SCIENCE EXPLORE
An International Journal on Science and Technology
Volume I, Issue 1
October - December 2020

OPTICAL CHARACTER RECOGNITION BASED TEXT ANALYSER: A CASE STUDY

S Selvakanmani, T Chandrashekar,
Nitin David Federick
A Mohammad Jaffar
Velammal Institute of Technology, Chennai

Abstract

OCR based Text Analyzer is a web application that is based on OCR (Optical Character Recognition). It extracts text by either capturing an image using the device camera or by selecting an existing image on the device. The text from the image is identified and displayed on the website. The Google search results for the modified text is then displayed on the website with the help of a suitable web scraping technique. The text recognition is done using an open – source OCR engine developed by Google, namely, Tesseract. This web application can be used for simplified searching as the user does not have to type out all the search words in the search engine. For instance, the user can find the meanings, additional information etc., about the scanned text and also get the digital text from an image. Further improvements on the accuracy and additional features such as translate, text to speech, sharing of the scanned text etc., can be implemented.

Key words: Optical Character Recognition (OCR), Text recognition, Camera captured image analysis, Tesseract, Open Source, Web application

INTRODUCTION

The conversion of the printed or scanned text into an editable text is called Optical character recognition (OCR) method. The main aim of the proposed work is to help the users extract text from an image and to simplify the search process. This way the user does not have to enter all the words that need to be given in the search box. The web application allows your mobile phone or computer to act as a text scanner as it will recognize the text in an image and display the same. It is based on the Tesseract engine which is an optical character recognition engine developed by Google. The user can also adjust the image brightness and contrast in such a way that the text becomes more legible before the actual recognition process takes place. A crop feature is introduced to allow the required portion of the image to be selected by the user. The search functionality is the main entity that distinguishes this project in comparison to the existing systems. The google search results for the recognized text will be displayed in a new tab. The OCR applications developed by Google are not compatible with certain devices whereas this project supports multiple platforms as it functions as a website. Some of the existing systems do not have the option to select an existing image from storage. The OCR based Text Analyzer also lets the user copy the extracted text and then save it as a text document for further use. The scanned text can be edited and stored in a document for reference.

LITERATURE SURVEY

Optical Character Recognition (OCR) is defined as the process of digitizing a document image into its constituent characters [1]. It is a branch of computer vision and has a variety of applications. An OCR is not an atomic process but comprises various phases such as acquisition, pre-processing, segmentation, feature extraction, classification and post-processing [1]. For good quality and high accuracy character recognition, OCR techniques expect high quality o rhigh resolution images with some basic structural properties such as high differentiation of text from background [2]. In the existing systems, the text recognition based applications lack the key feature which is the search functionality. In some of these applications, the user can only select an existing image and cannot use their device camera for real-time image capture. Handwriting character recognition is a very tough job due to different writing styles of users as well as different pen movements by the user for the same character [1].Apart from this, these systems are platform-specific and may also depend on the device specifications. Some of the systems have used an inferior optical character recognition algorithm which makes it less accurate and inefficient. The code for existing mobile applications must be rewritten for other platforms and hence it becomes a tedious process.

- **Lack of search functionality:** In applications like Text Scanner and Text Fairy for the Android platform, the user will only be able to extract text from the image and will not get the search results for the selected text.
- **Platform- specific: The** existing systems are platform dependent and thus they are not accessible to everyone. For instance, an android application is only available to an-droid users.
- **Device compatibility:** Applications like Google Lens, Text Fairy etc., do not have support for some mobile devices as they have certain limitations regarding the kernel version and camera accessibility.

The proposed system has been designed to overcome the limitations of the existing systems. It is a computer vision based web application that will be able to run on any device. It extracts text from images using the Tesseract engine and may also allow the user to modify the selected text if necessary. The search feature will help the user gather more information on the selected text. Further improvements on the accuracy and additional features such as translate, text to speech, sharing of the extracted text etc., can be added. The main advantages of the proposed system are:

- **Search feature: The** user will have the option of searching for the selected text.

The search results will be displayed in a new tab. This helps in getting more information about a particular text. For instance, if a question is scanned from an image, the solution for that question can be easily searched.

- **Cross-platform:** The web-based application means it can be accessed on any de- vice with a camera. It is also supported on various operating systems and hence the developers do not need to write different code for different platforms.
- **Tesseract OCR Engine:** This system implements the Tesseract.js code which is based on an open-source optical character recognition engine developed by Google. It allows the application to have better accuracy and efficiency compared to the other algorithms used in existing systems.

The flow of the work is narrated in the given form. Section 3 describes the architecture of the proposed work whereas Section 4 narrates about the working of Tesseract OCR engine.

Section 5 depicts the implementation of the proposed work and Section 6 concludes the work followed by the references.

ARCHITECTURE

An industrial OCR application combines algorithms studied in detail by different re- searchers in the area of image processing, pattern recognition, machine learning, lan- guage analysis, document understanding, data mining, and other, artificial intelligence domains[3]. The system architecture consists of a step by step pipeline.

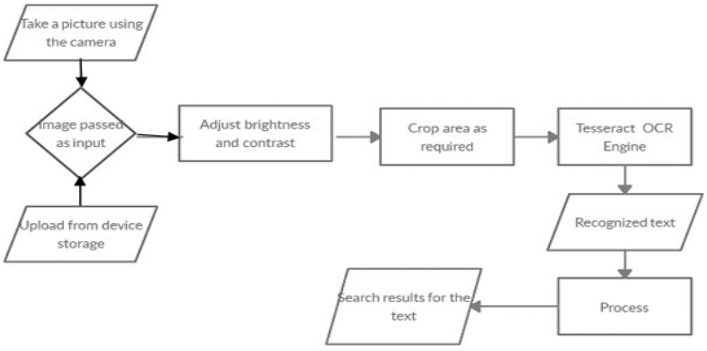

Figure 1: Architecture diagram of the Proposed work

The user's first step is to select an image from which text is to be extracted. The image can either be taken using the device camera or it can be uploaded from the memory. For designing an effective application related to the OCR, we must be considering the difficulties that may arise in each phase to obtain a high character recognition rate [2].

This is addressed in the next step which is to adjust the image brightness and contrast such that the text becomes more legible and can be detected more accurately. The user can crop out the selected region and it is passed to the Tesseract OCR Engine for processing. The input image is fed to the Preprocessing module, that converts input epigraphic image to Gray scale, enhances the image by reducing noise, and performs binarization [4]. The pre-processed epigraph is sent to the segmentation module which samples out the characters from the document and then the Feature Extraction module extracts relevant feature s[4].

Finally, Classification module labels the test character using a classifier and the classified characters are sent to Post processing module which maps the classified character to the present character [4]. The text is recognized and displayed as output. The final step is to search for the selected text and the results will be displayed in a new tab.

TESSERACT OCR ENGINE

Complex backgrounds, variations of text layout and fonts, and the existence ofuneven illumination, low resolution and multilingual content present a much greater challenge than clean, well-formatted documents[5].Solving these problems requires the application of advanced computer vision and pattern recognition techniques[5]. Tesseract is one such engine that helps with this. Tesseract OCR is an optical character reading engine developed by HP laboratories in 1985 and open sourced in 2005[6]. Since 2006 it is developed by Google. The latest Tesseract version is Tesseract 4. It adds a new neural net (LSTM) based OCR engine which is focused on line recognition but also still supports the legacy Tesseract OCR engine which works by recognizing character patterns[6]. Now it is developed and maintained by Google and provides support for various languages.

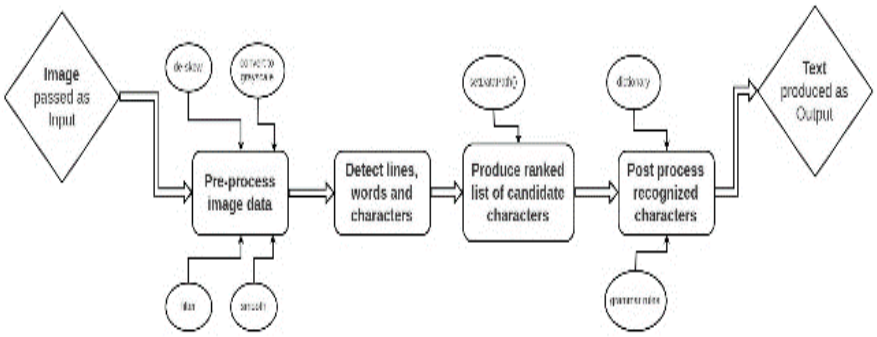

Figure 2: Tesseract OCR Engine

Generally, OCR works as follows:

i. Pre-process image data, for example: convert to gray scale, smooth, de-skew, filter[6].Tesseract is not able to extract the text with 100% accuracy for colour images. So to get more accurate result the same image is converted to the gray scale image and OCR is performed on this image [7].
ii. Detect lines, words and characters [6].
iii. Produce a ranked list of candidate characters based on trained data set [6].
iv. Post process recognized characters, choose the best characters based on confidence from previous step and language data [6]. Language data includes a dictionary, grammar rules, etc. Postprocessor can employ a spell checker and dictionary, probabilistic models like Markov chains and n-grams to improve the accuracy[1].

The outlines are gathered together from the gray scale, purely by nesting, into *Blobs*. Blobs are organized into text lines, and the lines and regions are analysed for fixed pitch or proportional text[8]. The text lines are broken into words differently according to the kind of character spacing. Fixed pitch text is chopped immediately by character cells whereas proportional text is broken into words using definite spaces and fuzzy spaces[8].Recognition then proceeds as a two-pass process and in the first pass, an attempt is made to recognize each word in turn and the satisfactory word is passed to an adaptive classifier as training data[8]. The adaptive classifier then gets a chance to more accurately recognize text lower down the page[8].Since the adaptive classifier may have learned something useful too late to make a contribution near the top of the page, a second pass is run over the page, in which words that were not recognized well enough are recognized again and a final phase resolves fuzzy spaces, and checks alternative hypotheses for the x-height to locate small cap text[8].

IMPLEMENTATION

The web based application is built using HTML, CSS and JavaScript. The steps are based on obtaining the image, performing pre-processing on the image, applyingan algorithm for character recognition and post-processing. The proposed system has the following modules:

IMAGE PROCESSING

The first module is the image processing module which includes selecting the input image, adjusting the image brightness and contrast, and cropping the required portion of the image. The image is either loaded from the device memory by selecting the ‘upload’ option or it can be captured using the device camera by selecting the ‘take picture’ option. Some

documents might come with embedded background images which often mislead the algorithms of character detection. For example, small dots or sharp edges from the background image are often bound-boxed as characters and passed to the next stage of the OCR pipeline, which causes an error chain[9].Thus, the image brightness and contrast can be adjusted to make the text more legible and to differentiate it from its background. This is done using glfx.js, an image effects library powered by WebGL. It uses your graphics card for image effects that would be impossible to apply in real-time with JavaScript alone.

The cropping functionality is performed to help the user select a certain area of text. It is implemented using the Jcrop, a jQuery plugin for image cropping. It combines the ease-of-use of a typical jQuery plugin with a powerful cross-platform DHTML cropping engine that is faithful to familiar desktop graphics applications[10]. The modified image is thus sent to the Tesseract OCR engine for extracting the text.

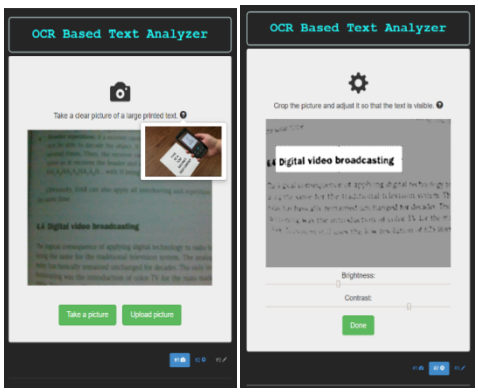

Figure 3: Image Processing

5.2 OPTICAL CHARACTER RECOGNITION

The image is passed through the Tesseract Optical Character Recognition engine to obtain the text from it. This is done by using Tesseract.js which is a JavaScript port of the Tesseract OCR engine[6]. Tesseract.js can run either in a browser and on a server with NodeJS which makes it available on a lot of platforms [6]. The main Tesseract.js functions (ex. recognize, detect) take an 'image' parameter, which should be something that is like an image.On a browser, an image can be an img, video, canvas element, a File object or a Blob object[6]. The extracted text is displayed for the user after recognition. The text accuracy and correctness depends on the clarity of the input image.

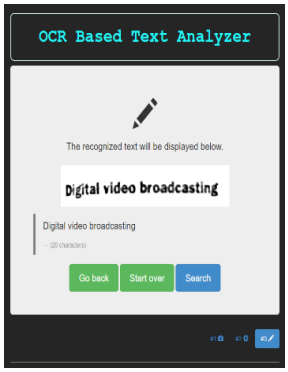

Figure 4: Analyzing using OCR

5.3 SEARCH

The final module is the search module which allows the user to get the google search results for the recognized text and image analysis. A new window containing the search results will appear and the user can navigate through this window and browse the obtained search results. This helps the user find the meanings of certain words as well as show additional information about the recognized text. For instance, the user can scan a question from a book and instead of typing out the entire question in the search engine, the user can extract the text and select the search option to get the answer to the selected question. The recognized text can also be copied and saved in a document. Further modules such as translation, text to speech, saving the text into a document etc., can be added.

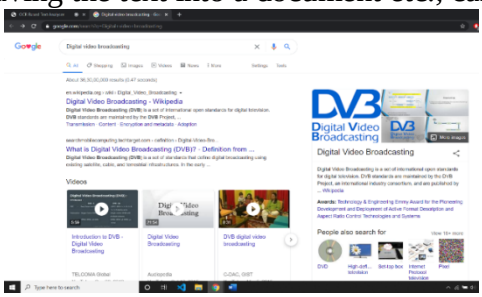

Figure 5: Searching using Google Search Engine

CONCLUSION

The 'OCR based Text Analyser' is a modern web application which aims to provide cross platform support across a wide range of operating systems and devices. It helps users extract text from images or from real time environments at any time or place. This application has a simple user interface and can be accessed by anyone with an internet connection. It gives the user control over the image and extracts the text from it. The search functionality makes this application stand out from all the other existing OCR applications.

FURTURE WORK

Optical Character Recognition is one of the many areas of computer vision that is being researched. There have been vast improvements and findings in the recognition of handwritten text. Thus, further additions can be made to the application in view of optimizing it with modern character recognition algorithms. Many other modules such as text to speech, translation, sharing of documents etc., can be included in the near future. However, the accuracy for text recognition in low resolution images can be improved as well as the recognition of handwritten text can be made possible with the help of modern algorithms.

REFERENCES

[1] N. Islam, Z. Islam and N. Noor, "A Survey on Optical Character Recognition System," *Journal of Information & Communication Technology-JICT,* vol. 10, no. 2, December 2016.

[2] K. Hamad and M. Kaya, "A Detailed Analysis of Optical Character Recognition Technology," *International Journal of Applied Mathematics, Electronics and Computers,* vol. 4, pp. 244-244, December 2016.

[3] I. Marosi, "Industrial OCR approaches: architecture, algorithms, and adaptation techniques," *Proc. SPIE 6500, Document Recognition and Retrieval XIV,* January 2007.

[4] S. A and H. K. G , "Recognition of Historical Records Using Gabor and Zonal Features," *Signal & Image Processing : An International Journal,* vol. 6, pp. 57-

69, August 2015.

[5] Q. Ye and D. S. Doermann, "Text Detection and Recognition in Imagery: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 37, June 2015.

[6] "Tesseract OCR with Java with Examples," [Online]. Available: https://www.geeksforgeeks.org/tesseract-ocr-with-java-with-examples/.

[7] C. Patel, A. Patel and D. Patel , "Optical Character Recognition by Open source OCR Tool Tesseract: A Case Study," *International Journal of Computer Applications,* vol. 55, pp. 50-56, October 2012.

[8] R. Smith, "An Overview of the Tesseract OCR Engine," *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007),* vol. 2, pp. 629-633, 2007.

[9] M. Shen and H. Lei, "Improving OCR performance with background image elimination," *2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD),* pp. 1566-1570, 2015.

[10] [Online].Available:https://github.com/newerton/yii2-jcrop. 2014