ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 3, March 2018

OCR for Devanagari Script Using Tensorflow Technology

Darshan Kawade[1], Neha Kaunds[1], Vedang Date [1], Rajendra Pawar [2]

B.E Student, Dept. of Information Technology, MAEER's MIT College of Engineering, Savitribai Phule Pune University, Pune, Maharashtra, India[1]

Asst. Professor, Dept. of Information Technology, MAEER's MIT College of Engineering, Savitribai Phule Pune University, Pune, Maharashtra, India[2]

ABSTRACT:It is a very complex task to recognize characters from a given image and store it in an editable format so that it can be used or modified later as per convenience. Scanned text documents or pictures stored in mobile phones as well as pictures captured by mobile phones are the main focus of this application. OCR of Devanagari script presents a wide range of challenges that are not seen when it comes to Latin based scripts. The purpose of this application is to recognize the text inimage given as input in order to reuse it later. This application will allow its users to perform actions such as converting it into an editable word format which can be modified later. Some other features such as transliteration and translation have also been added. This application is very useful as it saves time without the need of retyping the whole document itself. It can also be used for cross lingual information retrieval with the help of transliteration and translation modules.

KEYWORDS: Devanagari script, OCR, TensorFlow technology, MobileNet model

I. INTRODUCTION

Today, computer has become a major part of every individual's life. Previously, data storage was done in books, manuscripts, files, etc. Digitalization has brought a new revolution in the human life. Computers have now become the basis of human life. The world is slowly tending to the urge of digitalization. It is much easier to store and retrieve data from computers instead of books and manuscripts. Information stored in computers can be backed up and be preserved for more period of time.Optical Character Recognition(OCR) is a process using which one can optically recognize a character. In simple terms, this process helps to convert an input image having printed Devanagari characters to a textual format. It is mainly used for information entry from printed data record. OCR helps to store the data in the image or scanned document to be edited, copied and searched as needed. It also stores the data in a much more compact way unlike other resources such as books.

This paper presents an idea for buildingan open source library for images containing digital fonts in Devanagari script with the help of a Google's powerful classifier for mobile devices having restricted resource named MobileNet by using TensorFlow library. We have also added extra features along with OCR like transliteration and translation of the Devanagri script to English. TensorFlow is an open source Google library used for the purpose of deep learning. The approach for this OCR system is based on various steps such as preprocessing, segmentation, training and classification. Due to use of Deep Neural Network (DNN), the performance of this system has been increased. Input to this system are in the form of images containing digital fonts in Devanagari script. The output of this system will be a text document which can be edited.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 3, March 2018

II. LITERATURE SURVEY

In [1], authors have illustrated template matching and structure analysis approach for R&D. Some comments have also been made on recent techniques applied toOCR and also listed some problems.In [2], authors highlight various approaches for segmentation of the Devanagari word. It suggests to locate the header line by calculating the maximum number of black pixels in the given image. The columns in the image having no black pixel will be then taken as the threshold for separating the lines and words from a given image.In [3], authors have introduced a schema for description of various shapes present in Devanagari characters and its application in their recognition for printed Devanagari text.Authors have suggested to segment ascenders, descenders and core components in order to simplify the process of classification in [4].In [5], authors explained an approach for dealing with the noise found in the digital images. They suggested to use median filter for preprocessing of images as this method helps in excellent noise reduction. They have also highlighted the segmentation of line followed by segmentation of words followed by segmentation of characters. For the segmentation of line, they have used horizontal scanning from top to the last row which contains all white pixels before a black pixel is found. They have segmented words using vertical scanning. Segmentation of characters is done by locating the header line and replacing it with white pixels and dividing the word into upper, middle and lower zone by vertical scanning.
Authors used Artificial Neural Network and Nearest Neighbor Approach to recognize characters from scanned images with the help of three layers in [6]. In [7], authors deal with an OCR error detection and correction technique for Indian script like Bangla. In [8], authors suggested various techniques for character segmentation and have also made a comparative study of these techniques.
In [9], authors have described the TensorFlow dataflow model and demonstrated the TensorFlow performance for various real-world applications. In [10], Authors describe the TensorFlow interface and also its implementation which is built at Google. In [11], Authors have proposed a new model architecture called MobileNets which is based on depth wise separable convolutions.
In [12], authors proposed a system for named entity transliteration for Devanagari script to English using Support Vector Machine (SVM). They have suggested the overall logical flow into preprocessing of raw data, training of bilingual corpus using degree two of polynomial linear function and testing of additional data.

III. STRUCTURE OF THE PROPOSED OCR SYSTEM

Diagrammatic representation of the proposed OCR system is given as in fig. 1. For training, deep learning has been used. Deep learning can be achieved by using Google's TensorFlow. The input for training this system is a folder containing images of each character of Devanagari script where the folder represents the class for classification. The model is then trained for over 4000 iterations. This trained data is then used for classification. During the testing phase, the input image is scanned followed by preprocessing and segmentation respectively. Each character thus produced is given as an input to the trained model. The class with highest confidence represents that character which is then printed in the text document.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 3, March 2018

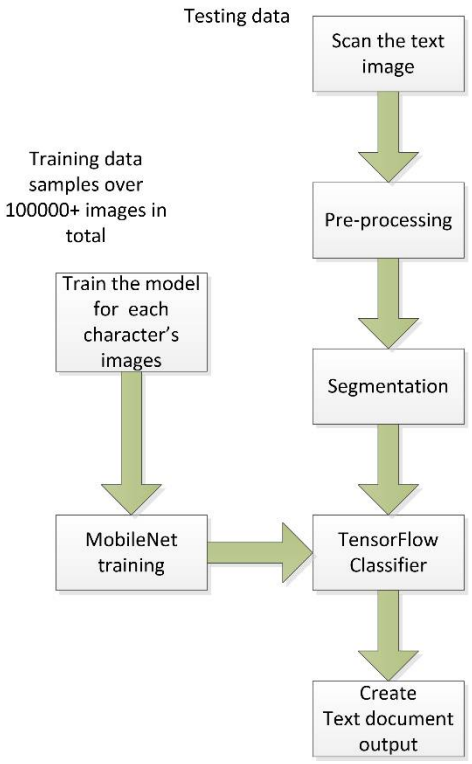

Fig.1. Methodology

A. PRE-PROCESSING

All phases of pre-processing are shown in fig. 2. With the help of bilateral filter applied on complex images multiple times, the input image is then converted to grey scale, refer fig. 2. A copy of the resultant grey scale image is inverted to handle various challenges with respect to font and background color. Apply thresholding techniques like median blur followed by Gaussian blur on both the images. Add a black color border of one pixel width to both images and use flood fill technique to flood the pixels with white pixels. Now, calculate black pixels in both the images and discard the one with less black pixels. The image is thus converted to a pure binary image that can be accessed as grey scale image with only two values 0 for black and 255 for white.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 3, March 2018

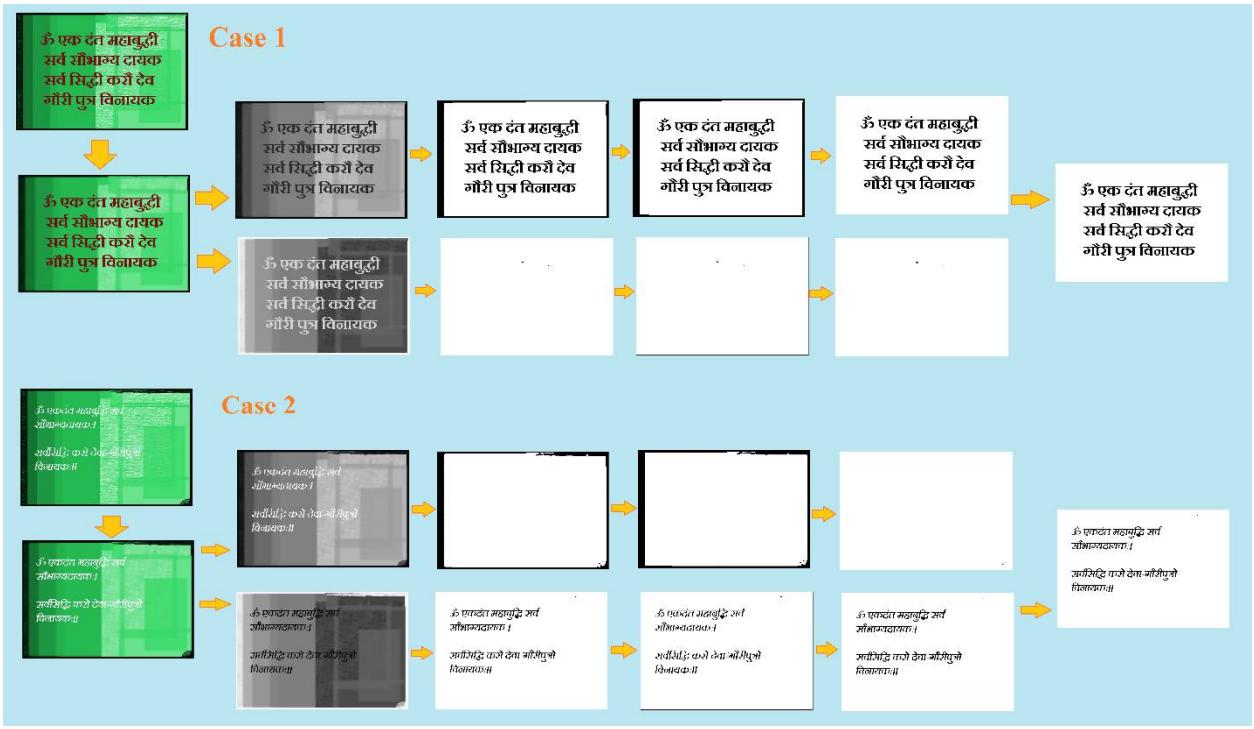

Fig. 2. Phases of preprocessing

B. SEGMENTATION

The performance of the OCR systems depends on the segmentation techniques used. Unlike Latin based scripts, while dealing with Devanagari script we face various challenges [4].Segmentation of Devanagari script is hard. Segmentation is used to divide an image into its constituent parts consisting of lines followed by words and then to each character present in the input image. It is very important to divide the image into characters as the classifier is modeled and trained to classify only characters. This phase is also considered to be a crucial one as it consists of tasks such as separating the jodaksharas (compound letters) and identifying the kaana, maatra and ukars (ascenders and descenders).
By default, the value for black color is 0 in pixels while that for white is 255. But we intend to invert the value of pixel such as 0 and 255 represent white and black respectively. These values are inverted to make calculations simpler in order to calculate pixel density in a row. Larger the value more black pixels are present in that row. Calculating density is same as calculating mean of all pixel values in a row. For segmentation, we need to identify the gap between two lines containing text and height of lines containing text. After calculating mean of each row, the value of mean will be minimum (i.e. mostly from 0 to 4) where there is gap. To calculate the mean of each row, get pixel value of each pixel by traversing the row first from left to right and then the column from top to bottom. At the end of each row calculate the value of mean.In this paper, two methods are introduced for calculating mean of every row. This mean value of each row is then stored in an array, where array index say “i”, itself represent the line number “i” containing text.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 3, March 2018

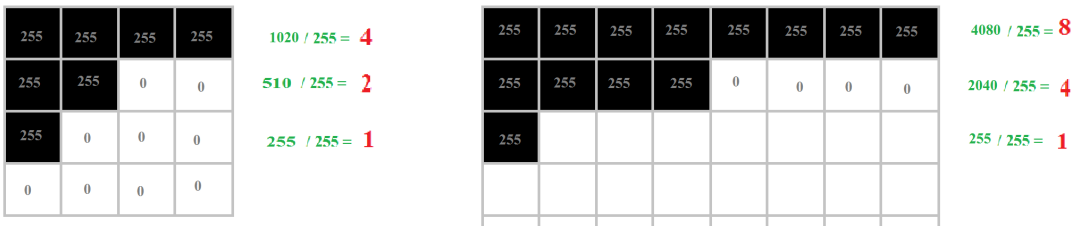

Fig. 3. Method 1 for calculating mean

Method 1 illustrated in fig. 3 returns the number of pixels in a row. This method is not useful for taking decisions in segmentation process. This method shows how two images of different resolutions having similar image returns different output. As output depends on resolution/image size, it becomes complex task to perform segmentation on these output values using this method.

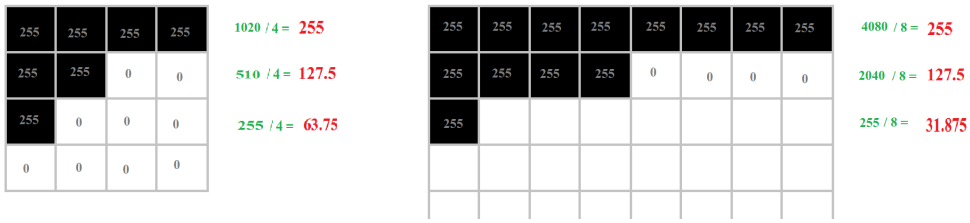

Fig. 4. Method 2 for calculating mean

Method 2 illustrated in fig. 4 returns the density of pixels i.e. 255 represents row with all black pixels (example, Shirorekha of word) and 127.5 represents row with half black pixels. The only possible real values in this case ranges from 0 to 255 (including both 0 and 255).This method helps to take sensitive decisions in segmentation process.This method is the best solution for our proposed model.

1) SEGMENTATION OF LINE: Mean of eachrow is an integer value that is always between 0 and 255 including both 0 and 255. Calculate frequency of each mean value. In order to separate lines from the input image, a threshold value has to be set. In this case as the input image is always converted to binary format the minimum mean is 0. However even if there is a binary image with some noise in each gap between the lines containing text, the value on minimum mean of all means will be less than 5 most of the time but may not be 0. Get the mean value between 0 and 5 (both including 0 and 5) that has maximum frequency. Set this mean value as threshold. This threshold value helps to identify the background of image and it also represents the gap between two lines.If the threshold mean value is not 0 but less than 5(i.e. for example if threshold mean is 4),then the row is treated as a row with no black pixels (i.e. gap between two lines); hence dealing with noise if present.The mean of each row is stored in an array. The index number of array represents the row number. The array is traversed to compare the current and previous mean value with threshold value to find the start and end of a line. This helps to crop the lines. Ifascenders and descenders of line are not joined then they are correctly identified and joined with their respective core line. To do this the distance between each line and height of each line is calculated and used to join separated ascenders and descenders to their respective core letters. Refer step 2 & 3 in fig. 5.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 3, March 2018

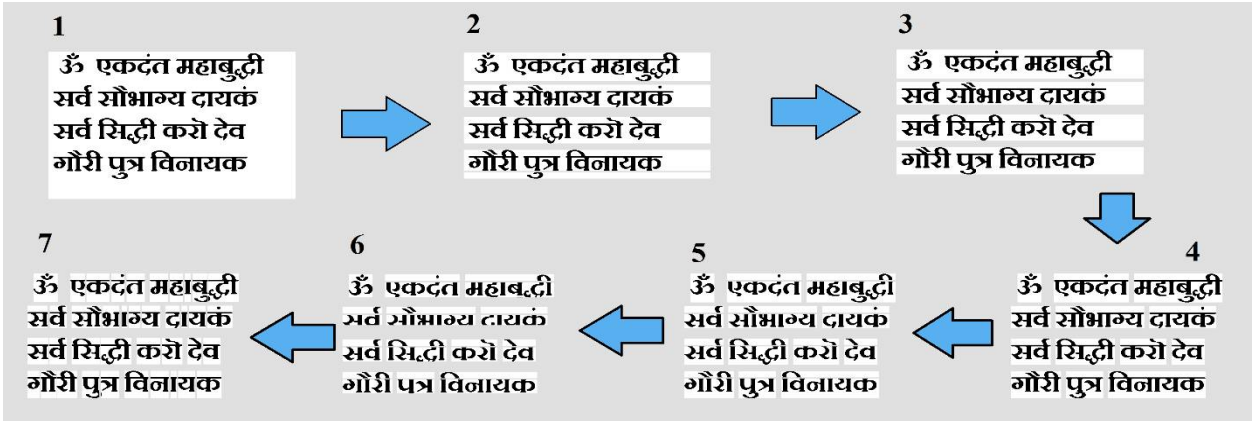

Fig. 5. Stages in Segmentation

2) *SEGMENTATION OF WORD:* To separate words from a line, rotate the line by -90 degree and apply the same logic that is used to separate lines. The output ofthis is then rotated back by 90 degree and we get segmented words. Refer step 4 in fig. 5.

3) *SEGMENTATION OF LETTER:* Segmentation of letters begins with the identification ofnormal word. If shirorekha is present then it is a normal word, else can be anumber, symbol or character like *Om*. For normal word the shirorekha is removed from the word image and it is rotated -90 degree. Apply the same logic which was used to separate lines for separating letters but skip the starting 20% width of image which denotes the descender of the letter.Refer step 5 to 7 in fig. 5.

Identification of a kana and a jodakshar is a challenging task. If the segmented letter is a kana then it must be associated as metadataof previous segmented letter. If the width of letter image(say letter17.png) is very less but greater than the width of a kana then we may need to merge it with the next letter image (say letter18.png). This is special case of a jodakshar (compound letter). The resultant of this must be given a separate name (say letter17A.png). However this may falsely merge two letters that is not a compound letter. To overcome this, the jodaksharas are identified based on the confidence given by trained model. Fig. 6 explains the two cases given by model.

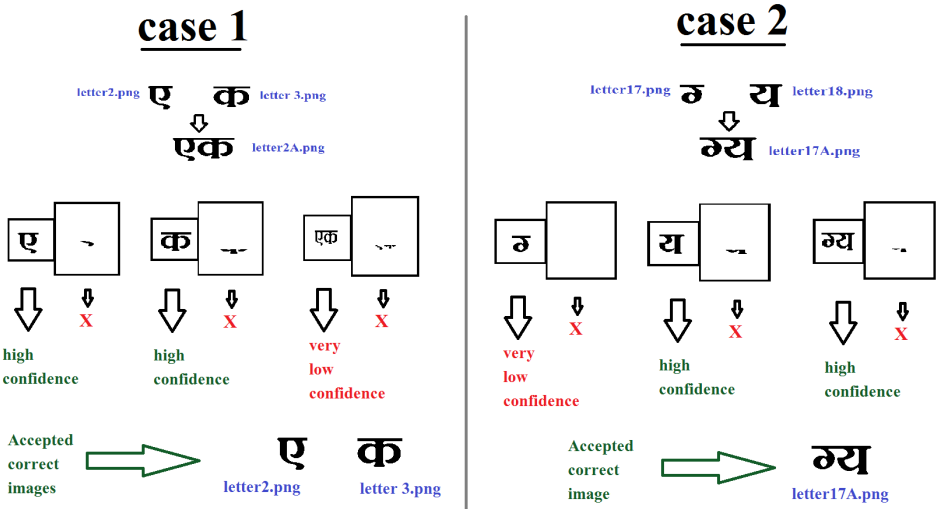

Fig.6. Cases in joining separated letters

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 3, March 2018

Iterate the model through every image, identify the shirorekha and separate the ascender. Some ascenders have their some part below their shirorekha (for example, ascenders pronounced as 'e', 'ee', 'o' and 'au'),so the ascender part is identified first and separated from core letter (fig.7). Further to this, if kana or an ascender is not present then create a copy of bottom 20% part.

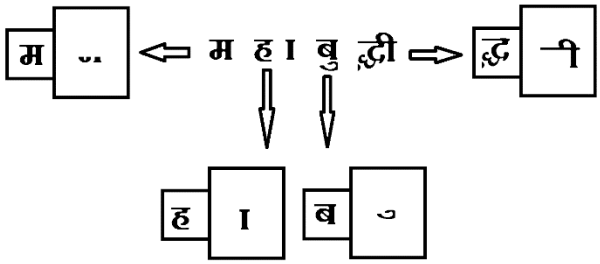

Fig. 7. Separating core letter from ascender and descender.

Fig.8 represents the case to deal with the possibility of descender. Three image copies are created consisting of original letter, upper 80% of original letter and image containing only lower 20% of original letter. Based on the confidence given by the classifier, the best solution is then considered which leads to zero error rate.

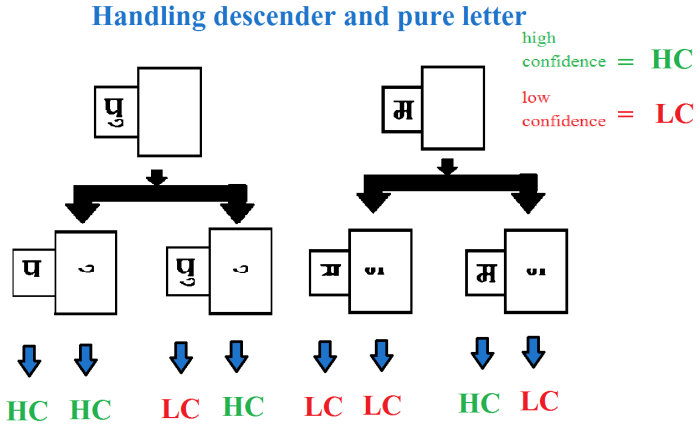

Fig. 8. Handling possibility of descender

IV. TRAINING AND CLASSIFICATION

TensorFlow library is used to train and build the machine learning model. It is an open source deep learning library which is provided by Google. Numerical computations are done using data flow graphs. The nodes present in the graph represent mathematical operations while the edges represent the tensors between them. Tensors are the data that move between the nodes which actually are multi-dimensional arrays consisting of real values. TensorFlow uses Convolutional Neural Network (CNN) for image recognition due to which the process of feature extraction is eliminated. Four separate models are trained for improving performance. 1st for ascender, 2nd for descender, 3rd for core and compound letters and 4th for symbols and numbers. Accuracy of MobileNetmodel and Inception model are compared in Table I. These are the results generated by training both models on twelve classes of Devanagari vowels i.e. 'a','aa','e','ee' to 'ah' where each class contains over three hundred and fifty to four hundred images. The value of accuracy is given by the model itself after completion of training.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 3, March 2018

TABLE I
INCEPTION VS MOBILENET OVER DIFFERENT ITERATIONS

Number of iterations	MobileNet model (in %)	Inception model (in %)
500	96.0	85.5
1000	96.4	87.6
1500	97.5	89.3
2000	96.8	91.4
2500	97.5	92.4
3000	97.5	92.1
3500	97.5	92.8
4000	97.5	93.4

TensorFlow library also provides a feature called TensorBoard to generate graph on trained models. Fig. 9 and Fig. 10 shows graphs generated for Inception and MobileNet model respectivelyover four thousand iterations.

Fig. 9. Accuracy of Inception Model over 4000 iterations

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 3, March 2018

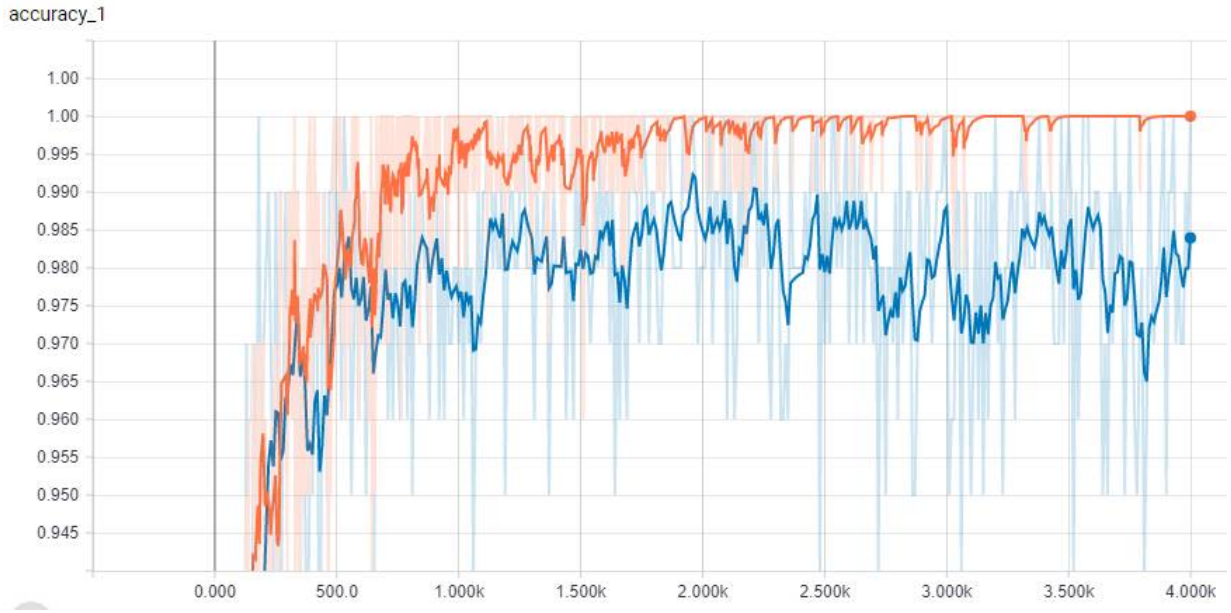

Fig. 10.Accuracy ofMobileNet over 4000 iterations

V. CLASSIFICATION

Classification is based on MobileNet training. MobileNet is a computer vision model. It is designed for devices having restricted resources that are small, low-power, low-latency models. MobileNet effectively maximizes accuracy.This model predicts the correct answer as one of its top 5 guesses. The one with highest confidence is taken as the output. If presence of ascender-or kana is detected then it can be core letter or compound letter. If not then it can be core or compound letter with or without descender, or it can be symbol.

VI. TRANSLATION

Translation is defined as the process of converting the source language to target language by preserving the meaning of the sentence. For this module, the system uses Googletrans which is a free and unlimited python library implemented by Google Translate API. This library has methods detect and translate which is called by the Google Translate Ajax API. We convert the output given by the OCR system to English script by making use of this API and store it in a text document.

VII. TRANSLITERATION

Transliteration is the process of transforming the source language to a target language by swapping the letters in order to preserve the pronunciation of the source language. This helps in cross lingual information retrieval. In our approach we have implemented the following algorithm for transliterating the Devanagari characters to English letters.
Step1: Map all the Devanagari characters to their respective English letter pronunciation and store the UNICODE of all consonants, vowels, nukta and virama.
Step2: Strip the document into words and then into letters. Check the range of letters.
Step3: Store the value of index in a variable.

a. In case of a consonant, check whether the next character is nukta, vowel or virama
   i. In presence of a nukta, jump to the next character.
   ii. In presence of a vowel, map the Devanagari character with its English letter.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 3, March 2018

iii. In presence of a virama, store it as it is.

b. In case of a vowel, map the character with its English letter.

Step4: Check whether the current character is the last letter.

a. If it is, then keep the word as it is.

b. If not, then append an 'a' to the output.

REFERENCES

1. S. Mori, C.Y. Suen, K. Yamamoto,"Historical Review of OCR Research and Development", Proceeding IEEE, 80, no 7, pp. 1029-1058, July 1992.
2. Veena B, R.M.K Sinha, "A Complete OCR for Printed Hindi Text in Devanagari Script", Proceedings of Sixth International Conference on Document Analysis and Recognition, IEEE Publiation, 2001.
3. Veena B, R.M.K Sinha, "On how to describe shapes of Devanagari characters and use them for recognition",Proceedings - Fifth International Conference on Document Analysis and Recognition, IEEE Publication, held at Bangalore from Sept 21-23, 1999, pp. 653-656.
4. Suryaprakash Kompalli, Sankalp Nayak, Srirangaraj Setlur, Venu Govindaraju, "Challenges in OCR of Devanagari documents", Proceedings – Eight International Conference on Document Analysis and Recognition, IEEE Publications, 2005.
5. Raghuraj Singh, C. S. Yadav, Prabhat Verma, Vibhash Yadav, "Optical Character Recognition (OCR) for Printed Devnagari Script Using Artificial Neural Network", International Journal of Computer Science & Communication, Vol. 1, No. 1, January-June 2010, pp. 91-95.
6. Honey Mehta, Sanjay Singla, Aarti Mahajan, "Optical character recognition (OCR) system for Roman script & English language using Artificial Neural Network (ANN) classifier", International Conference on Research Advances in Integrated Navigation Systems (RAINS), 2016.
7. B.B. Chaudhuri, U. Pal, "OCR error detection and correction of an inflectional Indian language script", Proceedings of the 13th International Conference on Pattern Recognition, 1996.
8. Ankita Srivastav, Neha Sahu, "A review to different approaches used for Devanagari characters segmentation", 3rd International Conference on Computing for Sustainable Global Development (INDIACom), 2016.
9. Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, Xiaoqiang Zheng, "TensorFlow: A System for Large-Scale Machine Learning", Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16), November 2–4, 2016.
10. Martin Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems", Preliminary White Paper by Google Inc, pp. 1-19, Nov 2015.
11. Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications", Preliminary White Paper by Google Inc, 2017.
12. P. H. Rathod, M.L. Dhore, R. M. Dhore, "Hindi and Marathi to English Machine Transliteration Using SVM", International Journal on Natural Language Computing (IJNLC) Vol. 2, No. 4, August 2013.