WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Tesseract OCR Engine - A Summary

S.Sabariraj[1], Dr.S.Thilagavathi[2]

22bscs247sabarirajs@skacas.ac.in,thilagavathis@skacas.ac.in

Student[1], Department of Computer Science, Sri Krishna Adithya College of Arts and Science, Kovaipudur, Coimbatore.

Faculty[2], Assistant Professor, Department of Computer Science, Sri Krishna Adithya College of Arts and Science, Kovaipudur, Coimbatore.

ABSTRACT :

It's a long-held desire to replicate human abilities in a machine, such as reading. Over the course of three decades, there has been a great deal of study focused on optical character recognition, which is the mechanical imitation of human reading. It is the mechanical or computer conversion of scanned images containing printed, typewritten, or handwritten te t. Tesseract, an open-source optical character recognition program developed by Google, is presented in this paper. We will provide an overview of the algorithms utilized at each level of the Tesseract pipeline. We will pay close attention to the features that set Tesseract apart from other OCR systems, whether they are new or just diIerent.

Intíoduction – HISTORY :

The mechanical or electronic conversion of pictures of printed, handwritten, or typed te t into machine-encoded te t is known as optical character recognition. This is the simplest way to digitize handwritten and printed materials so that They are more easier to search, store more compactly, display and modify online, and can be utilized for a variety of further processing jobs including te t mining and language translation.

Tesseract is an open-source OCR engine that was developed at HP between 1984 and 1994 .At HP Labs in Bristol, Tesseract started oI as a PhD research project.

and gained traction as a potential software and/or hardware add-on for HP's range of scanners with flatbeds. The commercial OCR engines available at the time were in their infancy and performed appallingly on all but the highest quality prints, which served as motivation.

Tesseract had a considerable accuracy advantage over the commercial engines following a collaborative project between HP Labs in Bristol and HP's scanner division in Colorado, but it was never released as a product. Its ne t round of research was conducted by HP Labs in Bristol as an OCR for compression study. Improvements in rejection e ciency were prioritized over base-level accuracy in the work. Development came to a halt completely by the end of 1994, marking the end of this project. After being sent to UNLV for the 1995 Annual Test of OCR Accuracy[1], the engine demonstrated its superiority over competing commercial engines. HP made Tesseract available as open source at the end of 2005. It can be found at http://code.google.com/p/tesseract-ocr as of right now.

Aíchitectuíe :

Tesseract never required its own page layout analysis because HP already had independently developed page layout analysis technology that was used in products (and hence not made available for open-source). Because of this, Tesseract believes that its input is a binary image with defined polygonal te t sections that may be optional Although processing is done in a conventional, step-by-step manner, there are a few odd steps that may still be in place today. A linked component analysis is the first step, when component outlines are saved.

Using Otsu's approach, the image is first transformed into a binary version by Adaptive Thresholding [3]. Page layout analysis, which is the following stage, is utilized to e tract the te t blocks within the document, as seen in fig.2

The tt is separated into words using definite and fuzzy spaces in the following stage, which also detects the baselines of each line.

The character outlines are taken out of the te t in the following stage. The process of te t recognition then begins with two passes. The static classifier is used for word recognition in the first phase. An adaptive classifier receives every word that is delivered with satisfaction as training data [4]. The page is scanned again with the recently acquired adaptive classifier.

In the last stage, fuzzy spaces are resolved and diIerent hypotheses for the -height are checked in order to locate smallcap te t.

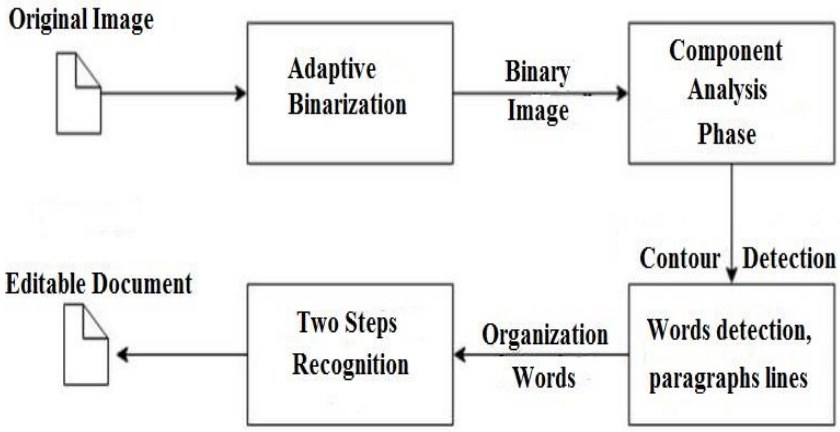


Figure 1: Architecture of Tesseract OCR engine

PAGE LAYOUT :

One of the initial stages of OCR is page layout analysis, which separates te t from non-te t portions of an image and breaks multi-column te t into columns. The Tesseract page layout analysis relies on the concept of tab-stop detection in a document picture. The steps are as follows:

- The vertical lines and the photos are detected by Leptonica's morphological processing. Before the cleaned image is sent to linked component analysis, these components are eliminated from the input image.
- After being located, the potential tab-stop related components that appear to be near a te t region's edge are organized into tab-stop lines.
- Runs of similarly categorized connected components are grouped into Column Partitions (CP) by scanning the connected components from left to right and top to bottom, with the restriction that no CP may cross a tab stop line.
- Te t blocks are composed of chains of te t CPs that are further subdivided into groups with consistent linespacing. A candidate region is now represented by each chain of CPs. Several heuristic principles dictate the areas' reading sequence.

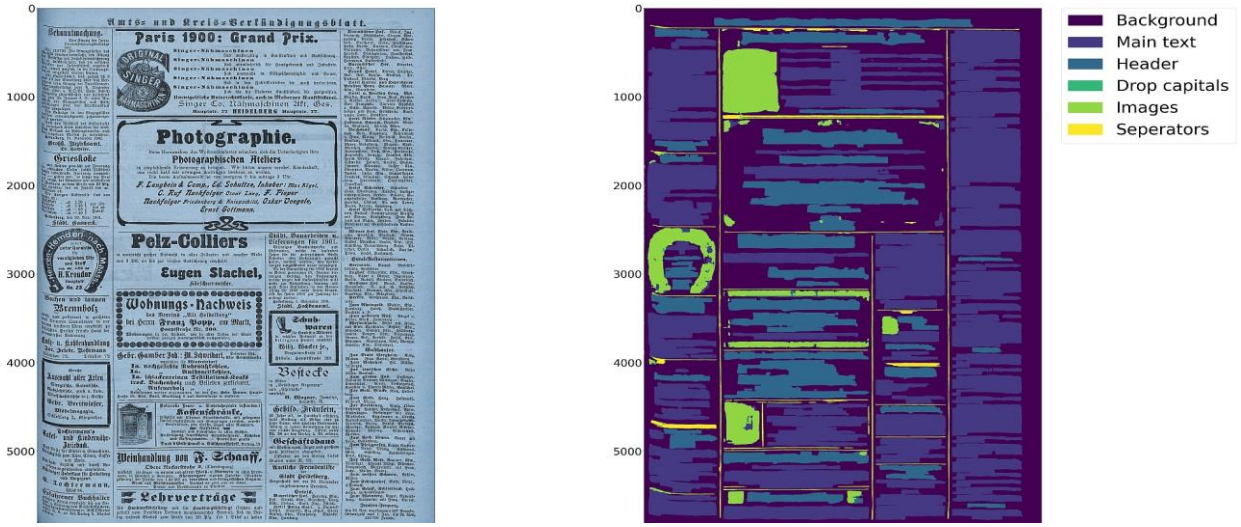


Figure 2: Page Layout Analysis

CHARACTER CLASSIFICATION :

STATIC CLASSIFICATION :

The elements of a polygonal appro imation of a shape's contour are the features that are utilized for classification. Each element of the polygonal appro imation is used to create a 4-dimensional feature vector (x , y) position, direction, and length) during training. These feature vectors are then grouped to create archetypal ones, hence the term Tesseract.

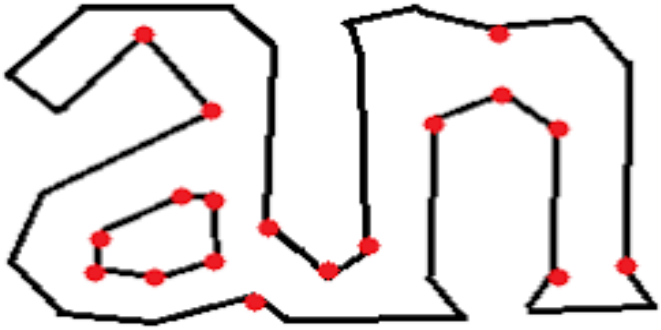
The polygon's constituents are divided into shorter, equal-length segments during recognition, removing the length dimension from the feature vector. The approach of matching minor features to large prototypes can handle degraded image recognition with ease. The primary issue is that calculating the distance between an unknown and a prototype requires a significant amount of computing power.

Adaptive Classification:

In documents with a restricted number of typefaces, a more font-sensitive adaptive classifier is employed, trained by the static classifier's output, to achieve higher discrimination. For adaptive classification, Tesseract employs the same features and classifier as the static classifier.
The adaptive classifier can only contribute considerably less toward the top of the page if it is implemented during the first run because it learns during that run. Consequently, the page is run through a second time, this time recognizing terms that the first pass did not catch efficiently.
The baseline/ -height normalization increases resilience to noise specks and facilitates the differentiation of upper and lower case characters. The elimination of font aspect ratio and a certain amount of font stroke width is the primary advantage of character moment normalization. Additionally, it simplifies the recognition of subscripts and superscripts; nonetheless, certain upper- and lower-case letters still need the use of an e tra classifier feature.

Woíd Recognition :

Any character recognition engine will recognize a word and determine how to divide it up into its component characters. First, the line finding segmentation output is classified. The remaining steps in the word recognition process are limited to non-fi ed pitch te t.



Linguistic Analysis :

Relatively little linguistic analysis is present in Tesseract. The best word string in each of the following categories is selected by the linguistic module (also known as the permuter) whenever the word recognition module is thinking about a new segmentation. most often used word, top dictionary word highest digit, Top classifier choice word, Top UPPER case word, and Top lower case word (with optional starting upper).
DiIerent segmentations of words may contain varying numbers of characters. Even in cases when a classifier is used, it is di cult to compare these terms directly.
Tesseract makes assertions about generating probabilities, but it doesn't. Tesseract solves this issue by producing two integers for every character classification. The normalized distance from the prototype is subtracted from the first, known as the confidence. Because of this, it can be considered a "confidence" in the sense that higher numbers are preferable, but it is still a distance because the farther anything is from zero, the greater the distance. The normalized distance from the prototype multiplied by the entire outline length of the unknown character yields the second result, known as the rating. Character ratings within a word can be summarisedin a meaningful way because of the overall outline length.

Tíaining Data :

The classifier was not trained on damaged characters because it can identify damaged characters with ease. Actually, just 20 samples totaling 94 characters from 8 fonts in a single size and 4 attributes (normal, bold, italic, and bold italic) were used to train the classifier, for a total of 60160 training samples.
This stands in stark contrast to other reported classifiers, like Baird's 100-font classifier with 1175000 training e amples and the Calera classifier with almost a million samples.

Findings :

Tesseract was used as "HP Labs OCR" in the 4th UNLV annual test [1] of OCR accuracy, but since then, a lot of changes have been made to the code, including retraining and a conversion to Unicode. findings from a current Tesseract version (shown as 2.0) are compared with the initial 1995 findings (represented as HP) in Table 1. Along with the number of mistakes (Errs), the percent error rate (%Err), and the percent change from the 1995 results (%Chg) for both character errors and non-stopword errors, all four of the 300 DPI binary test sets used in the 1995 test are displayed. [1] The most recent findings can be seen at http://code.google.com/p/tesseract-ocr.

Demonstíation :

The Tesseract engine receives an image with te t (fig.4) as input

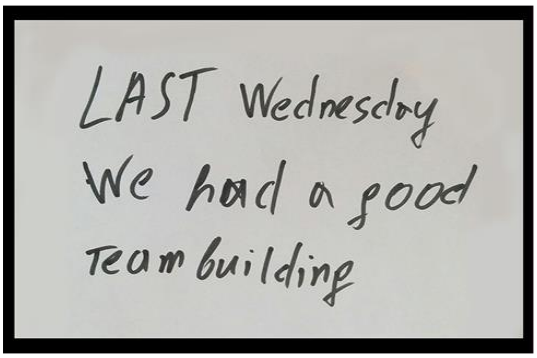


Figure 4 : sample image

The Tesseract command requires two inputs: thename of the te t-containing picture file and the output te t file containing the e tracted te t (fig.5).

X

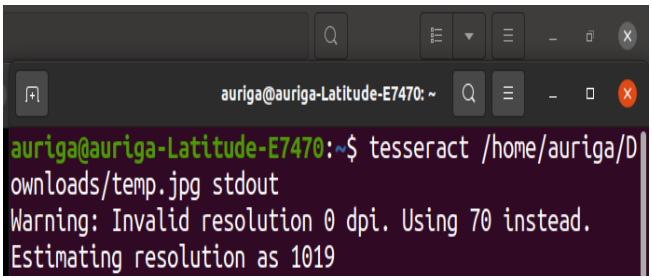


Figure 5 : Employing Terminal to run the Tesseract engine

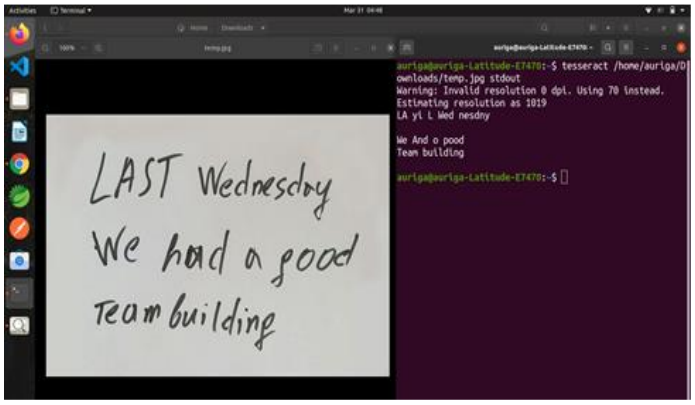


Figure 6 : Output

Final Thoughts and Upcoming Projects :

In terms of accuracy, Tesseract is currently lagging behind the top commercial engines, having lain dormant for almost a decade. Its peculiar feature selection is arguably its greatest asset. Its primary flaw is presumably that the classifier is fed a polygonal approximation rather than the raw contours. After internationalization, accuracy could likely be much increased by carefully incorporating a character n-gram model based on a Hidden-Markov-Model and perhaps even improvingthe chopper.

Recognitions

The ISRI group at UNLV for contributing their tools and data, John Burns and Tom Nartzer for working to make Tesseract open source, and Luc Vincent, Igor Krivokon, Dar-Shyang Lee, and Thomas Kielbus for their feedback on the paper's content are all appreciated.

REFERENCES :

1. Leptonica image processing and analysis library.http://www. leptonica.com.
2. Tesseract ocr, github:. https://github.com/tesseract-ocr.
3. Nobuyuki Otsu. A threshold selection method from gray-level histograms. Automatica, 11(285-296):23–27, 1975.
4. P.J. Rousseeuw, A.M. Leroy, RobustRegression and Outlier Detection, Wiley-IEEE, 2003.
5. S.V. Rice, G. Nagy, T.A. Nartker, Optical Character Recognition: An Illustrated Guide to the Frontier, Kluwer Academic Publishers, USA 1999, pp. 57-60.
6. Nobuyuki Otsu. A threshold selection method from gray-level histograms. Automatica, 11(285-296):23–27, 1975.
7. R Smith. An overview of the tesseract ocr engine. in proceedings of document analysis and recognition.. icdar 2007. In IEEE Ninth International Conference, 2007.
8. Ray Smith. A simple and efficient skew detection algorithm via text row accumulation. In Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on, volume 2, pages 1145–1148. IEEE, 1995
9. Ray Smith, Daria Antonova, and Dar-Shyang Lee. Adapting the tesseract open source ocr engine for multilingual ocr. In Proceedings of the International Workshop on Multilingual OCR, page 1. ACM, 2009.
10. Raymond W Smith. Hybrid page layout analysis via tab-stop detection. In 2009 10th International Conference on Document Analysis and Recognition, pages 241–245. IEEE, 2009.
11. Wikipedia. Optical character recognition. https://en.wikipedia.org/ wiki/Optical_character_recognition.