

INTELLIGENT TEXT EXTRACTION FROM PDF

ABSTRACT

A *wrapper* is a program that automatically navigates a data structure such as a web site, selecting and extracting the relevant content and delivering it in the form of structured data (such as XML) into databases and other applications. The *Lixto Visual Wrapper* [3] allows a user to visually create wrappers to extract data from similarly structured web pages, or from web pages whose content changes over time. The wrapper is based on user-defined patterns exploiting the hierarchical structure of the HTML source.

The focus of this paper is to extend the functionality of the Lixto Visual Wrapper to PDF documents. This requires *understanding* the PDF document and converting it into an XML (or annotated HTML) format whose structure represents the logical meaning of the document.

KEYWORDS

data extraction, document understanding, PDF, XML, logical structure, geometric structure, unstructured documents

1. INTRODUCTION

The *Lixto Visual Wrapper* [3], a product of research carried out by the Information Systems Department of TU Wien, currently makes use of the tree structure inherent in HTML documents to locate relevant data. Extraction patterns are modelled in the *Elog* web extraction language [2] which locates an element's position in the HTML parse tree and allows additional constraints to be specified. As the user is fully guided through this process by a visual interface, the complexity of the wrapper program is completely hidden.

Much of the information on today's web is published in the form of PDF documents. Documents can be analysed on two levels; the *geometric* level and the *logical* level; each with their corresponding structure. Only the geometric structure is explicitly available to us in the PDF format*. The logical structure must be rediscovered by the process of *document understanding*, which attempts to reverse the document-authoring process by examining the *layout conventions* that have been used.

2. IMPLEMENTATION

2.1 Line-finding and clustering (or segmentation) to form blocks

In order to segment a page into blocks that represent the smallest meaningful elements, the OCR community has traditionally relied upon techniques based on low-level operations on a bitmap image of the page. Typically these consist of several smoothing operations followed by connected components analysis [1] [4]. As PDF already contains higher-level data in the form of text blocks and co-ordinates, we have decided to use the technique of *clustering* instead of segmentation to group similar text blocks if they are sufficiently close to each other. Currently the threshold distance is a function of the average point size of text in the page. We have found that this algorithm, even in its most basic form, works sufficiently well for most of our purposes.

In PDF files, lines of text are usually comprised of separate text blocks of no more than a few characters. We have implemented a line-finding algorithm that merges text blocks that have similar vertical co-ordinates

* PDF 1.4 and later allow the use of *tags* to denote the logical structure in a document [8]. As these must be added manually at the authoring stage, they are seldom found in documents encountered on the web.

and inserts an inter-word space if there is a sufficiently large horizontal gap between them. As our algorithm is run after the clustering stage, it avoids the risk of “jumping across columns”.

2.2 Classification of blocks and determining relationships

The next stage in our development process is to classify the blocks we have created in Section 2.1, and begin to extract information about the structure of the document. The techniques proposed by most papers include the use of rules, grammars, or the combination of both [1] [5], to represent the *layout conventions* that have been used in the document-authoring process. An example of a rule in our proposed system is that a text block of *higher status* (e.g. larger or bolder font) directly above a paragraph must be a heading of some sort.

2.3 Understanding of paragraphs and tables

A method of understanding multi-level paragraph and heading structure is presented in [9]. It assigns symbols to various indentation patterns and uses *indentation grammars* to form an understanding of the overall structure. A novel method is presented in [6], which analyses the start, middle and end co-ordinates of each line. These co-ordinates are, in fact, used for segmenting the page, but the information gained could also provide an understanding of paragraph structure.

Table understanding is a rather more complex task, and one that we have not yet had the opportunity to investigate in detail. We aim to detect tables in PDF documents by examining rules (grid-lines) and *whitespace density graphs* [9] for tables without rules. Regions with tables should exhibit a clear “grid-like” structure.

3. PROTOTYPING AND FUTURE WORK

We are currently using *PDFBox* [7], an open-source Java library, to parse the low-level data in the PDF file, and return a set of objects and their attributes in XML format. *XMillum* [10] is used to visualize our results.

We are currently investigating various approaches to implementation, including using the *Acrobat* API to obtain the PDF data and interact with the user, and the use of logic programming and spatial reasoning languages to represent our rules. We plan to define a document structure ontology for our logical labelling.

Our plan for the future is to integrate fully with the Lixto Visual Wrapper, to present a fully visual interface for the user to select data and define wrapper programs by integrating with a bitmap rendition of the PDF.

REFERENCES

1. Aiello, M. et. al., 2002, Document understanding for a broad class of documents. In *International Journal of Document Analysis and Recognition*, Vol. 5 No. 1, pp. 1-16.
2. Baumgartner, R. et. al., 2001, The Elog Web Extraction Language. *Logic for Programming, Artificial Intelligence, and Reasoning*. Havana, Cuba, Springer Lecture Notes in Computer Science 2250, pp. 548-560.
3. Baumgartner, R. et. al., 2001, Visual Web Information Extraction with Lixto. In *Proceedings of the 27th International Conference on Very Large Data Bases*. Rome, Italy, pp. 119-128.
4. Hadjar, K. et. al., 2004, Xed: a new tool for eXtracting hidden structures from Electronic Documents. *International Workshop on Document Image Analysis for Libraries*. Palo Alto, CA, USA, IEEE Computer Society, pp. 1-16.
5. Klink, S. and Kieninger, T., 2001. Rule-based Document Structure Understanding with a Fuzzy Combination of Layout and Textual Features. In *International Journal on Document Analysis and Recognition*, Vol. 4, No. 1, pp. 18-26.
6. Lovegrove, W. and Brailsford, D., 1995. Document analysis of PDF files: methods, results and implications. In *Electronic Publishing*, Vol. 8, Nos. 2 & 3, pp. 207-220.
7. PDFBox, <http://www.pdfbox.org>
8. Planet PDF – What is tagged PDF? <http://www.planetpdf.com/enterprise/article.asp?ContentID=6067>
9. Rus, D. and Summers, K., 1997. Geometric Algorithms and Experiments for Automated Document Structuring. In *Mathematical and Computer Modelling*. Vol. 26, No. 1, pp. 55-83.
10. XMillum, <http://xmillum.sourceforge.net>