Journal of Computational Social Science (2022) 5:861–882
https://doi.org/10.1007/s42001-021-00149-1

RESEARCH ARTICLE

OCR with Tesseract, Amazon Textract, and Google Document AI: a benchmarking experiment

Thomas Hegghammer[1]

Received: 23 June 2021 / Accepted: 6 October 2021 / Published online: 22 November 2021

Abstract

Optical Character Recognition (OCR) can open up understudied historical documents to computational analysis, but the accuracy of OCR software varies. This article reports a benchmarking experiment comparing the performance of Tesseract, Amazon Textract, and Google Document AI on images of English and Arabic text. English-language book scans ($n = 322$) and Arabic-language article scans ($n = 100$) were replicated 43 times with different types of artificial noise for a corpus of 18,568 documents, generating 51,304 process requests. Document AI delivered the best results, and the server-based processors (Textract and Document AI) performed substantially better than Tesseract, especially on noisy documents. Accuracy for English was considerably higher than for Arabic. Specifying the relative performance of three leading OCR products and the differential effects of commonly found noise types can help scholars identify better OCR solutions for their research needs. The test materials have been preserved in the openly available "Noisy OCR Dataset" (NOD) for reuse in future benchmarking studies.

Keywords OCR · Cloud computing · Benchmarking

I am grateful to the three anonymous reviewers and to Neil Ketchley for valuable comments. I also thank participants in the University of Oslo Political Data Science seminar on 17 June 2021 for inputs and suggestions, as well as Eddie Antonio Santos for helping solve technical questions related to the ISRI/Ocreval tool. Supplementary information and replication materials are available at https://github.com/Hegghammer/noisy-ocr-benchmark.

✉ Thomas Hegghammer
thomas.hegghammer@ffi.no

[1] Norwegian Defence Research Establishment (FFI), Kjeller, Norway

Introduction

Few technologies hold as much promise for the social sciences and humanities as optical character recognition (OCR). Automated text extraction from digital images can open up large quantities of understudied historical documents to computational analysis, potentially generating deep new insights into the human past.

But OCR is a technology still in the making, and available software provides varying levels of accuracy. The best results are usually obtained with a tailored solution involving corpus-specific pre-processing, model training, or postprocessing, but such procedures can be labour-intensive.[1] Pre-trained, general OCR processors have a much higher potential for wide adoption in the scholarly community, and hence their out-of-the box performance is of scientific interest.

For long, general OCR processors such as Tesseract ([27, 38]) only delivered perfect results under what we may call laboratory conditions, i.e., on noise-free, single-column text in a clear printed font. This limited their utility for real-life historical documents, which often contain shading, blur, shine-through, stains, skewness, complex layouts, and other things that produce OCR error. Historically, general OCR processors have also struggled with non-Western languages ([16]), rendering them less useful for the many scholars working on documents in such languages.

In the past decade, advances in machine learning have led to substantial improvements in standalone OCR processor performance. Moreover, the past 2 years have seen the arrival of server-based processors such as Amazon Textract and Google Document AI, which offer document processing via an application processing interface (API) ([43]). Media and blog coverage indicate that these processors deliver strong out-of-the-box performance[2], but those tests usually involve a small number of documents. Academic benchmarking studies exist ([37, 41]) but the predate the server-based processors.

To find out, I conducted a benchmarking experiment comparing the performance of Tesseract, Textract, and Document AI on English and Arabic page scans. The objective was to generate statistically meaningful measurements of the accuracy of a selection of general OCR processors on document types commonly encountered in social scientific and humanities research.

The exercise yielded specifications for the relative performance of three leading OCR products as well as the differential effects of commonly found noise types. The

[1] For pre-processing see, e.g, [3, 7, 13, 19, 42], and [44]. For model training, see, e.g., [4, 29, 33], and [45]. For postprocessing, see, e.g., [17, 35], and [39].

[2] See, for example, Ted Han and Amanda Hickman, "Our Search for the Best OCR Tool, and What We Found," *OpenNews*, February 19, 2019 (https://source.opennews.org/articles/so-many-ocr-options/); Fabian Gringel, "Comparison of OCR tools: how to choose the best tool for your project," *Medium.com*, January 20, 2020 (https://medium.com/dida-machine-learning/comparison-of-ocr-tools-how-to-choose-the-best-tool-for-your-project-bd21fb9dce6b); Manoj Kukreja, "Compare Amazon Textract with Tesseract OCR—OCR & NLP Use Case," *TowardDataScience.com*, September 17, 2020 (https://towardsdatascience.com/compare-amazon-textract-with-tesseract-ocr-ocr-nlp-use-case-43ad7cd48748); Cem Dilmegani, "Best OCR by Text Extraction Accuracy in 2021," *AIMultiple.com*, June 6, 2021 (https://research.aimultiple.com/ocr-accuracy/).

Table 1 Features of Tesseract, Textract, and Document AI

Name	Maintainer	Installation	Architecture	Languages	Cost
Tesseract	Tesseract OCR Project	Local	LSTM	116	Free
Textract	Amazon Web Services	Server-based	Undisclosed	6	$1.50 per 1000 pages
Document AI	Google Cloud Services	Server-based	Undisclosed	60+	$1.50 per 1000 pages

findings can help scholars identify better OCR solutions for their research needs. The test materials, which have been preserved in the openly available "Noisy OCR Dataset" (NOD), can be used in future research.

Design

The experiment involved taking two document collections of 322 English-language and 100 Arabic-language page scans, replicating them 43 times with different types of artificially generated noise, processing the full corpus of ~18,500 documents in each OCR engine, and measuring the accuracy against ground truth using the Information Science Research Institute (ISRI) tool.

Processors

I chose Tesseract, Textract, and Document AI on the basis of their wide use, reputation for accuracy, and availability for programmatic use. Budget constraints prevented the inclusion of additional reputable processors such as Adobe PDF Services and ABBYY Cloud OCR, but these can be tested in the future using the same procedure and test materials.[3]

A full description of these processors is beyond the scope of this article, but Table 1 summarizes their main user-related features.[4] All the processors are primarily designed for programmatic use and can be accessed in multiple programming languages, including R and Python. The main difference is that Tesseract is open source and installed locally, whereas Textract and Document are paid services accessed remotely via a REST API.

[3] As of September 2021, Adobe PDF Services charges a flat rate of $50 per 1000 pages (https://www.adobe.io/apis/documentcloud/dcsdk/pdf-pricing.html, accessed 3 September 2021). ABBYY Cloud costs between $28 and $60 per 1000 pages depending one's monthly plan and the total number of documents (see https://www.abbyy.com/cloud-ocr-sdk/licensing-and-pricing/, accessed 3 September 2021). By contrast, processing in Amazon Textract and Google Document AI costs $1.50 per 1,000 pages.

[4] For documentation, see the product websites: https://github.com/tesseract-ocr/tesseract, https://aws.amazon.com/textract/, and https://cloud.google.com/document-ai.

Data

For test data, I sought materials that would be reasonably representative of those commonly studied in the social sciences and humanities. This is to say historical documents containing extended text, as opposed to forms, receipts, and other business documents, which commercial OCR engines are primarily designed for, and which tend to get the most attention in media and blog reviews.

Since many scholars work on documents in languages other than English, I also wanted to include test materials in a non-Western language. Historically, these have been less well served by OCR engines, partly because their sometimes more ornate scripts are more difficult to process than Latin script, and partly because market incentives have led the software industry to prioritize the development of English-language OCR. I chose Arabic for three reasons: its size as a world language, its alphabetic structure (which allows accuracy measurement with the ISRI tool), and the complexity of its script. Arabic is known as one of the hardest alphabetic languages for computers to process ([14, 23]), so including it alongside English will likely provide something close to the outer performance bounds of OCR engines on alphabetic scripts. I excluded logographic scripts such as Hanzi (Chinese) and Kanji (Japanese) partly due to the difficulty of generating comparable accuracy measures and partly due to my lack of familiarity with such languages.

The English test corpus consisted of the "Old Books Dataset" ([2]), a collection of 322 colour page scans from ten books printed between 1853 and 1920 (see Fig. 1a and 1b and Table 2). The dataset comes as 300 DPI and 500 DPI TIFF image files accompanied by ground truth (drawn from the Project Gutenberg website) in TXT files. I used the 300 DPI files in the experiment.

The Arabic test materials were drawn from the "Yarmouk Arabic OCR Dataset" ([8]), a collection of 4587 Wikipedia articles printed out to paper and colour scanned to PDF (see Fig. 1c,d). The dataset contains ground truth in HTML and TXT files. Due to the homogeneity of the collection, a randomly selected subset of 100 pages was deemed sufficient for the experiment.

The Yarmouk dataset is suboptimal because it does not come from historical printed documents, but it is one of very few Arabic language datasets of some size with accompanying ground truth data. The English and Arabic test materials are thus not directly analogous, and in principle the latter poses a lighter OCR challenge than the former. Another limitation of the experiment is that the test materials only includes single-column text due to the complexities involved in measuring layout parsing accuracy.

Noise application

Social scientists and historians often deal with digitized historical documents that contain visual noise ([18, 47]). In practice, virtually any document that existed first on paper and were later digitized—which is to say almost all documents produced before around 1990 and many thereafter—is going to contain some kind of

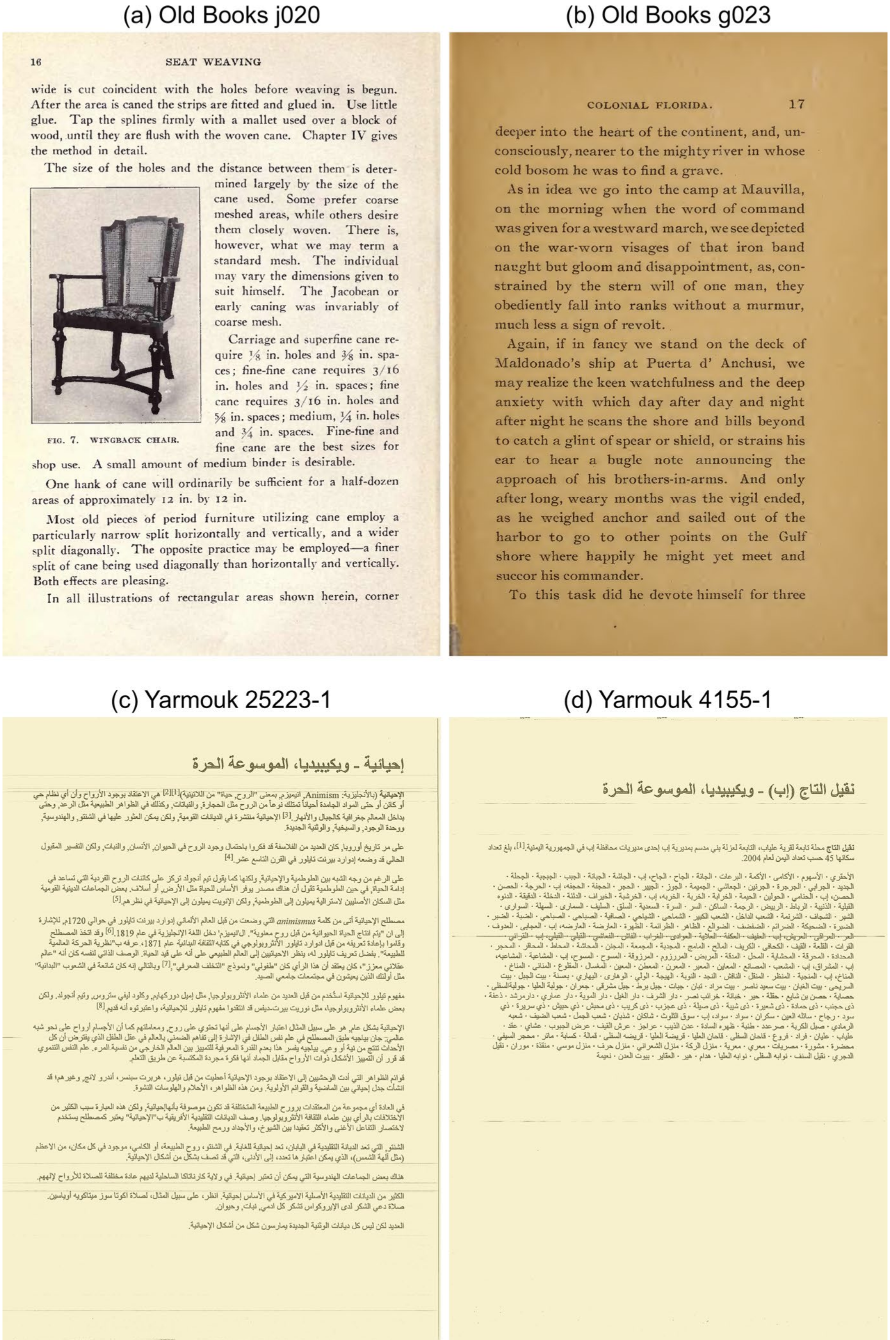


Fig. 1 Sample test documents in their original state

noise. Sometimes it is the original copy that is degraded; at other times the document passed through a poor photocopier, an old microfilm, or a blurry lens before reaching us. The type and degree of noise will vary across collections and individual documents, but most scholars who use archival material will encounter this problem at least occasionally.

A key objective of the experiment was, therefore, to gauge the effect of different types of visual noise on OCR performance. To achieve this, I programmatically applied different types of artificial noise to the test materials, so as to allow isolation of noise effects at the measurement stage. Specifically, the two dataset were duplicated 43 times, each with a different type of noise filter. The R code used for noise generation is included in the Appendix.[5]

I began by creating a binary version of each image, so that there were two versions—colour and greyscale—with no added noise (see Fig. 2a and b). I then wrote functions to generate six ideal types of image noise: "blur," "weak ink," "salt and pepper," "watermark," "scribbles," and "ink stains" (see Fig. 2c-h). While not an exhaustive list of possible noise types, they represent several of the most common ones found in historical document scans.[6] I applied each of the six filters to both the colour version and the binary version of the images, thus creating 12 additional versions of each image. Lastly I applied all available combinations of two noise filters to the colour and binary images, for an additional 30 versions.

This generated a total of 44 image versions divided into three categories of noise intensity: 2 versions with no added noise, 12 versions with one layer of noise, and 30 versions with two layers of noise. This amounted to an English test corpus of 14,168 documents and an Arabic test corpus of 4400 documents. The dataset is preserved as the "Noisy OCR Dataset" ([12]).

Processing

The experiment aimed at measuring out-of-the-box performance, so documents were submitted without further preprocessing using the OCR engines' default settings.[7] While this is an uncommon use of Tesseract, it treats the engines equally and helps highlight the degree to which Tesseract is dependent on image preprocessing.

The English corpus was submitted to all three OCR engines in a total of 42,504 document processing requests. The Arabic corpus was only submitted to Tesseract and Document AI—since Textract does not support Arabic—for a total of 8800 processing requests.

[5] There are other ways of generating synthetic noise, notably the powerful tool DocCreator ([15]). I chose not to use DocCreator primarily because it is graphical user interface-based, and I found I could generate realistic noise more efficiently with R code.

[6] It would be possible to extend the list of noise types further, to include 10–20 different types, but this would increase the size of the corpus (and thus the processing costs) considerably, probably without affecting the broad result patterns. Since the main aim here is not to map all noise types but to compare processors, I decided on a manageable subset of noise types.

[7] The only exception was the setting of the relevant language libraries in Tesseract.

(a) Original


16 SEAT WEAVING

wide is cut coincident with the holes before weaving is begun. After the area is caned the strips are fitted and glued in. Use little glue. Tap the splines firmly with a mallet used over a block of wood, until they are flush with the woven cane. Chapter IV gives the method in detail.

The size of the holes and the distance between them is determined largely by the size of the cane used. Some prefer coarse meshed areas, while others desire them closely woven. There is, however, what we may term a standard mesh. The individual may vary the dimensions given to suit himself. The Jacobean or early caning was invariably of coarse mesh.

FIG. 7. WINGBACK CHAIR.

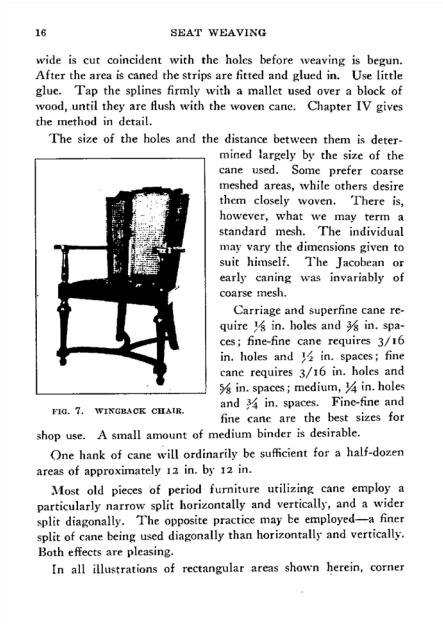
Carriage and superfine cane require ⅛ in. holes and ⅜ in. spaces; fine-fine cane requires 3/16 in. holes and ½ in. spaces; fine cane requires 3/16 in. holes and ⅝ in. spaces; medium, ¼ in. holes and ¾ in. spaces. Fine-fine and fine cane are the best sizes for shop use. A small amount of medium binder is desirable.

One hank of cane will ordinarily be sufficient for a half-dozen areas of approximately 12 in. by 12 in.

Most old pieces of period furniture utilizing cane employ a particularly narrow split horizontally and vertically, and a wider split diagonally. The opposite practice may be employed—a finer split of cane being used diagonally than horizontally and vertically. Both effects are pleasing.

In all illustrations of rectangular areas shown herein, corner

(b) Binary


16 SEAT WEAVING

wide is cut coincident with the holes before weaving is begun. After the area is caned the strips are fitted and glued in. Use little glue. Tap the splines firmly with a mallet used over a block of wood, until they are flush with the woven cane. Chapter IV gives the method in detail.

The size of the holes and the distance between them is determined largely by the size of the cane used. Some prefer coarse meshed areas, while others desire them closely woven. There is, however, what we may term a standard mesh. The individual may vary the dimensions given to suit himself. The Jacobean or early caning was invariably of coarse mesh.

FIG. 7. WINGBACK CHAIR.

Carriage and superfine cane require ⅛ in. holes and ⅜ in. spaces; fine-fine cane requires 3/16 in. holes and ½ in. spaces; fine cane requires 3/16 in. holes and ⅝ in. spaces; medium, ¼ in. holes and ¾ in. spaces. Fine-fine and fine cane are the best sizes for shop use. A small amount of medium binder is desirable.

One hank of cane will ordinarily be sufficient for a half-dozen areas of approximately 12 in. by 12 in.

Most old pieces of period furniture utilizing cane employ a particularly narrow split horizontally and vertically, and a wider split diagonally. The opposite practice may be employed—a finer split of cane being used diagonally than horizontally and vertically. Both effects are pleasing.

In all illustrations of rectangular areas shown herein, corner

(c) Blur


16 SEAT WEAVING

wide is cut coincident with the holes before weaving is begun. After the area is caned the strips are fitted and glued in. Use little glue. Tap the splines firmly with a mallet used over a block of wood, until they are flush with the woven cane. Chapter IV gives the method in detail.

The size of the holes and the distance between them is determined largely by the size of the cane used. Some prefer coarse meshed areas, while others desire them closely woven. There is, however, what we may term a standard mesh. The individual may vary the dimensions given to suit himself. The Jacobean or early caning was invariably of coarse mesh.

FIG. 7. WINGBACK CHAIR.

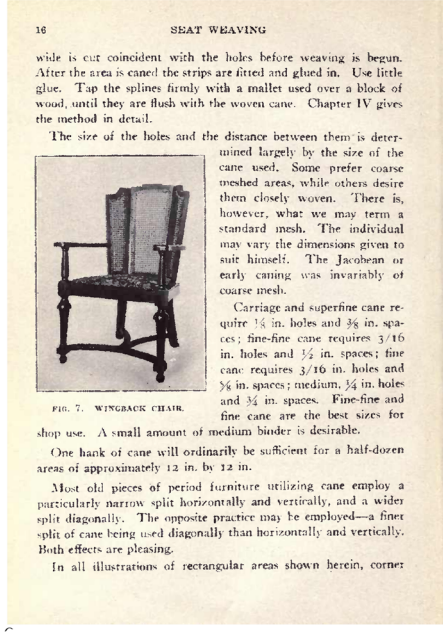
Carriage and superfine cane require ⅛ in. holes and ⅜ in. spaces; fine-fine cane requires 3/16 in. holes and ½ in. spaces; fine cane requires 3/16 in. holes and ⅝ in. spaces; medium, ¼ in. holes and ¾ in. spaces. Fine-fine and fine cane are the best sizes for shop use. A small amount of medium binder is desirable.

One hank of cane will ordinarily be sufficient for a half-dozen areas of approximately 12 in. by 12 in.

Most old pieces of period furniture utilizing cane employ a particularly narrow split horizontally and vertically, and a wider split diagonally. The opposite practice may be employed—a finer split of cane being used diagonally than horizontally and vertically. Both effects are pleasing.

In all illustrations of rectangular areas shown herein, corner

(d) Weak ink


16 SEAT WEAVING

wide is cut coincident with the holes before weaving is begun. After the area is caned the strips are fitted and glued in. Use little glue. Tap the splines firmly with a mallet used over a block of wood, until they are flush with the woven cane. Chapter IV gives the method in detail.

The size of the holes and the distance between them is determined largely by the size of the cane used. Some prefer coarse meshed areas, while others desire them closely woven. There is, however, what we may term a standard mesh. The individual may vary the dimensions given to suit himself. The Jacobean or early caning was invariably of coarse mesh.

FIG. 7. WINGBACK CHAIR.

Carriage and superfine cane require ⅛ in. holes and ⅜ in. spaces; fine-fine cane requires 3/16 in. holes and ½ in. spaces; fine cane requires 3/16 in. holes and ⅝ in. spaces; medium, ¼ in. holes and ¾ in. spaces. Fine-fine and fine cane are the best sizes for shop use. A small amount of medium binder is desirable.

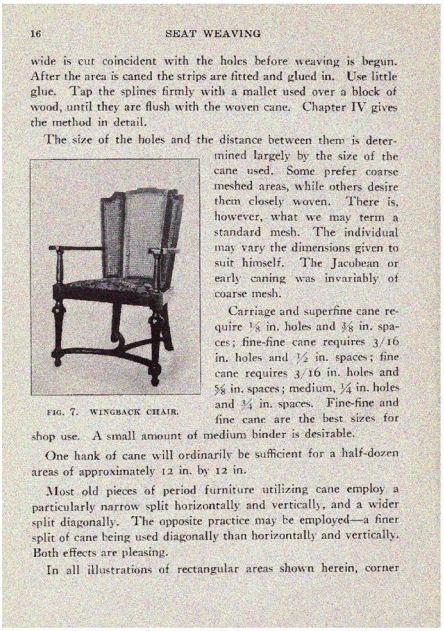
One hank of cane will ordinarily be sufficient for a half-dozen areas of approximately 12 in. by 12 in.

Most old pieces of period furniture utilizing cane employ a particularly narrow split horizontally and vertically, and a wider split diagonally. The opposite practice may be employed—a finer split of cane being used diagonally than horizontally and vertically. Both effects are pleasing.

In all illustrations of rectangular areas shown herein, corner

Fig. 2 Sample test document ("Old Books j020") with noise applied

(e) Salt and pepper



16 SEAT WEAVING

wide is cut coincident with the holes before weaving is begun. After the area is caned the strips are fitted and glued in. Use little glue. Tap the splines firmly with a mallet used over a block of wood, until they are flush with the woven cane. Chapter IV gives the method in detail.

The size of the holes and the distance between them is determined largely by the size of the cane used. Some prefer coarse meshed areas, while others desire them closely woven. There is, however, what we may term a standard mesh. The individual may vary the dimensions given to suit himself. The Jacobean or early caning was invariably of coarse mesh.

FIG. 7. WINGBACK CHAIR.

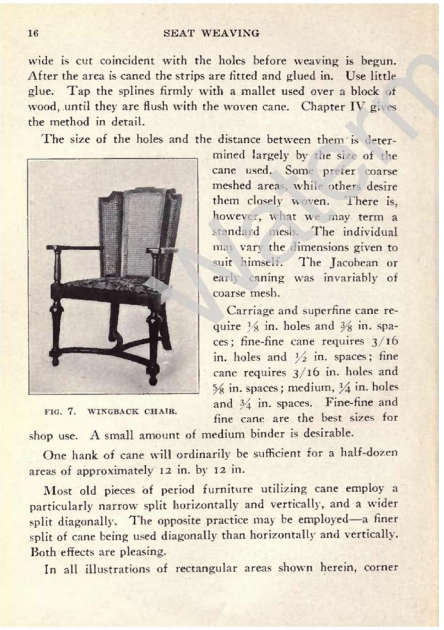
Carriage and superfine cane require 1/8 in. holes and 3/8 in. spaces; fine-fine cane requires 3/16 in. holes and 1/2 in. spaces; fine cane requires 3/16 in. holes and 5/8 in. spaces; medium, 1/4 in. holes and 3/4 in. spaces. Fine-fine and fine cane are the best sizes for shop use. A small amount of medium binder is desirable.

One hank of cane will ordinarily be sufficient for a half-dozen areas of approximately 12 in. by 12 in.

Most old pieces of period furniture utilizing cane employ a particularly narrow split horizontally and vertically, and a wider split diagonally. The opposite practice may be employed—a finer split of cane being used diagonally than horizontally and vertically. Both effects are pleasing.

In all illustrations of rectangular areas shown herein, corner

(f) Watermark



16 SEAT WEAVING

wide is cut coincident with the holes before weaving is begun. After the area is caned the strips are fitted and glued in. Use little glue. Tap the splines firmly with a mallet used over a block of wood, until they are flush with the woven cane. Chapter IV gives the method in detail.

The size of the holes and the distance between them is determined largely by the size of the cane used. Some prefer coarse meshed areas, while others desire them closely woven. There is, however, what we may term a standard mesh. The individual may vary the dimensions given to suit himself. The Jacobean or early caning was invariably of coarse mesh.

FIG. 7. WINGBACK CHAIR.

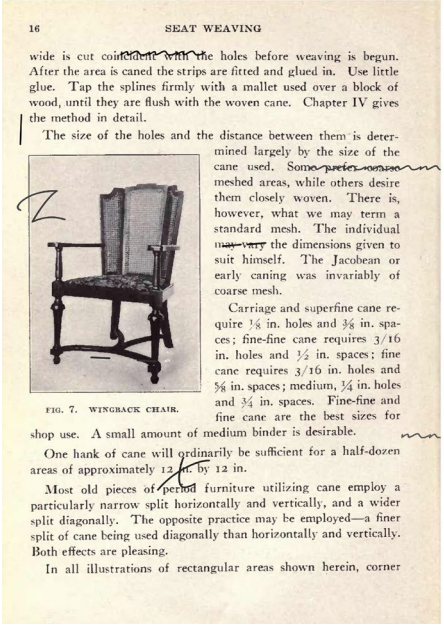
Carriage and superfine cane require 1/8 in. holes and 3/8 in. spaces; fine-fine cane requires 3/16 in. holes and 1/2 in. spaces; fine cane requires 3/16 in. holes and 5/8 in. spaces; medium, 1/4 in. holes and 3/4 in. spaces. Fine-fine and fine cane are the best sizes for shop use. A small amount of medium binder is desirable.

One hank of cane will ordinarily be sufficient for a half-dozen areas of approximately 12 in. by 12 in.

Most old pieces of period furniture utilizing cane employ a particularly narrow split horizontally and vertically, and a wider split diagonally. The opposite practice may be employed—a finer split of cane being used diagonally than horizontally and vertically. Both effects are pleasing.

In all illustrations of rectangular areas shown herein, corner

(g) Scribbles



16 SEAT WEAVING

wide is cut coincident with the holes before weaving is begun. After the area is caned the strips are fitted and glued in. Use little glue. Tap the splines firmly with a mallet used over a block of wood, until they are flush with the woven cane. Chapter IV gives the method in detail.

The size of the holes and the distance between them is determined largely by the size of the cane used. Some prefer coarse meshed areas, while others desire them closely woven. There is, however, what we may term a standard mesh. The individual may vary the dimensions given to suit himself. The Jacobean or early caning was invariably of coarse mesh.

FIG. 7. WINGBACK CHAIR.

Carriage and superfine cane require 1/8 in. holes and 3/8 in. spaces; fine-fine cane requires 3/16 in. holes and 1/2 in. spaces; fine cane requires 3/16 in. holes and 5/8 in. spaces; medium, 1/4 in. holes and 3/4 in. spaces. Fine-fine and fine cane are the best sizes for shop use. A small amount of medium binder is desirable.

One hank of cane will ordinarily be sufficient for a half-dozen areas of approximately 12 in. by 12 in.

Most old pieces of period furniture utilizing cane employ a particularly narrow split horizontally and vertically, and a wider split diagonally. The opposite practice may be employed—a finer split of cane being used diagonally than horizontally and vertically. Both effects are pleasing.

In all illustrations of rectangular areas shown herein, corner

(h) Ink stains

16 SEAT WEAVING

wide is cut coincident with the holes before weaving is begun. After the area is caned the strips are fitted and glued in. Use little glue. Tap the splines firmly with a mallet used over a block of wood, until they are flush with the woven cane. Chapter IV gives the method in detail.

The size of the holes and the distance between them is determined largely by the size of the cane used. Some prefer coarse meshed areas, while others desire them closely woven. There is, however, what we may term a standard mesh. The individual may vary the dimensions given to suit himself. The Jacobean or early caning was invariably of coarse mesh.

FIG. 7. WINGBACK CHAIR.

Carriage and superfine cane require 1/8 in. holes and 3/8 in. spaces; fine-fine cane requires 3/16 in. holes and 1/2 in. spaces; fine cane requires 3/16 in. holes and 5/8 in. spaces; medium, 1/4 in. holes and 3/4 in. spaces. Fine-fine and fine cane are the best sizes for shop use. A small amount of medium binder is desirable.

One hank of cane will ordinarily be sufficient for a half-dozen areas of approximately 12 in. by 12 in.

Most old pieces of period furniture utilizing cane employ a particularly narrow split horizontally and vertically, and a wider split diagonally. The opposite practice may be employed—a finer split of cane being used diagonally than horizontally and vertically. Both effects are pleasing.

In all illustrations of rectangular areas shown herein, corner

Fig. 2 (continued)

The Tesseract processing was done in R with the package `tesseract` (v4.1.1). For Textract, it was carried out via the R package `paws` (v0.1.11), which provides a wrapper for the Amazon Web Services API. For Document AI, I used the R package `daiR` (v0.8.0) to access the Document AI API v1 endpoint. The processing was done in April and May of 2021 and took an estimated net total of 150–200 h to complete. The Document AI and Textract APIs processed documents at a rate of approximately 10–15 s per page. Tesseract took 17 s per page for Arabic and 2 seconds per page for English on a Linux Desktop with a 12-core, 4.3 Ghz CPU and 64GB RAM.

Measurement

Accuracy was measured with the ISRI tool ([30]) in Eddie Antonio Santos's (2019) updated version—known as Ocreval—which has UTF-8 support. ISRI is a simple but robust tool that has been used for OCR assessment since its creation in the mid-1990s. Alternatives exist ([1, 5, 46]), but ISRI was deemed sufficient for this exercise.

ISRI compares two versions of a text—in this case OCR output to ground truth—and returns a range of measures for divergence, notably a document's overall character accuracy and word accuracy expressed in percent. Character accuracy is the proportion of characters in a hypothesis text that match the reference text. Any misread, misplaced, absent, or excess character is considered an error and subtracted from the numerator. This represents the so-called Levenshtein distance ([20]), i.e., the minimum number of edit operations needed to correct the hypothesis text. Word accuracy is the proportion of non-stopwords in a hypothesis text that match those of the reference text.[8]

Character and word accuracy are usually highly correlated, but the former punishes error harder, since each wrong character detracts from the accuracy rate.[9] In word accuracy, by contrast, a misspelled word counts as one error regardless of the number of wrong characters that contribute to the error. Moreover, in ISRI's implementation of word accuracy, case errors and excess words are ignored.[10]

[8] ISRI only has an English-language stopword list (of 110 words), so in the measurements for Arabic, stopwords are included in the assessment. All else equal, this should produce slightly higher accuracy rates for Arabic, since oft-recurring words are easier for OCR engines to recognize.

[9] ISRI's character accuracy rates can actually be negative as a result of excess text. OCR engines sometimes introduce garbled text when they see images or blank areas with noise, resulting in output texts that are much longer than ground truth. Since excess characters are treated as errors and subtracted from the numerator, they can result in negative accuracy rates. In the corpus studied here, this phenomenon affected 4.6 percent of the character accuracy measurements, and it occurred almost exclusively in texts processed by Tesseract.

[10] This also means that ISRI's word accuracy tool does not yield negative rates. As Eddie Antonio Santos explains, "The wordacc algorithm creates parallel arrays of words and checks only for words present in the ground truth. It finds 'paths' from the generated file that correspond to ground truth. For this reason, it only detects as many words as there are in ground truth"; private email correspondence, 1 September 2021. However, the word accuracy tool returns NA when the hypothesis text has no recognizable words. This occurred in 9.4 percent of the measurements in this experiment, again almost exclusively in Tesseract output. These NAs are treated as zeroes in Figs. 4,5,6

Figure 3 provides some examples of what character and word error rates may correspond to in an actual text. I will return later to the question of how error matters for analysis.

Which of the two measures is better depends on the type of document and the purpose of the analysis. For shorter texts where details matter—such as forms and business documents—character accuracy is considered the more relevant measure. For longer texts to be used for searches or text mining, word accuracy is commonly used as the principal metric. In the following, I, therefore, report word accuracy rates, transformed to word error rates by subtracting them from 100. Character accuracy rates are available in the Appendix.

Results

The main results are shown in Fig. 4 and reveal clear patterns. Document AI had consistently lower error rates, with Textract coming in a close second, and Tesseract last. More noise yielded higher error rates in all engines, but Tesseract was significantly more sensitive to noise than the two others. Overall, there was a significant performance gap between the server-based processors (Document AI and Textract) on one side and the local installation (Tesseract) on the other. Only on noise-free documents in English could Tesseract compete.

We also see a marked performance difference across languages. Both Document AI and Tesseract delivered substantially lower accuracy for Arabic than they did for English. This was despite the Arabic corpus consisting of Internet articles in a single, very common font, while the English corpus contained old book scans in several different fonts. An analogous Arabic corpus would likely have produced an even larger performance gap. This said, Document AI represents a significant improvement on Tesseract as far as out-of-the-box Arabic OCR is concerned.

Disaggregating the data by noise type shows a more detailed picture (see Figs. 5 and 6). Beyond the patterns already described, we see, for example, that both Textract and Tesseract performed somewhat better on greyscale versions of the test images than on the colour version. We also note that all engines struggled with blur, while Tesseract was much more sensitive to salt & pepper noise than the two other engines. Incidentally, it is not surprising that the ink stain filter yielded lower accuracy throughout since it completely concealed part of the text. The reason we see a bimodal distribution in the bin + blur" filters on the English corpus is that they yielded many zero values, probably as a result of the image crossing a threshold of illegibility. The same did not happen in the Arabic corpus, probably because the source images there had crisper characters at the outset.

Implications

When is it worth paying for better OCR accuracy? The answer depends on a range of situational factors, such as the state of the corpus, the utility function of the researcher, and the intended use case.

(a) Ground truth

16 SEAT WEAVING

wide is cut coincident with the holes before weaving is begun. After the area is caned the strips are fitted and glued in. Use little glue. Tap the splines firmly with a mallet used over a block of wood, until they are flush with the woven cane. Chapter IV gives the method in detail.

The size of the holes and the distance between them is determined largely by the size of the cane used. Some prefer coarse meshed areas, while others desire them closely woven. There is, however, what we may term a standard mesh. The individual may vary the dimensions given to suit himself. The Jacobean or early caning was invariably of coarse mesh.

Carriage and superfine cane require ⅛ in. holes and ⅜ in. spaces; fine-fine cane requires 3/16 in. holes and ½ in. spaces; fine cane requires 3/16 in. holes and ⅜ in. spaces; medium, ¼ in. holes and ¾ in. spaces. Fine-fine and fine cane are the best sizes for shop use. A small amount of medium binder is desirable.

One hank of cane will ordinarily be sufficient for a half-dozen areas of approximately 12 in. by 12 in.

Most old pieces of period furniture utilizing cane employ a particularly narrow split horizontally and vertically, and a wider split diagonally. The opposite practice may be employed—a finer split of cane being used diagonally than horizontally and vertically. Both effects are pleasing.

In all illustrations of rectangular areas shown herein, corner

Old Books j020

(b) ~5% word error (~8% character error)

16 SEAT WEAVING

wide is cut coincident with the holes before weaving is begun. After the area is caned the strips are fitted and glued in. Use little glue. Tap the splines firmly with a mallet used over a block of wood, until they are flush with the woven cane. Chapter IV gives the method in detail.

The size of the holes and the distance between them is deter- mined largely by the size of the cane used. Some prefer coarse | meshed areas, while others desire . Fie them closely woven. There is, =_ however, what we may term a ee Lf standard mesh. The individual : te may vary the dimensions given to | 23 iS La suit himself. The Jacobean or . c a early caning was invariably of _| coarse mesh.

| Carriage and superfine cane re- / quire 14 in. holes and 34 in. spa- | ces; fine-fine cane requires 3/16 <q in. holes and 1% in. spaces; fine 7 ood cane requires 3/16 in. holes and = 5@ in. spaces; medium, 14 in. holes hic. 7 owiehe come and 34 in. spaces. Fine-fine and fine cane are the best sizes for shop use. A small amount of medium binder is desirable.

One hank of cane will ordinarily be sufficient for a half-dozen areas of approximately 12 in. by 12 in.

Most old pieces of period furniture utilizing cane employ a particularly narrow split horizontally and vertically, and a wider split diagonally. The opposite practice may be employed—a finer split of cane being used diagonally than horizontally and vertically. Both effects are pleasing.

In all illustrations of rectangular areas shown herein, corner

Old Books j020 with col + wm, Tesseract

(c) ~10% word error (~8% character error)

16 SEAT WEAVING

wide is cut coinbident with the holes before weaving is begun. After the area is caned the strips are fitted and glued in. Use little glue. Tap the splines firmly with a mallet used over a block of wood, until they are flush with the woven cane. Chapter IV gives | the method in detail.

The size of the holes and the distance between them is deter- mined largely by the size of the cane used. meshed areas, while others desire Z them closely woven. There is, however, what we may term a standard mesh. The individual mariary the dimensions given to suit himself. The Jacobean or early caning was invariably of coarse mesh.

Carriage and superfine cane re- quire 1/2 in. holes and 3/8 in. spa- ces; fine-fine cane requires 3/16 in. holes and 1/2 in. spaces; fine cane requires 3/16 in. holes and in. spaces ; medium, 1/4 in. holes and 3/4 in. spaces. Fine-fine and FIG. 7. WINGBACK CHAIR. fine cane are the best sizes for shop use. A small amount of medium binder is desirable.

One hank of cane will ordinarily be sufficient for a half-dozen areas of approximately 12 n. by 12 in.

Most old pieces of period furniture utilizing cane employ a particularly narrow split horizontally and vertically, and a wider split diagonally. The opposite practice may be employed--a finer split of cane being used diagonally than horizontally and vertically. Both effects are pleasing.

In all illustrations of rectangular areas shown herein, corner

Old Books j020 with col + weak + scrib, Textract

(d) ~20% word error (~37% character error)

pA SRA WEAVING 0000.0

= wide is cut coi(@idetl Wit| the holes| before weaving is begun. After the area is canded the strips are fitted and glued in. Usellttle co > >| glue. |Tap| the splines| firmly with a mallet used |over a block of "_- wood, until they are flush with the woven cane. Chapter| IV gives G the method |adetil: oS

Is | The size of the holes and| the distance |between them| is deters eae ess anaes as 2s mined: largely| by| the size of the| ae oe es ie UA Ree _ so \ cee meshed |areas, while others desire bes one f ae me them| closely| woven. There is| a- !f ER BB | however, what we may term a | fo BER ea | standard mesh. The individual | nati 23S | naevery the dimensions given to Sones qT L 22M | suit himself| The Jacobean or 2 co mwa =| darly caning| was invariably of ok eB *s me ae b coarse amesh. _e _H 5 Ee S *eh a

vege § pe as Carriage and superfine cane ree Co es Es- | quire 14 in holes and 4 in. spa- eee Ben Pa | ces; fine-fine |cane requires 3/16| Cop | "-_]in holes and 34 in. spaces| fine 9 > PO MB | cale requires 3/26 holes and NSP og or | 5 in. spaces ; medium, 4 in. holes _2 and 34 in. spaces. Fine-fine and| "BIG 7. WINGBACK CHAIRE! 0 Soc 0 Eee et ge co fine cane |are the best sizes for| shop use| A small amount of medium binder is desirable Ann

One hank of cane will ordinarily be sufficient for a half-dozen bs 4 areas of srosimaely 2 ACT 8 pepe fete " | _ * |" age

o ae "Most old |pieces| of period furniture| utilizing| cane employ aes ae particularly narrow split horizontally |and vertically, and a wider ie wee ©» split diagonally. |The opposite practice may| be| employed—a finer split of cane being used diagonally than horizontally and vertically. ae Both effects|are pleasing. PSs Re CEES 3 US OLS ae

|

|In all illustrations |of rectangular areas shown |herein, corner age

Old Books j020 with bin + snp + scrib, Tesseract

Errors hand annotated in green. Omitted text unmarked.

Fig. 3 Examples of word error effects

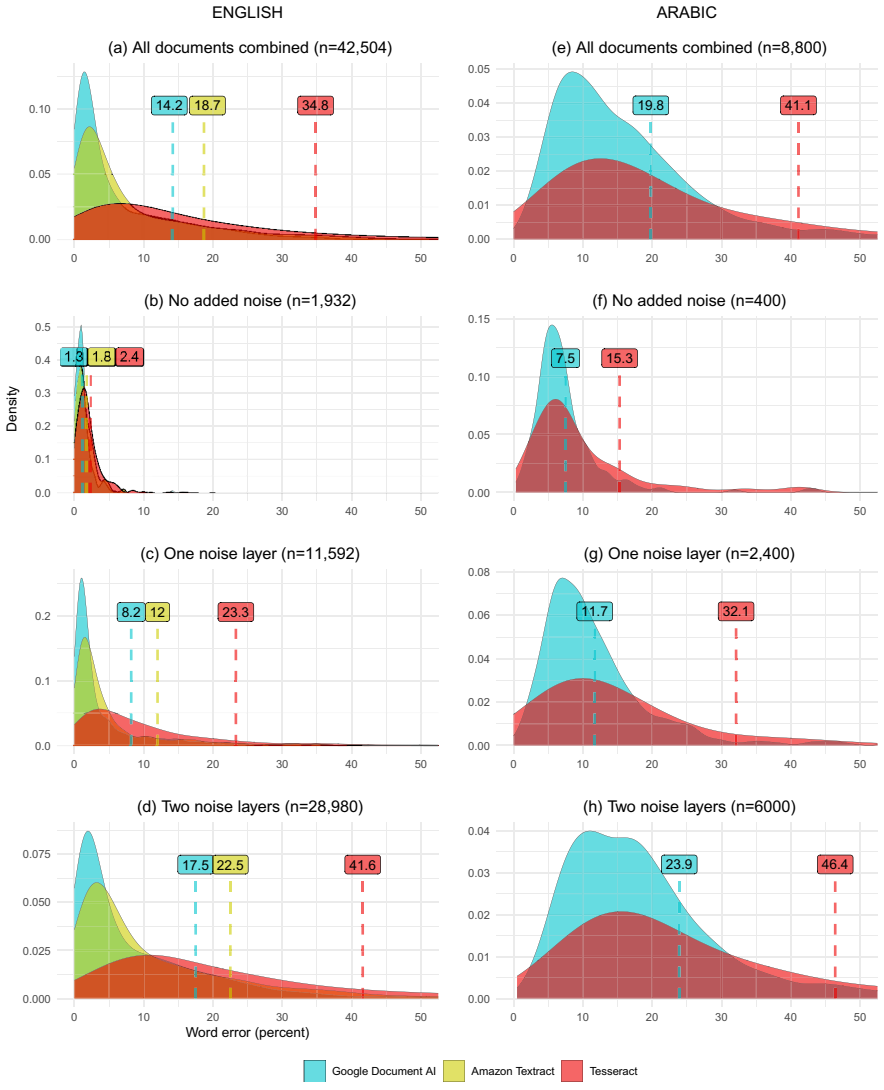


Fig. 4 Word error rates by engine and noise level for English and Arabic documents

Much hinges on the corpus itself. As we have seen, accuracy gains increase with noise and are higher for certain types of noise. Moreover, if the corpus contains many different types of noise, a better general processor will save the researcher relatively more preprocessing time. Unfortunately we lack good tools for (ground truth-free) noise diagnostics, but there are ways to obtain some information about the noise state of the corpus ([10, 21, 28]). Finally, the size of the dataset matters, since processing costs scale with the number of documents while accuracy gains do not.

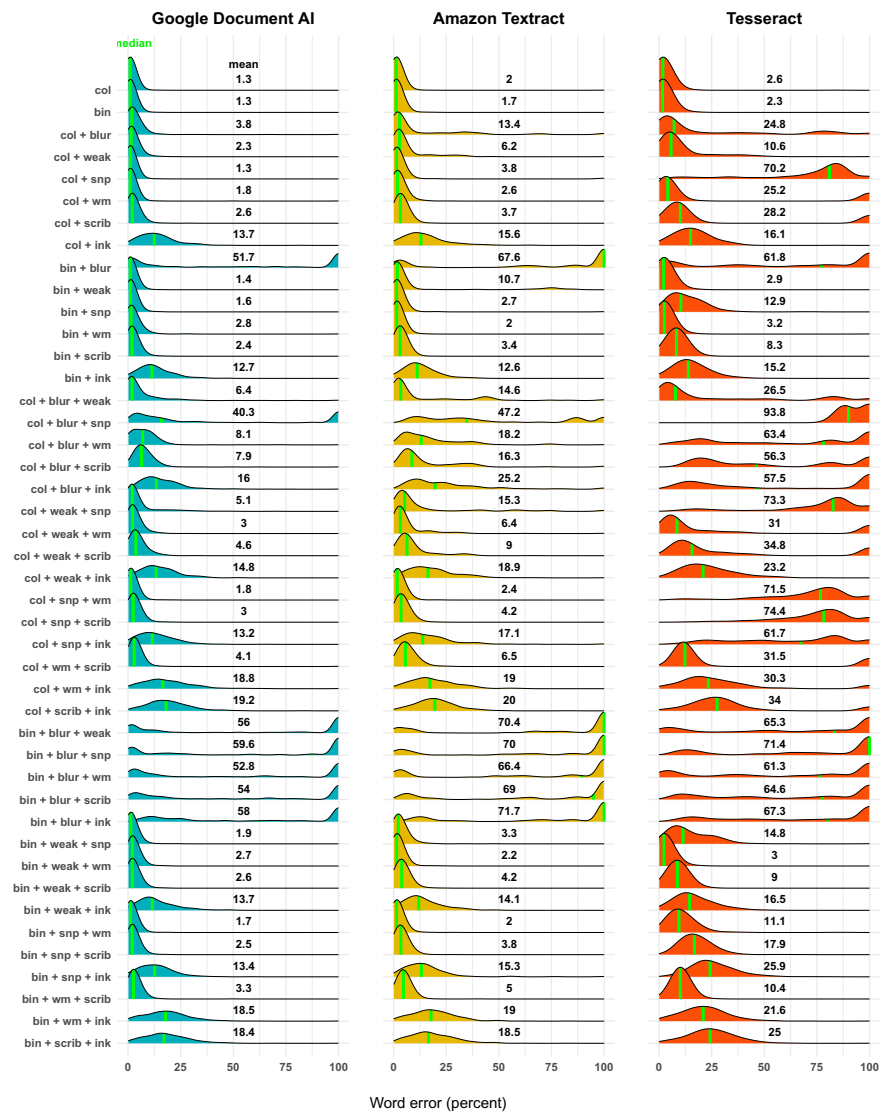


Fig. 5 Word error rates by engine and noise type for English-language documents

The calculus also depends on the economic situation of the researcher. Aside from absolute size of one's budget, a key consideration is labour cost, since cloud-based processing is in some sense a substitute for Tesseract processing with additional labour input. The latter option will thus make more sense for a student than for a professor and more sense for the faster programmer.

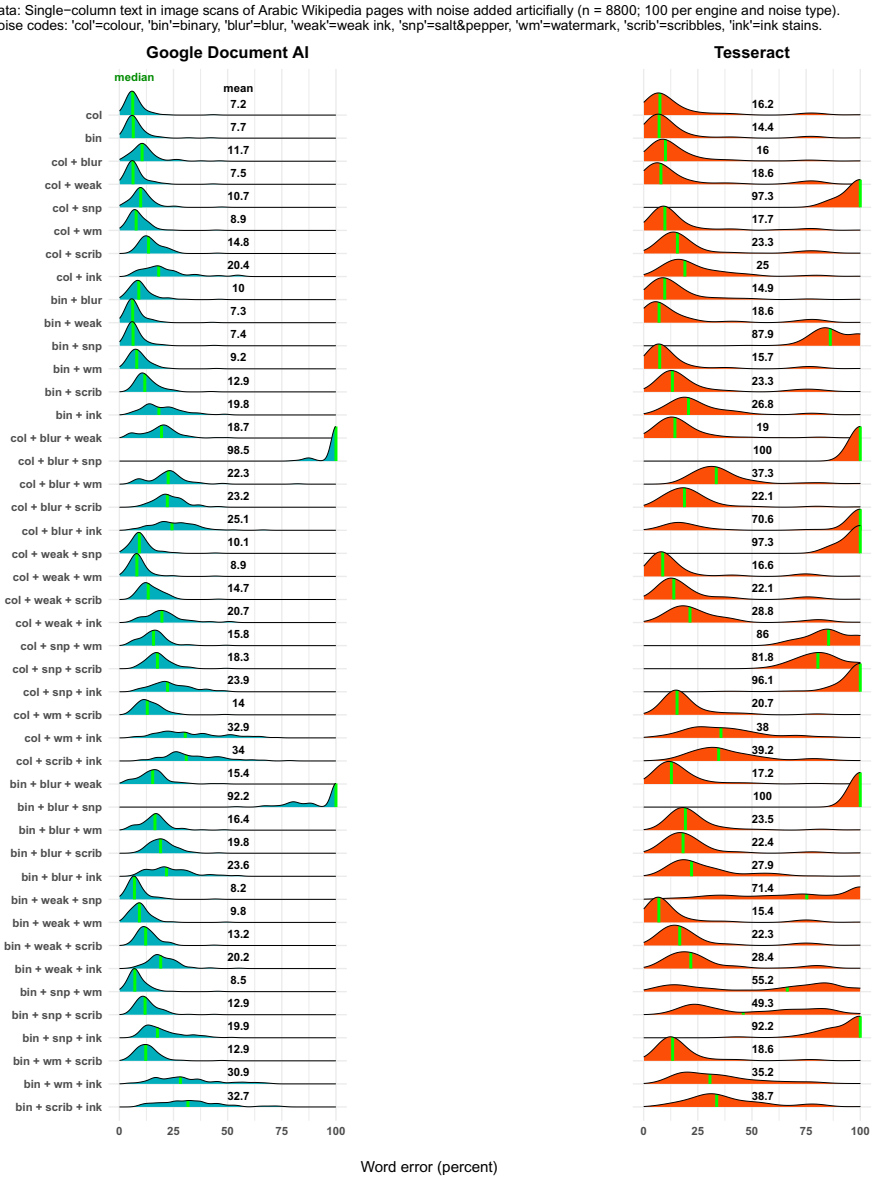


Fig. 6 Word error rates by engine and noise type for Arabic-language documents

Last but not least is the intended use of the OCRed text. If the aim is to recreate a perfect plaintext copy of the original document for, say, a browseable digital archive, then every percentage point matters. But if the purpose is to build a topic model or conduct a sentiment analysis, it is not obvious that a cleaner text will always yield better end results. The downstream effects of OCR error is a

Table 2 Composition of Old Books corpus

Title	Author	Year	Pages	Words
Engraving of Lions, Tigers, Panthers, Leopards, Dogs,&C.	Thomas Landseer	1853	8 (28)	3983
The Corset and the Crinoline	William Barry Lord	1868	30 (254)	9633
Horton Genealogy	George Firman Horton	1876	34 (316)	11744
Historical Sketches of Colonial Florida	Richard Lewis Campbell	1892	30 (284)	4801
Half-Hours with the Highwaymen	Charles George Harper	1908	34 (422)	7695
Betrayed Armenia	Diana Agabeg Apcar	1910	39 (77)	15001
The Lusitania's Last Voyage	Charles Emelius Lauriat, Jr.	1915	23 (162)	3438
The Child of the Moat	Ian B. Stoughton Holborn	1916	30 (408)	7844
Seat Weaving	L. Day Perry	1917	57 (96)	12437
The Boy Apprenticed to an Enchanter	Padraic Colum	1920	37 (168)	7420

complex topic that cannot be explored in full here, but we can get some pointers by looking at the available literature and doing some tests of our own.

Existing research suggests that the effects of OCR error vary by analytical toolset. Broadly speaking, topic models have proved relatively robust to OCR inaccuracy ([6, 9, 26, 36]), with [40] suggesting a baseline for acceptable OCR accuracy as low as 80 percent. Classification models have been somewhat more error-sensitive, although the results here have been mixed ([6, 25, 34, 40]). The biggest problems seem to arise in natural language processing (NLP) tasks where details matter, such as part-of-speech tagging and named entity recognition ([11, 22, 24, 40]).

To illustrate some of these dynamics and add to the empirical knowledge of OCR error effects, we can run some simple tests on the English-language materials from our benchmarking exercise. The Old Books dataset is small, but similar in kind to the types of text collections studied by historians and social scientists, and hence a reasonably representative test corpus. In the following, I look at OCR error in four analytical settings: sentiment analysis, classification, topic modelling, and named entity recognition. I exploit the fact that the benchmarking exercise yielded 132 different variants (3 engines and 44 noise types) of the Old Book corpus, each with a somewhat different amount of OCR error.[11] By running the same analyses on all text variants, we should get a sense of how OCR error can affect substantive findings. This said, the exercise as a whole is a back-of-the-envelope test insofar as it covers only a small subset of available text mining methods and does not implement any of them as fully as one would in a real-life setting.

Sentiment analysis

Faced with a corpus like Old Books (see Table 2), a researcher might want to explore text sentiment, for example to examine differences between authors or over

[11] In all of the below, "OCR error" refers to word error rates computed with the ISRI tool.

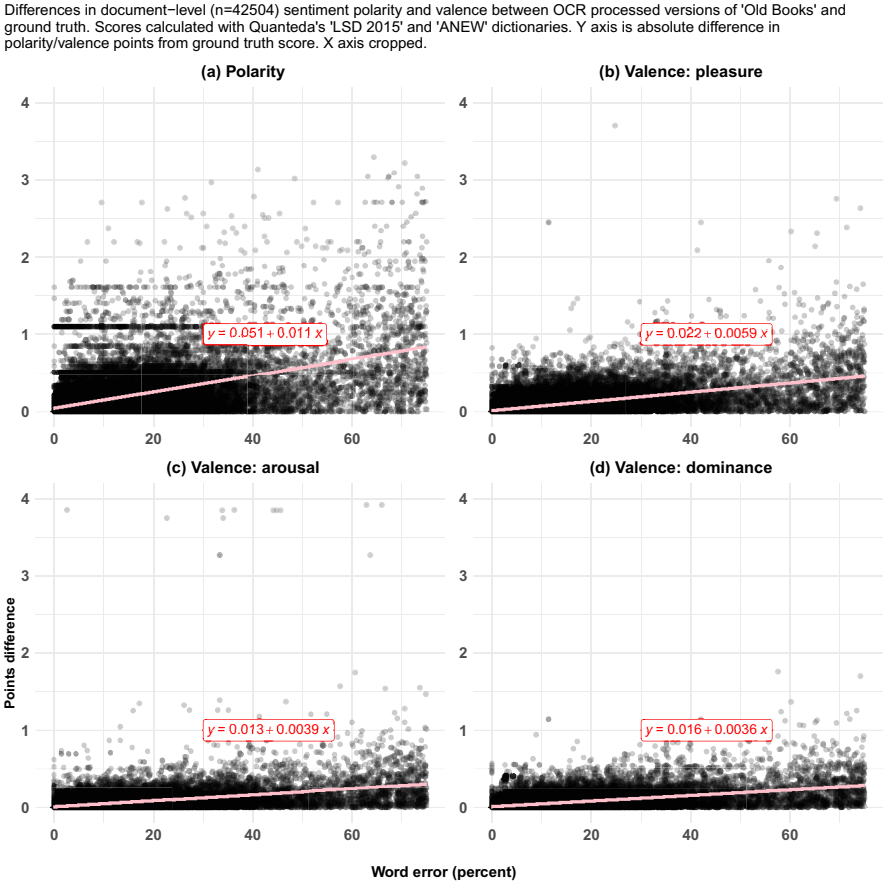


Fig. 7 OCR error and sentiment analysis accuracy

time. Using the R package `quantedas` LSD 2015 and ANEW dictionaries, I generated document-level sentiment polarity and valence scores for all variants of the corpus after standard preprocessing. To assess the effect of OCR error, I calculated the absolute difference between these scores and those of the ground truth version of the corpus. Figure 7a–d indicate that these differences increase only slightly with OCR error, but also that, for sentiment polarity, the variance is such that just a few percent OCR error can produce sentiment scores that diverge from ground truth scores by up to two whole points at the document level.

Text classification

Another common analytical task is text classification. Imagine that we knew which works were represented in the Old Books corpus, but not which work each document belonged to. We could then handcode a subset and train an algorithm to classify

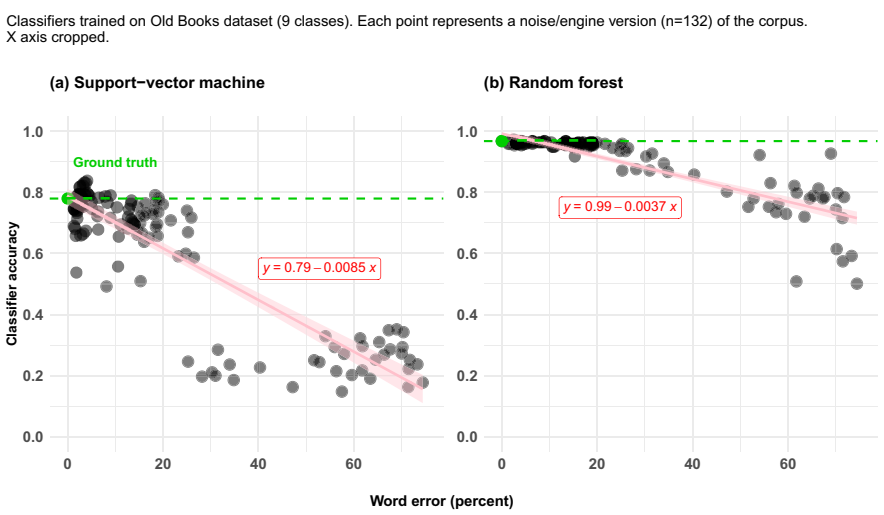


Fig. 8 OCR error and multiclass classifier accuracy

the rest. Since we happen to have pre-coded metadata we can easily simulate this exercise. I trained two multiclass classifiers—Random Forest and Support-Vector Machine—to retrieve the book from which a document was drawn. To avoid imbalance, I removed the smallest subset ("Engraving of Lions, Tigers, Panthers, Leopards, Dogs,&C.") and was left with 9 classes and 314 documents. For each variant of the corpus I preprocessed the texts, split them 70/30 for training and testing, and fit the models using the `tidymodels` R package. Figure s 8a, b shows the results. We see that OCR error has only a small negative effect on classifier accuracy up to a threshold of around 20% OCR error, after which accuracy plummets.

Topic modelling

Assessing the effect of OCR error on topic models is more complicated, since they involve more judgment calls and do not yield an obvious indicator of accuracy. I used the `stm` R package to fit structural topic models to all the versions of the corpus. As a first step, I ran the `stm::searchK()` function for a k value range from 6 to 20, on the suspicion that different variants of the text might yield different diagnostics and hence inspire different choices for the number of topics in the model. Figure 9a shows that the k intercept for the high point of the held-out likelihood curve varies from 6 to 12 depending on the version of the corpus. Held-out likelihood is not the only criterion for selecting k, but it is an important one, so these results suggests that even a small amount of OCR error can lead researchers to choose a different topic number than they would have done on a cleaner text, with concomitant effects on the substantive analysis. Moreover, if we hold k still at 8—the value suggested by diagnostics of the ground truth version of the corpus—we see in Fig. 9b that the semantic coherence of the model decreases slightly with more noise.

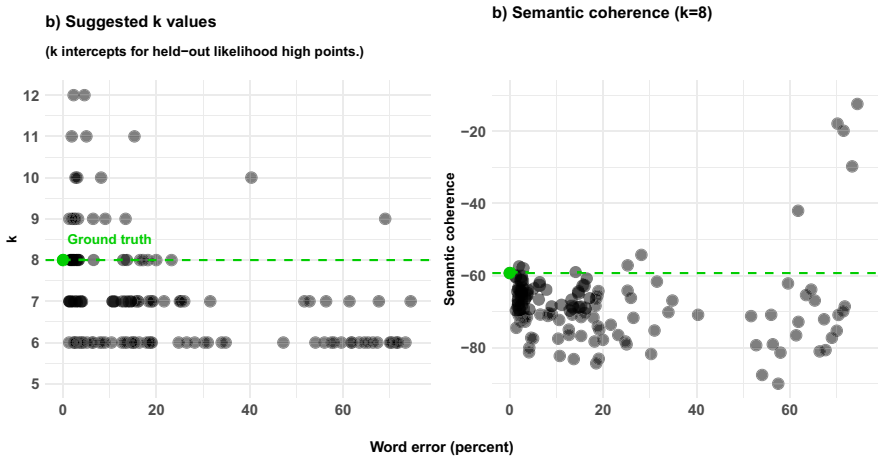


Fig. 9 OCR error and topic model fits

Named entity recognition

Our corpus is full of names and dates, so a researcher might also want to explore it with named entity recognition (NER) models. I used a pretrained `spaCy` model (`en_core_web_sm`) to extract entities from all non-preprocessed versions of the corpus and compared the output to that of the ground truth text. In the absence of ground truth NER label data, I treated `spaCy`'s prediction for the ground truth text as the reference point and calculated the F1 score (the harmonic average of precision and recall) as a metric for accuracy. For simplicity, the evaluation included only predicted entity names, not entity labels. Figure 10 shows that OCR error affected NER accuracy severely. In a real-life setting these effects would be partly mitigated by pre- and postprocessing, but it seems reasonable to suggest that NER is one of the areas where the value added from high-precision OCR is the highest.

Broadly speaking, these tests indicate that OCR error mattered the most in NER, the least in topic modelling and sentiment analysis, while in classification there was a tipping point at around 20 percent OCR error. At the same time, all the tests showed some accuracy deterioration even at very low OCR error rates.

Conclusion

This article described a systematic test of three general OCR processors on a large new dataset of English and Arabic documents. It suggests that the server-based engines Document AI and Textract deliver markedly higher out-of-the-box accuracy than the standalone Tesseract library, especially on noisy documents. It also

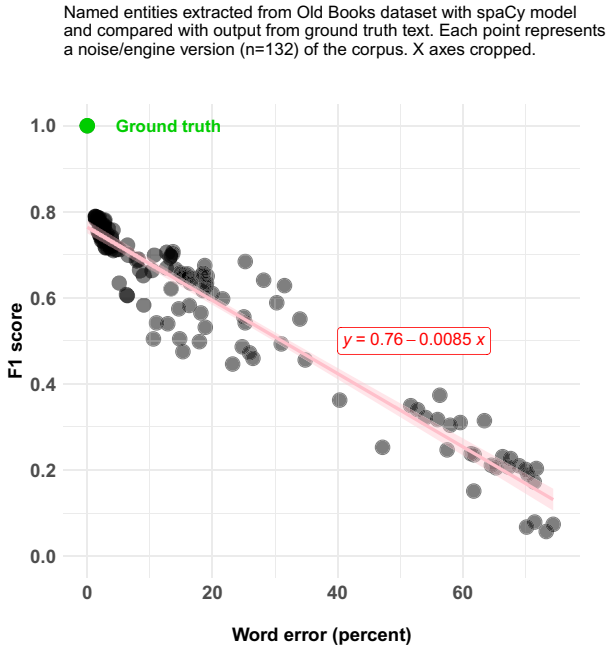


Fig. 10 OCR error and named entity recognition accuracy

indicates that certain types of "integrated" noise, such as blur and salt and pepper, generate more error than "superimposed" noise such as watermarks, scribbles, and even ink stains. Furthermore, it suggests that the "OCR language gap" still persists, although Document AI seems to have partially closed it, at least for Arabic.

The key takeaway for the social sciences and humanities is that high-accuracy OCR is now more accessible than ever before. Researchers who might be deterred by the prospect of extensive document preprocessing or corpus-specific model training now have at their disposal user-friendly tools that deliver strong results out of the box. This will likely lead to more scholars adopting OCR technology and to more historical documents becoming digitized.

The findings can also help scholars tailor OCR solutions to their needs. For many users and use cases, server-based OCR processing will be an efficient option. However, there are are downsides to consider, such as processing fees and data privacy concerns, which means that in some cases, other solutions—such as self-trained Tesseract models or even plain Tesseract—might be preferable.[12] Having baseline

[12] Amazon openly says it "may store and use document and image inputs [...] to improve and develop the quality of Amazon Textract and other Amazon machine-learning/artificial-intelligence technologies" (see https://aws.amazon.com/textract/faqs/, accessed 3 September 2021). Google says it "does not use any of your content [...] for any purpose except to provide you with the Document AI API service" (see https://cloud.google.com/document-ai/docs/data-usage, accessed 3 September 2021), but it is unclear what lies in the word "provide" and whether it includes the training of the processor.

data on relative processor performance and differential effects of noise types can help navigate such tradeoffs and optimise one's workflow.

The study has several limitations, notably that it tested only three processors on two languages with a non-exhaustive list of noise types. This means we cannot say which processor is the very best on the market or provide a comprehensive guide to OCR performance on all languages and noise types. However, the test design used here can easily be applied to other processors, languages, and noise types for a more complete picture. Another limitation is that the experiment only used single-column test materials, which does not capture layout parsing capabilities. Most OCR engines, including Document AI and Textract, still struggle with multi-column text, and even state-of-the-art tools such as Layout Parser ([32]) require corpus-specific training for accurate results. Future studies will need to determine which processors deliver the best out-of-the-box layout parsing. In any case, we appear to be in the middle of a small revolution in OCR technology with potentially large benefits for the social sciences and humanities.

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Alghamdi, Mansoor A., Alkhazi, Ibrahim S., & Teahan, William J. (2016). "Arabic OCR Evaluation Tool." In *2016 7th International Conference on Computer Science and Information Technology (CSIT)*, 1–6. IEEE.
2. Barcha, Pedro. (2017). *Old Books Dataset*. GitHub: GitHub Repository. https://github.com/PedroBarcha/old-books-dataset.
3. Bieniecki, W., Grabowski, S., & Rozenberg, W. (2007). "Image Preprocessing for Improving Ocr Accuracy." In *2007 International Conference on Perspective Technologies and Methods in MEMS Design*, 75–80. IEEE.
4. Boiangiu, C.-A., Ioanitescu, Radu, & Dragomir, Razvan-Costin. (2016). Voting-Based OCR System. *The Proceedings of Journal ISOM, 10*, 470–86.
5. Carrasco, R. C. (2014). "An Open-Source OCR Evaluation Tool." In *Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage*, 179–84.
6. Colavizza, G. (2021). Is your OCR good enough? Probably so. Results from an assessment of the impact of OCR quality on downstream tasks. *KB Lab Blog*. https://lab.kb.nl/about-us/blog/your-ocr-good-enough-probably-so-results-assessment-impact-ocr-quality-downstream.

7. Dengel, A., Hoch, R., Hönes, F., Jäger, T., Malburg, M., Weigel, A. (1997) "Techniques for Improving OCR Results." In *Handbook of Character Recognition and Document Image Analysis*, 227–58. World Scientific.
8. Doush, I. Abu, A., Faisal, & Gharibeh, A. H. (2018). "Yarmouk Arabic OCR Dataset." In *2018 8th International Conference on Computer Science and Information Technology (CSIT)*, 150–54. IEEE.
9. Grant, P., Sebastian, R., Allassonnière-Tang, M., & Cosemans, S. (2021). Topic modelling on archive documents from the 1970s: global policies on refugees. *Digital Scholarship in the Humanities, March*.https://doi.org/10.1093/llc/fqab018
10. Gupta, A., Gutierrez-Osuna, R., Christy, M., Capitanu, B., Auvil, L., Grumbach, L., Furuta, R., & Mandell, L. (2015). "Automatic Assessment of OCR Quality in Historical Documents." In *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 29. 1.
11. Hamdi, A., Jean-Caurant, A., Sidere, N., Coustaty, M., & Doucet, A. (2019). "An Analysis of the Performance of Named Entity Recognition over OCRed Documents." In *2019 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, 333–34. IEEE. https://ieeexplore.ieee.org/document/8791217.
12. Hegghammer, T. (2021). Noisy OCR Dataset. *Repository details TBC*.
13. Holley, R. (2009). How Good Can It Get? Analysing and Improving OCR Accuracy in Large Scale Historic Newspaper Digitisation Programs. *D-Lib Magazine, 15*(3/4)
14. Jain, M., Mathew, M., & Jawahar, C. V. (2017). Unconstrained Scene Text and Video Text Recognition for Arabic Script. arXiv:1711.02396.
15. Journet, Nicholas, Visani, Muriel, Mansencal, Boris, Van-Cuong, Kieu, & Billy, Antoine. (2017). Doccreator: a new software for creating synthetic ground-truthed document images. *Journal of Imaging, 3*(4), 62.
16. Kanungo, T., Marton, G. A., & Bulbul, O. (1999). Performance Evaluation of Two Arabic OCR Products. In *27th AIPR Workshop: Advances in Computer-Assisted Recognition*, 3584:76–83. International Society for Optics; Photonics.
17. Kissos, I., & Dershowitz, N. (2016). "OCR Error Correction Using Character Correction and Feature-Based Word Classification." In *2016 12th IAPR Workshop on Document Analysis Systems (DAS)*, 198–203. IEEE.
18. Krishnan, R., & Babu, D. R. R. (2012). A Language Independent Characterization of Document Image Noise in Historical Scripts. *International Journal of Computer Applications, 50*(9), 11–18.
19. Lat, A., & Jawahar, C. V. (2018). "Enhancing Ocr Accuracy with Super Resolution." In *2018 24th International Conference on Pattern Recognition (ICPR)*, 3162–67. IEEE.
20. Levenshtein, V. I, and others. (1966). Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. In *Soviet Physics Doklady*, 10:707–10. 8. Soviet Union.
21. Lins, R. D., Banergee, S., & Thielo, M. (2010). "Automatically Detecting and Classifying Noises in Document Images." In *Proceedings of the 2010 ACM Symposium on Applied Computing*, 33–39.
22. Lopresti, D. (2009). Optical Character Recognition Errors and Their Effects on Natural Language Processing. *International Journal on Document Analysis and Recognition (IJDAR)* 12 (3): 141–51. http://www.cse.lehigh.edu/~lopresti/tmp/AND08journal.pdf.
23. Mariner, M. C. (2017). Optical Character Recognition (OCR). *Encyclopedia of Computer Science and Technology* (pp. 622–29). CRC Press.
24. Miller, D., Boisen, S., Schwartz, R., Stone, R., & Weischedel, R. (2000). "Named Entity Extraction from Noisy Input: Speech and OCR." In *Sixth Applied Natural Language Processing Conference*, 316–24. https://aclanthology.org/A00-1044.pdf.
25. Murata, M., Busagala, L. S. P., Ohyama, W., Wakabayashi, T., & Kimura, F. (2006). The Impact of OCR Accuracy and Feature Transformation on Automatic Text Classification. In *Document Analysis Systems VII*, edited by Horst Bunke and A. Lawrence Spitz, 506–17. Berlin, Heidelberg: Springer Berlin Heidelberg. https://link.springer.com/chapter/10.1007/11669487_45.
26. Mutuvi, S., Doucet, A., Odeo, M., & Jatowt, A. (2018). "Evaluating the Impact of OCR Errors on Topic Modeling." In *International Conference on Asian Digital Libraries*, 3–14. Springer.
27. Patel, C., Patel, A., & Patel, D. (2012). Optical character recognition by open source OCR tool Tesseract: a case study. *International Journal of Computer Applications, 55*(10), 50–56.
28. Reffle, U., & Ringlstetter, C. (2013). Unsupervised Profiling of OCRed Historical Documents. *Pattern Recognition, 46*(5), 1346–57.

29. Reul, C., Springmann, U., Wick, C., & Puppe, F. (2018). "Improving OCR Accuracy on Early Printed Books by Utilizing Cross Fold Training and Voting." In *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, 423–28. IEEE.
30. Rice, S. V, & Nartker, T. A. (1996). The ISRI Analytic Tools for OCR Evaluation. *UNLV/Information Science Research Institute, TR-96* 2.
31. Santos, E. A. (2019). "OCR Evaluation Tools for the 21st Century." In *Proceedings of the 3rd Workshop on the Use of Computational Methods in the Study of Endangered Languages Volume 1 (Papers)*, 23–27. Honolulu: Association for Computational Linguistics. https://www.aclweb.org/anthology/W19-6004.
32. Shen, Z., Zhang, R., Dell, M., Lee, B. C. G., Carlson, J., & Li, W. (Eds.). (2021). *LayoutParser: A Unified Toolkit for Deep Learning Based Document Image Analysis. arXiv Preprint*arXiv:2103.15348.
33. Springmann, U., Najock, D., Morgenroth, H., Schmid, H., Gotscharek, A., & Fink, F. (2014). "OCR of Historical Printings of Latin Texts: Problems, Prospects, Progress." In *Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage*, 71–75.
34. Stein, S. S., Argamon, S., & Frieder, O. (2006). "The Effect of OCR Errors on Stylistic Text Classification." In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 701–2. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.67.6791&rep=rep1&type=pdf.
35. Strohmaier, C. M, Ringlstetter, C., Schulz, K. U., & Mihov, S. (2003). Lexical Postcorrection of OCR-Results: The Web as a Dynamic Secondary Dictionary? In *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.*, 3:1133–33. Citeseer.
36. Su, J., Boydell, O., Greene, D., & Lynch, G. (2015). Topic Stability over Noisy Sources. *arXiv Preprint*arXiv:1508.01067. *https://noisy-text.github.io/2016/pdf/WNUT09.pdf.*
37. Tafti, A. P., Baghaie, A., Assefi, M., Arabnia, H. R., Zeyun, Y., & Peissig, P. (2016). "OCR as a Service: An Experimental Evaluation of Google Docs OCR, Tesseract, ABBYY FineReader, and Transym." In *International Symposium on Visual Computing*, 735–46. Springer.
38. tesseract-ocr. (2019). Tesseract OCR 4.1.1. GitHub Repository. *GitHub*.https://github.com/tesseract-ocr/tesseract.
39. Thompson, P., McNaught, J., & Ananiadou, S. (2015). "Customised OCR Correction for Historical Medical Text." In *2015 Digital Heritage*, 1:35–42. IEEE.
40. van Strien, D., Beelen, K., Ardanuy, M., Hosseini, Kasra, McGillivray, B., & Colavizza, G. (2020). "Assessing the Impact of OCR Quality on Downstream NLP Tasks." INSTICC; SciTePress. https://doi.org/10.5220/0009169004840496.
41. Vijayarani, S., & Sakila, A. (2015). Performance Comparison of OCR Tools. *International Journal of UbiComp (IJU), 6*(3), 19–30.
42. Volk, Martin, Furrer, Lenz, & Sennrich, Rico. (2011). Strategies for Reducing and Correcting OCR Errors. *Language Technology for Cultural Heritage* (pp. 3–22). Springer.
43. Walker, J., Fujii, Y., & Popat, A. C. (2018). "A Web-Based Ocr Service for Documents." In *Proceedings of the 13th IAPR International Workshop on Document Analysis Systems (DAS), Vienna, Austria*. Vol. 1.
44. Wemhoener, D., Yalniz, I. Z., & Manmatha, R. (2013). "Creating an Improved Version Using Noisy OCR from Multiple Editions." In *2013 12th International Conference on Document Analysis and Recognition*, 160–64. IEEE.
45. Wick, C., Reul, C., & Puppe, F. (2018). Comparison of OCR Accuracy on Early Printed Books Using the Open Source Engines Calamari and OCRopus. *J. Lang. Technol. Comput. Linguistics, 33*(1), 79–96.
46. Yalniz, I. Z., & Manmatha, R. (2011). "A Fast Alignment Scheme for Automatic OCR Evaluation of Books." In *2011 International Conference on Document Analysis and Recognition*, 754–58. https://doi.org/10.1109/ICDAR.2011.157.
47. Ye, Peng, & Doermann, David. (2013). "Document Image Quality Assessment: A Brief Survey." In *2013 12th International Conference on Document Analysis and Recognition*, 723–27. IEEE.