How far has Optical Character Recognition Softwares have come.*

Evaulation of OCR software

Sampson Zhao

27 April 2022

Abstract

Optical Character Recognition (OCR) has been used since the 1930s as a way to recognize standardized text from a photo. This technology has greatly developed throughout the ages, and is now widely used in data entry from printed data, like passports, invoices, etc. Though the earlier versions of OCR required the text to have a set font for recognition, modern iterations of this has developed enough to recognize multiple fonts as well as different writing systems. This project focuses on the accuracy of modern OCR systems, like blank and blank, and will compare the accuracy of the produced text to the originals. (conclusion) Though the softwares...

Keywords: OCR, Tesseract, Magick, Accuracy, Historical Texts, Improvements in OCR, AI related to OCR.

Contents

*Code and data are available at: https://github.com/Sampsonzhao/OCR-final.

1 Introduction

OCR was first developed as a technology in the 1910s, as it was primarily used in tandem with reading devices for the blind "The History of OCR, Optical Character Recognition : Schantz, Herbert f : Free Download, Borrow, and Streaming" (1982), with the main premise of OCR being using some optical device to capture characters on a document, and transform it into a manner where computer software can then recognize the text. Though earlier iterations of OCR were used in the development of text documents to audio readers for the blind or visually impaired, the technology is now widely used as a shortcut in data entry, as documents like passports, cheques, and other printed documentations, can be quickly and accurately transformed, effectively digitizing the document to allow for editing and searching electronically. Though earlier iterations of the software required documents to be in particular fonts for the software to recognize the text, more modern and advanced systems have developed significantly, to the point where these softwares are able to recognize texts in a variety of fonts within the same documen, as well as recognizing different writing systems, like Cyrillic, Arabic, etc. (List of most of the available languages that are recognized by Google's OCR software here).

OCR development has been progressing fairly well, as the accuracy of the software from 19th century to the 20th century has seen an increase of 81% accuracy to 99% accuracy. Though this is comparing the developments of the technologies that these softwares are based off of, there was a fundamental shift in the style of recognition. OCR developed from a character-to-character recognition style to a more pattern recognition style, signifying the advancements of computational power, as effectively they changed from a character-to-character recognition to more of a line-by-line recognition. Though the recognition of text improved drastically back then, there were multiple variables that caused the 1% of texts to fail, and that is related to the pre-processing of the image. Factors like wrinkles, slanting, normalization of ratio and scales, were all factors that attributed to failed OCR reads, these were considered artifacts that the software could not recognize, effectively showing garbage code as a result. Though these factors have improved since the early 2000s, OCR is still continuing on the development of optimizing the overall recognition software.

Ultimately, this paper looks into the development of OCR software that are open access today, effectively looking at the accuracy of the text that is produced by the OCR software, when compared to the original text. We are able to use this information to track the accuracy of these software, which would be useful in digitizing older texts. Here, we would look at the difference in processed and unprocessed images of text documents from different titles and compare the accuracy of OCR, as well as testing the limits of OCR. The continued development of OCR aims to do this as one of it's main applications, as it would allow for further digitization of older texts, allowing more of this information to be share on the web. As many older texts have only photo renditions of the book online, OCR can be used to scan these pictures and create a searchable and interactable version of the text.

While many old texts show faded or blurred text, due to the age of the documents themselves, this posed an issue for both old and newer versions of OCR, as the pre-processing of the image would result in the software returning errors alongside the text. There are tweaks that are currently in development, as development of application-specific OCR has come up as one of the main projects developed by government entities, like recognizing license plate numbers or invoice specifics. These developments allow for the specialization of the software, which trims down on the processing time needed for the software to recognize these texts. The hope is that further development of application-specific OCR and effectively transition to further optimizing the recognition of older texts, allowing for the continued digitization of historical texts.

2 Data

2.1 Acknowledgement

This investigation uses the 'R' programming language (R Core Team 2020), and the 'R-Studio' Integrated development environment (RStudio Team 2020) as the primary source of data processing and the generation of the paper. The 'Tidyverse' package was used for manipulation of the datasets (Wickham et al. 2019). The

'Here' package was used to locate files produced from R scripts (Müller 2020).'KableExtra' package was used to enhance the overall appearance of tables(Zhu 2021). 'knitr' was used to tie all of the information on this document into one digestible PDF (Xie 2021)(Xie 2015)(Xie 2014). 'ggplot2' was used to create the bar plots in this document (Wickham 2016). 'reshape2' was used to modify data sets to better create the wanted plots, while simplifying the code needed for those graphs (Wickham 2007). 'Tesseract,' 'magick,' and 'magrittr' was used to implement the OCR recognition of the images, as well as slight modifications of the image for better recognition(Ooms 2022)(Ooms 2021)(Bache and Wickham 2020). 'diffr' was used to generate plots that compared the text files produced by the OCR from the previously mentioned packages and provide a visual comparison of two text documents(Muschelli 2017). 'RecordLinkage' and 'fuzzywuzzyR' were both used to generate a percentage difference between two text documents using different algorithms(Sariyar and Borg 2022)(Mouselimis 2021)(Inc 2014). 'reticulate' was used to install the correct python environment for package 'fuzzywuzzyR' to function properly(Ushey, Allaire, and Tang 2022).

2.2 Where this data comes from

While there is no direct rational behind the use of these particular title for this investigation, the overall goal of this investigation was to look at texts that were on different parts of the spectrum. Two books were used to represent the opposite sides of the spectrum, as well as a standard. In this investigation, the following books were used as sources for samples for the OCR:

- "Basic and Clinical Pharmacology 14th Edition" by Bertram E. Katzung
- "Life and Literature Book One Grade 7" by The Ministry of Education For Ontario
- "The Life and Work of Fredson Bowers" by G. Thomas Tanselle

Katzung represents more recent texts that have clean and non-intrusive text, alongside the fact that a digital copy of this book is available to retrieve the original text for comparison(Katzung and Katzung 2018). Tanselle is commonly used as a standard for testing OCR, as it was seen to be used in multiple sources that talked about the uses of OCR and tesseract (Tanselle and Battestin 1993). Life and Literature then represents the other end of the spectrum that was described in the introduction, looking at texts that may have faded texts due to aging as well as wear-and-tear, causing the text to become illegiable in some cases. Overall, the goal was met, where we have samples from different parts of texts that are still in circulation in some public libraries.

2.3 Data Selection and Concern

In this report, due to the sheer size of the books that were chosen as the samples for OCR (The Pharmacology Textbook has over 1200 pages in it, where as Life and Literature has over 300), 5 pages from each book were selected to represent a portion of the book. To remove any related bias, a random number generator was used to select page numbers for each book (except the standard, as that uses page 1 of the book.) The use of a random number generator makes sure that the page numbers selected in succession have no common variable between them, removing any influence from the writer, and thus removing a source of bias (Haahr 2019). For the case of the Pharmacology Textbook, because the pages have text split in half and to decrease the amount of artifacts that the OCR scans for, half of the page would also be selected. This would be done in the same manner as the random number generator used for the page selection of the books, where 1 would represent the left half of the page, and 2 would represent the right half of the page. Hence for this investigation, the following were chosen for OCR processing:

- "Basic and Clinical Pharmacology 14th Edition" by Bertram E. Katzung
  - Page 100, left half of page
  - Page 259, left half of page
  - Page 493, right half of page

  - Page 613, right half of page
  - Page 1099, right half of page
- "Life and Literature Book One Grade 7" by The Ministry of Education For Ontario
  - Page 65
  - Page 127
  - Page 183
  - Page 235
  - Page 318

For this investigation, the selected pages have been digitized via photo taken from an iPhone X. (These photos can be found in the /input/data folder of the github repository. These photos would then be cropped to limit the amount of white spaces and artifacts that would be introduced into the produced OCR text file. Three levels of processing will be tested:

- Raw/Unprocessed
  - This level would have no digital processing applied to the photo before running the photo through OCR. Cropping will be done on these photos, as it was found that if no cropping was done, the percentage of recognizable text would be very near 0%.
- Some Processing/Cleaning
  - This level will have some digital processing from the built-in image processing methods of Magick. This will be limited to converting the photo to gray-scale to limit the amount of artifacts picked up by OCR, enlarging the photo, increased brightness and contrast to attempt to remove shadows, as well as trimming additional white space.
- Advanced Processing/Cleaning
  - This level contains a larger amount of digital processing, as the photos were put through a 3rd-party software (The software used in this investigation was 'Scanner Pro' by Readdle Technologies Limited). The software is able to produce scanner-like images based off of the photos that are uploaded to the software, and is able to correct warping of the page, get rid of any shadows that are in the raw photo, as well as boosting the brightness and contrast to ensure that almost all text is clear and visible. While this is no different from physically scanning the pages through a printer/scanner, these were not available at the time of this investigation due to a malfunction.

2.4 Understanding the Data

3 Discussion

3.1 Problems with OCR

While OCR is a very convenient piece of software that allows us to quickly translate information from a photo into an interactable piece of digital text, there are still many flaws associated to the software itself. For example, while dealing with the images for Book 1, there was a issue related to the overall format of the text. OCR is not able to differentiate between full-page texts and split page text, hence it does not recognize the difference between them. This would then cause errors for the right half of the text, as the spacing between the two bodies of text throws off the OCR recognition, causing the corruption of the data on the right hand side. From this investigation, it was observed that the spacing between each character, as well as the spacing between bodies of text affect how Tesseract views the text itself. While this wouldn't be a big problem in normal circumstances, it would be an issue while looking into the further goals of OCR. The lack of flexibility in OCR, as well as the lack of understanding in sentence structures are one of the

downfalls of the software. Though these are easily ameliorated in the form of restricting recognition of text, as well as restricting white spaces, this may become an issue for how OCR can be used in text recovery and text recognition in older texts.

Based on a similar issue as mentioned above, we can also look into the artifacts that are introduced due to how the original text is captured. Though in Section 2.2, it was mentioned that all the photos have some level of cropping done to it. For Book 1 (Basic and Clinical Pharmacology), texts were limited to only half of the page, as there was no continuity in the sentence, as well as OCR not recognizing the white space used to separate the two columns of text. Though that played a large factor in generating large amounts of garbled data in the exported text file, artifacts on the edge of the photo, like the area marked in red in Figure 1, have been found to also cause some errors, as any sort of lines would be recognized as text by OCR. Hence, in the case of all of the photos that were taken in a similar manner, all of the edges were removed to make sure that there would not be false hits from the OCR scan.

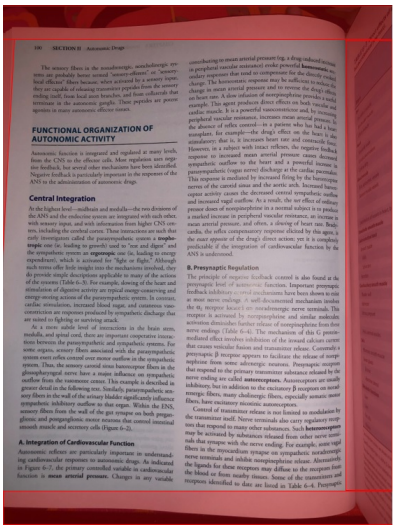

Figure 1: Edge of photo is recognized by Tesseract as text

In addition to cropping photos, there were various image manipulations done to further increase the accuracy of the OCR scans. If we look back at Figure ___ (Refer to chart generated from 02-percent_diff), we can see how important being able to clean and remove any artifacts from the photo, as we tend to see an positive correlation between cleaning and percent recognition. This brings into light the different photo manipulations needed, and how that might take away from the original text. In the case of comparing the raw photo to the semi-processed photo, we can see that even just removing colour and increasing the brightness and contrast, would generally be enough for the OCR software to recognize the text better. To be more accurate, the hypothesis for this phenomenon is more related to how the OCR software is unable to see flaws and artifacts if the brightness and contrast is increased. By modifying the photo slightly, the software is able to discern better the difference between text and random debris, allowing for increased accuracy in the text output. This is significant, due to the fact that not every photo is as clean as a scanner, as photos generally have large amounts of scrap data in the periphery, causing the OCR to incorrectly identify the characters, sometimes causing OCR scans to show up as errors.

The same could be said when comparing both the raw photos and the semi-processed photos to the photos that underwent advanced image processing, as a clear jump in accuracy was seen, when looking at Figure___ (Refer to chart generated from 02-percent_diff). This section of photos are most reminiscent to

3.2 Progress with OCR

In general, if this experiment was repeated, a suggestion can be made to increase the accuracy overall, by scanning the pages of texts, as well as preventing large shadows from being shown, as both of these factors have affected the overall percentage difference between the 3 levels of image processing. It is understandable that scanning certain books may not be possible, as it may incur damage to the physical item itself. As the premise of this paper is looking at how OCR can be used in the digitization of older materials. For example Book 2 (Life and Literature), this is an item from 1943. The age of the item as well as the physical condition of the item was considered to be fairly poor and would not have faired well if the spine was overextended. Scanning of many older literature would not be feasible, hence photos would be the next best alternative. While third-party softwares were available for this investigation, there have been progress in the overall OCR recognition, as there were mentions of OCR being able to discern text from uneven surfaces. While it was not apparent during this investigation, it is something that is progressing in the field of OCR, and would become a very powerful tool for those who wish to continue the digitization of many older texts. If used properly, image processing softwares could have OCR built into the overall application, allowing for less processing on the OCR side.

3.3 Further Development of OCR alongside the development of A.I.

4 Conclusion

5 Appendix

References

Bache, Stefan Milton, and Hadley Wickham. 2020. *Magrittr: A Forward-Pipe Operator for r.* https://CRAN.R-project.org/package=magrittr.

Haahr, Mads. 2019. "RANDOM.ORG - True Random Number Service." *Random.org.* https://www.random.org/.

Inc, SeatGeek. 2014. *fuzzywuzzy: Fuzzy String Matching in Python.* https://github.com/seatgeek/fuzzywuzzy.

Katzung, Betram G., and Bertram G. Katzung. 2018. *Basic &Amp; Clinical Pharmacology / by Betram g. Katzung.* McGraw-Hill.

Mouselimis, Lampros. 2021. *fuzzywuzzyR: Fuzzy String Matching.* https://CRAN.R-project.org/package=fuzzywuzzyR.

Müller, Kirill. 2020. *Here: A Simpler Way to Find Your Files.* https://CRAN.R-project.org/package=here.

Muschelli, John. 2017. *Diffr: Display Differences Between Two Files Using Codediff Library.* https://CRAN.R-project.org/package=diffr.

Ooms, Jeroen. 2021. *Magick: Advanced Graphics and Image-Processing in r.* https://CRAN.R-project.org/package=magick.

———. 2022. *Tesseract: Open Source OCR Engine.* https://CRAN.R-project.org/package=tesseract.

R Core Team. 2020. *R: A Language and Environment for Statistical Computing.* Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

RStudio Team. 2020. *RStudio: Integrated Development Environment for r.* Boston, MA: RStudio, PBC. http://www.rstudio.com/.

Sariyar, Murat, and Andreas Borg. 2022. *RecordLinkage: Record Linkage Functions for Linking and Deduplicating Data Sets.* https://CRAN.R-project.org/package=RecordLinkage.

Tanselle, G. Thomas, and Martin C. Battestin. 1993. *The Life and Work of Fredson Bowers.* Bibliographical Society of the University of Virginia.

"The History of OCR, Optical Character Recognition : Schantz, Herbert f : Free Download, Borrow, and Streaming." 1982. *Internet Archive.* [Manchester Center, Vt.] : Recognition Technologies Users

Association. https://archive.org/details/historyofocropti0000scha/page/n5/mode/2up.

Ushey, Kevin, JJ Allaire, and Yuan Tang. 2022. *Reticulate: Interface to 'Python'*. https://CRAN.R-project.org/package=reticulate.

Wickham, Hadley. 2007. "Reshaping Data with the reshape Package." *Journal of Statistical Software* 21 (12): 1–20. http://www.jstatsoft.org/v21/i12/.

———. 2016. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. https://ggplot2.tidyverse.org.

Wickham, Hadley, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, et al. 2019. "Welcome to the tidyverse." *Journal of Open Source Software* 4 (43): 1686. https://doi.org/10.21105/joss.01686.

Xie, Yihui. 2014. "Knitr: A Comprehensive Tool for Reproducible Research in R." In *Implementing Reproducible Computational Research*, edited by Victoria Stodden, Friedrich Leisch, and Roger D. Peng. Chapman; Hall/CRC. http://www.crcpress.com/product/isbn/9781466561595.

———. 2015. *Dynamic Documents with R and Knitr*. 2nd ed. Boca Raton, Florida: Chapman; Hall/CRC. https://yihui.org/knitr/.

———. 2021. *Knitr: A General-Purpose Package for Dynamic Report Generation in r*. https://yihui.org/knitr/.

Zhu, Hao. 2021. *kableExtra: Construct Complex Table with 'Kable' and Pipe Syntax*. https://CRAN.R-project.org/package=kableExtra.