Structure Extractor: Multilingual Extraction of Sections from Scientific Document

Ilia Kopanichuk
Antiplagiat
Moscow, Russia
kopanichuk@ap-team.ru

Artem Chashchin
Antiplagiat
Moscow, Russia
chashchin@ap-team.ru

Inga Ochneva
Antiplagiat
Moscow, Russia
ochneva@ap-team.ru

Andrey Grabovoy
Antiplagiat
Moscow, Russia
Moscow Institute of Physics and Technology
Moscow, Russia
grabovoy@ap-team.ru

Aleksandr Ogaltsov
Antiplagiat
Moscow, Russia
ogaltsov@ap-team.ru

Aleksandr Kildyakov
Antiplagiat
Moscow, Russia
kildyakov@ap-team.ru

Yury Chekhovich
Antiplagiat
Moscow, Russia
chehovich@ap-team.ru

***Abstract*—A scientific article usually has a good structure. The structure helps to guide both readers and journal editors. It also allows differentiated assessment of text reuse occurring in different sections of the article. Considering the wide use of plagiarism detectors in scientific practice, the task of automatic structure extraction from scientific articles becomes relevant in the plagiarism detection process. Most of the published articles and theses consist of the following sections: title, contents, introduction, methods, results and discussion, conclusions, bibliography, and appendices. In this paper we present a method to extract the structure of the scientific documents. Our solution processes formatted documents (pdf, doc, docx), extracts the text layer and the layout from them and outputs the borders of the aforementioned sections within the text layer. To identify section borders we use histogram-based gradient boosting trees. Some of the detected sections, namely introduction, methods, results and discussion, comprise the well-known IMRAD organizational structure of documents. Our solution is multilingual and can be scaled to support more languages by an unsupervised approach. We are also presenting a new custom dataset that consists of 73 documents with labeled sections in 30 languages. The solution achieves 0.87 average precision and 0.75 average recall per section on the dataset. The developed approach is used to determine the structure of articles in the production environment. It processes more than 55 pages per second on 1 CPU and is very helpful in tasks like table extraction, annotation extraction and machine generated text detection.**

I. Introduction

A. Motivation

Most of today's scientific articles are subject to a rigid narrative structure. The most common structure among scientific documents is IMRAD: introduction, methods, results, and discussion [1]. IMRAD plays an important role in different tasks such as summarization and plagiarism detection. In an era of digital information overload, the reuse of scientific documents has become an important aspect of scientific research. Text reuse search engines, which aim to identify text reuse in a large corpus of scientific documents, are essential tools for scientific research and innovation. The evaluation of detected borrowings in different parts of the article or dissertation should be done differently. For example, borrowing in the introduction, literature review, or methods is unlikely to be malicious. Whereas borrowing in the results, and even more so in the conclusions, is much more likely to be malicious. Models like [2], that generate queries for the search engines, would greatly benefit from automatic exclusion of less relevant parts of the text and inclusion of the parts where plagiarism is more likely to occur.

The multilingual nature of scientific research poses another challenge to text reuse search engines. While scientific articles are often written in English, borrowing ideas or reusing the text from literature in other languages is possible [3]–[6]. In this case the engines need resource-efficient and time-efficient solutions to detect such cases and to possibly decrease the amount of text that needs to be checked. Detecting different sections of the articles and only comparing some of them might help with the aforementioned challenges.

AI-generated text detection is another area where IMRAD is helpful [7], [8]. Differentiated approach to the sections allows to spend computing capacities on the most relevant parts of the text.

In this paper, we propose a novel approach to identify sections of multilingual scientific documents that can improve the performance of plagiarism detectors. It relies on histogram-based gradient boosting as well as regex-based filtering. There are two main classification models: the former identifies title and contents sections of a document, while the latter extracts the headers for the remaining sections. Our proposed approach has significant implications for research and innovation. By improving the performance of text reuse search engines, it allows researchers to identify and reuse scientific information more effectively, leading to accelerated scientific progress. In addition, our approach can facilitate the dissemination of scientific knowledge across languages and cultures, fostering international collaboration and interdisciplinary research. The time efficiency of the solution allows to use it in production.

B. Related work

Extracting structure from documents has been a relevant topic for several decades now. In [9], for example, the authors propose a method to extract preface, set of sections and references by analyzing connectives, idiomatic expressions and other linguistic clues. Their solution also summarizes the important parts of the document and thus generates an abstract.

The general problem of structure extraction has a promising approach in [10] where a deep learning framework is proposed to identify and classify document sections. The main network architectures used in the article are convolutional neural networks and recurrent neural networks. Firstly, each text line of the document is classified as either a header or a regular text. Then, another model classifies the headers on belonging to sections, sub-sections or sub-subsections. Finally, an algorithm is applied to detect the boundaries of all the sections, subsections and sub-subsections.

Some of the models are able to solve the structure extraction problem partially. For example, GROBID (GeneRation Of BIbliographic Data) library allows to extract headers (without matching them with the document sections) and references of a PDF document [11]. The authors employ deep learning models and CRF for this task. The output is a structured XML file containing extensive information about the document. Besides the main text, GROBID also extracts additional information sections (annex) that includes acknowledgments, funding information, appendices etc. It is also possible to retrieve page coordinates for the headers from the main text and annexes, as well as from general text [12]. Bibliography section is fully processed in the output XML file with the document names, authors, publication year and other relevant information extracted from raw text.

If names of the sections are known or can be obtained from the document, one could map them to the IMRAD structure. In [13], the authors extract an XML schema for the documents and, based on that information and features like in-text citation count, figure counts and table counts, map the section headings to IMRAD headings. Their method, however, does not achieve average precision higher than 22% on the three datasets from the experiments.

Document structure extraction can also be applied to documents of different nature and with a different structure. For example, in [14] a rule-based method is applied to detect the headings of French newspapers. In [15], BERT and Bi-LSTM models are applied to analyze the structure of privacy policies. And in [16] the language models are used to explore the structure of ad-buy forms and registration forms. Structre extraction is also applied to web pages because extracting structured text fields helps in many cases. For example, the authors of [17] use transformer for that task.

In some works a sentence-wise classification for IMRAD sections is applied. For example, [18] propose to classify each sentence separately as belonging to one or another section. A similar idea is echoed in [19] where authors apply various machine learning algorithms to map sentences from the abstracts to the corresponding sections they describe.

While, to our knowledge, no solutions based on large language models were proposed to extract document sections, there are LLM-based approaches to similar tasks. For example, in [20] the authors extract structured information from materials science and engineering texts with the help of fine-tuned GPT-3 and Llama-2. The tasks are linking dopants and host materials, cataloging metal-organic frameworks and general composition/phase/morphology/application information extraction. The models show favorable results in extracting complex structures from the document text.

Extraction of the document structure can also be used in keyword extraction [21], [22] where such model could be used to fully automate the process without the need to manually copy parts of the documents. Automating the process of extracting structural features of the document might also be useful in abstract generation models like [23] where an extractive summarization system is proposed to produce abstracts for articles in Turkish. The system identifies the most relevant sentences from each document section and produces an abstract according to the IMRAD structure. The results show that the readability of such structured abstracts is on par with or even better than the original abstracts.

A solution to a similar sub-problem of detecting a bibliography section was previously proposed by [24] where a regular-expression-based method that works with plain text is described. The main advantage of such neural-network-free method is small inference time which allows for its use in production and high load systems.

II. Methods

A. Problem statement

Let us formalize the structure extraction problem. Consider the dataset

$$\mathcal{D} = \{d_i, \mathbf{y}_i\}_{i=1}^{m},$$

where d_i is a document, which is described by the text t_i, the token sequence x_j^i and the layout $\mathbf{m}_j^i$ from the document metadata:

$$d_i = \{t_i, \left(x_1^i, \mathbf{m}_1^i\right), \ldots, \left(x_n^i, \mathbf{m}_n^i\right)\}, \quad \mathbf{y}_i = y_1^i, \ldots, y_n^i.$$

The problem is to build a mapping: $f : D \to Y$, where D is a document space and Y is a space of token label sequences, that would maximize the symbol-wise F_1 measure.

B. Data

In this study we have collected and labelled 3 datasets:

1) IMRAD Dataset: 3132 documents in 36 languages. It contains marked up IMRAD parts and headers.
2) DocStructure Dataset: 840 documents in 36 languages. This collection contains marked up title, contents, bibliography, appendix and headers.
3) DocFullStructure Dataset: 73 documents in 30 languages. It has all the sections marked up: title, contents,

introduction, methods, results, conclusion, bibliography, appendices

Datasets have been split into train, validation and test according to the following ratios:

1) 0.7 : 0.2 : 0.1 IMRAD
2) 0.8 : 0.1 : 0.1 DocStructure
3) 0 : 0 : 1 DocFullStructure

DocFullStructure is our final test dataset, and we open it for public use. This resource will allow to compare different structure extraction models and address the current lack of publicly released multi-language datasets for this problem setting. It is sourced from open-access sources and contains 73 scientific documents (articles and theses) in 30 languages with the following codes: bg, ca, cs, da, de, el, en, es, fi, fr, hr, hu, it, jp, kk, ko, ky, lt, mk, nl, no, pl, pt, ro, ru, sl, sr, tr, uk, zh. The main goal of using documents in different languages is testing how well our solution performs on documents with different templates and document structures.

The documents are collected from various online repositories. For each of the 30 languages 10 documents are found. Then they are randomly sampled with a pre-defined sample size for every language. For each document we extract the text layer and the page layout data. Section intervals are then annotated by the authors of this article.

The dataset is available in the supplementary materials. Included are document texts, layout data, annotated structure and source links.

Table I shows the information about the dataset. For each section there are average fractions of the text belonging to it as well as the number of documents having this section and the number of languages they are written in. For most of the languages there are documents with all the sections. The distributions of documents by size and page count are presented in Figure 1.

TABLE I. DocFullStructure dataset

Section	Avg text fraction	Docs	Langs
Title	< 0.01	68	29
Contents	0.04	69	30
Introduction	0.12	69	28
Methods	0.34	71	30
Results	0.26	67	29
Conclusions	0.04	69	30
Bibliography	0.09	70	30
Appendix	0.06	44	27

C. Experiment pipeline

The full pipeline of our solution to the problem of document sections identification is presented in Fig. 2. We obtain tokens and layout of each page of the document by DevExpress PDF API [25]. The token positions and their text content are then combined into boxes (i.e. parts of page that contain text) according to their relevance to the lines of the document. Geometric parameters (width, heigth, position, font family) and text content of boxes are combined into 45-dimensional

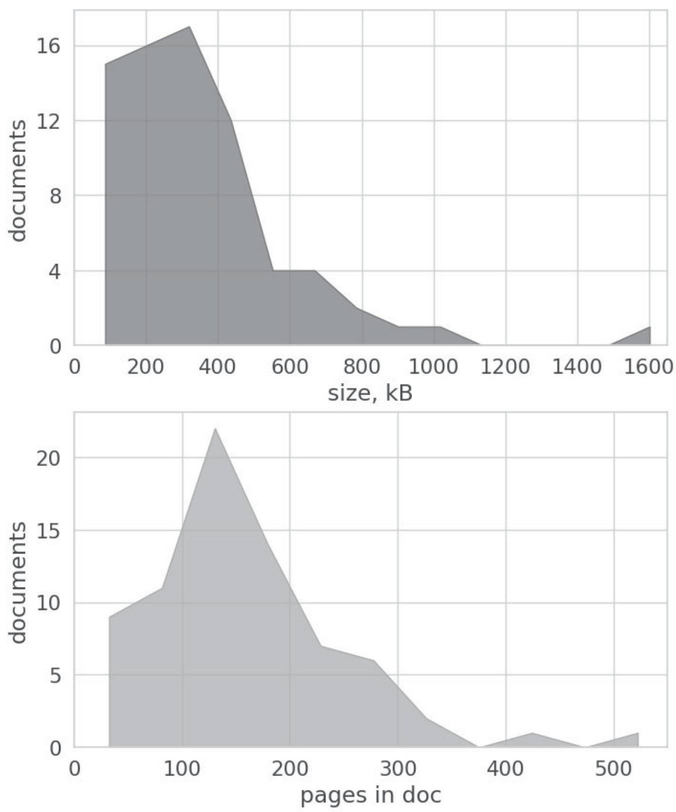

Fig. 1. Distribution of size and number of pages for documents in DocFullStructure Dataset

embeddings and used further as features for classification models. Another 26-dimensional feature space is generated for each page of the document.

After getting the features we use two classifiers to predict the sections in the text. The first model finds the title and contents of the document and leaves the remaining sections to be distinguished by the second model.

We train HistGradientBoosting [26] model (that we call TitleCont Selector) on page features from DocStructure dataset to predict whether a page is a title page, contents or another section from the document. The parameters for our TitleCont Selector model are shown in Table II.

TABLE II. TitleCont Selector model parameters

Name	Value
class weight	'balanced'
learning rate	0.126
max depth	8
max leaf nodes	37
min samples leaf	16
contents threshold	0.95
title threshold	0.75

With these parameters we reach precision of 0.92, recall of 0.89 for title detection and precision of 0.92, recall of 0.80 for contents detection. The reported results are achieved on DocStructure test dataset.

All the other sections are extracted in the following way. Another HistGradientBoosting model named Candidate Selector is trained on box features from IMRAD Dataset to predict if the box is a candidate for any section header. Candidate Selector parameters are shown in Table III:

TABLE III. CANDIDATE SELECTOR MODEL PARAMETERS

Name	Value
class weight	'balanced'
learning rate	0.172
max depth	12
max leaf nodes	46
min samples leaf	14
threshold	0.18

With these parameters we get precision of 0.05 and recall of 0.98 on IMRAD test dataset. Then we classify the candidates into headers based on the presence of the section keywords in them. The boxes after the section header are classified as the corresponding section until the header of the next section or until the limit of headers belonging to the section is exceeded. This limit for candidates is individual for each section and is in the range of [1, 40].

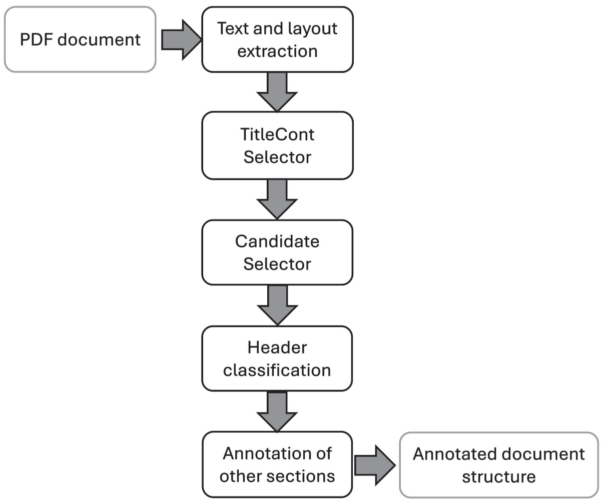

Fig. 2. Full solution pipeline

We tune both models using hyperopt package [27] with the negative F-measure as the optimization target. We consider F_1 for the TitleCont Selector and F_{30} for the Candidate Selector models. Due to a huge imbalance in candidate boxes distribution we remove all the pages without any candidates from IMRAD train dataset.

For comparison, we have decided to use GROBID library as one of the very few open-source solutions. Since it does not have a fully implemented means of extracting section boundaries from a document, we plug it into our pipeline. The GROBID model used for experiments employs deep learning models and CRF. The resulting "hybrid" model takes XML file produced by GROBID, extracts bibliography text, headers for the main text, headers for the annexes and coordinates for each text part, matches the extracted data with its positions in text and uses it to extract the section boundaries instead of Candidate Selector.

First, title and contents sections are extracted with the help of TitleCont Selector. Then, to get text positions of headers, we look for all the <head> elements in XML file, take their coordinates (page number and coordinates on the page) and match them with the text in the text layer. Bibliography text does not have any <head> elements, therefore a similar procedure is performed on the children of <div type="references"> element: the text coordinates are extracted and the text intervals are obtained. These text intervals for bibliography serve as the new pipeline's prediction of bibliography section.

After getting the text intervals of headers and bibliography we replace the output of Candidate Selector model with the headers from GROBID. Like in our original pipeline, they are then classified as document sections with the help of keywords. The annotation of the remaining text is done in the same way: the boxes between the headers belong to an earlier header's section. The aforementioned bibliography intervals are added to the final result.

Such a "hybrid" pipeline allows to compare the performance of a model that plays a key role in detecting IMRAD structure with another open-source solution. Integration of GROBID helps to get the same output as the original pipeline, so that the same metrics for both approaches could be calculated.

III. RESULTS

TABLE IV. FINAL METRICS ON DOCFULLSTRUCTURE DATASET

Section	Precision	Recall	F1
Title	0.97	0.88	0.92
Contents	0.99	0.84	0.91
Introduction	0.77	0.75	0.76
Methods	0.75	0.52	0.62
Results	0.81	0.26	0.40
Conclusion	0.85	0.82	0.83
Bibliography	0.88	0.95	0.91
Appendices	0.92	0.94	0.93

TABLE V. PERFORMANCE OF "HYBRID" PIPELINE WITH GROBID ON DOCFULLSTRUCTURE DATASET

Section	Precision	Recall	F1
Introduction	0.34	0.39	0.36
Methods	0.13	0.09	0.10
Results	0.08	0.08	0.08
Conclusion	0.09	0.11	0.10
Bibliography	0.79	0.32	0.46
Appendices	0.54	0.53	0.54

As presented in Table IV, our solution works well for title, contents, conclusion, bibliography and appendices as we achive F_1 higher than or equal to 0.83. With sections like conclusion and appendices, the models attain similar results for all the three metrics: precision, recall and F_1. With some other sections like title, contents and bibliography, the good

TABLE VI. TIME METRICS ON DOCFULLSTRUCTURE DATASET. 2500 MHZ CPU, NO GPU

	Avg	Min	Max
Time, s/doc	2.6	0.1	10.4
Size, pages/doc	155	8	547

TABLE VII. F_1 MEASURE BY LANGUAGE

F1 range	Languages
0.4-0.6	sr, nl
0.6-0.7	de, ro, pl, uk, es, no
0.7-0.8	ko, pt, en, zh, mk, cs, bg, ca, sl, fr, da
0.8-0.9	jp, it, kk, hr, ru, tr, el, hu
0.9-1.0	ky, fi, lt

performance is attained due to high precision (0.97 for title, 0.99 for contents) or high recall (0.95 for bibliography).

For introduction pages precision and recall have slighly decreased in value (0.77 and 0.75 respectively), and for methods the recall has dropped to 0.52. The performance on the results section is the lowest, with both recall and F_1 dropping below 0.5.

The reason for the performance decline lies in the complex nature of the concept of methods section. While other sections in the main part of the document often have fixed names like "Introduction" and it is relatively easy to find such names in the text, the section of methods is frequently called anything but "Methods". For example, sections (or chapters) of theses might contain the names like "Brief overview of <topic name>", or the section titles might be highly correlated with the title of the thesis. Screening headline candidates for the full variety of possible keywords results in a higher false positive rate.

Another source of sections misclassification arises from processing of Candidate Selector results and subsequent classification of headers. If a header in the document does not belong to the set of our sections (because it's a different section or as a result of incorrect prediction), its corresponding text might get attibuted to the previous section. For example, if the header for methods is not found, its content might get classified as introduction. On the other hand, the same classification approach can result in insufficient labeling of text when a section contains many subsections with headers. The model is just going to attribute the subsections to this section until the header limit is reached. Overall, these two sides of the issue should be balanced with a proper selection of hyperparameters, so that the model achieves good quality and both misclassification and insufficient labeling are minimized. The performance of our pipeline indicates that we are moving in the right direction, and further tuning might enhance the quality even more.

Our approach, however, surpasses the GROBID pipeline for every section (Table V). While the "hybrid" model shows moderate performance on the sections of introduction, bibliography and appendices, the sections of methods, results and conclusion have all the three metrics below 0.13. Among the reasons of the good performance on bibliography is its parsing by GROBID. That allows for a more precise prediction.

We have also calculated the speed of document processing with our solution. Table VI shows that the average processing time is around 2.6 seconds per document, with the average document length being 155 pages. Even for a huge document with 547 pages it outputs the result in less than 11 seconds with the average speed near 60 pages per second on a one 2500 MHz CPU without any GPU support or multithreading.

Despite the fact that our solution is multilingual, its quality differs quite significantly on different languages. Not all the scientific documents follow the typical IMRAD structure or use the common names for the sections. For example, some documents in DocFullStructure dataset have "Inleiding" and "Conclusie" (Dutch for introduction and conclusion) as a title of several sections: the introduction and conclusion for the whole text and then additional introductory and final subsections in other sections. Since the requirements for scientific documents (e.g. theses and articles) vary from country to country, this is going to affect the solution's performance on different languages. Table VII shows the F_1 range for documents written in different languages. The lowest values for F_1 (in the range of 0.4-0.6) are attained at Serbian and Dutch documents, while much better quality is attained on such languages as Russian, Kyrgyz, Finnish, Lithuanian and others. The Serbian documents follow the typical IMRAD structure, and the solution's performance can likely be improved by adding more data in Serbian language in the training sets and in keywords. The Dutch documents have a different ordering of the sections that is mentioned earlier: every method and result section has its own introductions and conclusions, therefore the solution fails to properly annotate them all with the limit of headers belonging to each section. More experiments on hyperparameter tuning might enhance the performance in such cases. Also, improving the list of all possible document sections for the solution and allowing for nested structure of the sections could help in this case.

While the presented solution to the problem of document structure extraction still allows for more improvements, the performance quality is very good for production, and improved future models can be tested on the released dataset. By releasing the dataset, we hope to encourage more research on this subject. One of the ways to significantly increase the detection quality is to use transformer-based models like LayoutLM [28]. This approach, however, has its downsides as it also increases the inference time and resource consumption. Adding more lightweight classifiers to select the pages that need to be processed with LayoutLM is one of the options that allows its potential use in production.

IV. CONCLUSION

By using histogram-based gradient boosting and unsupervised regex-based approach we produce a model that extracts sections from well-structured scientific documents. It achieves favorable results on most of the document sections while still maintaining moderate performance on challenging sections like methods. The application named Structure Extractor is created on the basis of this model pipeline, and due to its high performance speed it is possible to use the application in production. It is already embedded in our text reuse detection system which helped to improve the quality of other production applications: a detection of a machine generated text, annotation extraction, and table extraction services. The false positive rate of these services decreased by more than 2.5 times with the use of Structure Extractor. To mitigate potential adversarial attacks, Structure Extractor is used in ensemble with modules that detect specific sections of the documents, and postprocessing of the results is performed to correct the detection errors.

While the model produces quality results, there are a few limitations for it. Firstly, multilanguage settings are constrained by IMRAD keywords. While the approach isn't restricted to any specific language or group of languages and can be easily scaled, one needs to add language-specific keywords for the section headers to add the support of a new language. This process could potentially be automated with another model translating the most common keywords to different languages, but construction of the fully-automatic system is beyond the scope of this article.

Secondly, our solution shows a drop in quality on documents with non-standard section structure. If the names of the headers do not contain IMRAD keywords, there will be significant false negative and false positive rates for sections. This situation is less common in scientific documents except for the methods section, so generally they are affected very little by it. Possible solutions to scaling the model to any document type include extensive use of document layout information and visual information, but that would also require more resources and more computation time.

Future study on the topic includes further comparison of the model performance to other solutions with potential implementation of approaches proposed in other articles. Extracting the document sections with large language models would be an interesting task. While this approach certainly has its own drawbacks in terms of resources consumption and postprocessing of the LLM output, it could potentially help with speeding up the annotation process for document sections. Using other approaches in an ensemble with the current models can also help to improve the overall performance of the solution.

In terms of the performance improvement it will also be useful to further assess the differences between scientific document structure in different languages. Increasing lists of keywords and refining the filtering procedure of headers can allow for a better extraction of sections like results and methods.

Another direction of research is detection of page headers and footers and integration of such detector into the solution pipeline. While some headers and footers contain only page numbers and do not interfere with the header extraction process, others repeat the names of the sections or provide bibliographical references. That disrupts the general "flow" of the section causing them to get misclassified. Having a separate section entity for headers and footers could help a lot with filtering false positive cases and improving performance on other structure-related tasks.

References

[1] L. Sollaci and M. Pereira, "The introduction, methods, results, and discussion (imrad) structure: a fifty-year survey," *Journal of the Medical Library Association: JMLA*, vol. 92,3, pp. 364–7, 2004.

[2] D. Shodiev, I. Kopanichuk, A. Chashchin, A. Grabovoy, A. Kildyakov, and Y. Chekhovich, "Ensembling models for the generation of queries to an altering search engine using reinforcement learning," in *2023 Ivannikov Ispras Open Conference (ISPRAS)*. IEEE, 2023, pp. 144–149.

[3] A. Asvarov and A. Grabovoy, "The impact of multilinguality and tokenization on statistical machine translation," in *2024 35th Conference of Open Innovations Association (FRUCT)*. IEEE, 2024, pp. 149–157.

[4] K. Avetisyan, G. Gritsay, and A. Grabovoy, "Cross-lingual plagiarism detection: Two are better than one," *Programming and Computer Software*, vol. 49, no. 4, pp. 346–354, 2023.

[5] O. Bakhteev, Y. Chekhovich, A. Grabovoy, G. Gorbachev, T. Gorlenko, K. Grashchenkov, A. Ivakhnenko, A. Kildyakov, A. Khazov, V. Komarnitsky *et al.*, "Cross-language plagiarism detection: A case study of european languages academic works," in *Academic Integrity: Broadening Practices, Technologies, and the Role of Students: Proceedings from the European Conference on Academic Integrity and Plagiarism 2021*. Springer, 2023, pp. 143–161.

[6] K. Grashchenkov, A. Grabovoy, and I. Khabutdinov, "A method of multilingual summarization for scientific documents," in *2022 Ivannikov Ispras Open Conference (ISPRAS)*. IEEE, 2022, pp. 24–30.

[7] G. Gritsai, I. Khabutdinov, and A. Grabovoy, "Multi-head span-based detector for ai-generated fragments in scientific papers," *arXiv preprint arXiv:2411.07343*, 2024.

[8] G. Gritsai, A. Voznyuk, A. Grabovoy, and Y. Chekhovich, "Are ai detectors good enough? a survey on quality of datasets with machine-generated texts," *arXiv preprint arXiv:2410.14677*, 2024.

[9] K. Sumita, K. Ono, and S. Miike, "Document structure extraction for interactive document retrieval systems," in *Proceedings of the 11th annual international conference on Systems documentation*, 1993, pp. 301–310.

[10] M. M. Rahman and T. Finin, "Unfolding the structure of a document using deep learning," *arXiv preprint arXiv:1910.03678*, 2019.

[11] "Grobid," https://github.com/kermitt2/grobid, 2008–2025.

[12] GROBID. Annotation guidelines for the 'segmentation' model. [Online]. Available: https://grobid.readthedocs.io/en/latest/training/segmentation/

[13] I. Ahmed and M. T. Afzal, "A systematic approach to map the research articles' sections to imrad," *IEEE Access*, vol. 8, pp. 129 359–129 371, 2020.

[14] N. Gutehrlé and I. Atanassova, "Processing the structure of documents: Logical layout analysis of historical newspapers in french," *Journal of Data Mining and Digital Humanities*, no. Digital humanities in languages, 2022.

[15] S. Liu, F. Zhang, B. Zhao, R. Guo, T. Chen, and M. Zhang, "Appcorp: a corpus for android privacy policy document structure analysis," *Frontiers of Computer Science*, vol. 17, no. 3, p. 173320, 2023.

[16] Z. Wang, Y. Zhou, W. Wei, C.-Y. Lee, and S. Tata, "A benchmark for structured extractions from complex documents," *arXiv preprint arXiv:2211.15421*, 2022.

[17] Q. Wang, Y. Fang, A. Ravula, F. Feng, X. Quan, and D. Liu, "Webformer: The web-page transformer for structure information extraction," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 3124–3133.

[18] H. Houngbo and R. E. Mercer, "An automated method to build a corpus of rhetorically-classified sentences in biomedical texts," in *Proceedings of the first workshop on argumentation mining*, 2014, pp. 19–23.

[19] S. Ribeiro, J. Yao, and D. A. Rezende, "Discovering imrad structure with different classifiers," in *2018 IEEE International Conference on Big Knowledge (ICBK)*. IEEE, 2018, pp. 200–204.

[20] J. Dagdelen, A. Dunn, S. Lee, N. Walker, A. S. Rosen, G. Ceder, K. A. Persson, and A. Jain, "Structured information extraction from scientific text with large language models," *Nature Communications*, vol. 15, no. 1, p. 1418, 2024.

[21] C. Zhang, L. Zhao, M. Zhao, and Y. Zhang, "Enhancing keyphrase extraction from academic articles with their reference information," *Scientometrics*, vol. 127, no. 2, pp. 703–731, 2022.

[22] M. Danilevsky, C. Wang, N. Desai, J. Guo, and J. Han, "Kert: Automatic extraction and ranking of topical keyphrases from content-representative document titles," *arXiv preprint arXiv:1306.0271*, 2013.

[23] A. E. Özkan Çelik and U. Al, "Structured abstract generator (sag) model: analysis of imrad structure of articles and its effect on extractive summarization," *International Journal on Digital Libraries*, pp. 1–15, 2024.

[24] A. Ogaltsov, "Language-free regular expression search of document's references," in *Recent Trends in Analysis of Images, Social Networks and Texts*, E. Burnaev, Ed. Cham: Springer International Publishing, 2022, pp. 45–54.

[25] DevExpress, ".net ui controls: components for developers of mobile, desktop, web, bi reporting apps," 2023. [Online]. Available: https://www.devexpress.com/

[26] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf

[27] J. Bergstra, D. Yamins, and D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *Proceedings of the 30th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, S. Dasgupta and D. McAllester, Eds., vol. 28,1. Atlanta, Georgia, USA: PMLR, 17–19 Jun 2013, pp. 115–123. [Online]. Available: https://proceedings.mlr.press/v28/bergstra13.html

[28] Y. Xu, M. Li, L. Cui, S. Huang, F. Wei, and M. Zhou, "LayoutLM: Pre-training of text and layout for document image understanding," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, aug 2020. [Online]. Available: https://doi.org/10.1145%2F3394486.3403172