, typewritten or printed text into machine-editable text form is defined as Optical Character Recognition (OCR). Automatic text recognition using an OCR is the process of converting images containing text into the equivalent string of characters. OCR is one of the most challenging topics in the field of pattern recognition [1]. OCR technologies are used for various purposes like storing documents, searching text, information retrieval from paper-based documents, documenting library materials etc. OCR can be categorized into 3 types: offline handwritten text recognition, online handwritten text recognition and machine printed text recognition.

Table 1: Nepali Consonant Characters

क	ख	ग	घ	ङ	च	छ	ज	झ	ञ
ट	ठ	ड	ढ	ण	त	थ	द	ध	न
प	फ	ब	भ	म	य	र	ल	व	
श	ष	स	ह	क्ष	त्र	ज्ञ			

Table 2: Nepali Vowel Characters

अ	आ	इ	ई	उ	ऊ	ए	ऐ	ओ	औ	अं	अः

Table 3: Nepali Numeric Characters

०	१	२	३	४	५	६	७	८	९

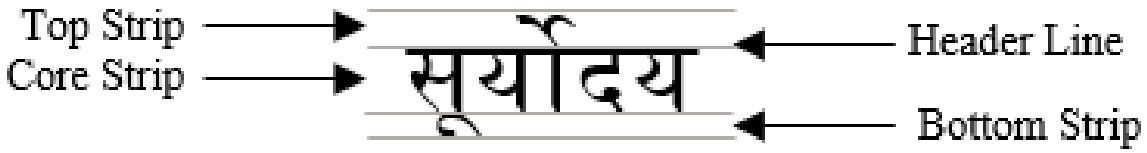

Fig.1: Nepali Word

Tesseract OCR engine is an open source OCR engine developed by the HP labs [2]. It was originally developed for English but later, extended to recognize other languages like Arabic, French, Italian, Catalan, Czech, Danish, Polish, Bulgarian, Russian Japanese, Chinese, Devanagari etc. Training the Tesseract OCR engine for any new script requires in-depth knowledge of the script and its character set. Tesseract needs a comprehensive and comparative study to deal with character recognition problem.

2. Problem Definition

Nepali is written from left to right direction. It is a phonetic and syllabic script, so, it does not have any lower or uppercase characters. The distinctive feature of Nepali script is the presence of a horizontal line on the top of all character known as the header line, shirorekha or diko [3]. The words can typically be divided into three strips: a core strip (middle zone), a top strip (upper zone), and a bottom strip (lower zone). When two or more characters appear side by side to form a word, the header lines touch and generate a bigger header line (figure 1).

Nepali character recognition using OCR faces many problems. One of the major problems is in segmenting characters. This problem arises due to the modifiers, combined characters, the variability of character size and so on [3].

Various approaches of character recognition can be applied to Nepali OCR such as gradient features, template-based formulation, and classification techniques like ANN, HMM, SVM are used [2-4].

3. Literature Review

The early attempt for Character Recognition was somewhat limited due to unavailability of powerful computing devices and digital image capturing devices [2]. Powerful hardware became commonplace during the 1980's which subsequently accelerated research in both online and offline OCR [1].

Image processing techniques and pattern recognition powered by artificial intelligence and various intelligent learning algorithms like ANN, HMM, fuzzy set reasoning etc. are commonly used in OCR [5]. The ultimate milestone of an OCR is to convert the printed or handwritten scanned document into machine-encoded text with negligible error. The Electronic device equipped with an OCR system can improve the speed of input operation and decrease possible human errors. OCR systems can decrease the use of keyboard and act as the interface between man and machine to a great extent. It also helps in office

automation leading to huge saving of time and human effort.

The Nepali language is written in the Devanagari script and hence the unique characteristics of the Devanagari script as reported in [5-7] apply to the Nepali printed text as well. Research in Devanagari OCR has been done extensively many people in the past. Similar achievements have been reported for the Bangla script which is similar to the Devanagari Script in many respects [6,7].

This study is focused on making use of the already available techniques in OCR with slight modifications necessary for the Nepali language.

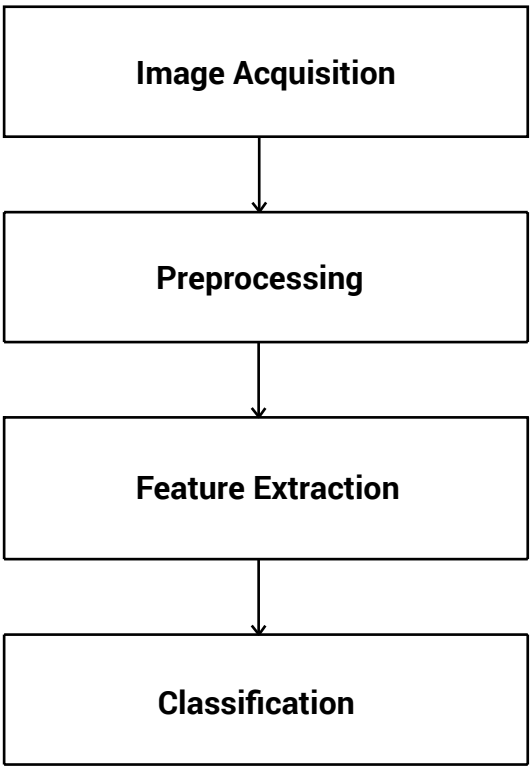

Fig.2: Steps of System Implementation

4. System Functioning

Hierarchical level of character recognition for the proposed system is divided into four subsections – image acquisition, preprocessing, feature extraction and classification (figure 2). The hierarchical model of is given below:

4.1. Image Acquisition

Images were acquired by typing characters in a word processor, printing and scanning in 300 DPI resolution. Character samples of different fonts were collected in PDF format.

4.2. Preprocessing

Raw image may contain some noisy pixels. They can be removed using the median filter. The resulting image is then ready for segmentation. Segmentation separates the digitized image into individual constituent character images.

4.3. Feature Extraction

After preprocessing of the image, feature vectors can be extracted for training and testing. Feature extraction plays an important role in the recognition of character to distinguish the character from image.

In Tesseract, the features for prototypes are 4 dimensional (x, y, angle, length) with 10-20 features in a prototype configuration. The features for unknown are 3 dimensional (x, y, angle) and for each character, there are approximately 50-100 features. The normalized features for the unknown are computed by tracing around the outline of the blob unit twice. The x, y position and the angle between the tangent line and a vector eastward from the center of the blob are saved as features.

4.4. Classification

Classification and recognition of a particular dataset depend on the selection of the features and classifiers that can recognize a particular character pattern. A classifier is called the 'heart' of pattern recognition system. It takes feature vectors and goes through all the decision-making process for recognition of patterns.

5. Training Tesseract

To train Tesseract for Nepali script, training data consisting of the following four files need to be created in the tessdata subdirectory of Tesseract installation directory:

- nep.inttemp
- nep.normproto
- nep.pffmtable
- nep.unicharset

The first three letters in each file, nep, represents the language code of Nepali in ISO 639-2 standard [8,9].

5.1. Training Data

Raw training data consists of 56 files, one for each Nepali font. Each file contains all the consonant characters in Nepali and was scanned from printed documents to 300 DPI TIFF

format.

5.2. Box File

Box file contains lists of characters, one per line, with the coordinates of the bounding box around the image. Initial box file generated by tesseract does not contain correct Unicode characters for Nepali, nor do they contain correct coordinates for bounding box (table 4). They are to be edited manually to fix this (table 4).

Table 4: Generated Box File

S	100	3147	159	3200	0
v	216	3146	273	3200	0
...	...	...	...	...	...

Table 5: Edited Box File

क	100	3147	159	3200	0
ख	216	3146	273	3200	0
...	...	...	...	...	...

5.3. Training File

Character images and corresponding box files are then used to create an exp.tr file, which contains the features of each character.

5.4. Character Set File Generation

Tesseract needs to know the following character properties: isalpha, isdigit, isupper, islower, ispunctuation before training the characters. This data is encoded in the unicharset file.

5.5. Font Properties

Font properties are used to provide font style information that will appear in the output when a font is recognized. This is provided in a text file names font_properties in the following format:

```
<font name> <italic> <bold> <fixed> <serif> <fraktur>
```

For a font named Mangal, the content of the file looks like:

```
Mangal 1 0 0 0 0
```

5.6. Clustering

After the extraction of character features of all the training pages, they are clustered using mftraining and cntraining commands. The mftraining command uses font_properties and unicharset files to generate inttempfile containing shape prototypes. The cntraining command generates the normproto data file that handles character normalization training for Tesseract.

5.7. Final Step

The final step combines the generated files in the previous steps. The required files are renamed by attaching a three-letter prefix, nep, for the Nepali language, to each file name.

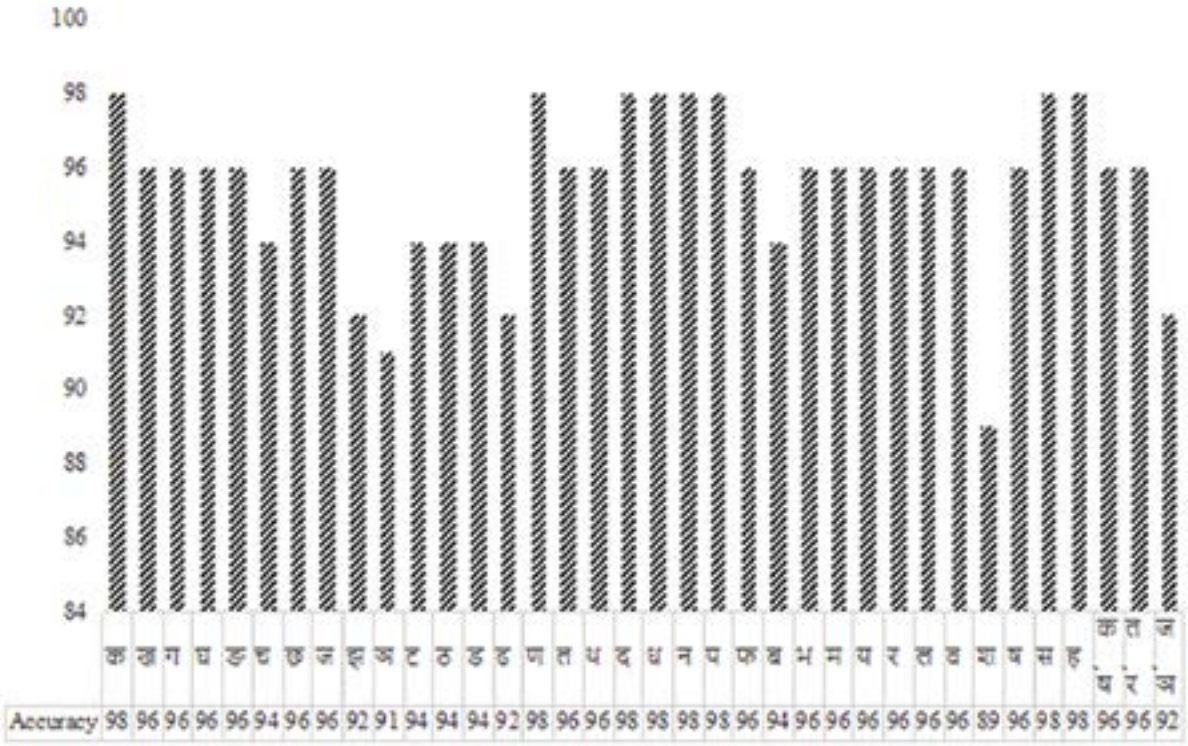

Fig.3: Training Accuracy for Individual Characters

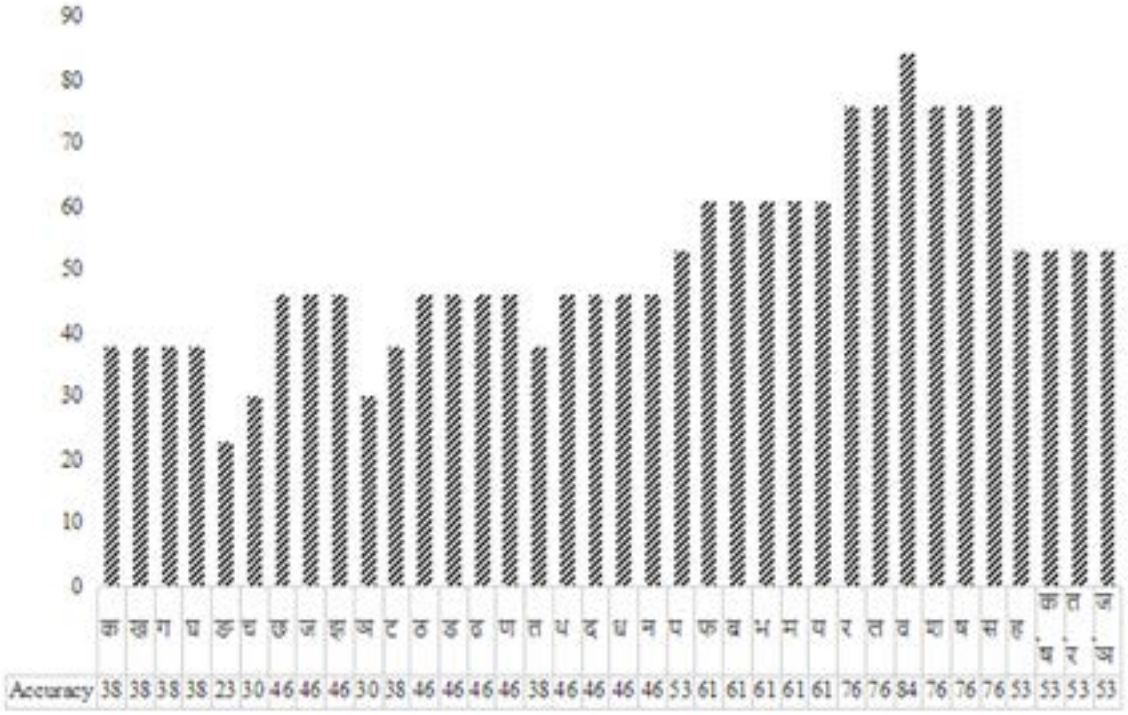

Fig.4: Testing Accuracy for Individual Characters

6. Evaluation and Result

The main aim of this study is to experimentally verify and evaluate the performance of Tesseract in optical character recognition of Nepali characters. The experiment is done in consonants of Nepali script, assuming the same procedure will be followed to the vowel and numerical digits. Dataset was created from fonts with 69 typefaces, containing 2520 individual character, for training and testing purposes. Training and testing were each carried out 10 times using random samples from the dataset.

Table 6 shows the training and testing performance for Tesseract. Each phase is carried out 10 times, and average values are shown in the table. 80% of the data was used to training and the remaining 20% was used for testing Tesseract engine. The result shows average training accuracy of 96% and average training time of 2.1 minutes. Similarly, it shows 69% average testing accuracy and 30 second average test completion time.

Similarly, individual character recognition accuracy for training data is displayed in figure 3 and for test data is displayed in figure 4. The figures show that overfitting is a major issue in Tesseract.

Table 6: Accuracy and Time

Phase	Average Time	Recognition Accuracy
Training	2.1 min	96%
Testing	30 sec	69%

7. Conclusion

Tesseract follows blob detection technique – it considers touching foreground pixels to be part of the same blob. For the individual character, a character component analysis is used to extract the character outline which is very useful because it does the OCR of an image with white text and black background. During the training phase, the segments of a polygonal approximation are used for features. In the recognition phase, features of a small, fixed length (in normalized units) are extracted from the outline and matched many-to-one against the clustered prototype features of the training data.

The Overall accuracy of 96% was obtained in the training phase and 69% in the testing phase. Similarly, average training time was 2.1 min and average test completion time was 30 sec. Analysis of individual character recognition data shows that Tesseract suffers from overfitting problem.

Character recognition, especially in a language like Nepali, has been a challenging research area for decades. Many research and various techniques have been carried out for the Devanagari script it uses, but due to the peculiarity of the script, the recognition accuracy has not been 100%. This can be addressed in future studies. Similarly, research on the word and sentence level Natural Language Processing (NLP) can also be carried out. A corpus of words can be created and used for translation. Handwritten Devanagari script recognition and multilingual character recognition is another comprehensive field of study for future researchers.

Reference

1. Y. Lu, Machine Printed Character Segmentation – an Overview, vol. 28, Pattern Recognition, 1995, pp. 67-80.
2. R. Smith, "An Overview of the Tesseract OCR Engine," Proc. of ICDAR 2007, 2007.
3. B. K. Bal, R. Pandey, S. Tuladhar and S. Shakya, "Interim Report on Nepali OCR," 2006.
4. M. Gunasekaram and S. Ganeshmoorthy, "OCR Recognition System Using Feed Forward and Back Propagation Neural network," Department of MCA, Park College

of Engg & Tech, Coimbatore.

5. D. Yadav, S. S. Cuadrado and J. Morato, “Optical Character Recognition for Hindi Language Using a Neural-Network Approcah,” J Inf Process Syst, vol. 9, no. 1, 2013.
6. B. K. Bal, “Scripts, Segmentation and OCR II Nepali OCR and Bangala Collaboration,” Jan 2009.
7. B. Chaulagain, B. B. Rai and S. K. Raya, “Final Report on Nepali Optical Character Recognition,” 07 2009.
8. R. Smith, D. Antonova and D.S. Lee, “Adapting the Tesseract Open Source OCR Engine for Multilingual OCR,” in Proceedings of the International Workshop on Multilingual OCR 2009, Barcelona, Spain, 2009.
9. ”Codes for the Representation of Names of Languages,” [Online]. Available: https://www.loc.gov/standards/iso639-2/php/code_list.php. [Accessed 2016-08-28].