SnapSearch: A Mobile Application using Tesseract Algorithm

Princess Razene M. Almacen
Lyceum of the Philippines University
Capitol Site Batangas City
princess.almacen@lpubatangas.edu.ph
ORCID No. 0000-0001-8683-3764

Lorenzo Gabriel D. Leynes
Lyceum of the Philippines University
Capitol Site Batangas City
lorenzoleynes@lpubatangas.edu.ph
ORCID No. 0000-0003-0744-7043

Frances Eunice C. Ramos
Lyceum of the Philippines University
Capitol Site Batangas City
frances.ramos@lpubatangas.edu.ph
ORCID No. 0000-0001-9539-8012

Abstract

SnapSearch is an Android Mobile Application developed using the Tesseract Algorithm to create an OCR English dictionary application. It allows the user to capture an English word from an image through a smartphone's camera and converts it into digital text wherein the application displays the definition or meaning of the word. Tesseract Algorithm uses Otsu's Method for adapted thresholding to convert the image into black and white image for a more accurate conversion. Oxford Dictionaries API was used as the dictionary source. Java is the programming language used to develop the application. Testing results show that the font style can affect the accuracy of the extraction of the text. Tesseract Algorithm can recognize handwritten, sans and sans serif, and decorative font styles. Sans and sans serif are the most accurate font styles recognized by the Tesseract Algorithm.

Keywords

Android Mobile Application, Optical Character Recognition, Oxford Dictionary API, Tesseract Algorithm,

1.0 INTRODUCTION

English is a rising global language, and can potentially be the international language of communication. (Crystal, 2003). It is the most popular language spoken in 101 countries, and is the official language in 35 countries. 1.5 billion out of 7.5 billion people in the world speak and learn English, and is spoken as their first language by 360 million people. (Gamio & Noack, 2015). It is also the most commonly studied foreign language in the world. (Lyons, 2017).

During this Information Age, people tend to use mobile devices anytime and anywhere as an access to learning. (Dahlstrom, E., Walker, J. D., & Dziuban C. , 2013) Due to advancements in technology, and the presence of mobile applications, a digital dictionary is more useful, than a traditional paper dictionary. Digital dictionary is easier, cheaper, and quicker than a printed dictionary. If there are any changes to lexicographic content and presentation, dictionary applications, as well as online dictionaries can be updated as often as needed, and all users can instantly benefit from the improved content or features right from the moment these become available. (Lew & Schryver, 2014) It is in this context that, the study SnapSearch was formulated.

SnapSearch is a mobile application which substituted for a printed English dictionary. It captured text from an image through a smartphone's camera, then showed the etymology and definition of the word. This application is a tool to help people learn the English language, especially those people who don't speak English as their first language.

The study focused on the application of the Tesseract Algorithm to create an OCR application which identifies and extract images. It was an algorithm which commonly used on recognizing text such as printed and digital. The images to be extracted may be taken from a smartphone's camera or pictures that are present in the device's gallery.

This mobile application required an internet connection to display the meaning of the extracted word and can only detect English words. It can detect multiple texts but defined a single word only. It needed proper ambient lighting in terms of capturing the image that will be converted to image. The application required a smart phone with an Android version that is equal or higher than v.4.4 also known as Android KitKat and a recommended 7 megapixels with f/2.2 (ration of focal length to aperture) or more for the back camera.

Objectives of the Study

This study sought to attain the following objectives:

1. To develop a mobile application that will detect English text from an image and give its English definition.
2. To use Tesseract Algorithm to detect text from an image.
3. To use a dictionary API to define the scanned text.

2.0 LITERATURE REVIEW

This section discusses various references related to the problem and the corresponding solution on how to define text using a smartphone's camera without typing the text itself

Tesseract

Optical Character Recognition has been a great help to mobile applications. Many types of OCR software available to market but most of them have proprietary licenses, which are not open source and free. The researchers opted to use Tesseract Algorithm because it is one of the pioneer OCR software developed. It is an open source and free software with Apache license developed by Ray Smith in 1984 as a PhD project sponsored by Hewlett-Packard and further developed by Google. (Springmann, 2015)

It recognizes text, whether written, or printed with any type of font, and extracts it to digital text. (Patel, C., et. al., 2012)

According to Vithlani and Kumbharana (2015), there are several GUI desktop applications using Tesseract as an OCR Tool: Free OCR, PDF OCR X, YAGF, gImageReader, VietOCR, OCRFeeder, Lector, Lime OCR, QTesseract and SunnyPage. There are also several GUI web OCR applications using Tesseract: Free OCR, i2OCR, CustomOCR, and WeOCR. They tested the accuracy of Tesseract alongside other OCR Tools, where some have proprietary licenses, and the results show that Custom OCR Online, a Tesseract web application, gives the highest recognition because it can identify the most characters and digits.

Another open source OCR software, OCRopus, incorporated the text line recognizer of Tesseract in its system. OCRopus used Tesseract algorithm for constrained text line finding in its layout analysis method. (Breuel, n.d.)

Tesseract Algorithm

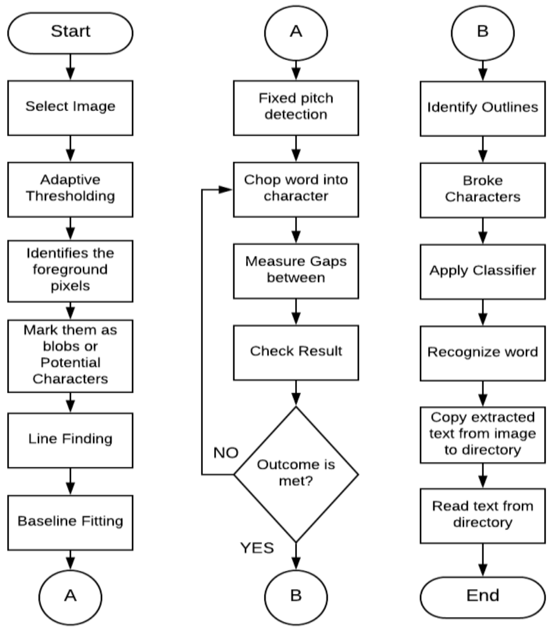


Figure 1 Tesseract Algorithm

In figure 2.0, the algorithm began by selecting an image either directly from your smartphone's camera or from gallery and will then proceed to adaptive thresholding. Adaptive thresholding performs the reduction of a grayscale image to a binary image then assumes that in an image there are foreground (black) pixels and background (white) pixels. Otsu's thresholding was used. After thresholding, the algorithm will proceed to search through the image, identify the foreground pixels, and marks them as "blobs" or potential characters. The next step is line finding. Blob filtering and line construction were the key parts of this process. To recognize a skewed page without having to de-skew, Line finding must be done, thus saving loss of image quality. Once the text lines have been defined, the baselines are fitted more precisely using a quadratic spline. This enabled Tesseract to handle pages with curved baselines, which are a common artifact in scanning, and not just at book bindings. Tesseract tests the text lines to know if the text is fixed pitch. When it detected fixed pitch text, Tesseract split the words into characters using the pitch, and measured the gaps between them. This procedure will repeat until a specific outcome is met. The outlines of the individual characters are then identified along with the broken or “incomplete” characters. To easily distinguish upper and lower-case characters, improve immunity to noise specks, and relative distances between the “broken” segments, the results were then classified using adaptive classifier. The results of which are matched with the prototypes of the “learned” characters, leading the words as recognized. (Smith, R. 2007)

Tesseract Architecture

Tesseract algorithm have several steps to work in a well-ordered way. Figure 1.0 shows the architecture of Tesseract:

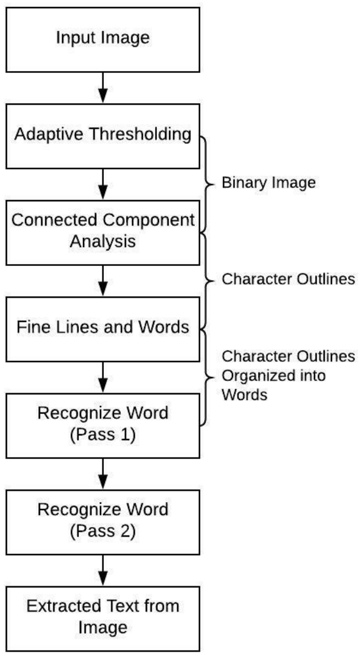


Figure 2 Tesseract Architecture

In figure 3.0, Tesseract Architecture uses Adaptive Thresholding to convert the image into binary images. Character outlines are extracted through Connected Component Analysis. Blobs were organized into text lines, and the lines and regions were analyzed for some fixed area or equivalent text size. Using definite spaces and fuzzy spaces, text was divided into words. The process for recognizing text was then started as two-pass process. The first pass was made to attempt to recognize each word from the text. Each word passed which was acceptable is passed to an adaptive classifier as training data. The adaptive classifier tried to recognize text in more accurate manner. As adaptive classifier has received some training data it has learn something new so final phase was used to resolve various issues and to extract text from images. (Patel, C., et. al., 2012)

API

Application Programming Interface, also known as API, is composed of routines, protocols, and tools for building software applications. It specifies how software components should interact and it is used to program graphical user interface components. Fundamentally, APIs make the development of programs by providing all the building blocks and resources. It is the interaction of software to another software. (Rana, V. K., 2016)

There are three types of APIs: (1) open APIs or public APIs which don’t have access restrictions and they are available for public use; (2) partner APIs which are private and used by strategic business partners, and need specific privilege to access them. And, (3) private APIs, also known as Internal APIs are not meant to be used outside the company and they are only used by internal systems for better productivity and reuse of services. (Rana, V. K., 2016)

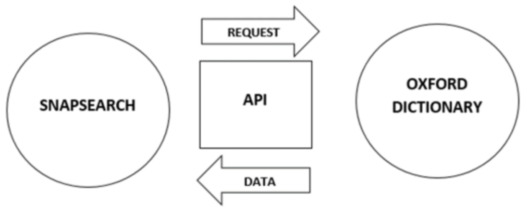


Figure 3 API Flowchart

Based on how systems or software interact with one another in building the application architecture, there are several APIs available for web service, hardware, operating systems, data structures, programmatic interfaces etc. A few of the standard API class libraries and frameworks are specifically helpful in performing specific actions and exchange of information. One of these, is Oxford dictionary API, which provides access to dictionary content. (Rana, V. K., 2016)

Oxford Dictionary

The Oxford Dictionary was published by the Oxford University Press, and it is the main historical dictionary of the English language. It provided a comprehensive resource to scholars and academic researchers, described usage in its many variations throughout the world, and traced the historical development of the English language. With all that said, Oxford Dictionaries API allowed easy access to world-renowned dictionary content.

Previous Researches

Wang and Yan (n.d.) developed a mobile visual text translator that can detect text from images and translate the text in real time by overlaying the translated text on top of the original one. The researchers explained that Tesseract algorithm is more suitable for the project than other OCR libraries such as ABBYY OCR, and Google Mobile Vision Text Recognition API because it was open source and has a wrapper for Android. Testing showed a successful text region detection rate of 89.3% and a successful Tesseract recognition rate of 71.6%.

Camarillas, Conway, Fernando, Jones, and Tangile (2017) proposed a mobile application developed using Tesseract Algorithm and engine to convert text in image form into digital format that can be easily edited and shared through various means.

In addition, Parwar, Goverdhan, Gajbhiye, Deshbhratar, Zamare, and Lohe (2017) used image text scanner, which is tesseract. The application enables the user to extract the image to text and able to translate, copy, edit, the text. The application also provides speak option.

Chakraborty, and Malik (2013) made an application that extracted text from images. The text is then converted into Braille. This application was useful for converting old valuable documents or books into Braille format. The researchers stated that Tesseract is one of the best OCR Engine offered because it is open source and free.

Furthermore, Bhaskar, Lavassar, and Green (n.d.) presented an algorithm for accurate recognition of text on a business card, given an Android mobile phone camera image of the card in varying environmental conditions. To optimize the image for input to the Tesseract OCR (optical character recognition) engine, the algorithm was implemented through MATLAB.

SnapSearch stood out from the existing similar mobile applications. Some of those apps available in the App Store and Google Play Store were WordSnap, Capture &

Translate, and Text Fairy. WordSnap is an iOS mobile application which instantly gives the meaning of a single word through a smartphone's camera. However due to its real time function of detecting the word and its meaning, there are limitations wherein it doesn't accurately detect the whole specific word. Meanwhile, Android mobile applications, Capture & Translate and Text Fairy, have very similar features with each other. They both scan a whole document then detect and convert the text into digital format, however Text Fairy could only scan a part of a document unlike Capture & Translate. The similar applications mentioned, specifically Capture & Translate and Text Fairy, have different features and objective compared to SnapSearch.

Flow of Application

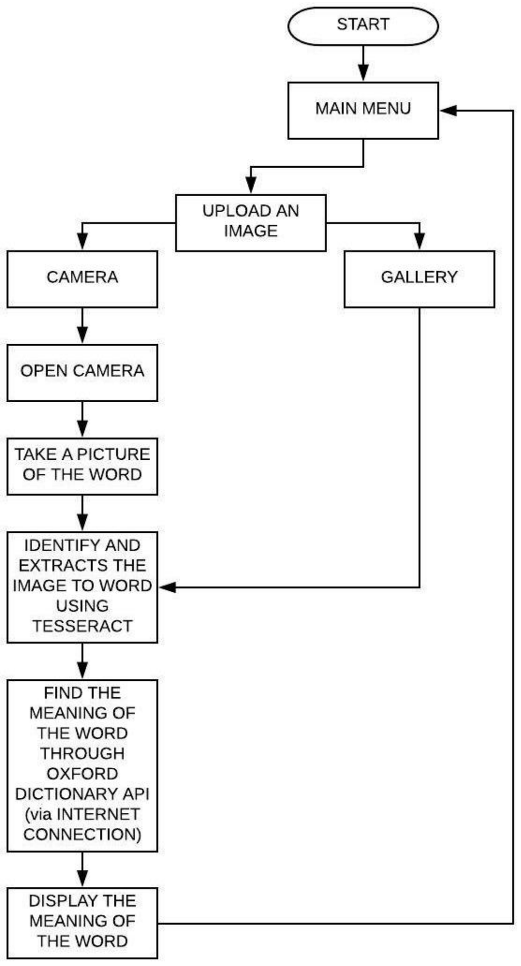

START
MAIN MENU
UPLOAD AN IMAGE
CAMERA
GALLERY
OPEN CAMERA
TAKE A PICTURE OF THE WORD
IDENTIFY AND EXTRACTS THE IMAGE TO WORD USING TESSERACT
FIND THE MEANING OF THE WORD THROUGH OXFORD DICTIONARY API (via INTERNET CONNECTION)
DISPLAY THE MEANING OF THE WORD

Figure 4 Application Flow

When the user starts, the application main menu screen will appear providing one (1) option named: Upload an Image as shown in figure 4.0. The first option will allow the user to select a source whether to upload directly on phone's built-in camera or upload via gallery. If the user selects camera, the application will let the user to take a picture of the word. It then identifies the word in image and extract into text format using tesseract algorithm. After extracting the image into text format, the word will search its meaning using Oxford's dictionary (API) as the search engine through an internet connection. While if the user selects Gallery, the application will enable the user to choose a picture from gallery.

3.0 METHODS

Java is the programming language used by the programmers to develop the application, which is object-oriented, developed by James Gasolin at Sun Microsystem. It is machine independent because instead of generating native machine code, byte-codes are generated. The compiled byte-codes are executed through the Java interpreter. (Sartipi, 1996)

The researchers programmed the mobile application through Android Studio, which was developed by Google. Based on IntelliJ IDEA, this is the official Integrated Development Environment (IDE) for the development of Android applications. To modify the build process, create multiple APKs with different features using the same project and modules, and reuse code and resources, the Android Studio uses Gradle build system. The researchers integrated the Android Virtual Device as an emulator to test and debug the application. The researchers incorporated Android Software Development Kit (SDK) to develop the application, and also used the built-in support of Google Cloud Platform.

Optical Character Recognition (OCR) is needed to be implemented for the application to function. To effectively develop the application, the researchers used the Tesseract Engine and its Algorithm. People who need to look up unfamiliar words from printed materials can benefit from this mobile application.

The researchers implemented the Oxford Dictionaries Application Programming Interface (API) to fetch the definition of the scanned text.

The minimum requirement of SnapSearch is Android 4.4 Kitkat.

Tesseract works best on images which have a DPI of at least 300 dpi, so it may be beneficial to resize images. The image is converted to black and white, but the result can be suboptimal, particularly of the image background consists of uneven darkness. Noise is a random variation of brightness or colour in an image, that can make the text of the image more difficult to read. Certain types of noise cannot be removed by Tesseract in the binarization step, which can cause accuracy rates to drop. A skewed image is when a page has been scanned when not straight. The quality of Tesseract's line segmentation reduces significantly if a page is too skewed, which severely impacts the quality of the OCR. Rotation feature is provided in the application to address this issue.

4.0 RESULTS AND DISCUSSION

The researchers tested the accuracy of the mobile application, Tesseract algorithm by scanning printed words. The words used were “nymph”, “spin”, “blitz”, “quick”, “vex”, “dwarf, and “joy”, which contain all the letters of the English Alphabet. Even though numbers can also be detected by tesseract, the researchers excluded the numbers since the mobile application only focuses in getting the definition of the word. The font type was categorized into four: Serif, Sans Serif, Decorative, and Script. The researchers tested the words by having five font styles from each font type. Each word was scanned ten times per font styles and divided into to two category, Upper case and Lower case. The accuracy was computed to obtain the overall accuracy of the application. The result is listed as follows:

Font Type	Upper Case	Lower Case	Average
Serif	96.30	90.86	93.58
Sans Serif	94.57	86.57	90.57
Decorative	66.86	55.14	61.00
Script	31.68	17.72	24.70

Table 1 Results of Testing

Table 1.0 shows the results of testing of accuracy of tesseract algorithm on different font family. The results listed on the table were the average for each font family. The font family highest accuracy rate was Serif, font type primarily used for books, and printed documents with 93.58%. The next is Sans Serif, another font type used for book and printed documents, with 90.57%. Following decorative with, 61.00%, while Script, fonts which are cursive with an unsatisfying result of 24.70%.

Font Type	Accuracy
Serif and Sans Serif	92.08
Script and Decorative	42.85
Overall Accuracy	**67.47**

Table 2 Accuracy of Font Types

As seen in Table 2.0, Serif and Sans Serif, font types which was primarily used for books and printed documents obtained the highest accuracy rate with 92.08%, following the Script and Decorative, font types commonly used for advertising and printed banners. Table 2.0 also shows the overall accuracy of the application with, 67.47 %.

Screenshots

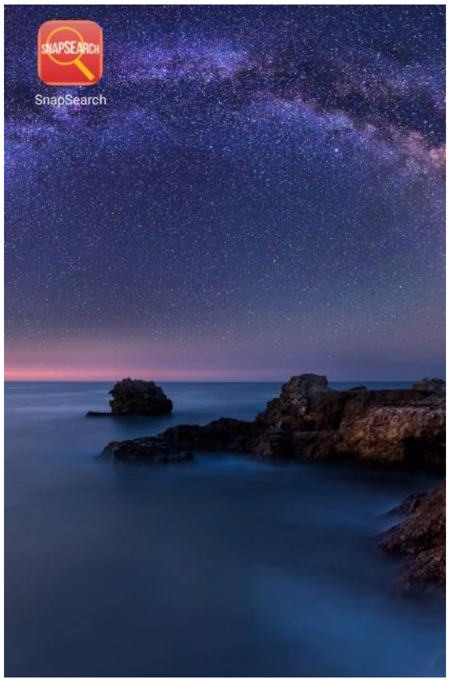
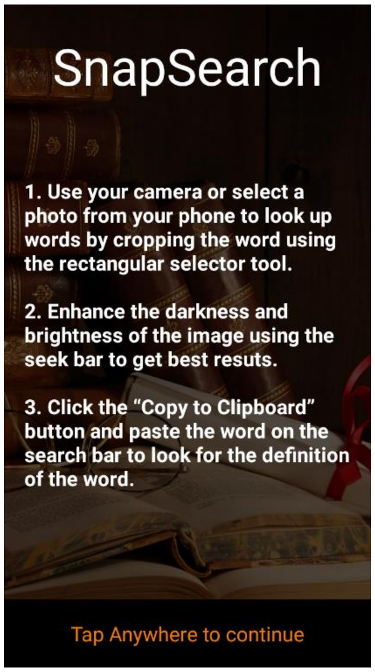


Figure 5 Launcher Icon

This is the icon of SnapSearch that the user will see on their home screen before they open the application.


SnapSearch

1. Use your camera or select a photo from your phone to look up words by cropping the word using the rectangular selector tool.

2. Enhance the darkness and brightness of the image using the seek bar to get best resuts.

3. Click the “Copy to Clipboard” button and paste the word on the search bar to look for the definition of the word.

Tap Anywhere to continue

Figure 6 Splash Screen

The splash screen contains the step by step instructions on how to use the application.

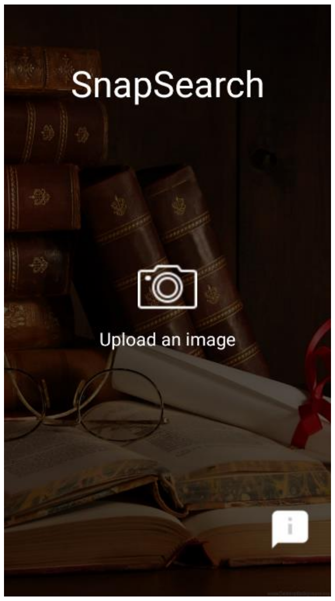


Figure 7 Main Menu Screen

This is the first screen that the user will see when the application is launched. The user can click anywhere to upload an image.

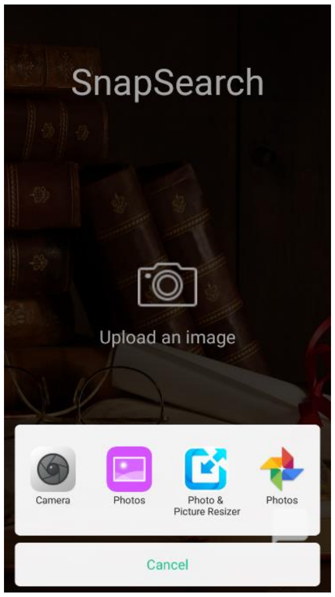


Figure 8 Select Source Screen

The user will select the source of the image where the text will be extracted from.

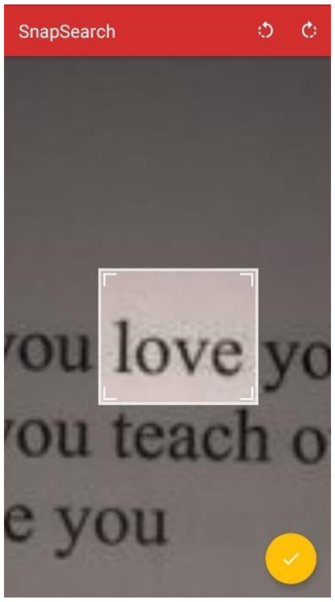


Figure 9 Crop and Rotate Screen

The user can adjust the rectangular selection tool to crop the background and focus on the desired text to be extracted.

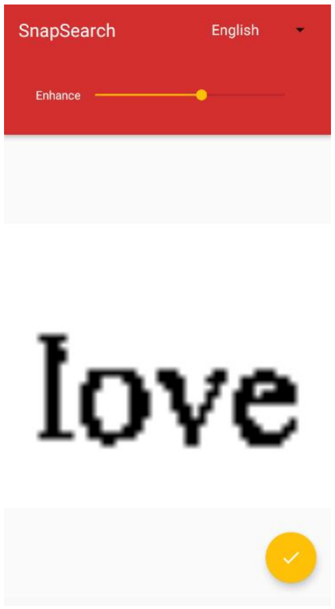


Figure 10 Picture Enhancement Screen

This screen shows the picture in black and white. The user can adjust the picture to make the text clearer for better binarization.

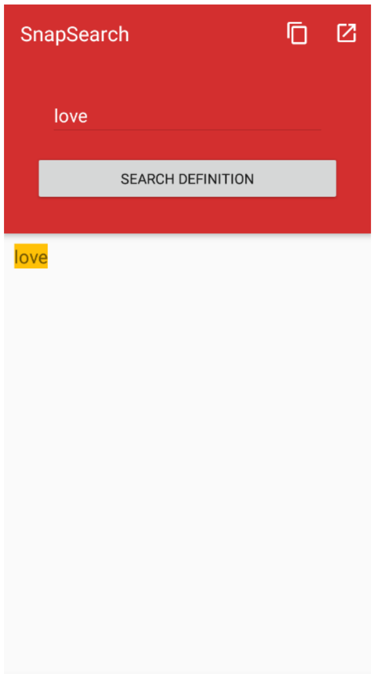
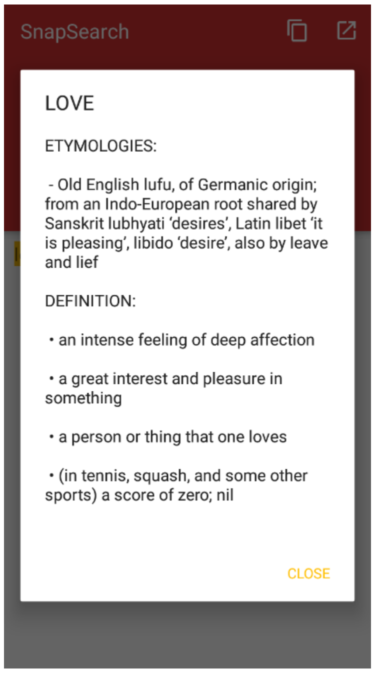


Figure 11 Search Screen

The scanned text will appear in the output view of the screen. The word needs to be copied to the clipboard and pasted on the search bar to find its meaning. The user needs to click the "Copy to Clipboard" button on the upper rightmost part of the screen and paste the word on the search bar. The "Search Definition" button enables the user search for the meaning of the word through the Oxford Dictionaries Application Program Interface (API).



LOVE

ETYMOLOGIES:

- Old English lufu, of Germanic origin; from an Indo-European root shared by Sanskrit lubhyati 'desires', Latin libet 'it is pleasing', libido 'desire', also by leave and lief

DEFINITION:

• an intense feeling of deep affection

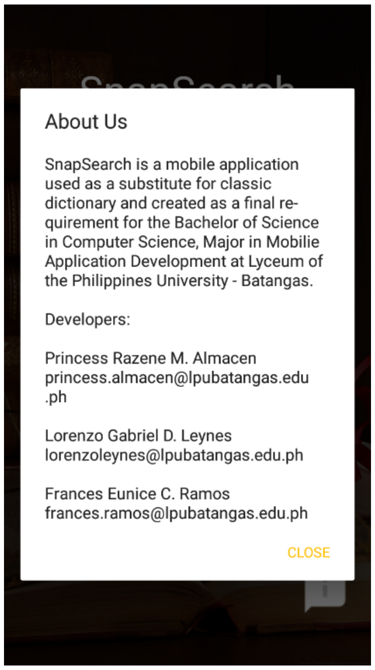
• a great interest and pleasure in something

• a person or thing that one loves

• (in tennis, squash, and some other sports) a score of zero; nil

Figure 12 Definition Screen

The definition, etymologies, and the word itself, will appear in a white pop-up screen.



About Us

SnapSearch is a mobile application used as a substitute for classic dictionary and created as a final requirement for the Bachelor of Science in Computer Science, Major in Mobilie Application Development at Lyceum of the Philippines University - Batangas.

Developers:

Princess Razene M. Almacen
princess.almacen@lpubatangas.edu.ph

Lorenzo Gabriel D. Leynes
lorenzoleynes@lpubatangas.edu.ph

Frances Eunice C. Ramos
frances.ramos@lpubatangas.edu.ph

Figure 13 About Us

The name and email address of the developers are shown in this screen.

5.0 CONCLUSIONS

In the overall composition of the mobile application, the researchers concluded that there are several factors that affect the accuracy: (1) the font styles, font sizes, and font groups of the captured word; (2) the specifications of the smart phone which includes, resolution of the camera, and noise reduction; (3) lighting conditions, angular positioning of the camera, and cropping result of the image which can affect the output of the binarization method; (4) poor internet connection is also one of the factors that affect the ability of the application to display the extracted text’s meaning.

As regards to font size and font style, the developers noticed that the larger text with no complicated font style that will be extracted from the image into text, the higher accuracy on detecting the word. The specifications of the smart phone especially in terms of the back camera have a vital role in the mobile application’s greater precision. Camera’s pixels and ratio of the focal length to aperture that have higher specification provides better result particularly if the text was captured with better lighting conditions. Natural ample light is advised to reduce the image noise which heightened the accuracy of thresholding in converting image to binary image. The user should also take a photo that is parallel to the word and crop it exactly along the text that will be extracted. The quality of the printed or digitized text with sharper word on the image is much easier to identify.

The researchers tested the application and obtained an overall accuracy rate of 67.46%. Serif fonts obtained the highest. Handwritten, particularly cursive words, are impossible to detect which could result to an error on the mobile application.

6.0 RECOMMENDATIONS

The developers recommend that the mobile application be improved by enabling to recognize not just English words, but also different languages even without Internet Connection. It is recommended to create a database for the history of words. It is also recommended to further develop this study to enhance the accuracy of the application in order to extract handwritten text from an image.

REFERENCES

[1] S. Bhaskar, N. Lavassar, and S. Green, Implementing Optical Character using Tesseract on the Android Operating System for Business Cards, n.d.

[2] T. M. Breuel. The OCRopus Open Source OCR System. https://pdfs.semanticscholar.org/8407/3be75980a6a8aabc11a00b4017f681362c08.pdf, n.d.

[3] J.I.V. Camarillas, R.J.M Conway, J.C.M. Fernando, J.S.F. Jones, and J.M.F. Tangile. Textor: A Mobile Application that Extracts Text From Image Format, 2017.

[4] P. Chakraborty, and A. Malik. An Open Source Tesseract based Tool for Extracting Text from Images with Application in Braille Translation for Visually Impaired, 2013.

[5] D. Crystal. English as a Global Language, Second Edition, 2003.

[6] E. Dahlstrom, J. D. Walker, and C. Dziuban. The ECAR Study of Undergraduate Students and Information Technology, Boulder, Colorado: EDUCAUSE Center for Applied Research. Retrieved from http://www.educause.edu/ecar, 2013.

[7] L. Gamio, R. Noack. The world's languages, in 7 maps and charts. https://www.washingtonpost.com/news/worldviews/wp/2015/04/23/the-worlds-languages-in-7-maps-and-charts/?utm_term=.1e6367a77d2b, April 2015.

[8] R. Lew, and G. M. D. Schryver. Dictionary Users in the Digital Revolution. International Journal of Lexicography, Volume 27, Issue 4, 1 December 2014, Pages 341–359, https://doi.org/10.1093/ijl/ecu011, August 2014.

[9] D. Lyons. How Many People Speak English, And Where Is It Spoken? https://www.babbel.com/en/magazine/how-many-people-speak-english-and-where-is-it-spoken/, July 2017.

[10] A. Parwar, A. Goverdhan, A. Gajbhiye, P. Deshbhratar, R. Zamare, and P. Lohe. Implementation to Extract Text from Different Images by Using Tesseract Algorithm. International Journal Of Engineering And Computer Science (IJECS), 2017.

[11] C. Patel, A. Patel, and D. Patel. Optical Character Recognition by Open Source OCR Tool Tesseract: A Case Study, 2012.

[12] V. K. Rana. Understanding API and their types. Retrieved from http://www.techulator.com/resources/15115-Understanding-API-and-their-types.aspx, January 2016.

[13] K. Sartipi. Java Programming Language. University of Waterloo, 1996.

[14] R. Smith. An Overview of the Tesseract OCR Engine. Proc. Ninth Int. Conference on Document Analysis and Recognition (ICDAR), IEEE Computer Society, 2007.

[15] U. Springmann. Module 6 Other OCR engines: ABBYY, Tesseract. http://www.cis.lmu.de/ocrworkshop/data/slides/m6-abbyy-tesseract.pdf, September 2015.

[16] P. Vithlani, and C. K. Kumbharana. Comparative Study of Character Recognition Tools. International Journal of Computer Applications (0975 – 8887) Volume 118 – No. 9, May 2015.

[17] W. Wang, and Q. Yan. Mobile Visual Text Translator. https://pdfs.semanticscholar.org/5c0b/4afae7c75c11915fe0c8488f67fea1e77034.pdf, n.d.