PDFX: Fully-automated PDF-to-XML Conversion of Scientific Literature

Alexandru Constantin
aconstantin@cs.man.ac.uk

Steve Pettifer
steve.pettifer@cs.man.ac.uk

Andrei Voronkov
voronkov@cs.man.ac.uk

School of Computer Science
The University of Manchester, United Kingdom
Oxford Road, M13 9PL

ABSTRACT

PDFX is a rule-based system designed to reconstruct the logical structure of scholarly articles in PDF form, regardless of their formatting style. The system's output is an XML document that describes the input article's logical structure in terms of title, sections, tables, references, etc. and also links it to geometrical typesetting markers in the original PDF, such as paragraph and column breaks. The key aspect of the presented approach is that the rule set used relies on relative parameters derived from font and layout specifics of each article, rather than on a template-matching paradigm. The system thus obviates the need for domain- or layout-specific tuning or prior training, exploiting only typographical conventions inherent in scientific literature. Evaluated against a significantly varied corpus of articles from nearly 2000 different journals, PDFX gives a 77.45 F1 measure for top-level heading identification and 74.03 for extracting individual bibliographic items. The service is freely available for use at `http://pdfx.cs.man.ac.uk/`.

Categories and Subject Descriptors

I.7.5 [**Document and Text Processing**]: Document Capture—*document analysis*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*text analysis*

General Terms

Algorithms, Design

Keywords

document structure analysis; PDF conversion; logical structure recovery; PDFX

1. INTRODUCTION

The recent increase in volume of the global research output has given rise to numerous initiatives that focus on automatic document processing. The predominant goal of these approaches has been to reduce the search space for potentially relevant information through means such as intuitive indexing and retrieval, document summarisation or discourse annotation. The added value brought by such services can be quite significant when considering the expeditious publication rate that certain fields of study enjoy, for instance biomedicine and the life sciences. Many such tools, however, work exclusively on plain-text and not camera-ready, typeset publications. This makes their performance dependent on data sources containing noise-free, accurate reconstructions of article narratives. Automating this pre-processing step requires programmatic access to the typographical layout of elements on page as well as to their logical/rhetorical function within the article. For this reason, analysis tools of scientific text often choose to couple themselves to data stores that have human-curated semi-structured representations of articles readily available, such as PMC [3], Scopus [7], DBLP [12] or arXiv [1]. This dependency deters the tools' widespread adoption because much of the information sought by researchers is made available solely within PDF publications with no alternative representation. Without the means to expand their reach to this highly popular format, many promising natural language processing and text mining solutions remain either undiscovered or of little use to potential users.

Notable previous efforts targeting the recovery of structure from the PDF are given in Table 1 along with their capabilities. Except for the machine learning solution SectLabel [13], the tools focus on geometrical analysis and either do not handle or are in their preliminary phases of logical structure recognition. In this paper we present a novel system called PDFX that focuses on logical structure, but handles its geometrical baseline as well. It aims to identify, extract and link these two structures together in order to facilitate an enhanced level of interaction with an article's contents. The method employed is rule-based, iterative and unrestricted with respect to the set of formatting templates that input articles need to adhere to. The only requirement is that they be full-text natively typeset PDF publications, as opposed to PDF images such as scans of paper documents. The 18 logical element types that the system can currently identify (listed in Table 2) cover the principal parts of a typical research article. They are ultimately stored in an XML file with a tag hierarchy that closely follows the JATS standard[1]. The semi-structured nature of the XML serves as a convenient, quick-access route to any of the articles's components.

2. SYSTEM AND METHODS

PDFX carries out a two-stage process in order to address structure recovery. The first stage constructs a geometrical model of the article's contents to determine the spatial organisation of textual and graphical units on page. The second stage draws upon the

[1]ANSI/NISO Z39.96-2012 standard, JATS: Journal Article Tag Suite - `http://jats.niso.org/` - formalised from the NLM Archiving and Interchange Tag Suite

DocEng'13, September 10–13, 2013, Florence, Italy.
ACM 978-1-4503-1789-4/13/09.
http://dx.doi.org/10.1145/2494266.2494271.

Tool	Output	Geometrical Structure	Logical Structure
pdftotext -bbox [2]	XHTML	pages, words (w/ coordinates)	-
pdftohtml -xml [2]	XML	fontspecs, pages, lines (w/ coordinates, font info), emphasis	-
pdftohtml -c [2]	HTML+CSS	paragraphs; CSS positioning instructions	not explicit
pdf2xml (1) [9, 10]	XML	pages, lines, words (w/ coordinates, font info, rotation, emphasis)	-
pdf2xml (2) [16]	XML	pages, font blocks (size, face, colour), lines (w/ coordinates), images	-
pdftohtmlEX [17]	HTML+CSS	fontspecs, lines, words; CSS positioning instructions	not explicit
pdfextract [11] (work in progress)	XML	pages, columns, lines, regions (w/ coordinates, font info, implicit font face)	title, header, footer, body, reference
LA-PDFText [14]	Text/XML	text blocks (w/ font, line number, height)	title, author, abstract, section, section heading
PDFExtract [6, 5]	XML	fontspecs, pages, paragraphs (w/ coordinates), lines (w/ font info)	title, abstract, section, section heading, body, footnote
SectLabel /ParsCit [13, 8]	XML/HTML	not provided; may be used as input for better logical structure recognition	title, address, affiliation, author, footnote, category, keyword, copyright, body, (sub)section, (sub)section heading, figure, table, caption, construct, equation, list_item, note, reference, email, page
PDFX (this paper)	XML/HTML	logical elements with page and column attributes and block, column and page break markers	title, author, abstract, author footnote, body, (sub)section, (sub)section heading, figure, table, caption, figure/table reference, citation, reference, URI, email, side note, header/footer, page

Table 1: Tools for structure recovery from PDF documents and their capabilities

Front Matter	Body Matter	Back Matter / Others
title	body text	bibliographic item
author	(sub)section	(reference)
abstract	(sub)section heading	URI
author footnote	image	email
	table	side note
	caption	header/footer
	figure/table reference	page number
	bibliographic reference (citation)	

Table 2: Logical elements that PDFX can extract

first to identify different logical units of discourse based on their discriminative features.

The geometrical model baseline is constructed using a library from the Utopia Documents PDF reader [4]. Three core elements of the PDF are identified: pages, words and bitmap images. Each element is modelled as a separate object with its specific features, such as bounding box, orientation, textual content or font information. Document- and page-wise statistics are then gathered to guide the selection of constituent blocks for different logical elements further down the pipeline. Font frequency maps suggest common versus rare features (such as those of the core body text vs. those of a possible title), whilst a font difference between two neighbouring words marks a first level of separation between two distinct logical units. Adjacent words of similar font characteristics are then merged together to form a set of contiguous rectangular blocks with which logical structure inference will commence. An important aspect is that the merging parameters used are defined relative to the font size and font face of each word as well as to the spacing between consecutive words and lines. This approach facilitates tailoring for any logical element type and any article layout, being significantly more flexible than approximating hard-coded numerical parameters.

With the geometrical model and statistics in place, the second stage attempts to determine the semantic roles of the newly created blocks, possibly merging them into logical *regions* in the process. A sequence of steps aims to identify one logical element type at a time, across the whole article, by tagging regions with certain characteristics.

The first and most important step in the sequence is to identify the core body text along with the reading order of the article. Out of the set of merged blocks, those containing primarily words in the most frequent font of the article are tagged as *body regions*. The dominant body region shape is used to determine the column layout and the intended reading order. Tagging of the rest of the regions is afterwards carried out in a prioritised manner. The priority is dictated by an empirically determined level of difficulty in identifying each logical element type. The elements considered easiest to tag confidently are searched for first. The rest of the identification sequence is as follows: (1) images, (2) DOI, (3) authors, (4) title, (5) outsiders: headers, footers, side notes, page numbers, (6) top-level headings, (7) abstract, (8) captions, (9) lower-level headings, (10) author footnotes, (11) remaining regions, (12) bibliography and bibliographic items, (13) other body regions, (14) tables, (15) in-text references to figures, tables and bibliographic items; URIs and emails.

A trade-off between precision and processing time was made in the system design in that the above sequence does not reiterate. Instead of employing multiple passes until no more new useful information is gained at the end of a pass, we have opted confer each tag assigned to a region an associated binary confidence level. This level can be either 'confident', to mean that the region adheres to concrete rules of a specific element type or 'possible', to signify a partial conformation with these rules. Then, as region identification progresses and new `(tag, confidence)` information becomes available, two types of events may occur: (A) certain regions may have their tags and/or confidence levels changed to reflect their most likely function in the current context; (B) increasingly more difficult element types become identifiable because of new structural and semantic cues.

The XML output is constructed with the most likely tags of the different regions at the end of the processing sequence. The initially identified contiguous blocks, now encapsulated in logical regions,

```
<region class="TextChunk" page="2" column="2">
   [...] encapsulated in logical regions,
   <marker type="page" number="3"/>
   <marker type="column" number="1"/>
   <marker type="block"/>
   jointly reconstruct the rhetorical [...]
</region>
```

Figure 1: XML example of a PDFX region spanning two columns.

jointly reconstruct the rhetorical structure of the article. Information about the different regions and their organisation is represented using an XML format very close in schema to the JATS standard. The logical *section* elements are implied by the heading hierarchy, being added in and populated during the XML construction. As regions can span multiple blocks, columns or pages, their respective XML elements may contain tags that act as physical position markers in the original text. An excerpt from the processing output of this paper illustrates a region spanning two pages (Figure 1). The intruding figure itself was identified and skipped over when reconstructing the text stream. The *class* attribute of the region, set in accordance with DoCO[2], was added in order to facilitate interoperability with other services. DoCO is an ontology of both physical and logical components of bibliographic documents, well-suited for linking PDFX output to other text processing pipelines.

3. PERFORMANCE

We report the performance of PDFX over 3 datasets of articles readily available both in published PDF form and as ground-truth, manually constructed XML representations.

The first dataset was chosen for comparison purposes against the state-of-the-art. It comprises 39 articles[3] from the field of Computer Science taken from Luong et. al. [13]. The authors have also attempted logical recovery on this collection by means of machine learning techniques. Their tool, SectLabel, was employed on geometrical analysis outputs of the articles (in the form of XML files) provided by third-party OCR software. The second dataset was compiled to ensure a hands-down evaluation on a very wide range of document styles, in effect providing a lower-bound on PDFX's performance. It consisted of the latest publication of every distinct journal from a 2011 snapshot of the PMC Open Access Subset[4]. After filtering out what was considered outside the scope of the study, such as OCR documents, prefaces and supplementary data files, the set comprised 1943 articles in total, spanning an equal number of journals. We consider this collection to be highly relevant for the task of logical structure recovery and have made it available to download at `http://pdfx.cs.man.ac.uk/serve/PMC_sample_1943.zip` for future reference. The archive contains the PDFs, the gold standard XMLs and PDFX's corresponding output (1.5GB in size). Finally, the third dataset was chosen from a practical point of view, for being representative of yearly published research around the world. It was taken from all of Elsevier's publications from the year 2008, kindly provided under a research license by the publishers. We have filtered the collection in the same manner as the PMC dataset and randomly chose 50,000 articles for testing. In contrast to the PMC dataset, style change was not as common here (being the output of a single publisher, albeit covering many journals), but the topic coverage was significantly wider.

Our standard evaluation procedure was to obtain precision (P), recall (R) and F1 measures for the XML conversions, using the Ratcliff/Obershelp string comparison method [15] to count as a correct match any extracted element found to be at least 95% similar to its ground-truth counterpart. The results for the comparative evaluation are given in Table 3 and also illustrated in Figure 2. Results for the other two datasets are given in Table 4.

Table 3: Performance evaluation for the Luong et. al. dataset. Comparison of F1 scores obtained by SectLabel, reported in [13], and those of PDFX. Scores of elements at which PDFX outperformed SectLabel are in bold.

Category	SectLabel	PDFX
author	97.94	87.02
body	96.97	88.62
email	97.64	**100**
figure	79.93	54.50
figure caption	76.91	**80.05**
page	97.84	92.30
reference (bibliography)	99.50	98.00
top-level heading	93.51	**95.3**
second-level heading	91.39	**93.0**
third-level heading	81.69	42.96
table	79.59	57.00
table caption	80.69	77.92
title	100	100

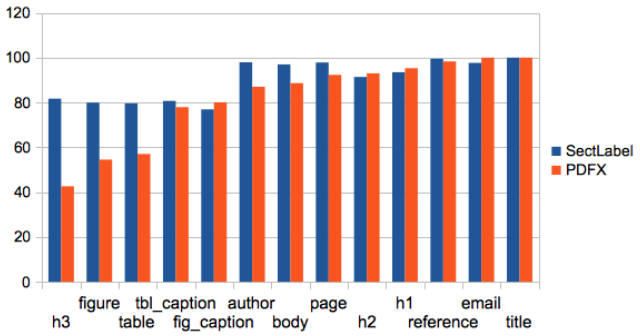

Figure 2: Bar graph view of the results in Table 3 - performance evaluation on the Luong et. al. dataset. Comparison of SectLabel [13] and PDFX.

The comparative evaluation yielded very promising results. Despite it being conducted on a dataset for which SectLabel had been trained, PDFX managed to keep up with the performance of its learned model counterpart and even outperform it at identifying four elements. At title identification, both systems performed flawlessly. PDFX was behind the SectLabel system for the other elements, particularly so for tables, figures and third-level headings. These elements mark the areas in which the visual analysis provided by an OCR system and prior learning of the layout specifics of articles prove valuable.

The automatic evaluation of the PMC dataset (Table 4), while satisfactory at times, was generally unforgiving. This is partly due to the dataset's substantial variety in terms of document styles but also to the ground-truth XMLs not necessarily matching the content of their respective PDF versions. Either because of character encoding issues, human error or intent (such as purposely leaving

[2]DoCO - `http:www.purl.org/spar/doco`

[3]The original set had 40 articles, but we were unable to retrieve one of them in natively typeset form and hence removed from the evaluation.

[4]The PubMed Central Open Access Article Subset - `http://www.ncbi.nlm.nih.gov/pmc/tools/openftlist/`

Dataset	Size	h3	table	h2	fig_tbl_ref	abstract	caption	author	citation	bib_item	h1	email	title
Elsevier	50000	83.35	28.78	82.03	89.1	62.01	82.86	94.63	75.46	86.08	90.5	97.61	96.7
PMC_sample	1943	6.05	13.27	27.19	27.52	32.41	54.53	61.65	63.10	74.03	77.45	79.67	85.42

Table 4: Performance results for the PMC and Elsevier datasets for a 0.95 similarity threshold. Precision and recall were computed for each individual article and averaged across each dataset.

out the Author Contribution section from the camera-ready PDF), the two variants differed at times. We ran the evaluation again at a 0.8 similarity threshold in addition to the 0.95 one in order to gain insight into the elements PDFX might still have identified correctly. We saw an average performance increase of 4 F1 points, with strong emphasis on elements with more textual content, abstracts and tables, hence more chances for discrepancies. These two elements saw a 9 and 13 F1 point increase respectively.

The highest results for the automated evaluation were obtained for the Elsevier dataset. This comes to confirm that the typesetting used across the collection is likely less varied on average, but also suggest that the curation level of Elsevier publications is stricter and their content, stylistically rich.

4. APPLICATIONS AND AVAILABILITY

The prime beneficiary of structure analysis of articles is likely to be the field of Text Mining, where knowledge relevant to a specific domain or task is typically searched for in vast document collections. Still, there are also notable use cases for a structure recovery system, when functioning as a personal tool. The following are real-world applications reported by PDFX users:

- *Accessibility support* - relieving the content of formatting additions such as headers or footers to help screen readers maintain fluency of discourse
- *Reading on a small screen* - easier consumption of content on mobile devices, facilitated through reading flow reconstruction
- *Document indexing* - improved through extraction of front-matter metadata
- *Literature recommendation* - improved through identification of bibliographic references
- *Ontology term recognition* - applied to manuscripts available only in PDF form
- *Support for tabular data extraction* - for populating specialised biomedical databases

PDFX is available at `http://pdfx.cs.man.ac.uk/` as an interactive web page and free-to-use programmatic web service. Submitted PDF articles are processed on-the-fly. The user is given three options of interacting with the output:

- Access/retrieve the generated XML version.
- View a reconstruction of the article in HTML form, using the generated XML. The core content of the original article is presented as a single-column stream of text, free from elements such as headers, footers or side notes, with figures and tables placed to the side.
- Download an archive containing the entire output, including rendered images, for offline viewing.

An example of this functionality is available at `http://pdfx.cs.man.ac.uk/example`. Input and output files for each processing job are stored for 24 hours since the time of submission, under randomly-generated URLs.

Acknowledgements. We are indebted to the Utopia Documents team, Stephen Wan, Alex Garnett, the ScienceWISE team, the NLP group at the National University of Singapore, the 2012 Biohackathon attendees as well as all the regular users of PDFX for all their support, valuable feedback and feature suggestions. We also thank Elsevier for providing the article collection for evaluation.

5. REFERENCES

[1] The arxiv e-print database - `http://arxiv.org`.

[2] The poppler pdf library `http://poppler.freedesktop.org/`.

[3] The pubmed central archive - `http://www.ncbi.nlm.nih.gov/pmc/`.

[4] Teresa K Attwood, Douglas B Kell, Philip McDermott, James Marsh, Steve Pettifer, and David Thorne. Utopia documents: linking scholarly literature with research data. *Bioinformatics*, 26(18):i568–i574, 2010.

[5] Øyvind Raddum Berg. High precision text extraction from pdf documents. *MSc - The University of Oslo*, 2011.

[6] Øyvind Raddum Berg, Stephan Oepen, and Jonathon Read. Towards high-quality text stream extraction from pdf: technical background to the acl 2012 contributed task. In *Proceedings of the ACL-2012 Special Workshop on Rediscovering 50 Years of Discoveries*, pages 98–103. Assoc. for Computational Linguistics, 2012.

[7] Judy F Burnham. Scopus database: a review. *Biomedical digital libraries*, 3(1):1, 2006.

[8] Isaac G Councill, C Lee Giles, and Min-Yen Kan. Parscit: An open-source crf reference string parsing package. In *Proceedings of LREC*, volume 2008, pages 661–667. European Language Resources Association (ELRA), 2008.

[9] Hervé Déjean. The pdf2xml project - `http://sourceforge.net/projects/pdf2xml/`.

[10] Hervé Déjean and Jean-Luc Meunier. A system for converting pdf documents into structured xml format. In *Document Analysis Systems VII*, pages 129–140. Springer, 2006.

[11] CrossRef Labs. The pdfextract project - `https://github.com/CrossRef/pdfextract`.

[12] Michael Ley. Dblp: some lessons learned. *Proceedings of the VLDB Endowment*, 2(2):1493–1500, 2009.

[13] Minh-Thang Luong, Thuy Dung Nguyen, and Min-Yen Kan. Logical structure recovery in scholarly articles with rich document features. *J. of Digital Library Systems. Forthcoming*, 2011.

[14] C. Ramakrishnan, A. Patnia, E. Hovy, et al. Layout-aware text extraction from full-text pdf of scientific articles. *Source code for biology and medicine*, 7(1):1–10, 2012.

[15] John W. Ratcliff and David Metzener. Pattern matching: The gestalt approach. *Dr. Dobb's Journal*, page 46, 1988.

[16] Matthew Talbert. Mobipocket.com pdf2xml - `https://launchpad.net/pdf2xml`.

[17] Lu Wang. The pdf2htmlex project - `http://coolwanglu.github.io/pdf2htmlEX/`.