

Content analysis & anonymization made simple

Anssi Jääskeläinen, Tuomo Räisänen; South-Eastern Finland University of Applied Sciences / Digitalia research center, Mikkeli, Finland

Abstract

Imagine the world of possibilities where all information would be easily findable and usable without the burden of copyrights, privacy issues or sensitive information. We at the Digitalia research center at the South-Eastern Finland University of Applied Sciences have elaborated on this issue further. Copyright issues we of course cannot affect due to legislation, but in our recent paper [4], we discussed the hidden information that could be unleashed by using a simple ocr with content analysis. Now we will go further by enhancing the content detection and including an anonymization in our service repertoire. We will also focus on making these services more easily usable by the interested community with the development of REST API.

Motivation

The amount of globally produced information has already reached Zettabyte level and will continue to grow [2]. Even though most of the produced digital information is strictly speaking just “rubbish”, data growth is inevitable. Therefore, a justified question should arise: How to identify the important information from the bulk?

In addition, a growing proportion of such data is both born digital and created by social communities that go beyond geographical and organizational borders. The collections are established through API-based strategies and to a certain extent preserved by collecting institutions. Yet, the contextuality of data provides a major challenge. As the information is not static documents but created in interaction, there is no single party to manage data and decide what is valuable. Who is responsible for the appraisal process? It is also in the nature of online media that people “over-share”, in other words, provide information which would not be openly shared in other public spheres. One example are lists where people answer to 30 personal questions about family matters – and even to a data scientist’s surprise, these answers are mixed in the data otherwise representing a non-harmful discussion topic. When there are 1,500,000 textual objects in the dataset, going through them manually is not an option.

While most of these services are owned by large commercial firms, their use has yet become an integral part of the information landscape and everyday life with large-scale economic, political and societal implications [3]. Having such importance, the data must be made re-usable e.g. for academic research purposes and at least partially preserved for later use. This calls for automated means to analyze the data and hide sensitive information.

It is already a great benefit to have information online in a machine-readable format, or at least in a format that a machine

can parse. Still, to be truly useful and findable the information should be equipped with adequate metadata, which should be based on thorough content analysis [4].

Digital information is also fragile, in a worst-case scenario change of just one byte can corrupt the whole file. Data storages become obsolete over time and the contained data needs to be curated, migrated and preserved again to remain discoverable, accessible and of value to its potential users [5]. Datasets can be lost through misfortune or accidentally discarded. In addition, many of them are kept in personal “archives” only. Sharing data may seem unlikely at the time of acquisition but this option should not be ruled out. For online-originated data, the major challenge is in “capturing enough content to provide meaning but also finding practical solutions to managing such large, diverse, and connected material” [5]. Today, the practices of managing such data are very much evolving. We believe there is a growing need for easy-to-use solutions supporting the re-use of data, considering the legal and ethical considerations.

Filling forms is a very common discomfort for users. If a user is e.g. registered into an online store and during the checkout is requested to fill the same information again it really annoys. The same is true with filling metadata. If a person is forced to fill a metadata for example for hundreds of documents, the most likely consequence will be frustration. An automated solution for this problem is a must-have functionality for every self-respecting institute that does some form of digital preservation. Human intervention is still needed to verify the quality of the algorithm generated metadata.

Sensitive and private information in the era of digital age is one of the biggest concerns. There is e.g. a Science magazine special issue about “The End of Privacy” which takes many perspectives towards this important issue¹. The well-known Paradox of Privacy refers to the gap between user attitudes and actual behavior. For instance, people who most appreciate information transparency and openness are less willing to be profiled by their online data.

Most of the readers are probably familiar with a problematic situation where a document or a piece of information should be shared with a community, but it just cannot be done legally due to some private or sensitive information that exists in the document. Sometimes this information might have been released despite sensitive aspects due to inattention or carelessness in manual labor. This might naturally lead to consequences such as compensation requirements or even being prosecuted for breaking the GDPR or other legislation. Manual anonymization is a possibility, but people have made mistakes and always will make mistakes, no matter what. Furthermore, anonymizing sensitive or private information manually is a very time consuming and repetitive task which further emphasizes the possibility for mistakes. From our opinion, a solution for this anonymization problem is needed. When publishing and

1 <http://science.sciencemag.org/content/347/6221>

manipulating such information it must fill e.g. GDPR lawful processing purposes [6]. This paper describes the developed workflow which automatically tries to anonymize documents based on predefined rules, wordlists and NER (Named Entity Recognition).

Finally, there is a general problem in accessing a demonstration, a proof of concept, etc. show-case solutions. For a demonstration and marketing purposes a web-based demonstration is an adequate solution, but no true government agency or enterprise feeds hundreds of documents into a web form one by one to be able to reach the target. Therefore, more sophisticated way of using the services is required. We intend to release our codes in GitHub so anyone can extend the solution to suit their own specific requirements. Finally, a platform independent access via REST API for the produced solutions will be offered for testing purposes. Currently, the API is still under development, but it will be up and running before the conference.

Approach

Digitalia - Research Center on Digital Information Management has been working on finding a solution for every problem behind the presented motivations. This chapter thoroughly presents the solutions with complete descriptions of the utilized applications. Everything is done without a single commercial product, by using Python with suitable extension libraries and open source applications.

Codes are released on our GitHub repository². This makes the presented solutions easily adoptable and modifiable by anyone who possess enough expertise.

Content analysis and metadata generation

Automated metadata generation is based on our existing content analysis approach which has been gradually extended since the last Archiving conference. Functionalities such as better language detection as well as better word class analysis have been introduced to the workflow. Sensitive and private information detection now utilizes better NER engine and can be further enhanced with user defined word lists and stop words.

The technical back-end of the analyzer is based on multiple Python libraries such as nltk (Natural Language Toolkit), polyglot, Finer and libvoikko. The last two are special libraries developed just for the Finnish language. The following list introduces the phases done during the content analysis.

1. File type is recognized by the file extension, but the recognition is secured with either the latest DROID or by Linux file command. If neither is available, the file type extension recognition is trusted.
2. The character encoding of the file is detected with chardet python library.
3. Language(s) and a confidence percentage(s) are checked with polyglot python library, which currently supports 196 languages out of the box. If polyglot fails, langid is used instead.
4. The content of the file is checked word by word and total words and unique words are calculated.
5. The base form of the word is resolved by using either finer or nltk.
6. The class of the word is resolved with polyglot or finer.

7. The stop words are recognized and marked based on multilingual stop word lists.
8. Named entities are recognized and tagged with polyglot or finer.
9. The class hierarchy of a word is resolved by using local install of the Finnish thesaurus and ontology which supports Finnish, English and Swedish.
10. A word cloud is drawn from the meaningful words with python wordcloud library.
11. Known related terms for the word are resolved by using the same installation as ninth step.
12. The top keywords are calculated based on the occurrences of the base word and word class.
13. A human readable report is generated, and the found metadata is embedded into the file.

Furthermore, if the analyzed content includes photographs or drawings, also these will be automatically analyzed by using our newest development which is based on Tensorflow and default Inception-v3. This work is still under development, but results will hopefully be demonstrated during the conference. Figure 1 demonstrates the enhanced report which is automatically generated by the script.

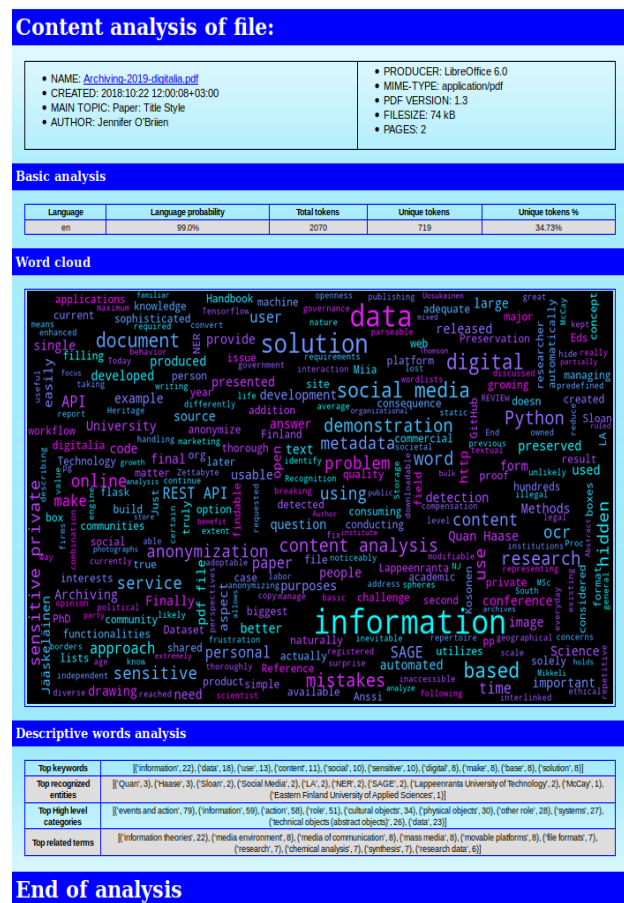


Figure 1. Enhanced human readable report

2 <https://github.com/Digitalia-Xamk>

Anonymization

Private, sensitive or security related information are the main reasons for keeping something out from the public internet. Even a single PII (Personally Identifiable Information) or a suspicion of such information inside a document makes it difficult to publish it online. One way to solve this issue is via anonymization, but it must be kept in mind that e.g. by combining information from different data sources an anonymized information can quite easily be re-identified [1].

In generally, there are at least three different approaches that can be followed when conducting anonymization. The first one would be to separate the text from the document and anonymize the plain text. Then the outlook of the original file should be re-created somehow. The second approach is based on drawing rectangles above the detected sensitive, private or otherwise to be hidden words. The third alternative would be to remove the text directly from the underlying bit stream without breaking the internal structure of the PDF file which is known to be very delicate.

Our current workflow is based on the solution number two which is drawing rectangles over the detected sensitive content. However, drawing just a box over a PDF file doesn't remove the text or make it inaccessible. The text behind the box can still be copy-pasted and therefore revealed. That is why our solution first draws the required rectangles over the detected words and then saves the complete page as an image. Then the new image is ocr'd again. With this approach, the hidden words become truly hidden and clearly marked. As a side effect the quality of the ocr might be reduced slightly and the file size will most likely increase. Still, during our large-scale tests, we have had multiple cases where the quality of the ocr has increased noticeably, instead of decreasing. This might be due to fact that the content we have been processing has been relatively old and ocr'd more than ten years ago. Secondly, we have been processing material without existing ocr information, where the ocr quality naturally increases. Formal ocr quality evaluation is not conducted thus it is not the point of this work to create statistically or scientifically valid evidences. The following list introduces the phases done during the anonymization.

1. PDF file language is recognized to be able to use appropriate detection methods.
2. Named entities are recognized with either polyglot or Finer.
3. PDF content is extracted and parsed with pdfminer python extension to get the exact x and y location of every single character and object inside the PDF file.
4. Extracted content is compared against NER data, word lists and user defined data to find the coordinates of the objects to be hidden.
5. Reportlab pdfgen is used to draw masking canvas over each PDF page with appropriate places hidden with a colored rectangle.
6. Python Pdfewriter is used to save the newly created masked PDF file.
7. PDF file is converted into png file with imagemagick
8. PNG file is re-ocr'd with tesseract.

Figure 2 presents the last page of the abstract we submitted for evaluation that has been run through the anonymizer. The

final output file is a validated archival graded PDF/A-3b file produced by Ghostscript. Text behind the black rectangles was identified by NER, text behind the red rectangles with the combination of Finnish first names and last names list and the text behind the blue rectangles was identified by a user defined list containing the words, Linux, Python and Github.

As it can be seen, the detection is not perfect, e.g. polyglot NER tends to identify the word "the" as a person if written with capital t, and some names were not detected at all. Something like this must, however, be expected when using general purpose NER program such as polyglot. If a language specific NER, such as Finer for the Finnish language or Stanford NER for the English language would be used, the accuracy would probably be much higher. As stated, we will be moving into more accurate NER tools later, but for now the accuracy is enough. Furthermore, machine made recognition will never be perfect, so any truly sensitive, private or security related information should stay that way despite any anonymization actions taken.

or carelessness in manual labor. This might naturally lead to consequences such as compensation requirements or even being prosecuted for breaking the GDPR or some other legislation.

People have always made mistakes and always will make mistakes, no matter what. Furthermore, anonymizing sensitive or private information is a very time consuming and repetitive task which further emphasizes the possibility for mistakes. Our opinion, also a solution for this anonymization problem is needed.

Finally there is a general problem in accessing demonstration, proof of concept, etc. show-case solutions. For a demonstration and marketing purposes a web based demonstration is an adequate solution, but no true government agency or enterprise feeds hundreds of documents into a web form one by one to be able to reach the target. Therefore more sophisticated way of using the services is required. Our solution is to build a REST API via which all of our services will be available.

Approach

We have a solution for every presented problem, which will be thoroughly presented in a final paper. As in all of our previous projects, everything is done without a single commercial product, by using code is released on our repository. This makes our solutions easily adoptable and modifiable by anyone who possess enough expertise. Implemented code and libraries that can be used to extend the functionality of basic

Automated metadata generation is based on our existing content analysis approach which has been greatly extended since the last conference. Functionalities such as better language detection and handling as well as better word class analysis has been introduced to the workflow. Sensitive and private information detection now utilizes better NER (Named Entity Recognition) engine and can be further enhanced with word lists. Finally, if submitted content contains photographs or drawings, also their content is automatically analyzed by using our newest development which is based on Tensorflow and default Inception-v3. Later on, we might start further training Inception with our own reference images.

For conducting anonymization there are at least two different approaches. First one would be to anonymize the plain text and re-create the outlook of the file afterwards. Second approach, which we are following, is based on drawing boxes above the detected sensitive, private or otherwise to be hidden words. As we should all know, this isn't solely enough. Just drawing a box over a pdf file doesn't actually remove or make the text inaccessible. Text behind the box can still be copy-pasted and revealed. That is why our solution first draws boxes over the detected words and then saves the complete page as an image. Then the new image is ocr'd again. This approach, the hidden words become truly hidden. As a side effect the quality of the ocr might reduce a bit though we have had multiple cases where also the quality of the ocr has actually increased noticeably.

We have an online demonstration site² where it is possible to test some of the developed functionalities. However, a demonstration site cannot be used for production purposes and therefore we will be implementing a REST API. At the time of

² <https://digitalia.xamk.fi/>

writing the API is still under construction and it will be build by using code with a Flask microframework³. API address and a more thorough description will be available when the final paper is written.

Results

As a result of running the scripts presented here, a final outcome will be either a pdf file with sensitive and private parts anonymized or a pdf file with embedded content analysis based metadata and a report. For the API part, it is too soon to write anything yet. If these solutions are used via demonstration site, a downloadable file is provided.

This paper demonstrates that conducting content analysis or anonymization doesn't have to be difficult or time consuming and can be done solely with open source products. Naturally we realize that algorithms make mistakes but after those mistakes have been identified those are much simpler to fix than human behavior which currently is the biggest source of mistakes. Our solution allows archivists and researchers to convert digital data into a shareable form, therefore advancing data openness and further use.

References

- [1] Personal Archiving: Preserving Our Digital Heritage (Metford, NJ, 2019)
- [2] L. McCay-Peet & Quan-Haase, What is social media and what questions can social media research help us answer? In L. & Quan-Haase (Eds.) SAGE Handbook of Social Media Research Methods, pp. 13-26. (SAGE, LA, 2017)
- [3] L. & OCR: Unleash the hidden information. Proc. Archiving 2018, pp. 63. (2019).
- [4] Data Storage, Curation and Preservation. In L. & Quan-Haase (Eds.) SAGE Handbook of Social Media Research Methods, pp. 161-176. (SAGE, LA, 2017)

Author Biography

has an IT MSc. (2005) and a PhD (2011) from Lappeenranta University of Technology. has an extensive knowledge of user experience and usability. His current interests are in information governance, format migration and development. holds a PhD (2008) from Lappeenranta University of Technology. She is an experienced researcher and trainer in knowledge management, open communities, social technologies, and social media. Her current interests are in the field of digital communication research and the related methods.

³ <http://flask.pocoo.org/>

Figure 2. Anonymized PDF page

Further research

Inspired by the feedback from the evaluators we decided to do some experiments with the solution number three. We browsed the Python libraries and encountered a Python package called pdf-redactor³. Usage of this software is based on the predefined find and replace pairs of strings. It has several limitations since the PDF format is an incredible complex data standard that has hundreds of capabilities. Most PDFs only embed the font information for characters used in the document.

³ <https://github.com/JoshData/pdf-redactor>

Since the redaction works by performing text substitution, your replacement text may contain characters that were not previously in the PDF. The workaround for this problem is to check the replacement text for new characters and replace them with some common characters like spaces.

When working directly with the bit stream, the workflow is simpler. This approach also has some computational benefits against the solution number 2, as extra ocr is avoided. For this paper, we didn't have enough time to measure the differences but those will be presented on a poster. The basic steps are as follows:

1. File language is recognized.
2. PDF content is extracted and parsed with pdfcrowd python package i.e. input stream is created.
3. Extracted content is compared against NER data, word lists and user defined data to find the objects to be replaced with the predefined strings.
4. Output stream is created and the file is generated using pdfcrowd.
5. Finally, the produced PDF file is converted into archival graded PDF/A-3b format with Ghostscript.

Despite its limitations, our first tests show that pdf-redactor seems to work quite decently. Figure 3 shows the author biography section of a research paper anonymized with this method. Author names and the word python is anonymized.

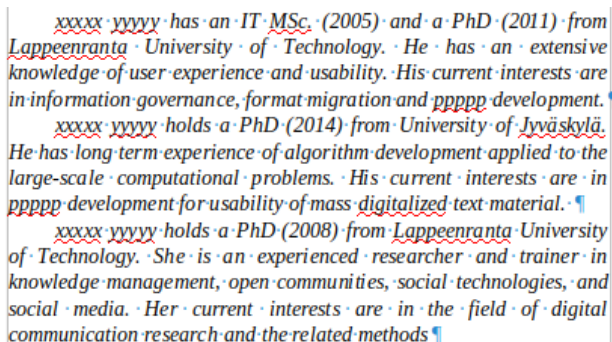


Figure 3. Text anonymized with xx pdf-redactor

We will continue testing this library and more thorough analysis will be presented on a poster.

API

For production purposes and API is one of the aspects that should be considered. When submitting the abstract we already knew that we will be creating a REST API. A new and shiny docker based API solution with a sophisticated management interface is under development by our hired consultants and will be released before the conference. Please contact us directly if you wish to get a test access to the API. Meanwhile, we also have an online demonstration site⁴ available, where it is possible to test some of the developed functionalities.

Results

As a result of running the scripts presented here, a final outcome will be either a PDF file with sensitive and private parts anonymized or a PDF file with embedded content analysis based metadata and a report. For the API part, it is too soon to write anything yet. If these solutions are used via demonstration site, a downloadable file is provided.

This paper demonstrates that conducting content analysis or anonymization doesn't have to be difficult or time consuming and can be done solely with open source products. Naturally we realize that algorithms make mistakes but after those mistakes have been identified those are much simpler to fix than human behavior which currently is the biggest source of mistakes. Our solution allows archivists and researchers to convert digital data into a re-shareable form, therefore advancing data openness and further use.

References

- [1] A. Narayanan, V. Shmatikov. Myth and fallacies of "Personally Identifiable Information". Communications of the ACM, 53, 6. Pp. 24-26, (2010)
- [2] Donald Hawkins, Personal Archiving: Preserving Our Digital Heritage (Medford, NJ, 2013)
- [3] L. McCay-Peet, A. Quan-Haase, What is social media and what questions can social media research help us answer? In L. Sloan & A. Quan-Haase (Eds), The SAGE Handbook of Social Media Research Methods, pp. 13-26. (SAGE, LA, 2017)
- [4] A. Jääskeläinen, L. Uosukainen. OCR: Unleash the hidden information, Proc. Archiving 2018, pg. 83. (2018).
- [5] A. Voss, I. Lvov, S.D. Thomson, Data Storage, Curation and Preservation. In L. Sloan & A. Quan-Haase (Eds), The SAGE Handbook of Social Media Research Methods, pp. 161-176. (SAGE, LA, 2017)
- [6] GDPR, Chapter II, Article 6, available. <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex%3A32016R0679>

Author Biography

Anssi Jääskeläinen has an IT MSc. (2005) and a PhD (2011) from the Lappeenranta University of Technology. He has an extensive knowledge of user experience and usability. His current interests are in information governance, format migration and Python development.

Tuomo Räisänen holds a PhD (2014) from University of Jyväskylä. He has long term experience of algorithm development applied to the large-scale computational problems. His current interests are in Python development for the usability of mass of digitized text material.

4 <https://digitalia.xamk.fi/>