PDFdigest: an Adaptable Layout-Aware PDF-to-XML Textual Content Extractor for Scientific Articles

Daniel Ferrés*, **Horacio Saggion***, **Francesco Ronzano**†, **and Àlex Bravo***
* Large Scale Text Understanding Systems Lab
TALN Research Group, DTIC Department
Universitat Pompeu Fabra, Barcelona (Spain)
{daniel.ferres,horacio.saggion,alex.bravo}@upf.edu
† Integrative Biomedical Informatics Group (IBI)
Research Programme on Biomedical Informatics (GRIB)
IMIM-UPF, Barcelona (Spain)
francesco.ronzano@upf.edu

Abstract

The availability of automated approaches and tools to extract structured textual content from PDF articles is essential to enable scientific text mining. This paper describes and evaluates the PDFdigest tool, a PDF-to-XML textual content extraction system specially designed to extract scientific articles' headings and logical structure (title, authors, abstract,...) and its textual content. The extractor deals with both text-based and image-based PDF articles using custom rule-based algorithms implemented with existing state-of-the-art open-source tools for both PDF-to-HTML conversion and image-based PDF Optical Character Recognition.

Keywords: Language Resources, Scientific Text Mining, Digital Libraries, Information Extraction, PDF Conversion

1. Introduction

Nowadays automated approaches to mine scientific literature are essential to support researchers - as well as any other interested actor - in taking full advantage of the huge number of articles available online: (Ware and Mabe, 2015) estimate that more than 2.5 million papers are published on the Web every year, and the percentage of publications distributed as Open Access content is constantly increasing. Even if XML-based formats are emerging, the majority of scientific articles are still accessed as PDF files. As a consequence, effective tools to extract structured textual content from PDF files represent a key technology to enable scientific text mining (Ronzano and Saggion, 2016). Such tools are essential components to develop a varied range of applications in different academic and industrial contexts, thus supporting the implementation of new, intelligent patterns to access scientific information (Accuosto et al., 2017). Once textual contents are extracted from scientific publications, they can be mined and enriched using existing Natural Language Processing (NLP) techniques such as Named Entity Recognition and Classification, Parsing, Relation Extraction, among others, thus helping researchers to improve their access to the scientific knowledge.

In this paper we present PDFdigest, a tool that extracts structured textual contents from scientific articles in PDF format. PDFdigest performs the conversion of files from PDF to XML format while preserving both the textual content and the layout details of the input PDF document.

The contributions of this paper can be summarized as follows:

- PDFdigest, a PDF to XML system that extracts the content of both text-based and image-based PDF files[1] by:
  – dealing with specific paper layouts as well as with text in multiple languages;
  – storing the layout of the original PDF in an HTML file, thus enabling the possibility of Visual Analytics over the original PDF document;
  – implementing a customizable approach to adapt the content extraction process to different PDF paper layouts by manually changing the extraction offsets and thresholds.
- An evaluation of the effectiveness of the text-based and image-based PDF textual content extraction algorithms with papers from a bilingual (English/Spanish) journal in the Natural Language Processing field (i.e. the SEPLN journal).

The rest of the paper is organized as follows: Section 2 provides an overview of the related work. Section 3 describes the main features of PDFdigest. The textual content extraction approach for text-based PDF files is described in Section 4. Section 5 introduces the approach to process image-based PDF files. Then, in Section 6 we evaluate the text-based and the image-based PDF extraction algorithms. In Section 7 we describe the online web-service demo. Finally, we discuss the key PDF mining advantages provided by PDFdigest (Section 8) and formulate our conclusions and future work plans (Sections 9 and 10 respectively).

2. Related work

During the last few years, several PDF-to-text extraction tools have been proposed, tailored to extract textual contents from PDF articles identifying their structural organization by spotting a set of common elements like the title, the authors' names and their affiliations, the abstract, one

[1] Web-service: `http://taln.upf.edu/pdfdigest`

or more sections and subsections, and the bibliographic entries included in the bibliography. Some examples of tools with such functionalities are:

- **PDFX**[2] (Constantin et al., 2013) that parses PDF files of scientific publications by exploiting several heuristics related to the layout and the lexical features of a paper to identify its structural elements;
- **Cermine**[3] (Tkaczyk et al., 2014) that processes the contents of PDF articles and properly classifies text zones as belonging to four general classes: metadata, references, body and other. In the metadata zone another classifier is responsible for the identification of the title of the paper, the authors, the affiliation, the keywords and other relevant information usually included in the header of an article. The zone classifiers exploited by Cermine are Support Vector Machines (Chang and Lin, 2011) that rely on lexical, geometrical, sequential and formatting features of the different zones of a paper to classify;
- **GROBID**[4] (Lopez, 2009) that exploits a chain of Conditional Random Field classifiers (from the Mallet library [5]) to extract a hierarchical set of structural elements from PDF papers;
- **SectLabel**[6] (Luong et al., 2012) that exploits a sequence tagging model (Conditional Random Fields) to associate to each sentence of a scientific paper a structural and rhetorical category chosen among a set of 23 structural categories (title, figure, section header, etc.) and 13 rhetorical categories (abstract, introduction, background, etc.);
- **SideNoter** (Abekawa and Aizawa, 2016) a PDF extraction tool that allows the possibility to refer the extracted text spans to the original layout of the PDF document processed.

GROBID, CERMINE and SectLabel are distributed as open-source software, while PDFX can be accessed as an on-line Web Service. At the time of writing, we are not aware of any implementation of Web Service available to process PDF papers by means of SideNoter.

3. System Description

The PDFdigest textual content extractor tool is a Java-based application that extracts some specific relevant logic and textual content from scientific articles in PDF format and stores its output in a file with XML format. It can detect both one-column and two-column articles in one or several languages for the same article (i.e. the SEPLN articles we consider in our evaluation combine Spanish and English). The extraction algorithm extracts the following parts of scientific articles:

- *titles*: main title, second title (in case it exists)
- *authors' names,*
- *authors' affiliation*
- *authors' email*
- *abstract(s)*
- *categories*
- *keywords*
- *sections' titles*: (including subsections and subsubsections).
- *sections' textual content*: (detected at paragraph level and can be associated also to subsections and subsubsections)
- *bibliographic references*: including the identification of each single bibliographic entry individually.
- other fields: such as *acknowledgements,* annexes, *author's biographies,* captions (from tables and figures), and other supporting information.

The extractor deals with both text and image-based PDF files with two approaches that share some similarities (see in Figure 1 a diagram that shows the PDFdigest architecture).

4. PDFdigest: Text-Based PDF-to-XML Extraction

The text-based PDF-to-XML extraction algorithm has 6 steps that are executed sequentially: 1) PDF to HTML conversion, 2) HTML tag properties and CSS properties' values statistics computation, 3) content filtering, 4) rule-based content detection and extraction, 5) language prediction, and finally, 6) XML generation.

4.1. PDF-to-HTML Conversion

The PDF to HTML conversion is performed using the pdf2htmlEX[7] extractor which converts each PDF file into an HTML file with the content structured in several HTML tag elements with associated CSS properties and inner textual content[8]. The HTML file includes DIV elements defining the position and style of small portions of the paper, preserving the paper's original layout by means of CSS properties.

4.2. HTML and CSS Tag Properties Statistics

The following phases of statistics computation and content extraction take profit of the generated HTML syntax and the CSS properties' values to extract the content. The JSoup[9] HTML parsing library is used in these phases. The second phase calculates some data statistics of the HTML document and its CSS properties (e.g. the most used textual font in the paper and its size).

[2] http://pdfx.cs.man.ac.uk/

[3] https://github.com/CeON/CERMINE

[4] https://github.com/kermitt2/grobid

[5] http://mallet.cs.umass.edu/

[6] https://github.com/knmnyn/ParsCit/tree/master/bin/sectLabel

[7] https://github.com/coolwanglu/pdf2htmlEX

[8] In some cases the pdf2htmlEX cannot extract textual content of PDF files that contain images instead of text.

[9] https://jsoup.org/

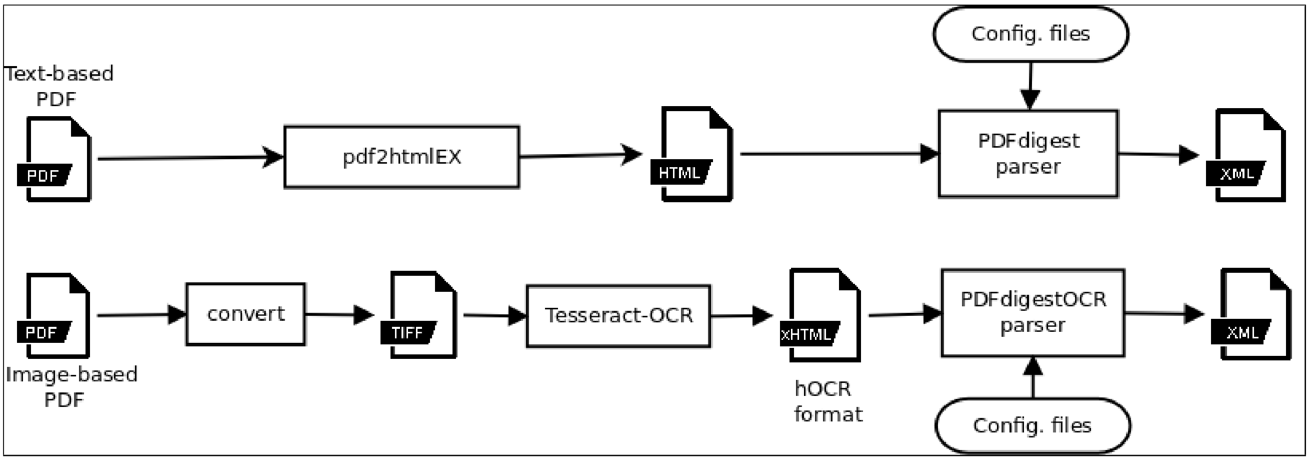


Figure 1: PDFdigest System Architecture for text-based and image-based PDF files.

4.3. Content Filtering

This phase filters out some textual content that is not extracted in the final XML file. The parts of the paper that are filtered out are: running heads, page numbers, footnotes and table contents. The following filtering strategies are the following:

- Pixel percentage margin-based filtering: this strategy filters out text that appears in the upper and lower margins of the document using the document height and predefined pixel percentage thresholds of this height for these margins.
- Text-based margin detection filtering: detects the top y-axis and bottom y-axis position of the basic textual content in the paper and uses it to filter out content in the top and bottom margins.
- Footnotes and table contents filtering functions.

4.4. Rule-Based Content Extraction

Then the rule-based extraction phase iterates over some specific tags of the HTML file and uses several manually generated rules to detect some specific content markers and consume its content. Each part of the textual content (i.e. title, abstract, acknowledgements) has its own extraction rules and consumption procedures. The rules are based on information from statistics, from the content markers detected previously, and from a set of language-dependent and content-specific regular expressions that can be manually modified or extended.
The rule-based extraction uses the following information:

1. Finite state machines that capture the regular logic structure of papers (i.e. abstract goes always after title detection, bibliographic entries go always after sections' textual content).
2. HTML attributes' statistics (i.e the most used textual font in the paper and its size).
3. CSS properties' values (e.g. values indicating the y-axis position of an element tag in the page).
4. Regular expressions that can cover several languages (currently the system supports Spanish and English). As an example, the regular expressions can find strings such as 'Abstract:' or 'Resumen:' expressions to detect the abstract titles in English and Spanish respectively.
5. HTML local element specific data (i.e. distance between the current and previous HTML tag elements).
6. Manually tuned offsets and thresholds (i.e. the maximum distance between the title and a line of text below the title to be considered part of the title).
7. Lists of language dependent hyphenated words extracted from the *Spanish LMF Freeling Lexicon* (UPC-TALP, UA-InterNostrum, UB-CLiC, and UPF-IULA , 2011) and the *Freeling*[10] dictionary for English (Padró and Stanilovsky, 2012). These lists are used to detect and mark real and false hyphens from the original text to the extracted one."Real hyphens" are the ones that appear at the end of a line but they coincide with a hyphen char from a hyphenated word (e.g. walkie-talkie). On the other hand, "false hyphens" are the ones appearing at the end of line but are not part of a hyphenated word.

The creation of rules to extract the content and the layout information was based on a set of examples from the following Computer Science conferences and journals: Computer Animation Virtual Worlds, ACM SIGGRAPH, Association for Computational Linguistics (ACL) Conference, and Language and Resources Evaluation Conference (LREC).

4.5. Language Prediction

The final phases consist in the language prediction of the recognized content and the generation of the output file in XML format. A set of language probabilities are calculated individually for each part of the content extracted and globally for the whole textual content extracted from the ar-

[10] http://nlp.cs.upc.edu/freeling/

ticles' sections. The language prediction is computed using the *optimaize language detector* [11] java API.

4.6. XML Generation and Validation

The XML generates and validates the XML output file. XML validation checking is performed with both JTidy[12] and SAX[13] Java parsers. The XML output contains a set of tags referring to the different logical sections of the paper (e.g. Abstract, Title, Author, Keywords, H1 (section title), Paragraph, BibEntry...). Each logical tag includes its related extracted content (including normalization of ligatures) within the "div" tags that relate the content to the original PDF2htmlEX HTML output "div" tags. Moreover each tag contains an attribute with the predicted language.

5. PDFdigestOCR: Image-Based PDF-to-XML Extraction

The image-based PDF-to-XML extraction is more difficult than the text-based one and needs special technologies to convert images into text. The image-based PDF-to-XML extraction algorithm has 5 steps executed sequentially: 1) image to hOCR format and textual lines layout bounding boxes extraction, 2) content filtering, 3) rule-based content detection and extraction, 4) language prediction, and finally, 5) XML generation and validation. The two last phases perform the same functionality described in subsections 4.5. and 4.6.

5.1. PDF-to-hOCR Conversion

This phase uses the Tesseract-OCR[14] state-of-the-art Optical Character Recognizer (OCR) engine to extract information from PDF files. First, PDF files are converted to images using the *convert*[15] utility. Then the OCR extracts the information of these files in a single file with the hOCR format using language specific pre-trained models. hOCR is a representation of text obtained from an OCR that stores text, style, layout information, and recognition confidence metrics in xHTML. The extraction of the bounding boxes of each textual line recognized by the OCR is crucial to detect useful data for the heuristics such as the distance between lines.

5.2. Content Filtering

This phase filters out running heads, page numbers and other textual contents in the top and bottom margins of the papers. The strategy used is the pixel percentage margin-based filtering (explained in Section 4.3.).

5.3. Rule-Based Content Extraction

Then the rule-based extraction phase, similar to the one presented in section 4.4., iterates over some specific tags of the hOCR file (the ones that have the attribute *class* equal to '*ocr_line*') and uses several manually generated rules to detect some specific content markers and consume its content. The rule-based extraction uses the following information (some points have been explained in Section 4.4.):

1. Finite state machines.
2. hOCR line-based bounding boxes data.
3. Regular expressions.
4. Manually tuned offsets and thresholds.
5. Lists of language dependent hyphenated words.

6. Evaluation

An evaluation of the quality extraction of both the PDFdigest text-based and the PDFdigestOCR image-based PDF-to-XML approaches was done with the creation of a gold standard set[16] of 27 bilingual (English/Spanish) SEPLN articles[17] that were manually annotated in order to spot: title(s), abstract(s), list of keywords, section headers (up to a depth of three levels), paragraphs, table captions, figure captions and bibliographic entries. Moreover, in order to evaluate the PDFdigestOCR image-based approach, the original dataset of 27 articles was also converted to image-based PDFs using the following Linux tools executed sequentially: 1) *pdftoppm* with a resolution of 300 DPI and 2) *convert* with the A4 layout format.

The PDFdigest text-based evaluation has been performed comparing the sets of "div" tags predicted to pertain to each class with the gold standard annotated "div" tags. The result of the PDFdigest text-based evaluation has an average F1 score of 0.917 (see global and specific results in Table 1).

Table 1: PDFdigest text-based evaluation

Class	Prec.	Recall	F1
Title	1.000	1.000	1.000
Abstract	0.921	0.928	0.890
Keywords	0.963	0.917	0.915
H1 -section title-	0.978	0.807	0.876
H2 -subsection title-	0.875	0.876	0.864
H3 -subsubsection title-	0.875	1.000	0.917
Paragraph	0.994	0.874	0.923
Table caption	0.549	0.879	0.664
Figure caption	0.584	0.922	0.691
BibEntry	0.923	0.915	0.919
Avg. weighted by class freq.	0.878	0.976	0.917

Most of the extraction errors of the PDFdigest text-based algorithm are due to: 1) papers without textual content extracted from the original PDF to the HTML file (includes cases in which wrong characters have been extracted and

[11] https://github.com/optimaize/language-detector

[12] http://jtidy.sourceforge.net/

[13] http://www.saxproject.org/

[14] https://github.com/tesseract-ocr/tesseract

[15] https://www.imagemagick.org

[16] The evaluation dataset with the gold annotated data and the PDF articles in text-based and image-based versions is avalaible for download at this site: http://taln.upf.edu/pdfdigest/resources.php

[17] SEPLN Journal http://journal.sepln.org

cases with only images of the full paper extracted), 2) papers that do not follow the logical structure of a regular paper (i.e. papers without abstract, papers without section titles,..).

On the other hand, the evaluation of the PDFdigestOCR algorithm for image-based PDF files uses the Normalized Levenshtein Similarity[18] between strings. The textual strings of each field extracted by the PDFdigestOCR are compared with the original strings generated by the pdf2htmlEX using the marked annotations of the gold dataset.

The results of the PDFdigestOCR image-based evaluation are reported in Table 2. The results of this evaluation show good performance in such elements as the Abstract, Title, Keywords, BibEntry and H3 fields.

Table 2: PDFdigestOCR image-based evaluation

Class	**Avg. Normalized Levenshtein Similarity**
Title	0.9183
Abstract[19]	0.9498
Keywords[20]	0.8856
BibEntry	0.9203
H1 -section title-	0.2628
H2 -subsection title-	0.4145
H3 -subsubsecion title-	0.9463

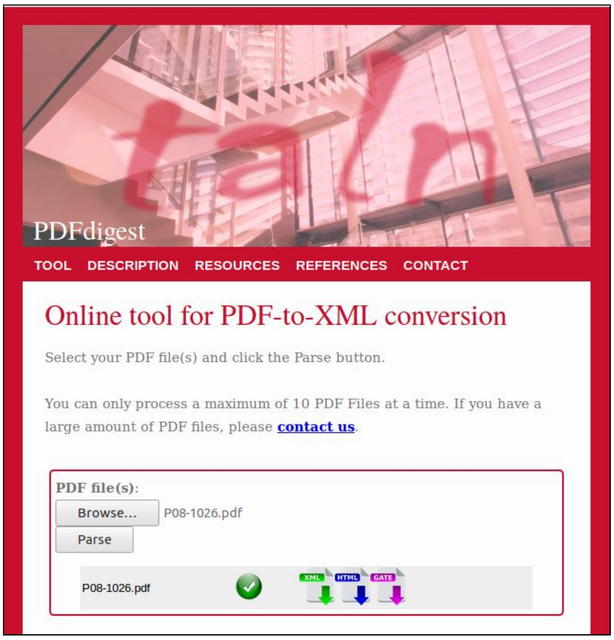
The extraction errors of the PDFdigestOCR image-based algorithm are due to: 1) OCR recognition errors (e.g. ”a.” for ”a”, ”a” for ”â”, or ”0 Name” for ”* Name”), 2) parsing and extraction errors due to bad OCR Recognitions (e.g, the error of recognizing ”0 Name” instead of ”* Name” can lead to extract a wrong section title header instead of a simple bullet list).

7. Online Demo

A web demonstration of the PDFdigest text-based PDF-to-XML tool can be accessed and tested in the following web address: `http://taln.upf.edu/pdfdigest`[21]. A screenshot of the web interface is shown in Figure 2. The demo allows to browse the local computer disk and select a set of PDF papers (up to a maximum of 10) to be converted to XML.

The output results are returned in 3 different files:

- A XML file with the content extracted and linked to the HTML file.
- An HTML file, generated by the pdf2htmlEX software, that clones the appearance of the original PDF.



Online tool for PDF-to-XML conversion

Select your PDF file(s) and click the Parse button.

You can only process a maximum of 10 PDF Files at a time. If you have a large amount of PDF files, please contact us.

PDF file(s): P08-1026.pdf

Figure 2: Screenshot of the PDFdigest tool online demo.

- A GATE XML file that includes tokenization and sentence segmentation calculated using the Dr Inventor pre-processing library[22] (Ronzano and Saggion, 2015).

8. Discussion

The PDFdigest tool can be easily adapted to different styles of scientific articles and languages. This adaptation can be realized by modifying the following data: 1) language-dependent regular expressions configuration files, 2) hyphenated words lists files, 3) the offsets and thresholds configuration files, and, in some special cases, 4) the finite states that define the logical structure of the paper style and the extraction and consumption rules themselves[23].

9. Conclusions

This paper describes and evaluates PDFdigest, an adaptable layout-aware PDF-to-XML textual content extraction tool. PDFdigest has been designed to extract textual content from both text-based and image-based PDF files. Although there are several fields that would need improvements in extraction, the evaluation of the PDFdigest text-based and the PDFdigestOCR image-based extractors shows a good performance for both algorithms in most of the PDF fields (classes) considered. The classes *Title*, *Abstract*, *Keywords*, and *Bibentry* achieved performances between 88% and 100% of average F1 (text-based) and average Normalized Levenshtein Similarity (image-based) in both approaches. This results indicate that with proper adaptation to paper layouts this tool can be valuable for scientific text mining.

[18] `https://github.com/tdebatty/java-string-similarity`

[19] Includes the detection of both the Abstract header (e.g. “*Abstract:*” in English) and the Abstract text.

[20] Includes the detection of both the Keywords header (e.g. ”*Keywords:*” in English) and the Keywords text.

[21] Note that the online demo only treats text-based PDF files and uses a set of predefined thresholds and offsets and it is not yet possible to modify them online to adapt to different layout styles.

[22] `http://backingdata.org/dri/library/`

[23] To make changes at the finite states and the extraction and consumption rules is necessary to modify the source code

10. Further Work

In the future, we plan to extend the set of content extraction rules to cover special cases of papers that do not follow the logical structure of the standard PDF articles. Further work could explore: 1) the addition of rules that can cover the extraction of more sophisticated layouts in the Computer Science conference papers, 2) an extension of the online demo that will permit the access to the configuration thresholds and offsets in order to allow the end user the option to adapt these parameters to new layouts, 3) the adaptation of the XML output format to the JATS standard[24], 4) the evaluation with different scientific datasets related to Computer Science research, and 5) the comparative evaluation with other state-of-the-art tools with the same evaluation datasets.

11. Acknowledgements

This work was partly funded by the TUNER project (TIN2015-65308-C5-5-R, MINECO/FEDER, UE) and the Spanish MINECO Ministry (MDM-2015-0502).

12. Bibliographical References

Abekawa, T. and Aizawa, A. (2016). SideNoter: Scholarly Paper Browsing System based on PDF Restructuring and Text Annotation. In Hideo Watanabe, editor, *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference System Demonstrations, December 11-16, 2016, Osaka, Japan*, pages 136–140. ACL.

Accuosto, P., Ronzano, F., Ferrés, D., and Saggion, H. (2017). Multi-level Mining and Visualization of Scientific Text Collections. In *Proceedings of the 6th International Workshop on Mining Scientific Publications (WOSP 2017)*.

Chang, C.-C. and Lin, C.-J. (2011). Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):27:1–27:27, May.

Constantin, A., Pettifer, S., and Voronkov, A. (2013). PDFX: Fully-automated PDF-to-XML Conversion of Scientific Literature. In *Proceedings of the 2013 ACM symposium on Document engineering*, pages 177–180. ACM.

Lopez, P., (2009). *GROBID: Combining Automatic Bibliographic Data Recognition and Term Extraction for Scholarship Publications*, pages 473–474. Springer Berlin Heidelberg, Berlin, Heidelberg.

Luong, M.-T., Nguyen, T. D., and Kan, M.-Y. (2012). Logical Structure Recovery in Scholarly Articles with Rich Document Features. *Multimedia Storage and Retrieval Innovations for Digital Library Systems*, 270:2.

Padró, L. and Stanilovsky, E. (2012). Freeling 3.0: Towards wider multilinguality. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Ronzano, F. and Saggion, H. (2015). Dr. inventor framework: Extracting structured information from scientific publications. In Nathalie Japkowicz et al., editors, *Discovery Science*, pages 209–220, Cham. Springer International Publishing.

Ronzano, F. and Saggion, H., (2016). *Knowledge Extraction and Modeling from Scientific Publications*, pages 11–25. Springer International Publishing, Cham.

Tkaczyk, D., Szostek, P., Dendek, P. J., Fedoryszak, M., and Bolikowski, L. (2014). CERMINE – Automatic Extraction of Metadata and References from Scientific Literature. In *Document Analysis Systems (DAS), 2014 11th IAPR International Workshop on*, pages 217–221. IEEE.

Ware, M. and Mabe, M. (2015). The STM Report: An overview of scientific and scholarly journal publishing.

13. Language Resource References

UPC-TALP, UA-InterNostrum, UB-CLiC, and UPF-IULA . (2011). *Spanish LMF Freeling Lexicon*. UPC-TALP, UA-InterNostrum, UB-CLiC, and UPF-IULA, 1, ISLRN 587-970-144-991-4.

[24] http://jats.niso.org