Optical Character Recognition for Hindi using a Convolutional Approach

Sonal Sannigrahi

***Abstract*— In this paper, an OCR method is built using a neural network based approach for Hindi with scope of extension to other Devanagari based scripts. With being one of the most widely spoken languages, Hindi text is becoming increasingly widespread and thus the need for an accurate OCR system for the same also arises. Due to features of the script, such as unclear character segmentation as compared to Roman scripts (English, French,...), OCR systems typically very accurate for the latter fails to provide a high accuracy rate for the former.**

For any general OCR system, the major steps include pre-processing, character segmentation, feature extraction, and lastly, classification and recognition. In this paper, the approaches taken in the pre-processing stage include conversion of gray scaled images to binary images by normalisation and rounding, image rectification, and segmentation of the document itself into lines, words, and characters (basic symbols). To recognise these characters, a Neural Network based classifier is used (trained upon a custom dataset curated for Hindi characters). Lastly, this paper uses convolutional layers for feature extraction. However three other feature extraction methods were also considered: histogram of projection based on mean distance, histogram of projection based on pixel value, and vertical zero crossing.

I. INTRODUCTION

Optical Character Recognition (OCR) is the process through which printed/scanned documents are converted into text (ASCII) that a computer can read, i.e converting an image of text into readable text. [1] In this section, we discuss the general flow control of a Hindi OCR system and further delve into the details of the Devanagari script itself from an OCR point of view.[2]

A. *An overview of OCR systems*

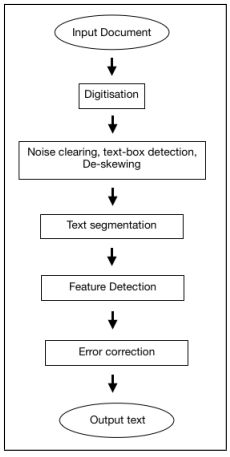


Fig. 1. Flow control for a Hindi OCR system

In general OCR systems, the segmentation and classification steps are considered to be the most important. During the segmentation process, key recognition units from the input image are recognised. In this paper, the key difference from general OCR systems is the classification process. After using careful segmentation, a deep neural network method is used to recognise and classify the characters.

The OCR system proposed here takes in an input image, after preprocessing produces isolated characters which are then fed into a neural network to return classified characters and finally are post processed to produces a text file.

B. *Specialities of the script*

One of the most important steps in building a successful OCR system is to have good character recognition. Here, we will use some structural properties of the Hindi language in order to produce useful segmentation.

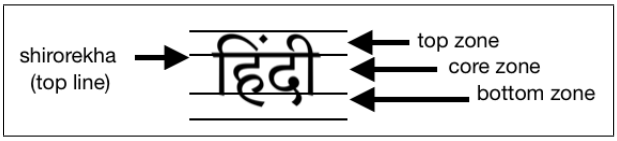


Fig. 2. Word Structure

From an OCR point of view, there are some structural properties, discussed later, of Hindi writing that would be useful in terms of good text segmentation.

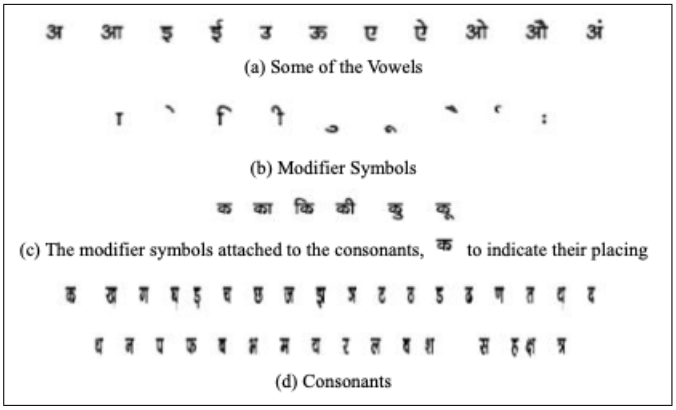


Fig. 3. Characters and Symbols used in Hindi

Looking at these structural components, it's clear to see that the complexity involved in Hindi text is quite high and thus careful segmentation must be performed to extract the character properly.

One of the major bottlenecks of this script is conjunctions. Not only can consonants be conjunct to form new characters, but modifier symbols can also be conjunct on the same character. This implies that there could be 2 modifier symbols on the same character and this character could be a conjunction of 2 other consonants. Furthermore, visually it is observable that the modifiers themselves are not uniform in their location either. It can be noticed that some modifiers are present only in the top zone while some present only in the bottom zone and any other combination of the three zones. This proves to be a difficult challenge to segment well and thus their structural properties were exploited in order to make a robust segmentation process.

In this OCR system, only single modifier and characters are taken into account but further approaches for segmentation are also discussed as possible extensions to this system.

II. SEGMENTATION

Keeping the model of the OCR system in mind, we first consider the input image. In order to make this input suitable for character classification, some pre-processing and text segmentation must be performed.

A. Pre-processing

Before applying any segmentation procedures, let us consider some preprocessing issues. Firstly, to detect lines with ease the image must be aligned correctly. In order to do this, a deskewing procedure is applied first.

Next, upon converting the input image into gray scale, an adaptive threshold method is used to convert the image into binary pixels allowing easy distinction between the background and the foreground. Upon testing various thresholding methods, applying a Gaussian Blur and proceeding with Otsu's threshold gave the best results. This methods allows us to consider inputs with varying lighting as well as those with noise. Below are some results:

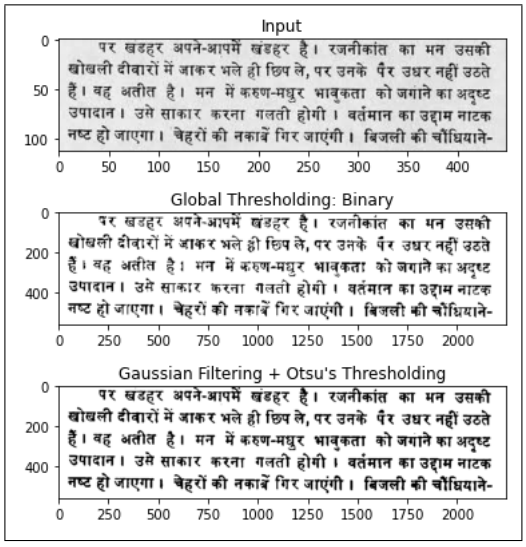


Fig. 4. Different Thresholding methods for Input

B. Text Segmentation

1) Line Detection: In order to detect lines in the input image, we perform a horizontal scanning method to the processed image, which corresponds to binarization through adaptive thresholding and text deskewing. This consists of scanning the entire image starting from the top till the last row of all white pixels after which a black pixel is detected is encountered and then it continues scanning until a similar white pixel row is first encountered signalling the start of a new line.

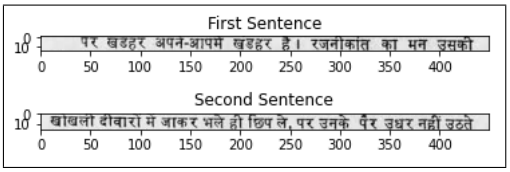


Fig. 5. Sentences Retrieved

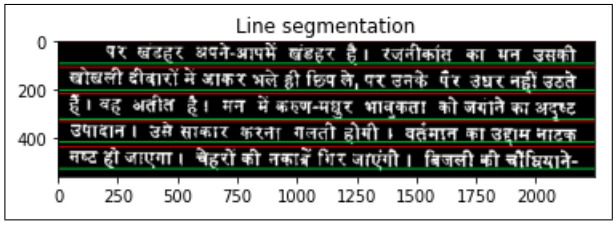


Fig. 6. Line Segmentation

2) Word Detection: From each line of the input image, the next step is to detect words. Due to the property of a strict header line in Hindi, we can exploit horizontal scanning in this case too but adapt it slightly differently. Here, a vertical scanning method is utilised. The implementation of this method can be understood as applying vertical scanning, as done for the lines, to the image rotated by 90 degrees clockwise and the final output is achieved by rotating the image 90 degrees counterclockwise after the segmentation process.

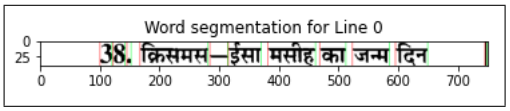


Fig. 7. Word Detection

3) Character Detection: Character segmentation is by far the most difficult part of preprocessing. Due to the structural complexities of Hindi, it is quite difficult to account for all possible conjunctions of characters. Furthermore, as vowels can be both independent and dependent on the previous characters, some further segmentation was used. Firstly, the header line was removed. Secondly, the word was divided into the three zones as described earlier. This aids in identifying the modifiers (the dependent vowels). Lastly, a vertical scanning method on the length of the word and width of the zone was used to obtain individual characters. For this project, the scope was limited to not include half

characters and conjugate characters as well.

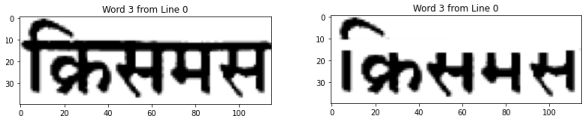


Fig. 8. **Left:** Original **Right:** Modified and ready for segmentation

As the header line wasn't completely removed for some specific examples, a secondary method was also employed to ensure that proper character segmentation is done. This process uses a threshold applied to column wise pixel sum. This is an efficient method as the header line occupies the least number of black pixels per column when compared to spaces with characters in them. Of course, this method doesn't remove the header line entirely but does create gaps between characters which once again allows us to use vertical scanning to separate.

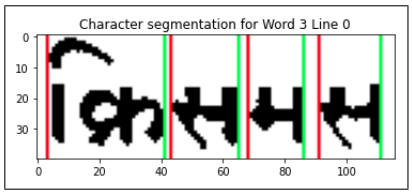


Fig. 9. Character Segmentation

III. FEATURE EXTRACTION AND CLASSIFICATION

Upon sufficient preprocessing and structural segmentation, a method to classify and recognise this these characters in terms of UTF-8 encoding is discussed. In order to do this, the chosen architecture was a convolutional neural network however other methods such as MLP networks, were also explored. Due to large number of trainable parameters from MLP networks, they were not chosen eventually.

A. Dataset Curated

The dataset used to train the CNN was made up of vowels, modifiers (conjunct with consonants), consonants, and digits. The dataset didn't include half characters as well as consonant conjunctions. As input to the CNN, each image was normalised to 32x16 and was made grayscale (hence 1 input channel). Below is a sample of the dataset:

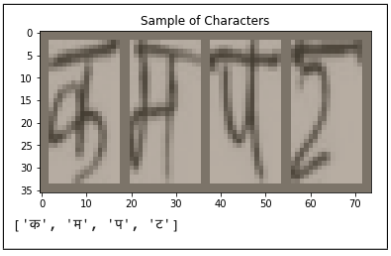


Fig. 10. Character Dataset with unicode labels

Due to the fact that the dataset was custom made for this project, there are some clear limitations. There were only 4 writers per label and hence there isn't a robust variety within each label. However UCI's Devanagari Character Dataset [9] provided good variety for the consonants and Nepali OCR Dataset provided additional characters such as the independent form of the vowels. The modifiers were collected by hand and were the only part that led to difficulties in training however were manageable. During the training process, it took approximately 3 hours to train for 164 epochs using the 12GB NVIDIA Tesla K80 GPU.

B. Architecture Used

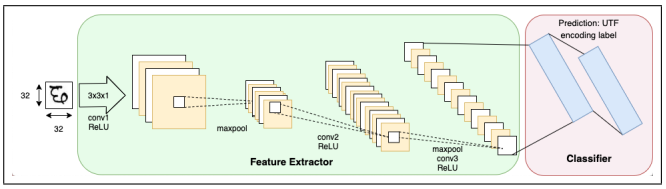


Fig. 11. Feature Extraction Model

A deep convolutional neural network was used in the character classification with a relatively standard model used widely in current convolutional architectures. A normalised 32x32 grayscale character is provided as input to the network following which the first 2D convolution is made with a kernel of size 3x3. The chosen non-linearity for the entire architecture was the rectified linear unit (ReLU) as it has proven to be most stable in this case. Next, a max pooling is performed to subsample from the current convolutional layer. The max pooling method checks for the maximum value on its local receptive field, multiplies it by a trainable coefficient, adds a trainable bias and generates output. This was chosen to help recognise sharp lines and maximise the presence of outlines in the image.

Following this subsampling layer, a second convolution is performed. Here, the two resultant feature maps (from convolution 1 and convolution 2) were not fully connected in order to reduce the total number of training parameters as a result of the limited dataset. A third convolution is used after which the feature extraction part of the model concludes.

In the classifier section, a few fully connected layers are used of sizes 128, 64, and the final number of classes to be classified. Here, to include more robustness a dropout method was included as well as batch normalisation to help increase model regularity. The output from these fully connected layers is a vector indicating a UTF label which is encoded as part of the post processing stage.

Here, some results obtained by the classifier are discussed. Shown below are the accuracy and error graphs obtained with training the network for the first 200 epochs. To allow

for larger number of epoch, a decaying learning rate was employed in the form of a SGD optimiser with a scheduler. Finally, an accuracy of 98.25% is reached.

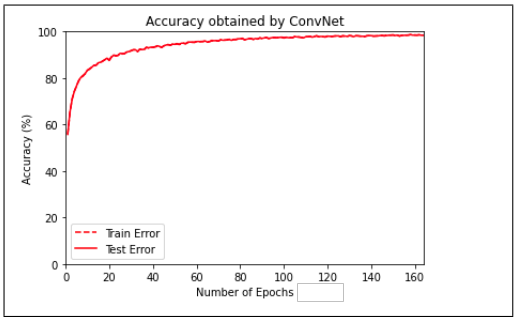


Fig. 12. Accuracy of the Model

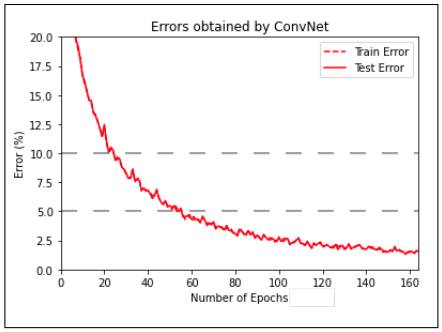


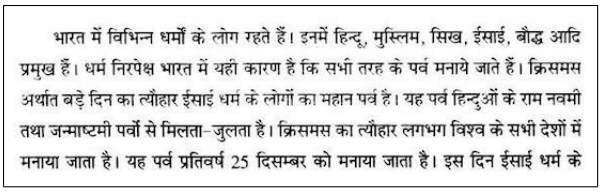
Fig. 13. Error of the Model

IV. POST-PROCESSING

In the final stage of the OCR process, the output text file is created. Here each individual character (from the segmentation process) is fed into the trained network, it's classified prediction is given to an utf-encoder, final these utf-encoded characters are placed back into words and sentences as was original segmented. This output is saved into a text file which is now machine readable.

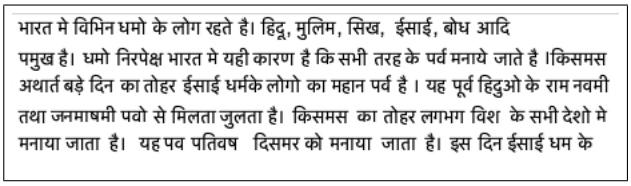
V. RESULTS

The complete OCR system was tested on a variety of Hindi documents with differing fonts, text sizes, and levels of noise. To accurately test the classier and the processing stages, the evaluation was done at two levels: character and paragraph level. From the validation set of the characters, an accuracy of 98.25 % was reached. However, due to special characters not yet recognised by the system, the paragraph level accuracy drops to 90% on average from the paragraphs tested. The variations in accuracy at the paragraph level arise primarily from the appearance of conjunctions in consonants and modifiers as well as the presence of bilingual text (for example English alphanumeric characters). Furthermore, only a few punctuation marks was considered by the dataset, the Hindi period which is represented by a solid black vertical line and commas. However, other punctuation marks were not trained and therefore are not recognisable by the classifier as well.



भारत में विभिन्न धर्मों के लोग रहते हैं। इनमें हिन्दू, मुस्लिम, सिख, ईसाई, बौद्ध आदि प्रमुख हैं। धर्म निरपेक्ष भारत में यही कारण है कि सभी तरह के पर्व मनाये जाते हैं। क्रिसमस अर्थात बड़े दिन का त्यौहार ईसाई धर्म के लोगों का महान पर्व है। यह पर्व हिन्दुओं के राम नवमी तथा जन्माष्टमी पर्वों से मिलता-जुलता है। क्रिसमस का त्यौहार लगभग विश्व के सभी देशों में मनाया जाता है। यह पर्व प्रतिवर्ष 25 दिसम्बर को मनाया जाता है। इस दिन ईसाई धर्म के

Fig. 14. Input Image



भारत मे विभिन धमो के लोग रहते है। हिदू, मुलिम, सिख, ईसाई, बोध आदि
पमुख है। धमो निरपेक्ष भारत मे यही कारण है कि सभी तरह के पर्व मनाये जाते है ।किसमस अथार्त बड़े दिन का तोहर ईसाई धर्मके लोगो का महान पर्व है । यह पूर्व हिदुओ के राम नवमी तथा जनमाषमी पवो से मिलता जुलता है। किसमस का तोहर लगभग विश का सभी देशो मे मनाया जाता है। यह पव पतिवष दिसमर को मनाया जाता है। इस दिन ईसाई धम के

Fig. 15. Result of OCR

From the above images, it is clear that the main limitation observed by this OCR system is the lack of a good and varied dataset. It can be seen in, for example, word 3 line 1 that the half character wasn't recognised. Furthermore, conjunct modifiers such as in word 2 were also not correctly recognised (missing the upper dot in this case). While these results are good for a first attempt at a complete OCR system, there are some improvements to be made.

A. Testing Robustness

The classifier was also tested on characters with varying levels of distortion filters applied in order to check the robustness of the classifier.

Following varying levels of distorting, the classifier successfully classified up to 100 % of the characters correctly, the only exceptions being examples in which even native speakers of the language couldn't tell the difference.

Here is a small sample of the distorted data:

Original Character	Distorted Character	Character Recognized
४	४	४
४	४	४
४	४	४
४	४	४
४	४	५

Fig. 16. Robustness Results

From this we can say that the character classifier developed here is quite robust and meets benchmark tests.

VI. CONCLUDING REMARKS

To summarise, this paper presented a novel method for an OCR system for Hindi text using a deep convolutional network reaching excellent results on character level recognition and great paragraph level results as well. Further extensions to this project would include developing a more robust segmentation algorithm to deal with conjunct character and

modifier, developing a wider dataset for such classification which could be made publicly available, amongst other things. Additionally, dictionary based approaches should also be looked into carefully to have better performance on word level recognition and thus improve overall accuracy. Certainly, the most difficult part of the project was finding suitable datasets online, and eventually they were not. So it remains that to have further advancements in character recognition for Indic texts, a large database of ground truth characters must be developed. Lastly, this paper considers printed text only, which allows for scanning based methods for segmentation. An extension would be then to consider new pattern recognition based approaches for handwritten text that may not be uniform. In conclusion. the method developed in this paper can have wide range applications like translation of images and also may be further extended to speech-to text encoding.

ACKNOWLEDGMENT

This paper acknowledges the support of the writers, Ashis Sannigrahi, Payel Sannigrahi, Prativa Sannigrahi, without whom this wouldn't have been possible. This is to thank them for their countless hours and effort.

REFERENCES

[1] Mori, S. et. al.: Historical Review of OCR Research and Development. Proceeding IEEE, Vol.80, No.7, 1992, pp.1029-1058.

[2] Bansal, V., Sinha, R.M.K.: Designing a Front End OCR System for Indian Script for Machine Trans- lation- A Case Study for Devanagari. Symposium on Machine Aids for Translation and Communication, New Delhi, India, 1996.

[3] Tellache, M., Sid-Ahmaed, M., Abaza, B.: Thinning algorithms for Arabic OCR. Communications, Computers and Signal Processing, 1993, IEEE Pacific Rim Conference on Vol.1, 1993, pp.248-251.

[4] Natarajan, P., MacRostie, E., Decerbo, M.: The BBN Byblos hindi OCR system. In Guide to OCR for Indic Scripts, V. Govindaraju and S. Setlur, Eds. New York: Springer-Verlag, 2009, pp.173-180.

[5] Kompalli, S., Setlur, S., Govindaraju, V.: Devanagari OCR using a recognition driven segmentation framework and stochastic language models. Int. J. Document Anal. Recognit., Vol.12, 2009, pp.123- 138.

[6] Jayadevan, R., Kolhe, Satish R., Patil, Pradeep M., Pal, U.: Offline Recognition of Devanagari Script: A Survey. IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews, Vol.41, No.6, 2011, pp.782-796.

[7] Pal, U.; Roy, P. P., Tripathy, N., Josep, L.: Multi-oriented Bangla and Devanagari text recognition. Pattern Recognition 43, 2010, pp.4124-4136.

[8] Shimizu, M., Fukuda, H., Nakamura, G.: A thinning algorithm for digital figures of characters. Image Analysis and Interpretation, 2000. Proceedings. 4th IEEE Southwest Symposium, 2000, pp.83-87.

[9] S. Acharya, A. K. Pant and P. K. Gyawali, "Deep learning based large scale handwritten Devanagari character recognition," 2015 9th International Conference on Software, Knowledge, Information Management and Applications (SKIMA), Kathmandu, 2015, pp. 1-6, doi: 10.1109/SKIMA.2015.7400041.

[10] Desai, Apurva A.: Gujarati hand written numeral optical character reorganization through neural network. Pattern Recognition 43, 2010, pp.2582-2589.59.

[11] Dhingra, K. D., Sanyal, S., Sanyal, P. K.: A robust OCR for degraded documents. In Lecture Notes in Electrical Engineering. Huang et al., Eds. New York: Springer-Verlag, 2008, pp.497-509.

[12] Pal, U., Chaudhuri, B.B.: Automatic Separation of Machine-printed and Hand-Written Text Lines. Document Analysis and Recognition. IEEE Proceedings of the Fifth International Conference on, 1999, pp.645-648.

[13] Govindaraju,V., Khedekar, S., Kompalli, S., Farooq, F., Setlur, S., Vemulapati, R.: Tools for enabling digital access to multilingual Indian documents. in Proc. 1st Int. Workshop Document Image Anal. Libraries, 2004, pp.122-133.

[14] Hinds, S.C., Fisher, J.L., D'Amato, D.P.: A document skew detection method using run-length encod- ing and the Hough transform. Pattern Recognition, IEEE Proceedings., 10th International Conference on Vol.1, 1990, pp.464-468.