

Docreader labeling system for line type classifier

Ilya S. Kozlov¹

¹*Ivannikov Institute for System Programming of the RAS, Alexander Solzhenitsyn st. 25, Moscow, 109004, Russian Federation*

Abstract

We develop the document analysis system, which is able to extract text and text metadata (such as font size and style), and restore the document structure. Some parts of the pipeline are based on machine learning thus requiring training and the labeled dataset, creating a training dataset is based on manual labeling. In this article, we describe an approach to the creation of a labeling system in the task of multiclass classification of document lines (paragraphs). The pipeline consists of several stages ranged from getting the source documents to getting a ready-to-learn dataset. An approach to the analysis of scanned documents and documents in docx and txt format is considered. In our work, we focus on intra-team labeling, thus we do not consider some problems, common for the crowdsourcing approach (such as unscrupulous annotators).

Keywords

document structure analysis, PDF documents, document analysis,

1. Introduction

As a rule, large documents are not uniform but split into smaller parts. The scientific articles are divided into sections, the novels divided into chapters, etc. The larger parts, in their turn, are divided into smaller parts, as subsections or paragraphs. Thus the document can be represented in the form of the tree.

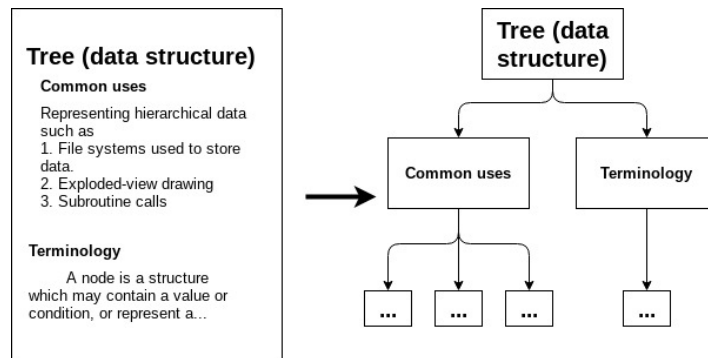


Figure 1: Document: how the reader sees it and in the form of a tree


Information Technologies: Algorithms, Models, Systems (ITAMS), September 14, 2021, Irkutsk

✉ kozlov-ilya@ispras.ru (I. S. Kozlov)

🆔 0000-0002-0145-1159 (I. S. Kozlov)



© 2021 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

The physical representation of this logical structure is the Table Of Contents, it serves to facilitate the navigation in the large texts and also can be hierarchical. Thus in this article we can consider the task of logical structure extraction and TOC extraction as synonymous.

Most of the methods of automatic TOC extraction are based on supervised machine learning techniques, thus requiring some labeled dataset. One of the ways of obtaining the labeled dataset is to ask human experts to label the data. These humans are called annotators. One can see main pipeline (in solid lines) and alternative way where feature extraction and classification preformed by the annotator (in dotted lines) in figure 2.

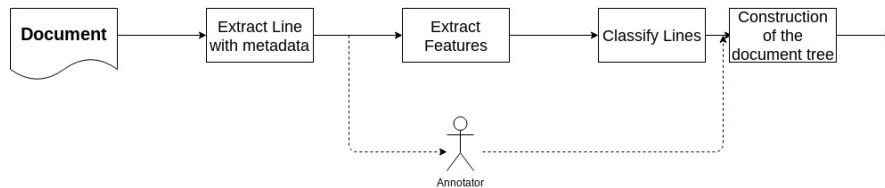


Figure 2: Document: line classification pipeline

The creation of the labeled dataset is not a one-time task:

- Each new type of documents has its own type of logical structure, thus it requires its own labeled dataset. For example, we can work with scientific articles but want to work also with financial documents
- The existing labeled dataset does not contain some important subcategory of documents. For example, the scientific articles dataset does not contain any biological articles. In this case, the quality of the logical structure extraction from biological articles probably will be low, and the best way to improve it is the expansion of the collection of labeled documents.
- Generally, the enlargement of the training set is the simplest and effective way to improve the quality of any supervised machine learning model.

This paper is organized as follows:

- Section 2 describes related work gave a short description of the current state of the task of TOC extraction and the existing approaches to the creation of the training dataset.
- Section 3 describes our approach to the creation of the dataset and the way to solve the problem, which emerges in the process of the creation of a labeled dataset for the task of TOC extraction.

2. Related Work

2.1. Logical Structure Extraction

One of the early surveys, considering the logical structure extraction task is [1], most of considered methods are rule-based. In the survey most of concepts are defined, such as tree structure

of the document.

In the 2008–2013 the series of TOC extraction competitions from fiction books [2, 3, 4, 5] were held. The competition is based on two complementary metrics: a title-based measure and a link-based measure. The rule-based and machine learning-based solutions were proposed.

In the 2019–2021 the series FinTOC competitions were held – TOC extraction competitions from financial reports [6, 7, 8]. There were two tasks – TOC extraction and title detection. The difference is that in the title detection participants were asked to classify each line of the document as "Title" or "Not Title". There were two datasets – English and French documents, thus there were potentially more than one winners. Let's briefly list the winner's approaches:

- The best solution [9] for title detection in the FinTOC 2019 was based on the LSTM.
- The best solution [10] for the TOC extraction task in the FinTOC 2019 was based on the decision tree classifier.
- The Best Title Detection for English in FinTOC 2020 was based on the neural networks [11].
- The Best TOC extraction for both English and French was based on Random Forest classifier [12].
- At the moment of writing the paper, the winner of the FinTOC-2021 has not published the solution yet.

The approach, used in the Docreader project is based on the XGBoost classifier [13] and described in [14].

We can conclude that most modern approaches to TOC extraction are based on machine learning methods and, accordingly, require a labeled dataset.

2.2. Labeling Systems

Depending on the specific labeling problem one can rely on crowdsourcing or do the job in-team. Each approach has advantages and disadvantages. Crowdsourcing allows the creation of large datasets with the assistance of external (and often working for the little money) annotators. One can use Amazon Mechanical Turk¹ or Yandex Toloka² as a crowdsourcing platform. The disadvantages of the crowdsourcing approach are includes:

- It is impossible to give data containing state or commercial secrets to outsourcing.
- Some annotators may work unscrupulous, some external quality control is required.
- You would be limited with the platform restrictions.

You may find more information about crowdsourcing in the [15, 16]

In the case of in-team labeling the work is performed by team members, who often work for much more fee then the crowdsourcers. On the other hand in the case of in-team labeling, some

¹<https://www.mturk.com/>

²<https://toloka.yandex.ru>

problems are not actual. Usually one may not worry about unscrupulous annotators, it is easier to organize labeling of the secret documents. There are many tools for labeling, one of the most powerful tools is a Labeling Studio[17] Label Studio has rich functionality, it is suitable for segmentation, object detection, image classification, etc. We have use Label Studio but face a relatively long waiting time for image updates (about 1 second)³. It is not a problem when we do segmentation tasks, because it takes much more than one second for one image. But in the case of image classification, the waiting time may be more than task completion time, so we use much more simple program ImageClassifier⁴.

3. Proposed Method

The text line classification pipeline is organized as follows (fig 2).

1. Extraction of the text lines with metadata (font size and style, indents, etc) from the document.
2. Extraction of the features from lines with metadata
3. Classification of the lines by their type
4. Construction of the document structure

We need labeled data to train the classifier 3 (and sometimes the feature extractor 2).

As a rule, the result of a labeling task are pairs of features X and the label y But the way how we extract lines with metadata and how we extract features may change, and we don't want to redo the data labeling task every time when any step of the pipeline is changed.

3.1. Persistent line id

We add special persistent id for each line with metadata, which is not changed during the change of our pipeline. Thus we are able to change our pipeline (for example feature extraction) and do not need to relabel the training dataset. The way how to build the persistent id is different for different kinds of documents.

Scanned documents: Scanned documents in fact are images, so line detection is a separate task, and we use Tesseract⁵ for this purpose. The change of the Tesseract version may lead to the change in the text line location algorithm and even to the change of number and the order of lines. In order to avoid the need to perform the labeling task each time when we change the Tesseract version, we save the found lines in the form of the bounding box and the extracted text.

³as of the beginning of 2019

⁴<https://github.com/dronperminov/ImageClassifier>

⁵<https://github.com/tesseract-ocr/tesseract>

- не устанавливать в технические средства ИС программное обеспечение, зараженное вирусами.

Нарушение настоящих требований влечет за собою гражданско-правовую и иную ответственность, предусмотренную действующим законодательством.

13. Требования к стандартизации, унификации и к гарантийному сроку

13.1. Средства защиты информации должны соответствовать соглашениям и стандартам, в части работы с клавиатурой, отображения информации на экране, вызова справочной информации, организации пользовательского интерфейса и т.п.

13.2. Требования к гарантийному сроку и (или) объему предоставления гарантий качества средств защиты информации.

Объем предоставления гарантий качества средств защиты информации – 100%.

Подрядчик гарантирует качество поставляемых средств защиты информации в полном соответствии с требованиями действующего законодательства Российской Федерации.

Гарантийный срок на компакт диски не менее 12 месяцев.

Гарантийный срок на средства защиты информации не менее 12 месяцев.

Директор

И.Ю. Кривошеева.

Figure 3: Line with bounding box (red frame)

In the future, we use the found bboxes as a result of the work of the Tesseract.

Txt document: Line in txt document is defined by the document itself and the number of the line. We define the line id as **md5sum(document) + "_" + line_id**

Docx: Docx is the Microsoft Word format. It represents by a zip archive with files in XML format. One of the files consists of paragraphs, each paragraph contains text and some meta information in the explicit form or as a reference to some style. Paragraphs located in file document.xml, styles defined in the style.xml file, the archive can hold other files also. The paragraph is defined by the XML which produces it and by the docx file itself. We define the line id as **md5sum(document) + "_" + md5sum(paragraph_xml)**

3.2. Creating tasks for annotators

We hope that the one who is interested in labeled data should maximally simplify the process for the annotators. The annotator should not install the strange software with the cumbersome installation instructions, the annotator should have direct access to the instruction. In our case, we reduce the task to the classification of the image. Annotator gets a zip archive with images of the bounding box (as in picture 3), the annotator instructions, docker file with all the dependencies. The task can be launched with two shell commands (in case if you have docker installed), after launching one can use a web browser to perform the tasks. We use ImageClassifier to create web interfaces for the labeling task. The annotator labels the document line by line, after the first line following the second line, and so on, so the context is holding. After finishing the labeling task annotator gets the archive with the results and is able to upload it into the tasks server. When all annotators have uploaded their tasks, task server merges the

answers and adds original documents. As a result, we obtain the collection of documents and labeled pairs line id and the label. The line id should be the same for each run of the pipeline, we describe how to build such id in the subsection 3.1.

3.2.1. Creating tasks images

We have to create images with a bounding box around the text line. The way how to create is different for the different kinds of the documents.

Scanned documents: A scanned document is a picture, we have the coordinates of the line from the Tesseract and have saved it. Thus the bounding box can be drawn with the help of the OpenCV [18] of the PIL [19] library.

Txt documents: Txt document containing only text lines without the metadata (such as font size or font style). Thus one may draw text with some image processing library and do not fear losing some important meta information.

Docx documents: Docx document is the most difficult one to obtain the image. Typically the docx document contains a lot of valuable information about the text formatting, so we do not want to draw the document as raw text and lose all the metainformation. On the other hand, the process of the drawings of the docx document is complicated, so only some large libraries are able to do it. The solution may be found in the modification of the internal XML. One may add information to the paragraph that should be concluded into the box. To obtain the coordinates of the box we tried to convert the modified document and the original one into images and subtract the second image from the first one, but note that the drawing of the box leads to the shift of the paragraphs. We also note that the box of the neighborhood paragraphs may be merged if both boxes have the same color. All the above forces us to use more complicated ways of creating images with bounding boxes for docx:

1. Create a pair of documents with the boxes. Each box in the first document have a unique color and the color of the box in the second document alternates.
2. Convert pair of the docx documents to the pair of pdfs and pdfs into list of images. See picture 4
3. We subtract the second image from the first one and obtain the image with only nonzero pixels in the former box.



Figure 4: Pair of images with boxes

4. Summary

The task of creating a training dataset occurs regularly in the process of machine learning-based system development. We describe our approach to the creation of the labeled dataset. We have described our approach to the creation of the labeled dataset, describe an approach that enables us to not redo data annotation with every change of our documents handling pipeline, described how to lead the task of line annotation to the task of image classification. We hope that the need for human labeling data is not one time task, but a regularly occurring problem, so the machine learning systems should enable to create such tasks easily.

References

- [1] S. Mao, A. Rosenfeld, T. Kanungo, Document structure analysis algorithms: a literature survey, in: Document Recognition and Retrieval X, volume 5010, International Society for Optics and Photonics, 2003, pp. 197–207.
- [2] G. Kazai, A. Doucet, M. Landoni, Overview of the inex 2008 book track, in: International Workshop of the Initiative for the Evaluation of XML Retrieval, Springer, 2008, pp. 106–123.
- [3] G. Kazai, A. Doucet, M. Koolen, M. Landoni, Overview of the inex 2009 book track, in: International Workshop of the Initiative for the Evaluation of XML Retrieval, Springer, 2009, pp. 145–159.
- [4] A. Doucet, G. Kazai, B. Dresevic, A. Uzelac, B. Radakovic, N. Todic, Setting up a competition framework for the evaluation of structure extraction from ocr-ed books, International Journal on Document Analysis and Recognition (IJ DAR) 14 (2011) 45–52.
- [5] A. Doucet, G. Kazai, S. Colutto, G. Mühlberger, Icdar 2013 competition on book structure extraction, in: 2013 12th International Conference on Document Analysis and Recognition, IEEE, 2013, pp. 1438–1443.
- [6] R. Juge, I. Bentabet, S. Ferradans, The fintoc-2019 shared task: Financial document structure extraction, in: Proceedings of the Second Financial Narrative Processing Workshop (FNP 2019), 2019, pp. 51–57.
- [7] N.-I. Bentabet, R. Juge, I. El Maarouf, V. Moulleron, D. Valsamou-Stanislawski, M. El-Haj,

The financial document structure extraction shared task (fintoc 2020), in: Proceedings of the 1st Joint Workshop on Financial Narrative Processing and MultiLing Financial Summarisation, 2020, pp. 13–22.

- [8] I. El Maarouf, J. Kang, A. Aitazzi, S. Bellato, M. Gan, M. El-Haj, The Financial Document Structure Extraction Shared Task (FinToc 2021), in: The Third Financial Narrative Processing Workshop (FNP 2021), Lancaster, UK, 2021.
- [9] K. Tian, Z. J. Peng, Finance document extraction using data augmentation and attention, in: Proceedings of the Second Financial Narrative Processing Workshop (FNP 2019), 2019, pp. 1–4.
- [10] E. Giguet, G. Lejeune, Daniel@ fintoc-2019 shared task: toc extraction and title detection, in: Proceedings of the Second Financial Narrative Processing Workshop (FNP 2019), 2019, pp. 63–68.
- [11] D. Premi, A. Badugu, H. Sharad Bhatt, AMEX-AI-LABS: Investigating transfer learning for title detection in table of contents generation, in: Proceedings of the 1st Joint Workshop on Financial Narrative Processing and MultiLing Financial Summarisation, COLING, Barcelona, Spain (Online), 2020, pp. 153–157. URL: <https://www.aclweb.org/anthology/2020.fnp-1.26>.
- [12] D. Kosmajac, S. Taylor, M. Saeidi, DNLP@FinTOC’20: Table of contents detection in financial documents, in: Proceedings of the 1st Joint Workshop on Financial Narrative Processing and MultiLing Financial Summarisation, COLING, Barcelona, Spain (Online), 2020, pp. 169–173. URL: <https://www.aclweb.org/anthology/2020.fnp-1.29>.
- [13] T. Chen, C. Guestrin, XGBoost: A scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’16, ACM, New York, NY, USA, 2016, pp. 785–794. URL: <http://doi.acm.org/10.1145/2939672.2939785>. doi:10.1145/2939672.2939785.
- [14] A. O. Bogatenkova, I. S. Kozlov, O. V. Belyaeva, A. I. Perminov, Logical structure extraction from scanned documents, Proceedings of the Institute for System Programming of the RAS 32 (2020) 175–188.
- [15] R. Gilyazev, D. Y. Turdakov, Active learning and crowdsourcing: A survey of optimization methods for data labeling, Programming and Computer Software 44 (2018) 476–491.
- [16] A. Drutsa, V. Farafonova, V. Fedorova, O. Megorskaya, E. Zerminova, O. Zhilinskaya, Practice of efficient data collection via crowdsourcing at large-scale, arXiv preprint arXiv:1912.04444 (2019).
- [17] M. Tkachenko, M. Malyuk, N. Shevchenko, A. Holmanyuk, N. Liubimov, Label Studio: Data labeling software, 2020-2021. URL: <https://github.com/heartexlabs/label-studio>, open source software available from <https://github.com/heartexlabs/label-studio>.
- [18] G. Bradski, The OpenCV Library, Dr. Dobb’s Journal of Software Tools (2000).
- [19] P. Umesh, Image processing in python, CSI Communications 23 (2012).