

Header Extraction from Scientific Documents

Kevin Yao

Advisors: Mario Lipinski, Bela Gipp, Jim Pitman

1. Introduction

1.1 Context

With the massive amount of published material becoming accessible to the public via the World Wide Web, a tool that can parse header information from research papers will be invaluable to systems concerned with storing and retrieving scientific publications. Given certain search criteria and metadata, we would like to have some way of finding and identifying a document that matches our aforementioned criteria. A simple way of identifying an article is by header information (Fig 1), e.g. title, authors, affiliation, published date, etc. Thus, our goal is to put together a tool that can efficiently and accurately extract header data from research papers.

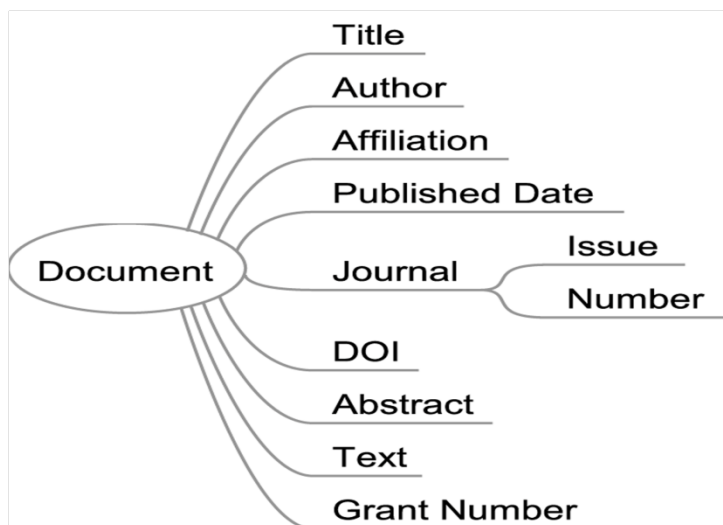


Fig. 1 Relevant header information

Additionally, natural language processing and various encoding issues add to the difficulty of accurately parsing header data.

1.2 Overview of Challenges

As with all text mining programs, the key is to output results reliably. Although, there appears to be a general guideline for how one should write research papers, not everyone conforms to the same format. This, in turn, is the essential problem to writing a parser tool. We must be able to infer header information from an implicit structure instead of looking for key words such as "Title" or "Author," because the truth is, no one labels their header sections.

1.3 Our Plan

To develop a tool that can parse header information not only accurately, but also efficiently, is a daunting task. Therefore, our plan is to survey the existing tools and then use these tools for our own implementation, building on top of and combining them if necessary. This report serves to highlight our current development and our outlook.

2. Methods& Tools

2.1 Overview of Concepts

To be able to parse information from scientific documents, one must be familiar with certain concepts in use today. We know, from looking at the available tools that are engaged with header extraction, that there are three fundamental approaches. The most widely distributed format of research papers is the Adobe Portable Document Format (pdf). Hence, one possible approach to the problem of header extraction is to conduct pattern matching via regular expressions, font size and structural matching after converting the pdf file to an accessible format such as txt or xml. Another popular method is to use a knowledge base, such as an existing online service like Google Scholar, to act as a cross-reference to your document. For instance, one may extract small amount of information from the document and query Google Scholar to fill in the remainder of header contents needed. Lastly, the final approach is the use of machine learning techniques, which is a more practical approach compared to the first approach. Current implementations with machine learning techniques use support vector machines (SVM) [1], hidden Markov models (HMM) [2] and conditional random fields (CRF) [3] to extract data logically. Naturally, one might also be interested in combining all of these approaches and merging the results.

2.2 Available Tools

There are many groups working on data extraction from scientific documents, including many groups that are developing tools for citation extractions. However, for our purpose of header extraction, the number of existing tools is fairly limited. Through a near-exhaustive search on the web, we discovered a small number of tools, most of which are not fully developed. Furthermore, to use the tools implemented by other organizations, the best option for us would be to use open-source tools, only then will we have complete control and knowledge of the tools. These requirements narrowed our choices to just a handful - notably, half of which are new endeavors, suggesting that there is a growing interest and need in the area of header extraction.

3. My Contribution

3.1 Comparison of the Tools

After researching and finding the parser tools online, I initially compiled a table of all the current available tools and examined their top-level behaviors - a simplified version (Fig 2) is available for viewing.

Name of Tool	Approach	Link	Open-Source?
Mendeley Desktop	SVM & web-based look-up	http://www.mendeley.com/	No
pdfssa4met	pdf2xml	code.google.com/p/pdfssa4met	Yes
HeaderParser Service	SVM	http://sourceforge.net/projects/citeseerx/	Yes
ParsCit	CRF	http://aye.comp.nus.edu.sg/parsCit/	Yes
GROBID - online	CRF	http://grobid.no-ip.org/ http://sourceforge.net/projects/grobid	Yes
pdfmeat	Queries Google Scholar, pdftotext	http://code.google.com/p/pdfmeat/	Yes
Zotero	Queries Google Scholar	http://www.zotero.org/	Yes
Paperpile	Web-based look-up	http://paperpile.com/beta/	Yes

Fig 2 Initial examination of existing tools along with their respective websites

As one can see, these tools are split between using machine learning techniques and querying a knowledge base. After this initial examination and resource-gathering, the next step was to test how well these tools performed on real research papers by conducting an informal test and a side-by-side comparison of the results (an excerpt available for viewing in Fig 3).

Name of Tool	Paper 1	Paper 2	Paper 3
Mendeley Desktop	Title, Authors, Journal, Year, Pages, URL, DOI	Title, Authors, Journal, Year, Issue, Pages, URL, DOI	Title, Authors, Journal, Year, Volume, Pages, Abstract, Keywords, URL, PMID
pdfssa4met	Title, Authors	FAILED	Title
ParsCit	Title, Authors, Affiliation, Email, Abstract	Title, Authors, Affiliation, Address, Email, Abstract	Title, Authors, Address, Email, Abstract,
HeaderParserService	Title, Authors, Affiliation, Address, Email, Abstract	Title, Authors, Affiliation, Address, Email, Abstract	Title, Authors, Affiliation, Address, Email, Abstract
GROBID - online	Title, Authors, Department, Institution, Address, Email, Abstract	Abstract	Title, Authors, Department, Institution, Address, Abstract
pdfmeat	Title, Authors, Year, Publisher, URL, Abstract	Title, Authors, ISSN, Year, Publisher, URL, Abstract	Title, Authors, Journal, Year, Volume, Number, Pages, ISSN, Publisher, Abstract, URL
Zotero	Title, Authors, Date	Title, Authors, Date, Pages	Title, Authors, Publication, Volume, Issue, Pages, Date, ISSN
Paperpile	Title, Authors, Abstract, Journal, Year	Title, Authors, Abstract, Journal, Pages, DOI, ISSN	Title, Authors, Journal, Abstract, Volume, Pages, Year, Month, ISSN, Pubmed ID

Fig 3 Performance of parser tools (showing three out of ten papers tested), green = correct, red = incorrect

The test was informal in the sense that the papers used were hand-picked (not entirely random) specifically due to their different styles. However, there is really no categorization for different styles of research papers, so the procedure as to how one might formally test the performance of tools is a difficult task. Fortunately, the purpose of this test is to get a general grasp of how these tools are different and to see if a hybrid approach of integrating these tools is a viable option. As one can see from Figure 3, we will definitely be able to benefit from integration of the tools because there is not a single tool that is perfect, so it will be to our advantage if we were to use multiple tools to fill in one another's deficiencies.

3.2 Integration of the Tools

Integration will not be a trivial task because the tools are all written in different languages and not all the tools can be run from the command line. Furthermore, we have no algorithm for merging the results of different tools. Thus, all we have done so far for this project is the designing of our future parser tool and the creation of the framework in Java. For this initial endeavor, we included four tools: pdfssa4met, ParsCit, HeaderParserTool, and pdfmeat. These four were chosen because they can be executed from the command line, which greatly eases our implementation of the Java program. The framework provides us with a way to call all the different tools it knows and for each of these tools, generate a uniformly formatted result set. This result set is implemented with a Java class that contains instance variables for each of our header entities: title, author, affiliation, published date, journal title, journal issue, journal number, doi, and the abstract. If an entity can have multiple entries, such as when a document has multiple authors, that instance variable will be implemented using a Java HashSet. Hence, with this setup for the result set, we will be able to access, retrieve, store and compare data using a much more systematic manner than if we had to deal with all the different raw output formats created by the different tools. This gives us the clear advantage of not needing to worry about each tool's individual implementations and let us focus on just the data outputted. Furthermore, to ensure that we do not lose all the auxiliary information when creating a result set from raw output, the result set will contain a HashMap that can store any other information we wish to extract from the raw output. One more advantage that arises is that a uniform result set allows us to merge outputs of different tools, so that we may benefit from filling in gaps using multiple tools. However, that is not to say that integrating result sets will be a trivial task, for there are many small nuisances one must consider in deciding which tool generated the "correct" data.

4. Conclusion

As this project is very early in its development, there are still many tasks we must accomplish before we have a precise and efficient header parser. Mainly, we must find ways to generate results from all the other relevant parser tools and figure out the best way to merge result sets. We hope that along with our current development, the tools that we are using will also improve, so that we will be able to push our precision closer and closer to 100 percent.

5. Acknowledgments

I would like to thank my advisors for giving me this opportunity to work on a part of their project and for aiding me with the resources necessary to accomplish my tasks. Also, I would like to acknowledge the effort put into the open source tools we have used for this project. Without the current development in header extraction, this project would not have been possible.

6. References

- [1] Hui Han, C. Lee Giles, Eren Manavoglu, Hongyuan Zha, Zhenyue Zhang, and Edward A. Fox. 2003. Automatic document metadata extraction using support vector machines. In *Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital libraries (JCDL '03)*. IEEE Computer Society, Washington, DC, USA, 37-48.
- [2] Erik Hetzner. 2008. A simple method for citation metadata extraction using hidden markov models. In *Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries (JCDL '08)*. ACM, New York, NY, USA, 280-284. DOI=10.1145/1378889.1378937 <http://doi.acm.org/10.1145/1378889.1378937>
- [3] Fuchun Peng and Andrew McCallum. 2006. Information extraction from research papers using conditional random fields. *Inf. Process. Manage.* 42, 4 (July 2006), 963-979. DOI=10.1016/j.ipm.2005.09.002 <http://dx.doi.org/10.1016/j.ipm.2005.09.002>