Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 6.017

IJCSMC, Vol. 7, Issue. 4, April 2018, pg.109 – 114

Handwriting Recognition of Diverse Languages

Mihir Shah[1], Sahil Mehta[2], Pranav Mody[3], Akhil Sen Roy[4], Sunil P. Khachane[5]

[1,2,3,4,5]Computer Department, MCT's Rajiv Gandhi Institute of Technology, Andheri West, Mumbai-400053, India

[1] shahmn96@gmail.com; [2] mehtasahil31@gmail.com; [3] pranavmodi96@gmail.com; [4] akhil.senroy@gmail.com; [5] sunil.khachane@mctrgit.ac.in

Abstract— Online Handwriting Recognition for Diverse Languages is a system which is used to recognize digital as well as handwritten inputs. In this paper we will compare two different methods which we used to serve the above purpose. The two methods are K-Nearest Neighbour(KNN) and Tesseract OCR. We begin this paper by introducing the main purpose of this paper which will include a summary of both KNN as well as Tesseract OCR (Optical Character Recognition). After that we will describe in detail the working of both methods, compare them according to their recognition accuracy. This system will also help in Licence plate number recognition, image extraction from images, extracting text from other types of documents, etc.

Keywords— Optical Character Recognition (OCR), k- Nearest Neighbour (k-NN), Binary Large Objects (BLOBS), Graphical User Interface (GUI), Portable Network Graphics (.png).

I. INTRODUCTION

This paper includes two different systems which were tested in order to recognize digital texts and more importantly handwritten inputs. The first system which we tested was based on k-NN algorithm. K nearest neighbours is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure (e.g., distance functions). K-Nearest Neighbours is one of the most basic yet essential classification algorithms in Machine Learning. It belongs to the supervised learning domain and finds intense application in pattern recognition, data mining and intrusion detection. **Tesseract** is an Optical Character Recognition (OCR) engine for various operating systems. It is an open source software which has a very good accuracy and with regular updates it is now able to recognize more than 100 languages. Accuracy of Tesseract OCR varies from 40% to 98%. Further we discuss about both systems in detail.

II. DESIGN AND IMPLEMENTATION (K-NN)

A. ARCHITECTURE:

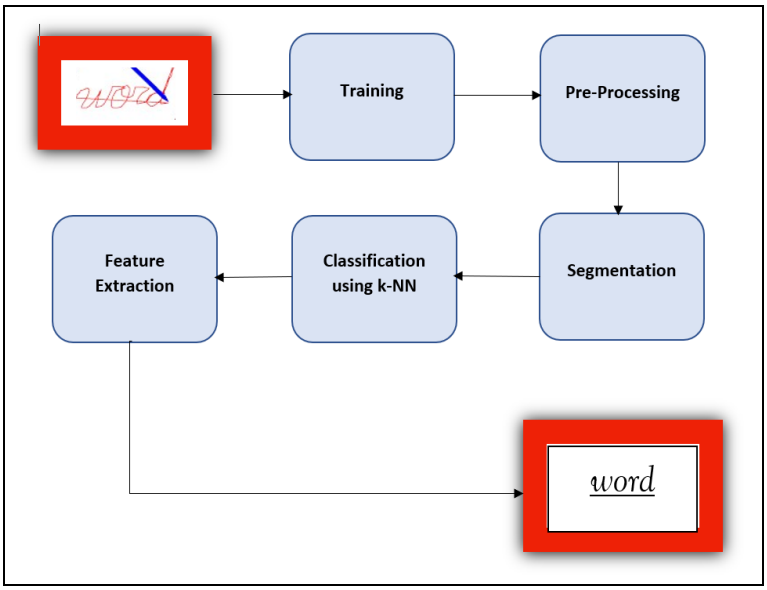

Fig. 1 Architecture of k-NN

There are five different phases in the architecture as follows:

1) **Training**: A training data set is taken as input and then it is segmented into individual characters. For each segment generated, target values are locked. This classification of training data is stored in ".BOX" files.
2) **Preprocessing:** The input image is first preprocessed i.e. uneven thickness; slant strokes and rotations are removed. This processed input is then passed for segmentation.
3) **Segmentation:** The preprocessed input is segmented into individual characters.
4) **Classification using k-NN:** This classification algorithm uses the results of trained data and on the basis of those results, it tries to recognize the input image using the distance formula of k-NN algorithm.
5) **Feature Extraction:** The classified input is then converted into and text and displayed to the user.

B. IMPLEMENTATION:

To perform this code, commands are given through the command prompt. The pseudo code for the k-NN is as follows:

Classify (**X**, **Y**, x) //**X**: training data, **Y**: class labels of **X**, x: unknown sample
for i = 1 to m **do**
    Compute distance d($\mathbf{X_i}$, x)
end for
Compute set I containing indices for the k smallest distances d($\mathbf{X_i}$, x).
Return majority label for {$\mathbf{Y_i}$ where i ∈ I}

While, testing we provided this code with two different images as inputs i.e. Image with Digital text and Image with handwritten text. The k-NN classifier is provided with a training data. After that an input image is been provided and this image undergoes Pre-processing, segmentation, classification using k-NN and feature extraction techniques after which the final output is been provided in ".png" format as well as on the command prompt.

RESULTS:

There were multiple input images which were provided to the k-NN based classification system. Some of the example images with their output images are as follows:

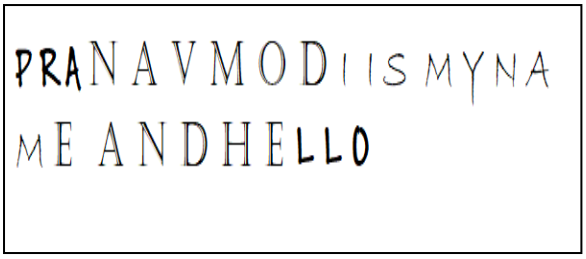

Fig. 2 Input image for k-NN

Fig. 3 Output image for Fig. 2

From the given input image, the accuracy obtained was 57.69%. This is the maximum accuracy which was obtained in the output for all the input images given. Even after training the system the accuracy didn't improve. This system was unsuccessful in recognizing the handwritten inputs. The maximum accuracy obtained in recognizing the handwritten inputs was less than 12%.

III.DESIGN AND IMPLEMENTATION (TESSERACT OCR)

A. ARCHITECTURE:

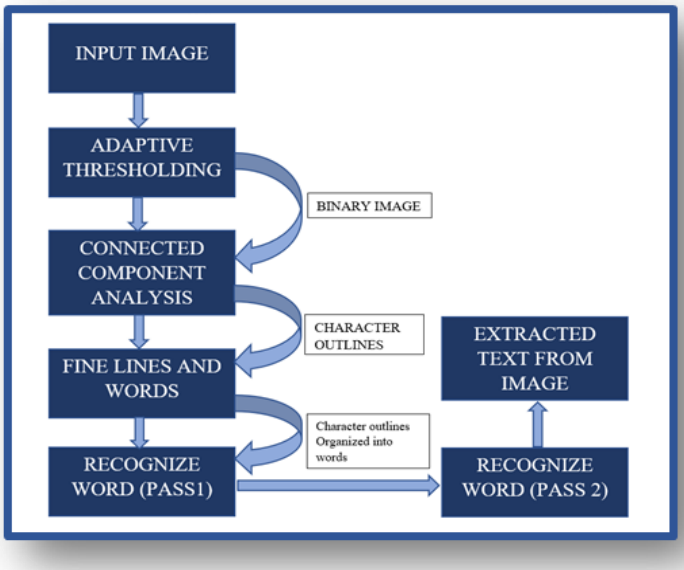

Fig. 4 Architecture of Tesseract OCR [1]

The Tesseract OCR works through various number of steps as follows:

1) **Select an input image:** This step includes human-machine interaction in which the user is expected to select a image in which the text is to be recognized. This image then goes through various processes after which the text inside the image is finally recognized.
2) **Adaptive Thresholding:** Adaptive thresholding is a technique which is used to segment an image by setting all pixels whose intensity values are above a threshold value to a foreground value. Eventually the pixels whose intensity values are below the threshold value are set to background value. This process is basically used to convert the input image to a binary image.
3) **Connected Component Analysis:** The advantage of this analysis is that by inspection of the nesting of outlines, and the number of child and grandchild outlines, it is simple to detect inverse text and recognize it as easily as black-on-white text.
4) **Conversion to BLOBS:** The Binary image is then converted into BLOBS (Binary Large Objects). BLOBS are divided into two consecutive functions. BLOB extraction and BLOB classification. Separation of different objects are called as BLOB extraction and classification of these different BLOBS is called as

BLOB classification. The term "LARGE" indicates that only the objects with a certain size are of importance. The other "Small" binary objects are considered as noise.

5) **Fine Lines and Words:** Outlines are gathered together, purely by nesting, into Blobs. Blobs are organized into text lines, and the lines and regions are analysed for fixed pitch or proportional text. Text lines are broken into words differently according to the kind of character spacing. Fixed pitch text is chopped immediately by character cells. Proportional text is broken into words using definite spaces and fuzzy spaces. [2]
6) **RECOGNIZE WORD (PASS 1):** In pass 1, an attempt is made to recognize each word. After a satisfactory recognition of the word, this word is passed to an adaptive classifier. Now, the advantage of passing the recognized word to the adaptive classifier is; that the classifier can provide more accuracy in recognition of the text lower down the page. As the recognized words are passed to the classifier the future recognitions of other text images become more and more accurate.
7) **RECOGNIZE WORD (PASS 2):** The adaptive classifier initially doesn't provide good accuracy in recognition of the text which are at the top of the page since it may have learned something very late to make any contributions in the beginning. Hence, this system runs a second pass over the image. The advantage here is that since the classifier has already recognized and learned a lot from pass 1, it can now make major contributions in providing very good accuracy in recognizing text at the top of the page and also recognize all those words in the entire image which were not properly recognized.
8) A final phase resolves fuzzy spaces and checks alternative hypotheses for the x-height to locate small cap text. [2]

B. IMPLEMENTATION:

In this system we have also tried to make a GUI (Graphical User Interface) to make it more user friendly.

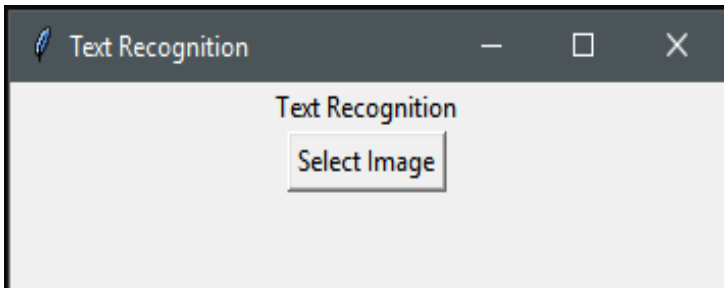

Fig. 5 User Friendly GUI for selecting an image

The Fig. 5 shows that this dialog box provides the user with an option to SELECT IMAGE from his device. When the user clicks on Select Image button it opens up the File explorer (For Windows) in the screen. The user then has to select the file, then the image in which the text has to be recognized. The Fig. 5 shows that on clicking the Select Image option various images that the user has stored are seen from his folders. Once the user selects an image from the files, the image's path is then passed to the code. This image is then the input image to the Tesseract OCR. The OCR will then perform all the steps and provide an output in a text format.

C. RESULTS:

We provided four different inputs i.e. Digital Input, Numbers, Handwritten Alphabets and a handwritten letter. The accuracy was calculated by comparing the output alphabet and the input alphabet or number.

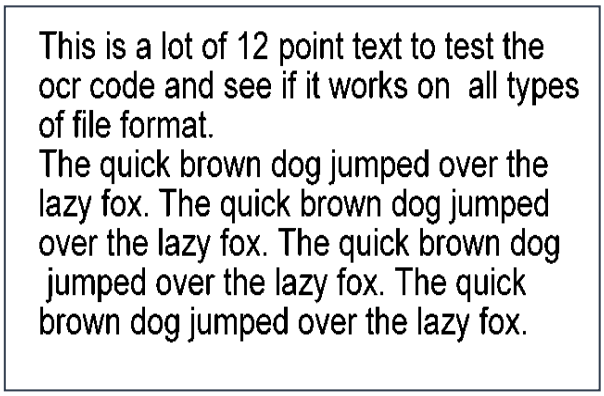

Fig. 6 Input Image 1 for Tesseract OCR

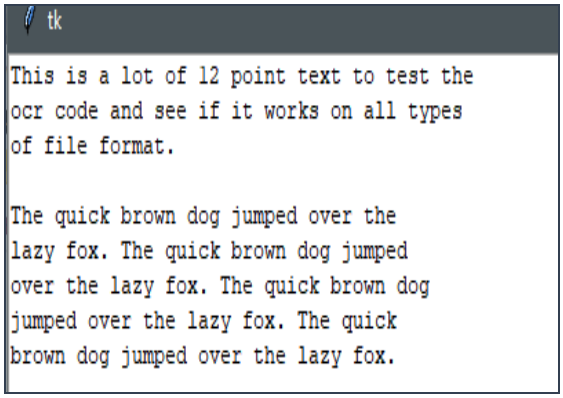

Fig. 7 Output Image for Input Image 1

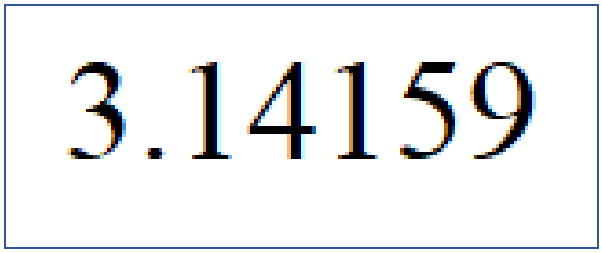

Fig. 8 Input Image 2 for Tesseract OCR

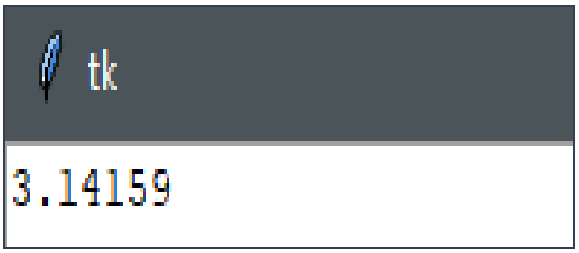

Fig. 9 Output Image for Input Image 2

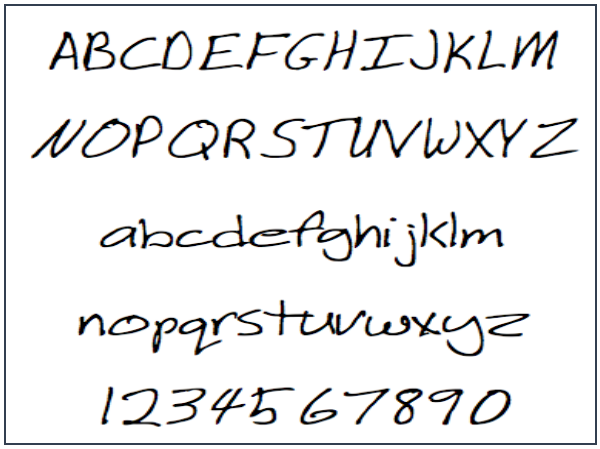

Fig. 10 Input Image 3 for Tesseract OCR

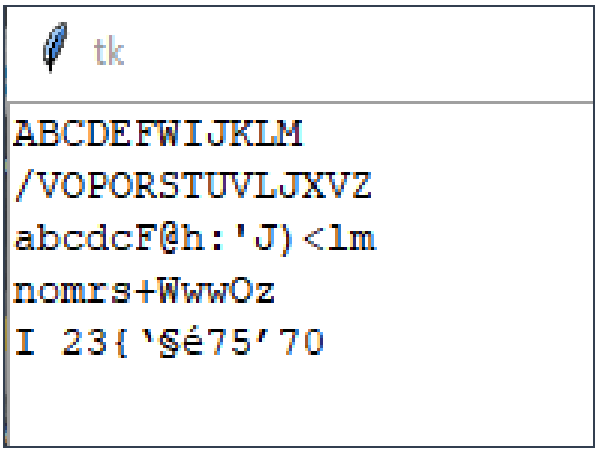

Fig. 11 Output Image for Input Image 3

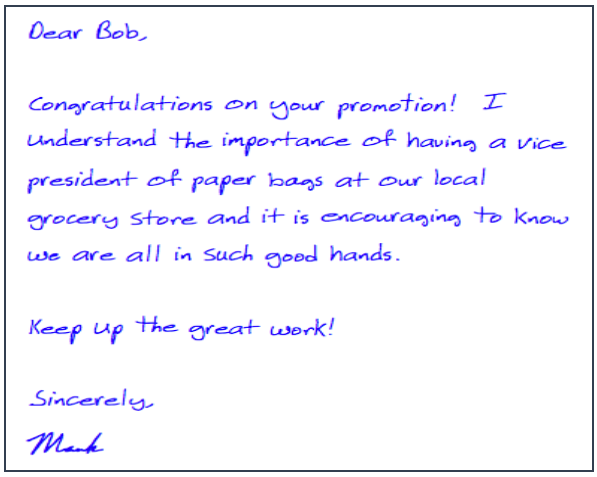

Fig. 12 Input Image 4 for Tesseract OCR

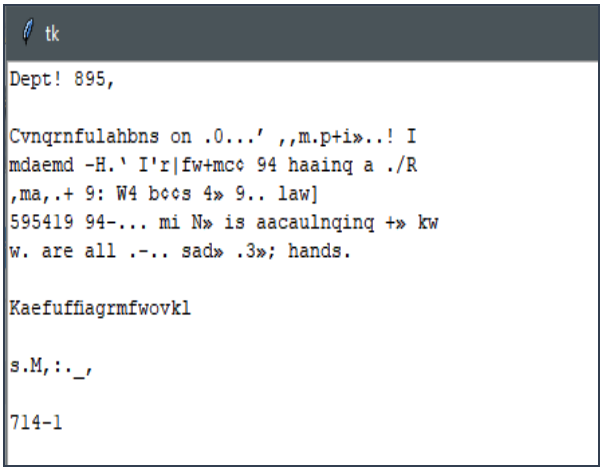

Fig. 13 Output Image for Input Image 4

In Fig. 10 are the alphabets and numbers in English language. These were the handwritten inputs to the tesseract. The alphabets were written with enough spacing and in upper as well as the lower case. Fig. 11 shows the output of the provided input. It was observed that the tesseract had provided the output with a very good accuracy rate. While in Fig. 12 the input was cursive handwritten English language. As seen in Fig. 13 the OCR did recognize a lot of characters with proper spacing but still had a low accuracy rate.

D. CALCULATION OF ACCURACY:

SR. NO.	INPUT IMAGE	OUTPUT IMAGE	ACCURACY (%)
1	Fig. 6	Fig. 7	98.72
2	Fig. 8	Fig. 9	100
3	Fig. 10	Fig. 11	62.90
4	Fig. 12	Fig. 13	14.65

The above table describes in percentage the accuracy rate of all the inputs.

IV. CONCLUSION AND FURTHER WORK

K-NN: By using this algorithm we can train the data but only up to a certain limit. It has been observed that after training the data again and again and executing the program the accuracy does not increase further. The maximum accuracy obtained here was only 57.69% but it had certain limitations.

a) It could not recognize any handwritten inputs which is the main drawback considering the purpose of our project.
b) Unfortunately, training the handwritten data also didn't help.

TESSERACT OCR: After a unsuccessful attempt of recognizing handwritten input and less accurate digital input using k-NN algorithm, we considered using the Tesseract OCR. Here it was observed that not only the accuracy of recognizing the digital input improved drastically by about 35-40% but the OCR could also recognize well handwritten input which served the main purpose of the project. Our attempts are to retrain the OCR so that the accuracy of recognizing the handwritten inputs also increase eventually.

While it is not possible to further train and improve the efficiency of k-NN, our attempts are to improve the efficiency of Tesseract OCR by retraining it. If this training becomes successful, it will become much more accurate in recognizing the handwritten input as well. Further, since OCR provides us with a large range of languages to be recognized, Tesseract OCR can be used for recognition of other languages too.

REFERENCES

[1] "*Optical Character Recognition by Open source OCR Tool Tesseract: A Case Study*." Authors: Patel, Chirag; Patel, Atul; Patel, Dharmendra. Publication: International Journal of Computer Applications, vol. 55, issue 10, pp. 50-56. Publication Date: 10/2012. Origin: CROSSREF. DOI: 10.5120/8794-2784.

[2] "*An Overview of the Tesseract OCR Engine"* Ray Smith Google Inc.

[3] 1. R. Plamondon and S. N. Srihari, *"On-line and off-line handwriting recognition: A comprehensive survey"*, IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, no. 1, pp. 63–84, Jan. 2000.

[4] *"Handwriting Profiling using Generative Adversarial Networks"* Arna Ghosh* and Biswarup Bhattacharya* and Somnath Basu Roy Chowdhury* Department of Electrical Engineering, Indian Institute of Technology Kharagpur 7 Nov 2016.

[5] *"Multi-Language Online Handwriting Recognition"*, Daniel Keysers, Thomas Deselaers, Henry A. Rowley, Li-Lun Wang, and Victor Carbune. IEEE Transactions on pattern analysis and machine intelligence, Vol. 39, No. 6, June 2017

[6] *"Preprocessing for real-time handwritten character recognition'*, Bartosz Paszkowski, Wojciech Bieniecki and Szymon Grabowski Katedra Informatyki Stosowanej, Politechnika LÃ o´dzka wbieniec@kis.p.lodz.pl

