Combining OCR Outputs for Logical Document Structure Markup

Technical Background to the ACL 2012 Contributed Task

Ulrich Schäfer **Benjamin Weitz**
DFKI Language Technology Lab
Campus D 3 1, D-66123 Saarbrücken, Germany
{ulrich.schaefer|benjamin.weitz}@dfki.de

Abstract

We describe how *paperXML*, a logical document structure markup for scholarly articles, is generated on the basis of OCR tool outputs. *PaperXML* has been initially developed for the ACL Anthology Searchbench. The main purpose was to robustly provide uniform access to sentences in ACL Anthology papers from the past 46 years, ranging from scanned, typewriter-written conference and workshop proceedings papers, up to recent high-quality typeset, born-digital journal articles, with varying layouts. *PaperXML* markup includes information on page and paragraph breaks, section headings, footnotes, tables, captions, boldface and italics character styles as well as bibliographic and publication metadata. The role of *paperXML* in the ACL Contributed Task *Rediscovering 50 Years of Discoveries* is to serve as fall-back source (1) for older, scanned papers (mostly published before the year 2000), for which born-digital PDF sources are not available, (2) for born-digital PDF papers on which the PDFExtract method failed, (3) for document parts where PDFExtract does not output useful markup such as currently for tables. We sketch transformation of *paperXML* into the ACL Contributed Task's TEI P5 XML.

1 Introduction

Work on the ACL Anthology Searchbench started in 2009. The goal was to provide combined sentence-semantic, full-text and bibliographic search in the complete ACL Anthology (Schäfer et al., 2011), and a graphical citation browser with citation sentence context information (Weitz & Schäfer, 2012). Since the ACL-HLT 2011 conference, the Searchbench is available as a free, public service[1].

A fixed subset of the Anthology, the *ACL Anthology Reference Corpus*[2] (ACL-ARC), contains various representations of the papers such as PDF, bitmap and text files. The latter were generated with PDFBox[3] and OCR (Omnipage[4]), applied to the PDF files or bitmap versions thereof. Its static nature as infrequently released reference corpus and low character recognition quality especially of older, badly scanned papers, made us to look for alternatives. For quick, automatic updates of the Searchbench index, a robust method for getting the text from old and new incoming PDF files was needed.

After a thorough comparison of different PDF-to-text extraction tools, a decision was made to process every PDF paper in the Anthology with ABBYY PDF Transformer[5], for various reasons. It ran stably and delivered good character recognition rates on both scanned, typewriter-typeset proceeding papers as well as on born-digital PDF of various sources, even on papers where PDFbox failed to extract (usable) text. Reading order recovery, table recognition and output rendering (HTML) was impressive and de-hyphenation for English text worked reasonably well. All in all, ABBYY did not deliver perfect results, but at that time was the best and quickest solution to get most of the millions of sentences from the papers of 46 years.

The role of this OCR-based approach in the ACL

[1] http://aclasb.dfki.de
[2] http://acl-arc.comp.nus.edu.sg
[3] http://pdfbox.apache.org
[4] http://www.nuance.com/omnipage
[5] http://www.abbyy.com

Proceedings of the ACL-2012 Special Workshop on Rediscovering 50 Years of Discoveries, pages 104–109,
Jeju, Republic of Korea, 10 July 2012. ©2012 Association for Computational Linguistics

Contributed Task *Rediscovering 50 Years of Discoveries* (Schäfer et al., 2012) is to serve as fall-back source when the more precise PDFExtract method (Berg et al., 2012) is not applicable.

2 Target Format

The focus of the Searchbench text extraction process was to retrieve NLP-parsable sentences from scientific papers. Hence distinguishing running text from section headings, figure and table captions or footnotes was an important intermediate task.

PaperXML is a simple logical document markup structure we specifically designed for scientific papers. It features tags for section headings (with special treatment of abstract and references), footnotes, figure and table captions. The full DTD is listed in the Appendix. A sample document automatically generated by our extraction tool is displayed in Figure 2 on the next page. In *paperXML*, figures are ignored, but table layouts and character style information such as boldface or italics are preserved.

3 Algorithm

Volk et al. (2010) used two different OCR products (the above mentioned Omnipage and ABBYY) and tried to improve the overall recognition accuracy on scanned text by merging their outputs. This approach adds the challenge of having to decide which version to trust in case of discrepancy. Unlike them, we use a single OCR tool, ABBYY, but with two different output variants, *layout* and *float*, that in parts contain complementary information. As no direct XML output mode exists, we rely on HTML output that can also be used to render PDF text extraction results in a Web browser.

3.1 Core rich text and document structure extraction

Our algorithm uses the *layout* variant as primary source. *Layout* tries to render the extracted text as closely as possible to the original layout. It preserves page breaks and the two-column formatting that most ACL Anthology papers (except the CL Journal and some older proceedings) share.

In the *float* variant, page and line breaks as well as multiple column layout are removed in favour of a running text in reading order which is indispensable for our purposes. However, some important layout-specific information such as page breaks is not available in the *float* format. Both variants preserve table layouts and character style information such as boldface or italics. Reading order in both variants may differ. A special code part ensures that nothing is lost when aligning the variants.

We implemented a Python[6] module that reads both HTML variants and generates a consolidated XML condensate, *paperXML*. It interprets textual content, font and position information to identify the logical structure of a scientific paper.

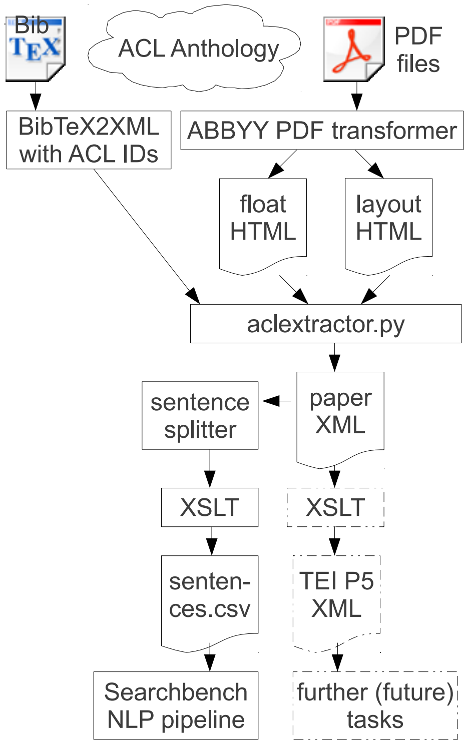

Figure 1: PDF-to-*paperXML* workflow

Figure 1 depicts the overall workflow. In addition to the two HTML variants, the code also reads BibTEX metadata in XML format of each paper. A rather large part in the document header of the generated *paperXML* addresses frontpage and bibliographic metadata. Section 3.2 explains why and how this information is extracted and processed.

Using XSLT[7], *paperXML* is then transformed into a tab-separated text file that basically contains one sentence per line plus additional sentence-related

[6] http://www.python.org

[7] http://www.w3.org/TR/xslt

```
<?xml version="1.0" encoding="UTF-8"?>
<article>
 <header>
  <firstpageheader>
   <page local="1" global="46"/>
   <title>Task-oriented Evaluation of Syntactic Parsers and Their Representations</title>
   <pubinfo>Proceedings ofACL-08: HLT,pages 46-54, Columbus, Ohio, USA, June 2008. ©2008 Association [...]</pubinfo>
   <author surname="Miyao" givenname="Yusuke">
    <org name="University of Tokyo" country="Japan" city="Tokyo"/>
   </author>
    [...]
  </firstpageheader>
  <frontmatter>
   <p><b>Task-oriented Evaluation of Syntactic Parsers and Their Representations</b></p>
   <p><b>Yusuke Miyao<footnote anchor="1"/>"   Rune Saetre<footnote anchor="1"/>"   Kenji Sagae
      <footnote anchor="1"/>"   Takuya Matsuzaki<footnote anchor="1"/>"   Jun'ichi Tsujii<footnote anchor="1"/>"** </b>
      ^Department of Computer Science, University of Tokyo, Japan * School of Computer Science, University of Manchester,
      UK *National Center for Text Mining, UK</p>
   <p>{yusuke,rune.saetre,sagae,matuzaki,tsujii}@is.s.u-tokyo.ac.jp</p>
  </frontmatter>
  <abstract>This paper presents a comparative evaluation of several state-of-the-art English parsers [...]</abstract>
 </header>
 <body>
  <section number="1" title="Introduction">
   <p>Parsing technologies have improved considerably in the past few years, and high-performance syntactic parsers are
     no longer limited to PCFG-based frameworks (Charniak, 2000; [...]</p>
  </section>
  <section number="2" title="Syntactic Parsers and Their Representations">
   <p>This paper focuses on eight representative parsers that are classified into three parsing frameworks:
    <i>dependency parsing, phrase structure parsing, </i>and <i>deep parsing.</i> [...] </p>
   <subsection number="2.1" title="Dependency parsing">
    <p>Because the shared tasks of CoNLL-2006 and CoNLL-2007 focused [...] </p>
    <p><b>mst </b>McDonald and Pereira (2006)'s dependency parser,<footnote anchor="1"/> based on the Eisner
      algorithm for projective dependency parsing (Eisner, 1996) with the second-order factorization.</p>
    <footnote label="1">http://sourceforge.net/projects/mstparser</footnote>
    <figure caption="Figure 1: CoNLL-X dependency tree"/>
   </subsection>
  [...]
   <subsection number="4.2" title="Comparison of accuracy improvements">
    <p>Tables 1 and 2 show the accuracy [...] </p>
    [...]
    <p>While the accuracy level of PPI extraction is the similar for the different parsers, parsing speed differs significantly.
     <page local="7" global="52"/> The dependency parsers are much faster than the other parsers, [...] </p>
    <table caption="Table 1: Accuracy on the PPI task with WSJ-trained
     parsers (precision/recall/f-score)" class="main" frame="box" rules="all" border="1" regular="False">
     <tr class="row"> [...]
    </table>
  <section title="Acknowledgments">
   <p>This work was partially supported by Grant-in-Aid for Specially Promoted Research (MEXT, Japan) [...]</p>
  </section>
  <references>
   <p>D. M. Bikel. 2004. Intricacies of Collins' parsing model. <i>Computational Linguistics, </i>30(4):479-511.</p>
   <p>T. Briscoe and J. Carroll. 2006. Evaluating the accuracy of an unlexicalized statistical parser on the PARC [...]</p>
    [...]
  </references>
 </body>
</article>
```

Figure 2: An example of an automatically generated *paperXML* version of the ACL Anthology document P08-1006. Parts are truncated ([. . .]) and some elements are imbalanced for brevity.

characteristics such as type (paragraph text, heading, footnote, caption etc.) page and offset. This output format is used to feed NLP components such as taggers, parsers or term extraction for the Searchbench's index generation. On the right hand side of the diagram, we sketch a potentional transformation of *paperXML* into TEI P5 for the Constributed Task. It will be discussed in Section 4.

The extraction algorithm initially computes the main font of a paper based on the number of characters with the same style. Based on this, heuristics allow to infer styles for headings, footnotes etc. While headings typically are typeset in boldface in recent publications, old publications styles e.g. use uppercase letters with or without boldface.

On the basis of this information, special section headings such as `abstract`, and `references` are inferred. Similarly, formatting properties in combination with regular expressions and Levenshtein distance (Levenshtein, 1966) are used to identify `footnotes`, `figure` and `table captions` etc. and generate corresponding markup.

A special `doubt` element is inserted for text fragments that do not look like normal, running text.

3.2 Bibliographic metadata and author affiliations

Conference or publication information can often be found on the first page footer or header or (in case of the CL journal) on every page. Our code recognizes and moves it to dedicated XML elements. The aim is not to interrupt running text by such 'noise'.

Publication authors, title and conference information as well as page number and PDF URL is commonly named bibliographic metadata. Because this information was partly missing in the ACL Anthology, special care was taken to extract it from the papers. In the *paperXML* generation code, author affiliations from the title page are mapped to author names using gazetteers, position information, heuristics etc. as part of the *paperXML* generation process. This experimental approach is imperfect, leads to errors and would definitely require manual correction. A solution would be to use manually corrected author affiliation information from the ACL Anthology Reference Corpus (Bird et al., 2008). This information, however, is not immediately available for recent proceedings or journal articles. Therefore, we developed a tool with a graphical user interface that assists quick, manual correction of author affiliation information inferred from previous publications of the same author in the Anthology by means of the ACL ARC data.

Independently from the *paperXML* extraction process, bibliographic metadata for each paper in the ACL Anthology has been extracted from BibTeX files and, where BibTeX was missing, the Anthology index web pages. We semi-automatically corrected encoding errors and generated easy-to-convert BibTeXML[8] files for each paper. Using the page number information extracted during the *paperXML* generation process, our code enriches BibTeXML files with page number ranges where missing in the ACL Anthology's metadata. This is of course only possible for papers that contain page numbers in the header or footer. The resulting BibTeXML metadata are available at DFKI's public SubVersioN repository[9] along with the affiliation correction tool.

4 Transformation to TEI P5

The ACL Contributed Task *Rediscovering 50 Years of Discoveries* (Schäfer et al., 2012) proposes to use TEI P5[10] as an open standard for document structure markup. The overall structure of *paperXML* is largely isomorphic to TEI P5, with minor differences such as in the position of page break markup. In *paperXML*, page break markup is inserted after the sentence that starts before the page break, while in TEI P5, it appears exactly where it was in the original text, even within a hyphenated word.

The Python code that generates *paperXML* could be modified to make its output conforming to TEI. Alternatively, transformation of *paperXML* into the TEI format could be performed using XSLT. Table 1 summarizes mapping of important markup elements. Details of the element and attribute structure differ, which makes a real mapping more complicated than it may seem from the table.

[8] `http://bibtexml.sourceforge.net`

[9] `http://aclbib.opendfki.de`

[10] `http://www.tei-c.org/Guidelines/P5`

TEI element	*paperXML* element
TEI	article
teiHeader	header
author (unstructured)	author (structured)
title	title
div type="abs"	abstract
front	header/abstract
body	body
back	(no correspondance)
div type="ack"	section title= "Acknowledgments"
div type="bib"	references
p	p
head	section title="..."
hi rend="italic"	i
hi rend="bold"	b
hi rend="underline"	u
del type="lb"	- (Unicode soft hyphen)
pb n="52"	page local="7" global="52"
table	table
row	tr
cell	td

Table 1: Element and attribute mapping (incomplete) between *paperXML* and TEI P5

5 Summary and Outlook

We have described a pragmatic and robust solution for generating logical document markup from scholarly papers in PDF format. It is meant as an OCR-based fall-back solution in the ACL Contributed Task *Rediscovering 50 Years of Discoveries* (Schäfer et al., 2012) when the more precise PDFExtract method (Berg et al., 2012) is not applicable because it can only handle born-digital PDF documents. Moreover, the approach can serve as fall-back solution where PDFExtract fails or does not produce markup (e.g. currently tables). Our solution has been shown to work even on typewriter-typeset, scanned papers from the 60ies. Correctness of the produced markup is limited by heuristics that are necessary to select at markup and layout borders, reconstruct reading order, etc. Levenshtein distance is used at several places in order to cope with variants such as those induced by character recognition errors. The approach is implemented to produce XML documents conforming to the *paperXML* DTD that in turn could be transformed to TEI P5 using XSLT.

Acknowledgments

This work has been funded by the German Federal Ministry of Education and Research, projects TAKE (FKZ 01IW08003) and Deependance (FKZ 01IW11003).

References

Berg, Ø. R., Oepen, S., & Read, J. (2012). Towards high-quality text stream extraction from PDF. Technical background to the ACL 2012 Contributed Task. In *Proceedings of the ACL-2012 main conference workshop on Rediscovering 50 Years of Discoveries.* Jeju, Republic of Korea.

Bird, S., Dale, R., Dorr, B., Gibson, B., Joseph, M., Kan, M.-Y., Lee, D., Powley, B., Radev, D., & Tan, Y. F. (2008). The ACL Anthology Reference Corpus: A reference dataset for bibliographic research in computational linguistics. In *Proceedings of the sixth international conference on language resources and evaluation (LREC-08).* Marrakech, Morocco.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, *10*(8), 707—710.

Schäfer, U., Kiefer, B., Spurk, C., Steffen, J., & Wang, R. (2011). The ACL Anthology Searchbench. In *Proceedings of the ACL-HLT 2011 system demonstrations* (pp. 7–13). Portland, OR.

Schäfer, U., Read, J., & Oepen, S. (2012). Towards an ACL Anthology corpus with logical document structure. An overview of the ACL 2012 contributed task. In *Proceedings of the ACL-2012 main conference workshop on Rediscovering 50 Years of Discoveries.* Jeju, Republic of Korea.

Volk, M., Marek, T., & Sennrich, R. (2010). Reducing OCR errors by combining two OCR systems. In *ECAI-2010 workshop on language technology for cultural heritage, social sciences, and humanities* (pp. 61–65). Lisbon, Portugal.

Weitz, B., & Schäfer, U. (2012). A graphical citation browser for the ACL Anthology. In *Proceedings of the eighth international conference on language resources and evaluation LREC-2012* (pp. 1718–1722). Istanbul, Turkey: ELRA.

Appendix: *paperXML* DTD

```
 <!-- paperXML DTD second version as of
      2009-10-16 Ulrich.Schaefer@dfki.de -->

<!ELEMENT article (header, body) >

<!ELEMENT header (file?, pdfmetadata?,
            ocrmetadata?, firstpageheader,
            frontmatter?, abstract) >

<!ELEMENT pdfmetadata (meta)* >

<!ELEMENT ocrmetadata (meta)* >

<!ELEMENT meta EMPTY >
<!ATTLIST meta name    CDATA #REQUIRED
               content CDATA #REQUIRED >

<!ELEMENT firstpageheader (page, title,
            subtitle?, pubinfo?, author*) >

<!ELEMENT frontmatter (p)* >

<!ELEMENT title (#PCDATA) >

<!ELEMENT subtitle (#PCDATA) >

<!ELEMENT pubinfo (#PCDATA) >

<!ELEMENT author (#PCDATA | org)* >
<!ATTLIST author surname    CDATA #IMPLIED
                 middlename CDATA #IMPLIED
                 givenname  CDATA #IMPLIED
                 address    CDATA #IMPLIED
                 email      CDATA #IMPLIED
                 homepage   CDATA #IMPLIED >

<!ELEMENT org EMPTY >
<!ATTLIST org name    CDATA #IMPLIED
              country CDATA #IMPLIED
              city    CDATA #IMPLIED >

<!ELEMENT abstract (#PCDATA | b | i | u |
                    footnote)* >

<!ELEMENT body (section*, references?,
            appendix*) >

<!ELEMENT section (subsection | p | footnote |
            table | figure | page | doubt)* >
<!ATTLIST section number CDATA #IMPLIED
                  title  CDATA #REQUIRED >

<!ELEMENT subsection (subsubsection | p | table|
           footnote | table | figure | doubt)* >
<!ATTLIST subsection number CDATA #IMPLIED
                     title  CDATA #REQUIRED >
```

```
<!ELEMENT subsubsection (p | footnote | table |
             figure | page | doubt)* >
<!ATTLIST subsubsection number CDATA #IMPLIED
                        title  CDATA #REQUIRED >

<!ELEMENT references (p | footnote | page |
             doubt)* >

<!ELEMENT appendix (p | footnote | table |
             figure | page | doubt)* >
<!ATTLIST appendix number CDATA #IMPLIED
                   title  CDATA #REQUIRED >

<!ELEMENT p (#PCDATA | page | b | i | u |
             footnote)* >

<!ELEMENT page EMPTY >
<!ATTLIST page local  CDATA #REQUIRED
               global CDATA #IMPLIED >

<!-- boldface -->
<!ELEMENT b (#PCDATA | i | u | footnote)* >

<!-- italics -->
<!ELEMENT i (#PCDATA | b | u | footnote)* >

<!-- underlined -->
<!ELEMENT u (#PCDATA | i | b | footnote)* >

<!ELEMENT footnote (#PCDATA) >
<!ATTLIST footnote label  NMTOKEN #IMPLIED
                   anchor NMTOKEN #IMPLIED >

<!-- text that is probably not sentential -->
<!ELEMENT doubt (#PCDATA) >
<!ATTLIST doubt alpha     CDATA #REQUIRED
                length    CDATA #REQUIRED
                tooSmall  CDATA #REQUIRED
                monospace CDATA #REQUIRED >

<!ELEMENT figure (#PCDATA | p)* >
<!ATTLIST figure caption CDATA #IMPLIED >

<!-- rest is HTML-like table markup -->
<!ELEMENT table (tr)* >
<!ATTLIST table caption CDATA #IMPLIED
                class   CDATA #IMPLIED
                frame   CDATA #IMPLIED
                rules   CDATA #IMPLIED
                border  CDATA #IMPLIED
                regular CDATA #IMPLIED >

<!ELEMENT tr (td)* >
<!ATTLIST tr class CDATA #IMPLIED >

<!ELEMENT td (p)* >
<!ATTLIST td class   CDATA #IMPLIED
             rowspan CDATA #IMPLIED
             colspan CDATA #IMPLIED >
```