A Survey on Devnagari Text Retrieval from Image

Mr. Rhunal Sawant[1] Mr. Pralhad Yesare[2] Mr. Pandurang Magdum[3]

[1,2,3]Department of Computer Engineering

[1,2,3]Rajendra Mane College of Engineering and Technology, Ambav, Mumbai University

Abstract— Devnagari OCR Software is used to read printed books, letters, agreements, etc in Hindi & Marathi Languages. It enables digitization of large amounts of text images in short time. All feature-extraction techniques as well as training, useful for the recognition are discussed in various sections of the paper. The main aim of our project is to develop an interactive system that will detect the devnagari characters and convert the devnagari image to text which will be in form of editable devnagari text.

Key words: OCR (Optical Character Recognition), Trained data

I. Introduction

Devnagari OCR is often used to convert paper books and documents into editable files. When one scans a paper page into a computer, it produces just an image file. The computer cannot understand the letters on the image, so you cannot search for words or edit it and have the words re-wrap as you type, or change the font, as in word processor. You would use OCR system to convert it into a text or word processor file so that you could do those things. The result is much more flexible and compact than the original image.

User gives input in the form of image, this image must contain devnagari text and this image process by proposed system and extract devnagari text from image with more accuracy and convert it into editable devnagari text.

II. Proposed System

The OCR system is simply converting Printed, Scanned document usually captured by digital camera or scanner into editable text. For the processing of that the characters in scanned text are matched with training dataset and after completion of matching OCR system gives editable devnagari text as result to user. Also it is useful for the student for making notes for particular topic. The current process consumes too much man hours and increase the overheads of the company for the data migration process. In the proposed system, the application takes in the input in the form of an image and extracts devnagari text from it. This automates the process of text extraction from the images.

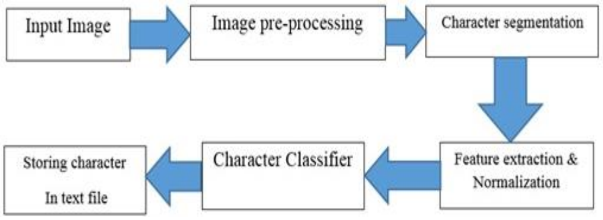

Fig. 1: System block diagram

III. Methodology

In order to achieve the best performance and accurate detection, we have designed and implemented tesseract algorithm.

A. *Tesseract Algorithm:*

Here we are using Tesseract OCR engine and JAVA language for developing project. We are begun with capturing Image and this image is then forward for processing. Text lines are broken into words according to the kind of character spacing. An attempt is made to recognize each word in turn. Each word that is satisfactory is passed to an adaptive classifier as training data. Adaptive classifier consists of training data. Training data contain different character and fonts. A final phase resolves fuzzy spaces. After recognizing words accurately finally output is display which is editable text in which user can make any change.

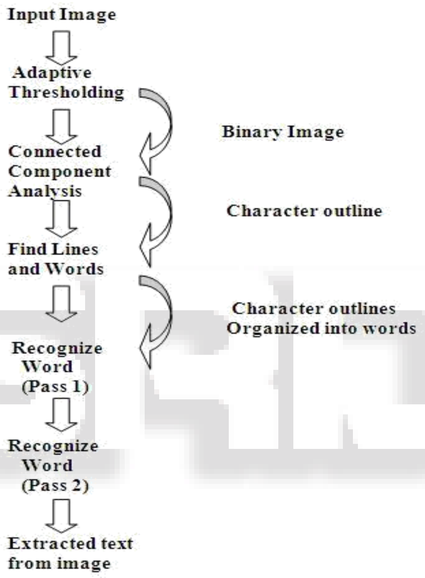

Fig. 2: Tesseract working

IV. Implementation

Here we are using net bean 7.0.1 tool for writing java source code for designing interface and creating various processes included in application. We begin with the conversion of printed, scanned text into editable form, this is performed by matching the characters of scanned document with training dataset, this task performed by the tesseract API which includes inbuilt functions for matching of scanned characters with training dataset. After processing of OCR engine the output is given to user in the form of editable text.

If the characters in scanned documents are not matched with training dataset then our OCR system gives garbage values as a result to the input to the user.

A. *Generation of Training Dataset:*

Here we are using jTessBoxEditor tool to generate training dataset. jTessBoxEditor is a box editor and trainer for Tesseract OCR. There is need to provide the TIFF/Box files as input to the editor. A box file is a text file that lists the characters in the training image, in order, one per line, with the co-ordinates of the bounding box around the image.

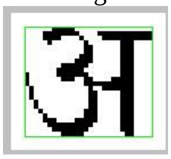

Fig. 3: Box view

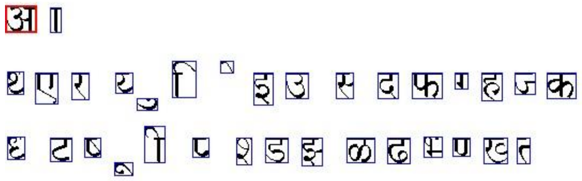

Fig. 4: Training of some Devnagari characters

Tesseract has a mode in which it will output a text file of the required format, but if the character set is different to its current training, it will naturally have the text incorrect. For each of your training image, box file pairs, run tesseract in training mode. The next step is to compute the character set. Tesseract needs to know the set of possible characters it can output. To generate the unicharset data file, use the unicharset_extractor program on the generated box files. A new requirement for training is a font_properties file. It provides font style information that will appear in the output when the output is recognized. JTessBoxEditor tool performs clustering, mftraining and cntraining on provided training image.

After performing all those processes, there is need to collect together all the files and rename them with a lang.prefix and then run combine_tessdata on them. Those all commands will be executed by jTessBoxEditor tool. This will result lang.traineddata file i.e. training dataset. After generating the training dataset, it is implemented in tesseract OCR.

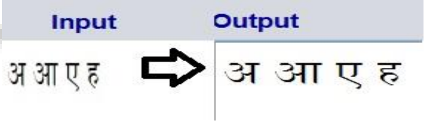

Fig. 5: Result of OCR

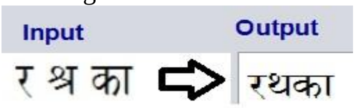

Fig. 6: Result of OCR

V. Conclusion

The proposed system will demonstrate the plan and implementation of building an application which will retrieve text from image with high accuracy and have many applications in industry. Training dataset is the most important part of this system. More training data may improve robustness and accuracy. If multiple fonts are trained then the system gives very less accuracy. The preliminary results on testing of the system show performance of more than 70% on printed texts on individual fonts. Further testing is currently underway for multi-font and hand printed texts. Most of the errors are due to inaccurate segmentation of symbols within a word. We are using only up to word level knowledge in our system.

Acknowledgement

No project is ever complete without the guidance of those experts who have already traded this before and hence became master of it and as a result, our leader. So we would like to take this opportunity to thank all those individuals who have helped us in visualizing this project.

We express our deep gratitude to our project guide and coordinator Mr. Magdum P.G. for providing timely assistant to our query and for his guidance in choosing this project and also for providing us all this details on proper presentation of this project.

We extend our sincere appreciation to all our Professor from RAJENDRA MANE COLLEGE OF ENGINEERING & TECHNOLOGY for their valuable time during the designing of the project. Their contributions have been valuable in so many ways that we find it difficult to acknowledge them individual.

We are also grateful to our HOD Mr. Naik L.S. for extending his help directly and indirectly through various channels in our project work.

References

[1] Pinal K Shah, Preeti K Dave, "Java Based Devnagari Script Recognition using JNI(Java Native Interface)", International Journal of Computer& Organization Trends – Volume2Issue1-2011.

[2] Sheetal A. Nirve, Dr. G. S. Sable, “Optical character recognition for printed text in Devnagari using ANFIS”, International Journal of Scientific & Engineering Research, Volume 4, Issue 10, October-2013ISSN 2229-5518

[3] Shrinath Janvalkar, Paresh Manjrekar, Sarvesh Pawar, Prof .Laxman Naik, “Text Recognition from an image”, Int. Journal of Engineering research and applications ISSN:2248-9622,vol.4, Issue 4(Version 1), April 2014,pp.

[4] Paul and B.B. Chaudhuri, “Indian script character recognition: A survey,” Pattern recognition, vol. 37, no.9, pp. 1887-1899, 2004.

[5] Veena Bansal & R.M.K. Sinha, “Segmentation of Touching Characters in Devnagari”, http://www.iitk.ac.in/ime/veena/PAPERS/stwo.pdf

[6] https://github.com/tesseract-ocr/tesseract/wiki/TrainingTesseract#training-procedure

[7] http://stackoverflow.com/questions/11628052/font-property-issue-while-training-tesseract-ocr-v-3-01

[8] https://sourceforge.net/p/tesseracthindi/wiki/OCR%20for%20Devanagari/

[9] https://blog.cedric.ws/how-to-train-tesseract-301