CSE 3200: Software Development Project 2

OCR Ultimate

Developed By:

Tanmoy Tapos Datta

Roll : 1307006

&

Arunima Mandal

Roll : 1307018

Supervised By:

Mahtab Ahmed,
Lecturer,
Department of Computer Science and Engineering ,
Khulna University of Engineering & Technology,
Khulna-9203, Bangladesh.

Acknowledgement

All praises goes to Almighty for his kindness & blessings.

Thanks to our project supervisor Mahtab Ahmed, Lecturer, Department of Computer Science and Engineering, Khulna University of Engineering & Technology for his untiring effort as well as strong support. He truly helped throughout the entire project with his correct decision & necessary advice we are able to complete this software development project.

Any suggestion, comment from teachers as well as seniors will be highly appreciated.

Table of Contents:

1. Objectives:

The aim of this project is to develop an android application which can show both real time and offline conversion of any handwritten text or printed text. The objective of this particular course is to make us known with the process of developing mobile application, to adapt with the systematic way of development. Basically we are using **Google Vision API**, **Google Tesseract & Open CV,** optical character recognition libraries powered by Google for detecting text from image.

2. Why use OCR Ultimate:

There are many cases such as converting an image into a machine readable text where the need of an OCR is felt. An OCR is Optical Character Recognition which converts an image into an electronic piece of text which can be read easily. Though OCR machine is very expensive and takes only scanned photos which further add to the cost, Google Drive gives you an option to do it for free. But what if you don't have a data connection all the time?

Yes, OCR apps are there in the market but they do need some improvements. Either they are not accurate or require a data connection so that you can upload your images to their servers and they'll convert it into a document. So today we have an app that is devoid of all these defects to some extent.

3. Introduction to Android:

Android is an operating system for mobile devices such as smartphones and tablet computers. It is developed by the Open Handset Alliance led by Google. It's is built on a Linux foundation. Google purchased the initial developer of the software, Android Inc., in 2005. The unveiling of the Android distribution on November 5, 2007 was announced with the founding of the Open Handset Alliance, a consortium of 84 hardware, software, and telecommunication companies devoted to advancing open standards for mobile devices.

This alliance shares a common goal of fostering innovation on mobile devices and giving consumers a far better user experience than much of what is available on today's mobile platforms. By providing developers a new level of openness that enables them to work more collaboratively, Android will accelerate the pace at which new and compelling mobile services are made available to consumers. Android is often symbolized by the green robot to the right.

Android has evolved rapidly since its launch. Google has named all projects after a dessert. The main releases are listed below, this is nothing you have to memorize, and it's just to illustrate the rapid pace of development and all the innovations. Android is developed "on Internet time", that is much faster than the old style of development (for example Windows releases which are typically several years apart).

4. Google Vision API:

Text recognition is the process of detecting text in images and video streams and recognizing the text contained therein. Once detected, the recognizer then determines the actual text in each block and segments it into lines and words. The Text API detects text in Latin based languages (French, German, English, etc.), in real-time, on device.

Recognized Languages

The Text API can recognize text in any Latin based language. This includes, but is not limited to:

- Catalan
- Danish
- Dutch
- English
- Finnish
- French
- German
- Hungarian
- Italian
- Latin
- Norwegian
- Polish
- Portuguese
- Romanian
- Spanish
- Swedish
- Tagalog

- Turkish

Text Structure

The Text Recognizer segments text into blocks, lines, and words. Roughly speaking:

- A **block** is a contiguous set of text lines, such as a paragraph or column,
- A **line** is a contiguous set of words on the same vertical axis, and
- A **word** is a contiguous set of alphanumeric characters on the same vertical axis.

The image below highlights examples of each of these in descending order. The first highlighted block, in cyan, is a Block of text. The second set of highlighted blocks, in blue, are Lines of text. Finally, the third set of highlighted blocks, in dark blue, are Words.

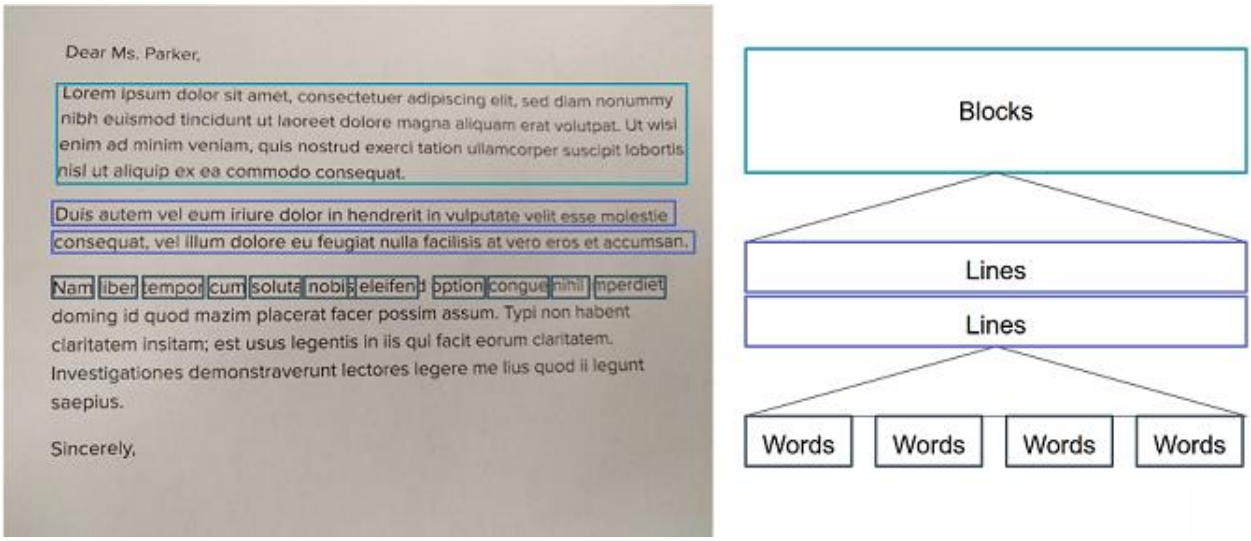

Fig 1: Working diagram of Vision API

5. Google Tesseract :

Tesseract is an optical character recognition engine for various operating systems. It is free software, released under the Apache License, Version 2.0, and development has been sponsored by Google since 2006. Tesseract is considered one of the most accurate open source OCR engines currently available.

- **History :**

  The Tesseract engine was originally developed as proprietary software at Hewlett Packard labs in Bristol, England and Greeley, Colorado between 1985 and 1994, with some more changes made in 1996 to port to Windows, and some migration from C to C++ in 1998. A lot of the code was written in C, and then some more was written in C++. Since then all the code has been converted to at least compile with a C++ compiler. Very little work was done in the following decade. It was then released as open source in 2005 by Hewlett Packard and the University of Nevada, Las Vegas (UNLV). Tesseract development has been sponsored by Google since 2006.

- **Features:**

  Tesseract was in the top three OCR engines in terms of character accuracy in 1995. It is available for Linux, Windows and Mac OS X, however, due to limited resources only Windows and Ubuntu are rigorously tested by developers.

  Tesseract up to and including version 2 could only accept TIFF images of simple one column text as inputs. These early versions did not include layout analysis and so inputting multi-columned text, images, or equations produced a garbled output. Since version 3.00 Tesseract has supported output text formatting, hOCR positional information and page layout analysis. Support for a number of new image formats was added using the Leptonica library. Tesseract can detect whether text is monospaced or proportional.

  The initial versions of Tesseract could only recognize English language text. Starting with version 2 Tesseract was able to process English, French, Italian, German, Spanish, Brazilian Portuguese and Dutch. Starting with version 3 it can recognize Arabic, Bulgarian, Catalan, Chinese (Simplified and Traditional), Croatian, Czech, Danish, Dutch, English, German (Antiqua and Fraktur script), Greek, Finnish, French, Hebrew, Hindi, Hungarian, Indonesian, Italian, Japanese, Korean, Latvian, Lithuanian, Norwegian, Polish, Portuguese, Romanian, Russian, Serbian, Slovak (standard and Fraktur script), Slovenian, Spanish, Swedish, Tagalog, Tamil, Thai, Turkish, Ukrainian and Vietnamese. Tesseract can be trained to work in other languages too.

  If Tesseract is used to process right-to-left text such Arabic or Hebrew the results are ordered as though it is left-to-right text.

  Tesseract is suitable for use as a backend, and can be used for more complicated OCR tasks including layout analysis by using a frontend such as OCRopus.

  Tesseract's output will be very poor quality if the input images are not preprocessed to suit it: Images (especially screenshots) must be scaled up such that the text x-height

is at least 20 pixels, any rotation or skew must be corrected or no text will be recognized, low-frequency changes in brightness must be high-pass filtered, or Tesseract's binarization stage will destroy much of the page, and dark borders must be manually removed, or they will be misinterpreted as characters.

6. OpenCV:

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 7 million. The library is used extensively in companies, research groups and by governmental bodies.

It has C++, C, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available. A full-featured CUDA and OpenCL interfaces are being actively developed right now. There are over 500 algorithms and about 10 times as many functions that compose or support those algorithms. OpenCV is written natively in C++ and has a template interface that works seamlessly with STL containers.

7. About our Project :

The name of our project is "**OCR Ultimate"**. It's typically a mobile based application which can extract texts both real time and offline (from images). User can hold the app in the source and get text in a clipboard instantly which is done through **Google Vision API** and can also can get extracted text from a captured image using **Google Tesseract.** But the image must be clear for getting proper result. This app doesn't work for blur images well.

We have used Google **Vision API**, for detcting text real time and **OpenCV,** an open source image processing library before detecting text via **Tesseract.** It has improved the detecting quality of **Google Tesseract**.

8. Features:

- User friendly graphical interface.
- Can get real time text extraction from any document.
- Can get text extraction from an image.
- One can choose image from the gallery. One should choose clear images because the app doesn't work for blur images.
- Get text from whether hand-written document or printed document.
- Before capturing an image, "Auto focus" switch button is there for getting clear images.
- If there is situation like one can't get enough light to capture clear image, then before capturing an image, "Use flash" switch button is there for getting clear images.
- After extracting the text, one can copy or edit it to any clipboard or document within the phone.
- The app needs no internet connection.

9. Workflow of OCR Ultimate:

The main design of our project is given below which describes the full functionality of our app:

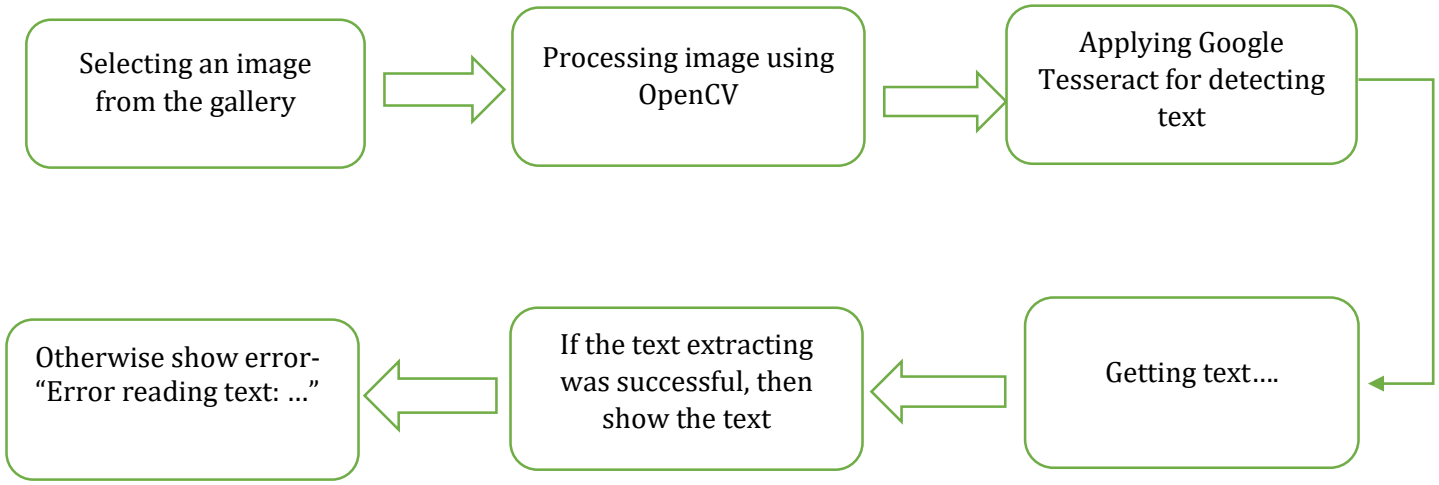

Fig-2: Work flow of OCR Ultimate

10. App Sample:

A. Main Page:

The app starts with a static page consisting of two image buttons. The Gallery button and the "Detect Text instantly" button.

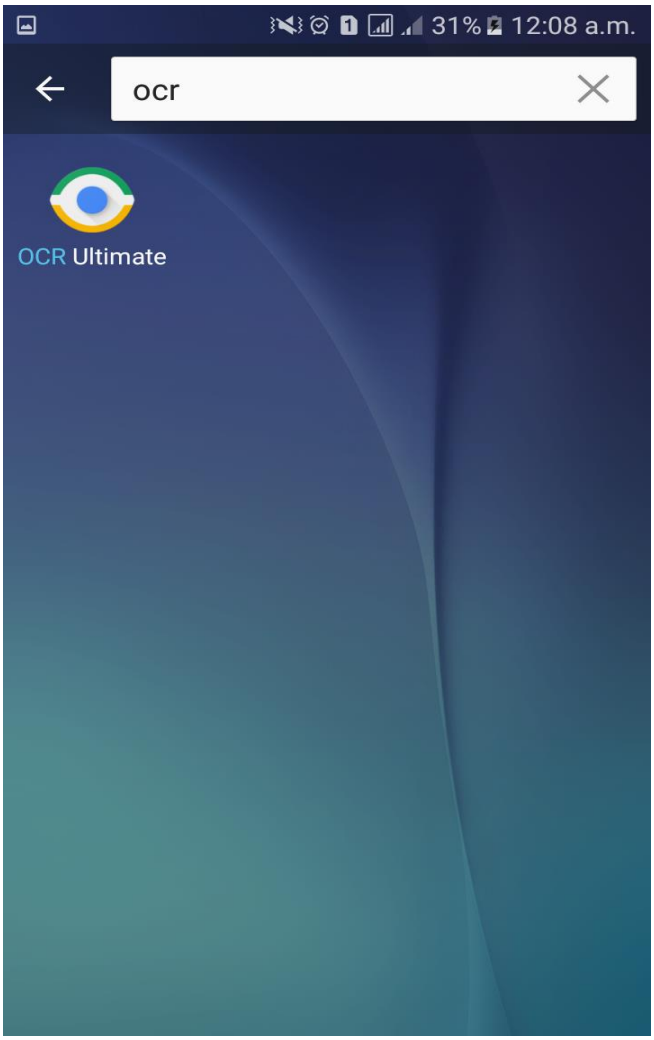

Fig 3: App icon in main menu

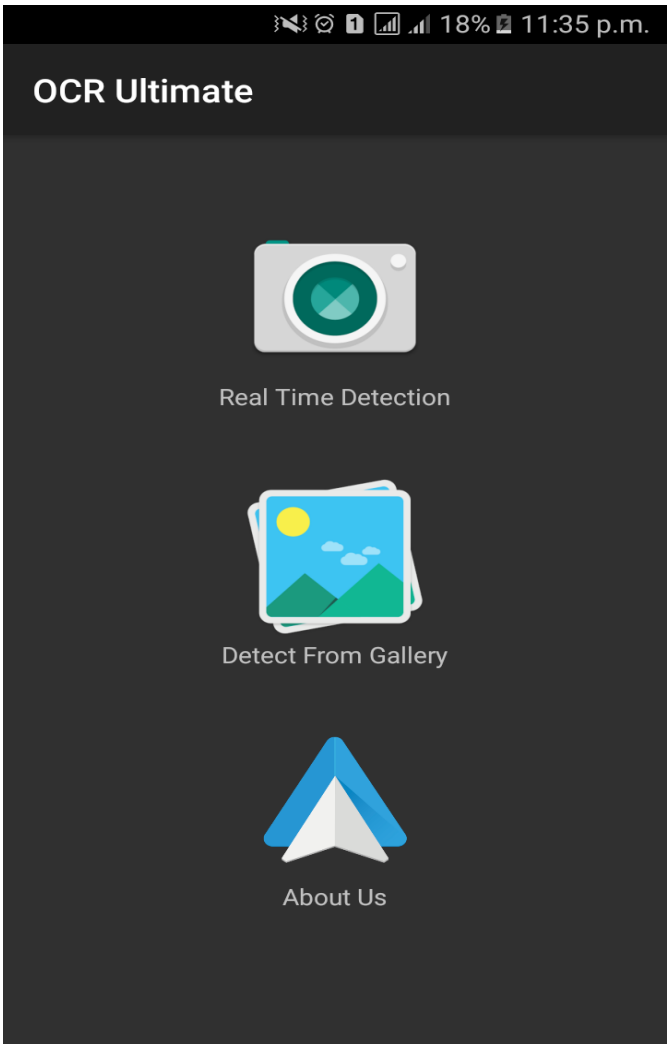

Fig 4: Home screen of App

Our project code consists of several classes and their corresponding packages. Most of the classes are shown in below screenshot.

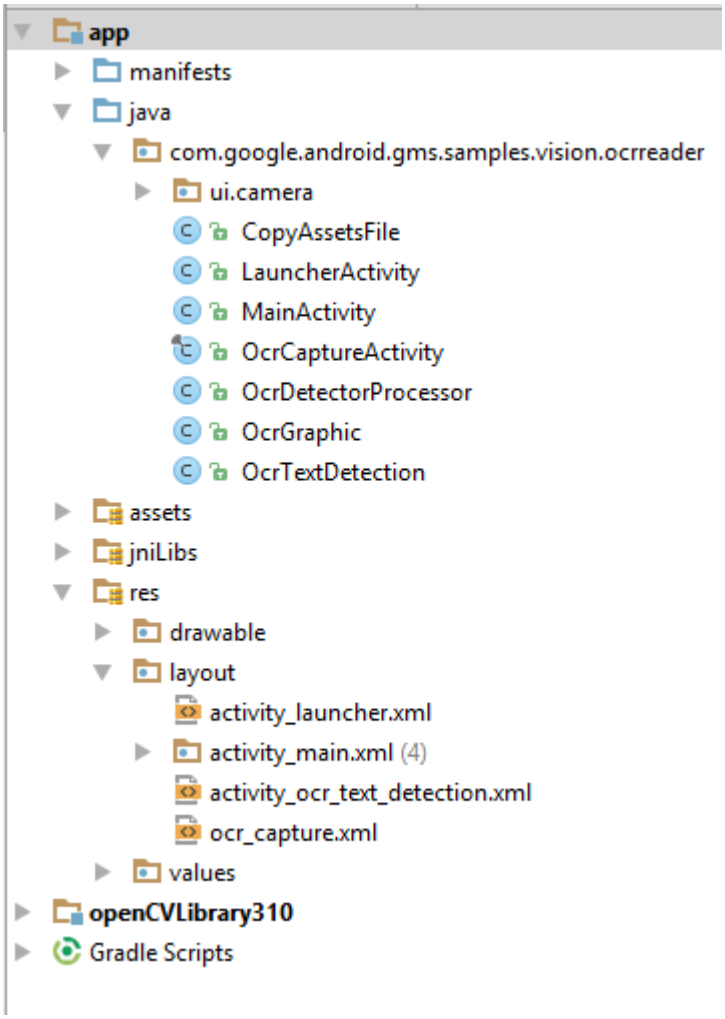

Fig 5: Sample class.

For detecting text from image we have used Tesseract. Init() method is used to declare Tesseract API. The sample code for detecting text through Tesseract is given below:

```
    @Override
    protected String doInBackground(Object... objects) {
        image = (Bitmap) objects[0];

        TessBaseAPI baseApi = new TessBaseAPI();
/ DATA_PATH = Path to the storage
/ lang = for which the language data exists, usually "eng"
        baseApi.init(context.getExternalFilesDir(null) + "/", "eng");
/ Eg. baseApi.init("/mnt/sdcard/tesseract/tessdata/eng.traineddata", "eng");
        image = image.copy(Bitmap.Config.ARGB_8888, true);
        baseApi.setImage(image);
        String recognizedText = baseApi.getUTF8Text();
        baseApi.end();
        return recognizedText;
    }
```

Fig 6: Tesseract code example.

But improving the detecting quality of text was not good at all. So, we have to use OpenCV for processing the image like this:

```
//This below code is for image processing using openCV.
Utils.bitmapToMat(bmp, imageMat);
Imgproc.cvtColor(imageMat, imageMat, Imgproc.COLOR_BGR2GRAY);
Imgproc.GaussianBlur(imageMat, imageMat, new Size(3, 3), 0);
Imgproc.threshold(imageMat, imageMat, 0, 255, Imgproc.THRESH_OTSU);
Utils.matToBitmap(imageMat,bmp);
```

Fig-7: OpenCV code example.

11. How to use the Application:

We have two modes for extracting text-

1. First one is Opening camera for detecting text –

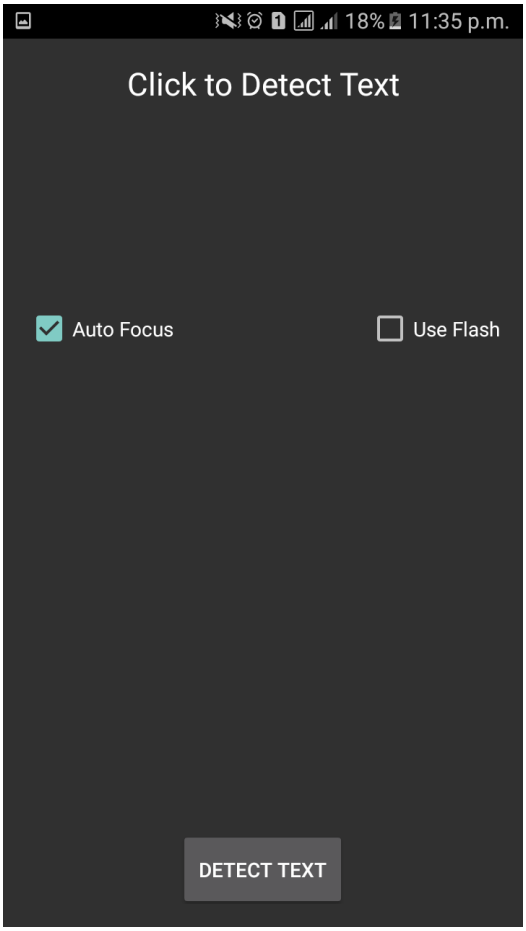

Fig 8: Detecting text using camera

After switching the button as necessary, when one click the “Detect text” button, instantly camera opens for detecting text. Then it instantly extract texts like -

Fig 9: Text detected

Then tap on any block & the result will be seen in a clipboard.

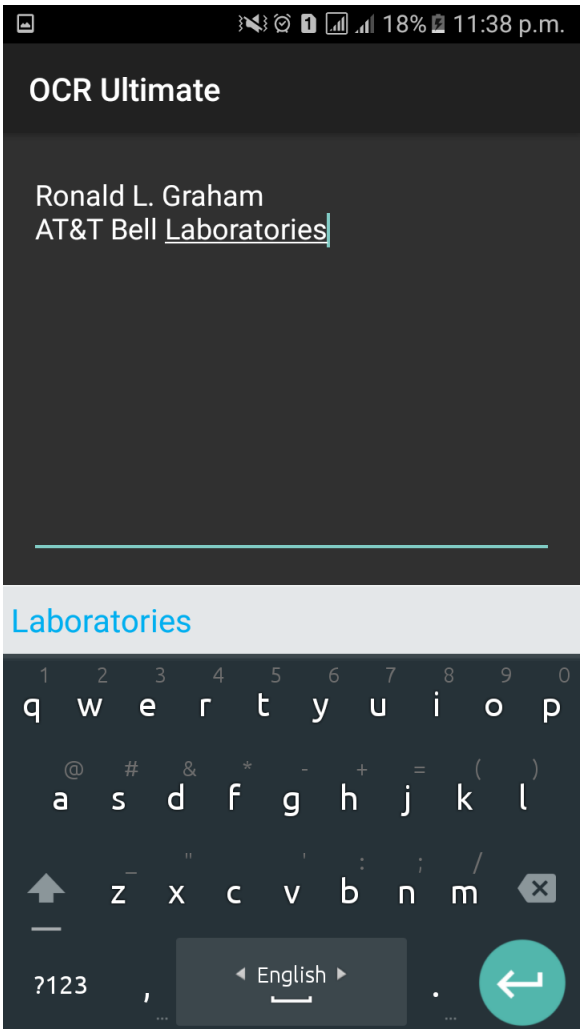

Fig 10: Result Text

2. Another mode is to select image from gallery. For that, this window will be open-

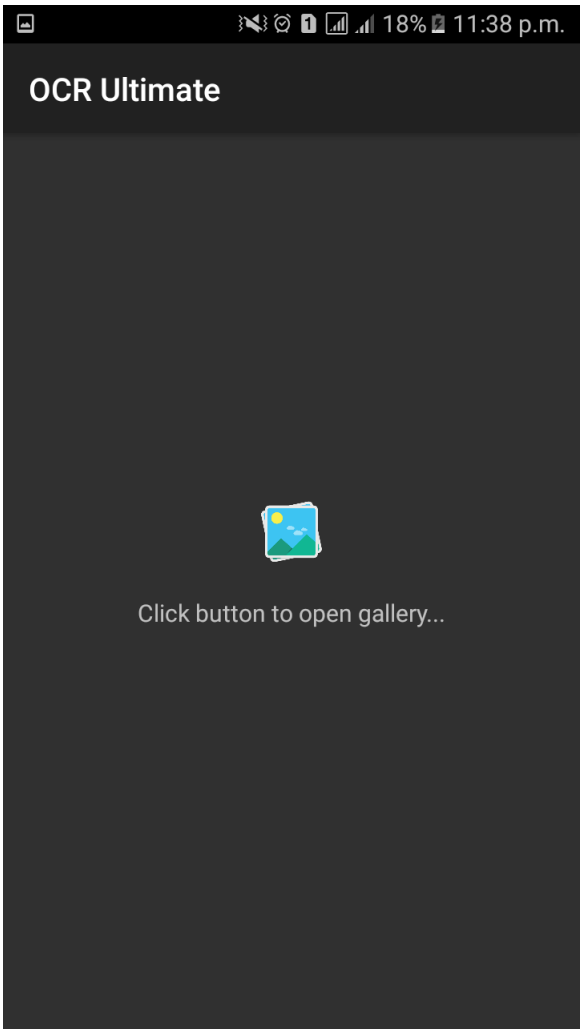

Fig 11: Selecting image from gallery

After choosing any image the app will process the image using Tessaract and will shoe "Getting text". We tested the image –

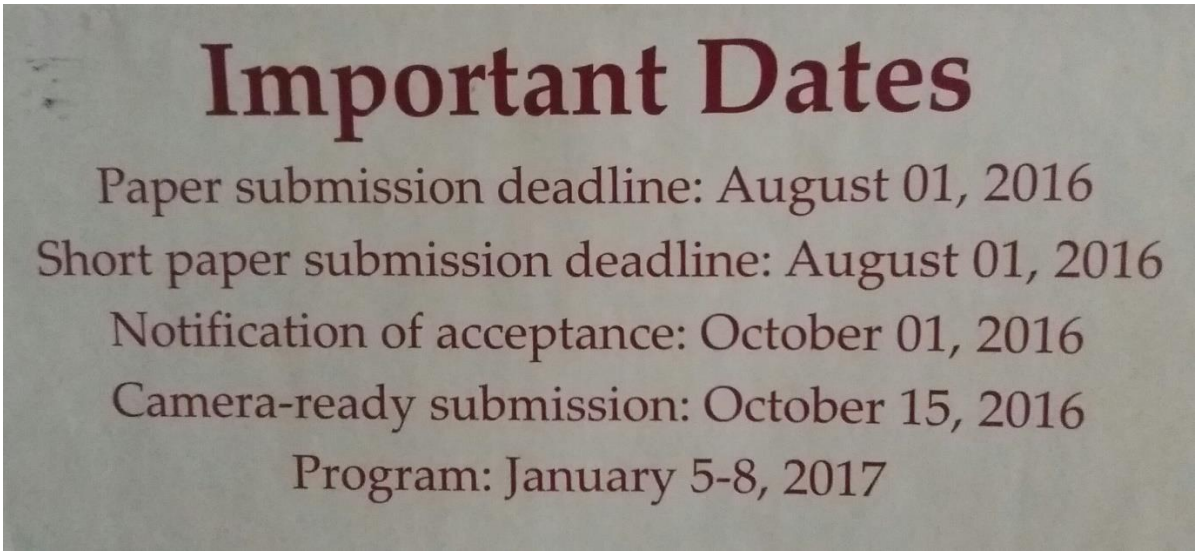

Fig 12 : Selected image

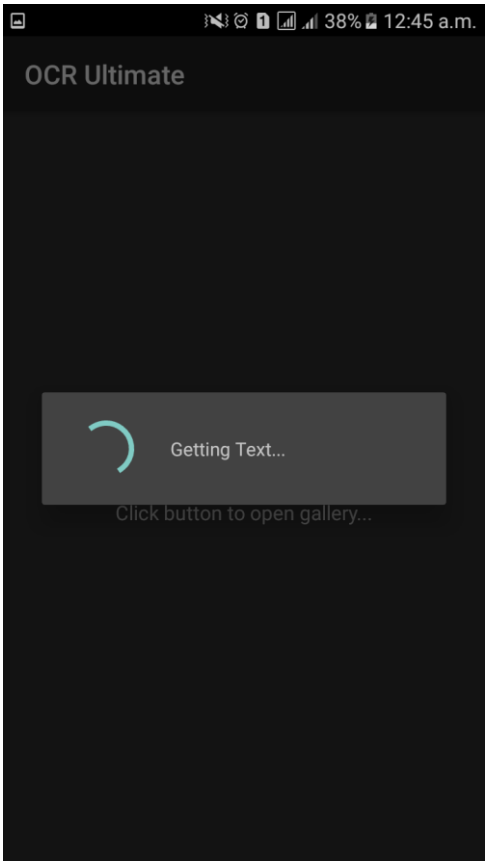

Fig 13: Analyzing image

12. Image Processing Steps:

Fig 14: Image processing steps

After analyzing the image, the result text is shown as below-

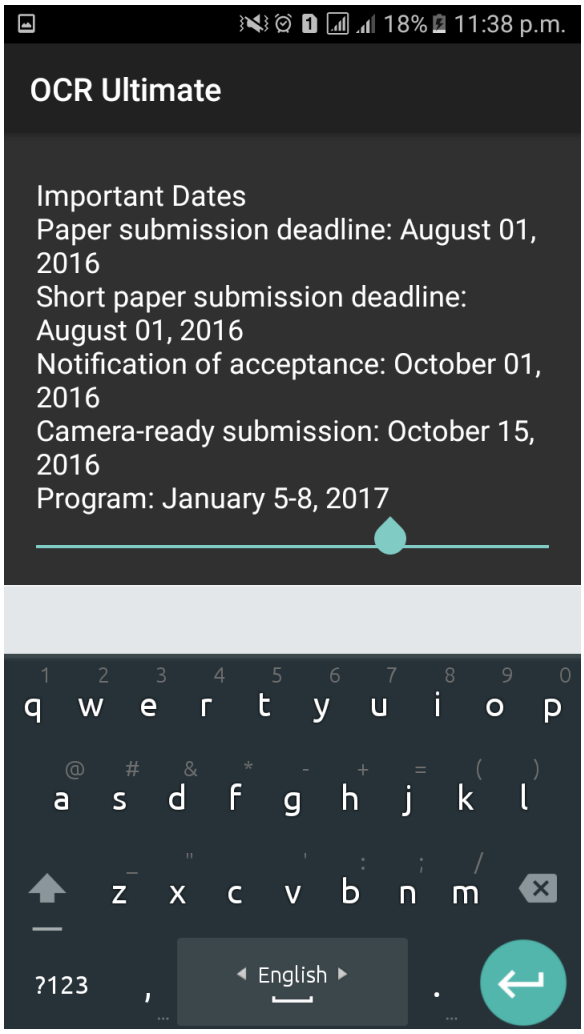

Fig 15: Result.

The same process for also acts for any hand-written document.

13. Application:

- When we need to call or save any mobile number from a visiting card or Bkash number from a printed copy in the shop or any mobile number printed or hand-written anywhere, to our phone, then we have to type the whole number in the dial pad by repeatedly seeing it from the source. It normally takes at least 8 second before calling or saving. But using our “Detect Text instantly” option, which is for real-time, one can just hold the app to the source and by just one click, the whole number will

be extracted in the screen. So this process is less disturbing and less time consuming than repeatedly seeing a number and type it.

- Again, there are situations one student anyhow missed one or two lectures and so he/she want to collect that from his/her friend. But we all feel comfortable in reading printed copy than handwritten copy of others. In this case our app is here to the rescue by scanning a image, extract all the text in a document which you can further edit or copy.
- When you need a copy of even some printed document which is messy, but you can't represent to others as this, then printing that document by a printer even doesn't help. Then you are goanna need this app for clean text which now you can print.

14. Limitations:

1. Only detect English language.

2. Sometimes Google Tesseract's performance is very poor.

3. Can select any one block at a time when detecting text instantly.

4. Doesn't work for blur images.

15. Future plan for Project:

We will make sure that this app can detect Bengali in future. Currently we are working on this. We are trying to increase the accuracy of detecting text more. We are trying our best for releasing this app to Google Play store.

16. Conclusion:

At First it was difficult for us to comprehend the concept of OCR. After studying a lot we came to know about Google Vision API & Google Tesseract .Our project supervisor Matlab Ahmed Sir helped us with the idea of using Open CV & Google Tesseract for detecting text and with his proper help we were able to create our project successfully in time.

17. References:

- http://opencv.org/
- Google Vision API
- OpenCV for Secret Agents - Joseph Howse.
- en.wikipedia.org