Automated recognition of highlighted text through a mobile application and image processing algorithms to enable better rates of recall

Shashank Rammoorthy
Bengaluru, Karnataka, India

Abstract

A mobile application that enables users to scan pages of textbooks and extract blocks of highlighted text was made. The application gathers highlighted text and presents these text blocks in a simple and intuitive user interface. Furthermore, the application periodically reminds users to revisit these notes (through the usage of push notifications), and quizzes them on the notes using machine learning algorithms.

Introduction

Highlighting of notes or textbooks is a widespread practice. It is considered an exercise in assigning varying degrees of emphasis, or sometimes urgency, to information presented in educational material.

The practice is common among students in secondary school and among university students of all ages. This is especially true of students studying a memorization-intensive subject - such as medicine or law. Highlighting also serves a broader purpose of refining information into the most important parts - the minimum effective dose, if you will. It helps in distilling informational material and can serve the dual purpose of minimising the time spent in rote memorisation and enabling broader comprehension (Fowler). Highlighting often serves as a summary of the text. It facilitates skimming and efficient revision (Turney).

However, there has been much resistance towards the practice of highlighting of late. Many an article have been published regarding the futility of highlighting as a tool for effective studying. Oftentimes, students mindlessly highlight parts of texts without real comprehension. It is hard to locate highlighted blocks of text in otherwise dense material, keeping track of these locations is often a task in itself.

Study techniques employed by students include re-reading the highlighted sections repeatedly in one sitting in the hopes of retention. These techniques are ineffective, inefficient and time-consuming. They inhibit retention rates. Often they are a barrier to understanding.

Highlighting is an often time-intensive chore. The highlighted text must be revisited later on to ensure deeper comprehension, but this is the step most students forgo. There is a definite need for an application that fills this void. An aid that summarizes all the highlighted text - and gives ready and mobile access to these essential parts of students' studying. A tool that brings up excerpts of lengthy and dense academic matter. A tool that aids in comprehension.

Studies in psychology have demonstrated how app notifications on mobile devices can trigger behavioural change. One hopes that habit formation can be triggered through the application presented in this paper. Frequency of phone usage has been shown to be highly influenced by the frequency and latency of push notifications - in other words, notifications sent from a remote server onto a handset (Kim).

Instead of attempting to ‘internalise’ or ‘cram’ all the highlighted segments of texts in one sitting - it is more efficient and useful to employ methods of spaced repetition. The application makes strategic use of push notifications in order to bring about comprehension and maximal performance on examinations for students.

Research

There are various methods through which the solution could be implemented - these include a desktop application, a web app where one can upload images or a smartphone application. However, the most effective from a convenience point of view would be a mobile application. A web application is an application run by a client on the browser.

Solutions like Quizlet (Sutherland) and Memrise have been quite effective in the classroom, but these do not always cater to the needs of the students. They do not give the students direction, because they are often not in total explicit control of the input material due to various constraints. These include the fact that students are often unwilling to devote hours simply transcribing from their textbooks to these web-based applications. Here, the highlighted text represents what the student perceives to be important text.

It further serves the purpose of reflection on the part of the student as to what will be the most efficient usage of his or her time.

The application was made with the Ionic Framework (using Angular 4) as the front-end component. It relied on the BottlePy Python microframework as the back-end. The Python code that dealt with the image processing component, that is, the extraction and identification of blocks of highlighted text, was made to run on the BottlePy framework. Additionally, Python code that dealt with contour detection and edge detection - to facilitate the ‘scanning’, as it were, of the pages of textbooks, were made to run from the backend.

These were all accessible to the frontend JavaScript implementation in Ionic using API calls with base-64 encoded images. After a batch of images has been chosen or taken through the application by the user, the base-64 encoded versions of the images are sent with the API call to the Bottle framework backend.

Methodology

The Ionic Framework was chosen for the development of the application due to factors including but not limited to - the fact that it enables the creation of a single codebase and multiple platform support, good community participation and the author's previous experience with the framework and building Angular 2 based applications.

Following is the workflow of the application:

1. User generates an image on the mobile device using the in-built utility in the Ionic Framework HTML5 Application.
2. Image is encoded as a base64 string using the toDataURL() method of the HTML5 Canvas element.
3. Base64 string is the parameter of the API Call to the Backend BottlePy framework. This is hosted on heroku.com, a Backend as a Service (BaaS) platform.
4. This base64 string is the input to the Python script hosted at the API endpoint. The image processing occurs here. This includes canny edge detection if needed, contour detection and adaptive thresholds.
5. The highlighted text extraction then occurs in the same function.
6. Lastly, an OCR API is called to recognize the text in the pixels extracted from the image. Both PyTesseract-OCR and Google Cloud Vision API were used in the testing stages. Google Cloud Vision API generally had the higher accuracy.
7. The text is sent back to the application, as a part of the HTTP response object.
8. The Python Natural Language Processing Library TextBlob is run on the text, and extracts entities. It also serves as a spell checker. The function implementation from TextBlob is straightforward and well-documented. Further details are enclosed in the Appendix.
9. As a part of the response to the Ionic Framework application, the following details are included:
    a. The highlighted text (user has the opportunity to edit and correct possible minor mistakes in recognition). This text, with a later version of the application, can be used in Machine learning datasets to improve accuracy of deep learning/machine learning algorithms that attempt OCR (Optical Character Recognition)

   b. The various entities - to enable the user to create a mindmap and deliberate over the relationship between said entities. Web scraping is conducted on Google search result pages with these query keywords. Additionally, these entities are used to perform a web search on the user's default search engine using the:
      i. BeautifulSoup Python library, useful for parsing HTML
      ii. Python WebBrowser module
10. These details (the response objects) are later rendered in Modal views in the Ionic application. They are essentially supplemental material from third-party providers, such as Wikipedia or Youtube.
11. The app makes use of Push notifications from provider OneSignal.com.
12. Later analyses of the data so obtained is beyond the scope of the current version of the project.

Working

This is a highly modular, decentralized application with no server-side code as of yet. I learnt quite a bit about how to build a software application from scratch through this independent research project.

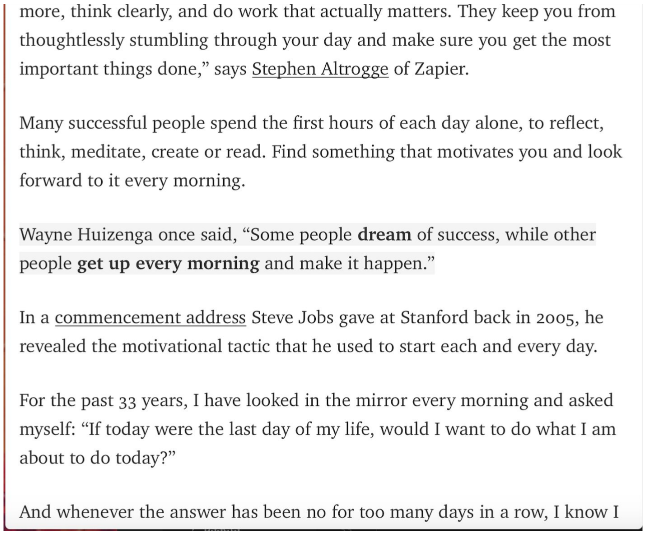
The Ionic application, which can be on either an iOS, an Android or a Windows device, works by calling API (Application Programming Interface) endpoints. Once an image is taken by the user, the application uploads it to Imgur, an online image sharing community as well as image hosting service. The images are uploaded, as base64 strings, to Imgur.com with the help of a POST request to the Imgur API. The URL is returned as a part of the API's response to the POST request.

This URL is later passed on another request to the PythonAnywhere Console, which is the location and host of the installation of the Web2Py framework for the project. The BottlePy framework is hosted locally as of now, but with the course of this project it will soon be loaded onto a Backend-as-a-Service such as Heroku.

On the Python-based web framework, the image is passed as a parameter to the *edgeDetect()* function as well as the *highlightRead()* function. The modified image is then passed through an OCR API, and text is passed in the response object.

This text is then picked up by the Ionic application, and is the basis of the app.

Attached are some miscellaneous images which relate to the project:



more, think clearly, and do work that actually matters. They keep you from thoughtlessly stumbling through your day and make sure you get the most important things done," says Stephen Altrogge of Zapier.

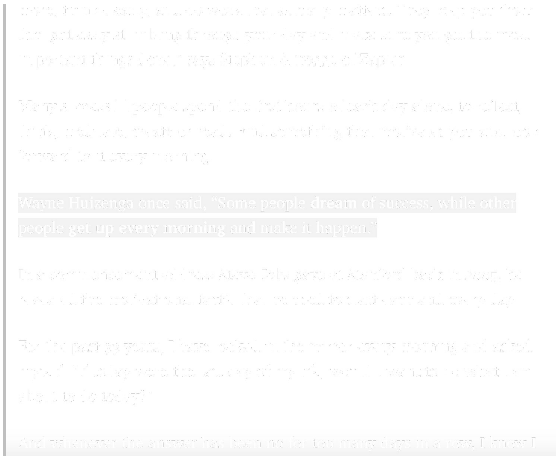
Many successful people spend the first hours of each day alone, to reflect, think, meditate, create or read. Find something that motivates you and look forward to it every morning.

Wayne Huizenga once said, "Some people **dream** of success, while other people **get up every morning** and make it happen."

In a commencement address Steve Jobs gave at Stanford back in 2005, he revealed the motivational tactic that he used to start each and every day.

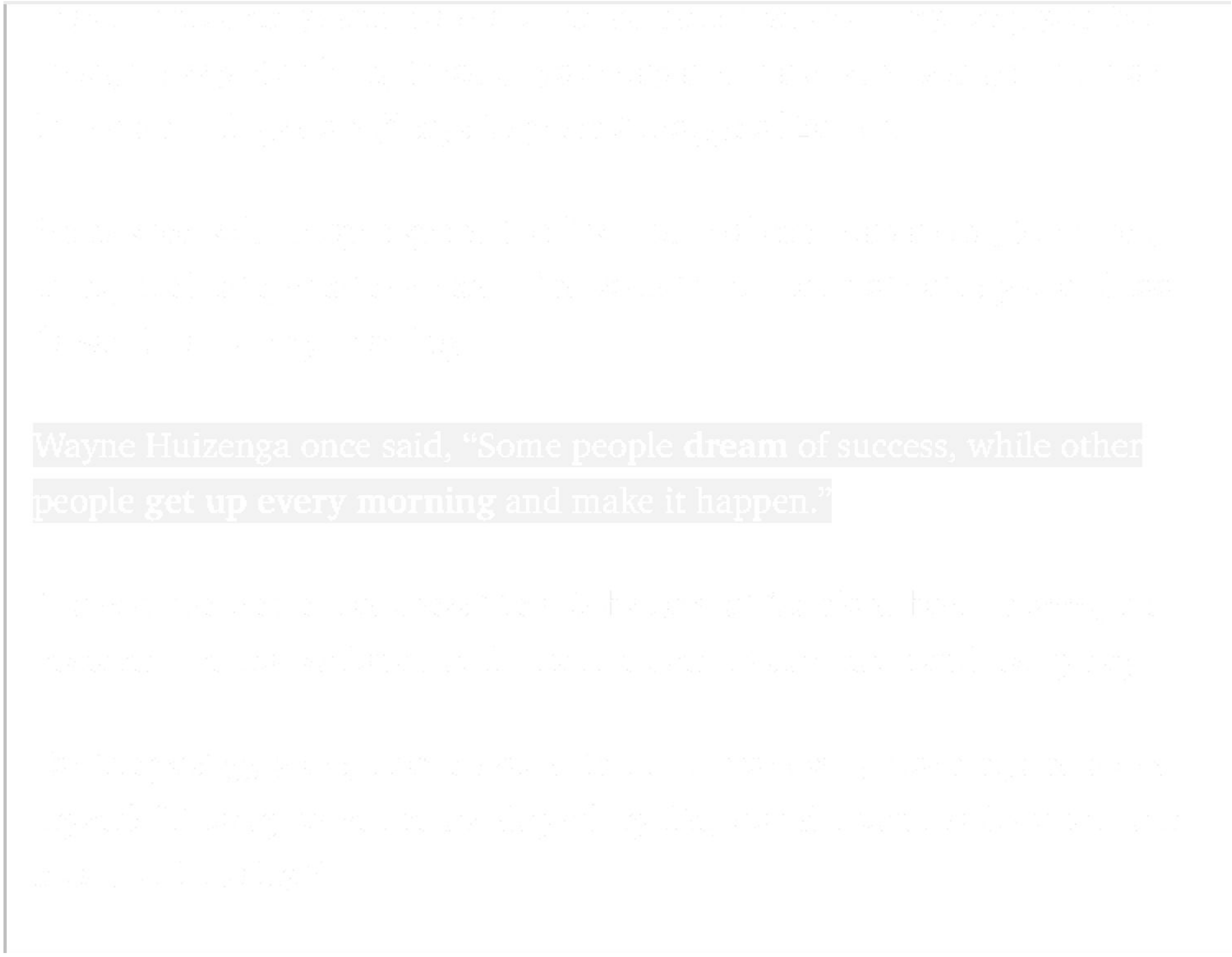
For the past 33 years, I have looked in the mirror every morning and asked myself: "If today were the last day of my life, would I want to do what I am about to do today?"

And whenever the answer has been no for too many days in a row, I know I



Wayne Huizenga once said, "Some people dream of success, while other people get up every morning and make it happen."

Time taken for program execution: 2.102708 seconds, these images represent the preprocessing Highlight extraction stages. Notice the presence of noise in the image.



Wayne Huizenga once said, "Some people **dream** of success, while other people **get up every morning** and make it happen."

Same image passed through the main processing algorithm, this image will now be passed to an OCR API

Future Development

The application serves a need, and is highly extensible. The application will be made installable on the various mobile platforms' stores.
With the text data and user input, one can build a comprehensive dataset that will presumably be valuable to data scientists and machine learning researchers in the future. The application can serve a need, and achieve similar goals as the reCaptcha system did and continues to do with digitising books and manuscripts. Data that will be collected is by no means sensitive, but nevertheless, users will be informed and will be able to opt-out. Evolutionary algorithms in the context of question generation will be explored in further depth, along with possible integrations of a TensorFlow based system for question generation to facilitate the proposed Spaced Repetition System.

Acknowledgements

I would like to thank Alexander Silgidjian from SAP Labs for his valuable help and insight. I am grateful to Todor Kolev for help with the brainstorming and suggesting optimal methods of going ahead with the application with his vast repertoire of knowledge relating to technology. Lastly, I would also like to thank Konstantin Delchev for his help and support throughout this project, and the Summer Research School for facilitating the research.

References

Abadi, Martín, et al. "Tensorflow: Large-scale machine learning on heterogeneous distributed systems." *arXiv preprint arXiv:1603.04467* (2016).

Fowler, Robert L., and Anne S. Barker. "Effectiveness of highlighting for retention of text material." *Journal of Applied Psychology* 59.3 (1974): 358.

John D. Hunter. **Matplotlib: A 2D Graphics Environment**, Computing in Science & Engineering, **9**, 90-95 (2007),DOI:10.1109/MCSE.2007.55

Jones E, Oliphant E, Peterson P, *et al.* **SciPy: Open Source Scientific Tools for Python**, 2001-, http://www.scipy.org/ [Online; accessed 2017-08-15].

Kim, Mijung. *The effects of external cues on media habit and use: Push notification alerts and mobile application usage habits*. Michigan State University, 2014.

Kim, Seul-Kee, So-Yeong Kim, and Hang-Bong Kang. "An Analysis of the Effects of Smartphone Push Notifications on Task Performance with regard to Smartphone Overuse Using ERP." *Computational intelligence and neuroscience* 2016 (2016).

Librenjak, Sara, Marijana Janjić, and Kristina Kocijan. "Sustainable vocabulary acquisition in Japanese classroom with the help of Memrise." *International Academic Conference on Global Education, Teaching and Learning in Budapest 2016*. 2016.

Loria, Steven. "TextBlob."

"Python Anywhere." *Python Anywhere*, Python Anywhere, PythonAnywhere.com.

Gupta, James. "Spaced Repetition: a Hack to Make Your Brain Store Information." *The Guardian*, Guardian News and Media, 23 Jan. 2016, www.theguardian.com/education/2016/jan/23/spaced-repetition-a-hack-to-make-your-brain-store-information.

"Quizlet." *Quizlet*, quizlet.com/.

Rosebrock, Adrian. "Creating a Face Detection API with Python and OpenCV ."*PyImageSearch*, 7 June 2015.

Rubens, Paul. "Technology | Spam Weapon Helps Preserve Books." *BBC News*, BBC, 2 Oct. 2007, news.bbc.co.uk/2/hi/technology/7023627.stm.

Rubinstein, Dana, and Hilary Greenbaum. "The Hand-Held Highlighter." *The New York Times*, The New York Times, 21 Jan. 2012, www.nytimes.com/2012/01/22/magazine/the-hand-held-highlighter.html.

Stéfan van der Walt, S. Chris Colbert and Gaël Varoquaux. **The NumPy Array: A Structure for Efficient Numerical Computation**, Computing in Science & Engineering, **13**, 22-30 (2011), DOI:10.1109/MCSE.2011.37

Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu and the scikit-image contributors. **scikit-image: Image processing in Python**, PeerJ 2:e453 (2014)

Sutherland, Andrew. "Quizlet." (2014).

Turney, Peter D. "Learning to extract keyphrases from text." *arXiv preprint cs/0212013* (2002).