Semantic Recognition of Digital Documents*

Paweł Betliński, Paweł Gora, Kamil Herba,
Trung Tuan Nguyen, and Sebastian Stawicki

Faculty of Mathematics, Computer Science and Mechanics
University of Warsaw
Banacha 2, 02-097 Warsaw, Poland
pbetl@mimuw.edu.pl, pawelg@mimuw.edu.pl, k.herba@students.mimuw.edu.pl,
nttrung@pjwstk.edu.pl, stawicki@mimuw.edu.pl

Abstract. The paper presents methods developed by the Methods of Semantic Recognition of Scientific Documents group in the research within the scope of the SYNAT project. It describes document representation format together with a proof of concept system converting scientific articles in PDF format into this representation. Another topic presented in the article is an experiment with clustering documents by style.

Keywords: content representation, documents semantics, clustering.

1 Introduction

SYNAT is a large scientific project aiming to create the universal hosting and scientific content storage and sharing platform for academia, education and open knowledge society. The project is carried out by the research consortium comprising major Polish science institutions.

The goal of our research team is to develop methods for semantic recognition of digital documents, e.g.:

- extracting key information, e.g. title, names of authors, affiliations, references,
- splitting document content into sections,
- developing common format for document representation.

The rest of the paper is organized as follows: first we introduce our document representation which is an XML-based format developed by the National Library of Medicine. We will refer to this format as "NXML". Particularly, we describe basic concepts from the tag suite and experiments conducted with the format in

* The authors are supported by the grant N N516 077837 from the Ministry of Science and Higher Education of the Republic of Poland and by the National Centre for Research and Development (NCBiR) under Grant No. SP/I/1/77065/10 by the strategic scientific research and experimental development program: "Interdisciplinary System for Interactive Scientific and Scientific-Technical Information".

the SYNAT project. In Section 2 we describe the NXML format and reference experiments (conducted by other research groups) which prove usefulness of the chosen representation. In the next section we present the process of generating NXML representation of the document from the PDF file.

In Section 4 we describe a procedure for clustering documents by styles. It is a preprocessing step which aims to facilitate generating document representation. Documents in each extracted cluster have similar layout, so in the further research it may be helpful in creating a domain knowledge base of document styles. We also point out how to use such base to improve the process of generating NXML representation.

Section 5 concludes the paper and describes plans for the future research.

2 Document Representation

2.1 Representation of the Documents

In the SYNAT project we have encountered a problem of data inconsistency. Our sources contain documents (books and articles) in various formats and this diversification is especially visible on two levels:

– File formats - image formats (jpg, tiff, bmp) representing bitmaps and pixels and documents exchange formats (ps, pdf) focused on presentation layer (used mainly to facilitate reading or navigating).
– Documents differ in layout (even within a given journal we can see layout evolution over time). Locating document meta information (such as title, abstract, keywords, authors or references) is a difficult task and depends on document "layout style".

Despite having documents in a file-form it is not a trivial task to automatically process them. To solve the problem we have decided to use an abstraction layer of document representation. More precisely, we convert all documents (independently on their origin and initial format) into an intermediate representation - familiar to architects, designers and developers of project further layers. The following subsection describes it in more details.

2.2 NLM Journal Archiving and Interchange Tag Suite

The National Center for Biotechnology Information (NCBI) of the National Library of Medicine (NLM) has faced the problem of creating common format for exchanging textual content like articles or books. They prepared a tool (Journal Archiving and Interchange Tag Suite) which simplifies document structure schema preparation process. From a technical point of view the tag suite is a set of XML schema modules (DTD modules) which can be combined to create a specific (well-fitted to a given task) XML tag set (XML schema).

NCBI/NLM has prepared (by use of their own tag suite) and made available several tag sets for specific purposes. The one on which we focused and which seems to meet our requirements is called "Journal Archiving and Interchange Tag Set". It is the most liberal among the NLM/NCBI tag sets in a sense that it allows to express the highest spectrum of document structural notions and concepts. As to the description that appears on the tag suite web page [NXML] this tag set is "Created to enable an archive to capture as many of the structural and semantic components of existing printed and tagged journal material as conveniently as possible, with no effort made to model any particular sequence or textual format". And as such it looks as a perfect match for SYNAT requirements for an intermediate representation which allows to convey documents structure and semantics. SYNAT codename for this format is NXML.

NXML documentation provides a brief description together with remarks on the usage of elements in the schema. Let us mention a few concepts from [NXML]:

<article-title>
"The full title of a journal article or other journal component such as a book review."

Remarks: "The <article-title> element is used in two contexts: as a part of a metadata concerning the article itself and as a part of a bibliographic reference metadata inside bibliographic citations (<element-citation> and <mixed-citation>)."

Best Practice: "Although this Tag Set cannot enforce either practice, retrieval performance will be enhanced if the subtitle is consistently placed within the <article-title> element (or the <source> element for book titles, proceedings titles, and other titles) for all cited material. Either with a <named-content> or as untagged text, the subtitle is easy to lose to searching."

<abstract>
"Summarized description of the content of a journal article."

<aff>
"Name of the institution or organization, such as university or corporation, which is the affiliation for a contributor such as the author or editor."

<conf-name>
"The full name of the conference, including any qualifiers, such as '43rd Annual'."

<contrib>
"Container element to contain the information (such as name and affiliation) about a single contributor to the article, for example, one author."

<ref-list>
"List of references (citations) for an article, which is often called 'References', 'Bibliography', or 'Additional Reading'."

<sec>
"A headed group of material; the basic structural unit of the article."
Remarks: "A very short article, or a simple article such as an editorial or an obituary, may contain nothing but paragraphs and other paragraph-level elements such as figures and tables. But most journal articles are divided into sections, each with a title that describes the content of the section, such as 'Introduction', 'Methodology', or 'Conclusions'."

2.3 Usefulness of the NXML Format

NXML conveys structural and meta information about the document which can be used by other teams in their work. Below we describe shortly two experiments conducted by other groups in the SYNAT project showing usefulness of the chosen representation.

Common Citations Experiment
The aim of the first experiment was to measure similarity between documents basing on common citations [ISMIS]. The general idea is that if documents have common references this might suggest that they treat about similar topics. Another variation of this concept is to look at papers which cite particular article (co-citations). In this experiment documents are used as nodes of a directed graph where references represent edges between them. NXML contains information about citations as particular elements defined in the XML schema which can be translated into the graph structure.

Semantic Clustering Experiment
Another experiment was a semantic clustering of scientific articles related to rough sets [RSKT]. Applied method grouped documents on the basis of their content and with assistance of DBpedia knowledge base [DBPedia]. In the experiment each document had to be converted into a vector representation. Some specific words occuring in certain parts of the document (page headers, footers, references, etc.) could have biased further analysis. Therefore only paragraphs, sections and their titles were extracted. Existence of separate elements in the NXML format representing these objects made this task relatively easy. We modified the program converting PDF files into the NXML format and added text output format containing only relevant parts of documents. Usage of selected kind of data improved the experiment results. They were better in comparison with case of using the whole document text as the input.

3 Retrieving Document Representation

In order to run a search engine or make further analysis and experiments on documents their content needs to be extracted and converted into a well structured format like NXML. We wrote a Python script based on the open-source library PDFMiner [PDFMiner] which is designed to process digital PDF files. Below we describe algorithm in more details.

3.1 Algorithm Steps

layout extraction - It is done by the PDFMiner library which obtains information about characters and their positions in a document. Basing on proximity in x- and y-axis characters are grouped into lines and blocks of text. In the third algorithm step text is classified as a semantic representation of the particular elements of a document.

validation - It is required because we cannot handle all encodings used in PDF files. Validation is done by couple heuristics involving statistics of alphanumeric characters in a document, etc.

layout conversion into the semantic representation - This is the main step in which layout elements are analyzed in order to obtain the actual content of the document. It involves analysis of properties of layout elements like common font, height and width of the line, vertical spaces between them, number of lines in a block of text, etc. Also some kind of analysis on the word level is done. For example words like 'introduction', 'references' or 'conclusion' are useful in detecting examples of headers and inferring the style of the font used in section headers. This knowledge can be applied to find the remaining headers.

conversion of the semantic representation into NXML - It is straightforward because in the third step we find parts of the text which have exact representation as the elements of our XML schema. This step can be replaced to produce some other output format as it was done in case of the experiment with classification of the rough set related documents (Section 2.3). In this particular case text representation was needed. It was easy to add functionality of producing text files containing only selected parts of the documents which were relevant for the experiment.

3.2 Layout Conversion

This step is most involved and it is split into small modules:

1. **Combining text lines which were accidentally split by the PDFMiner**.
2. **Determining number of columns in the text and their typical start and end positions** - This information is useful in finding the text flow on the page. In one of the next steps it allows us to find the paragraphs and order them correctly. Furthermore these statistics are used to determine which blocks of the text represent actual sections of the document. By that we can filter out (or leave for further processing) headers, titles, footers, etc. These unknown elements might be later analyzed by other modules to detect their semantic representation.
3. **Finding blocks of the same font** - It can be useful in finding emphasised elements and titles which are split into multiple lines, etc.
4. **Finding the article title** - We try to imitate human way of thinking. People usually do not have problem with detecting a title of the article. Our

heuristic is that it is usually on the first page before the actual start of the article text and is written with a "big" font and is clearly separated from other parts of a page.

5. **Splitting article into sections and paragraphs** - Here we use statistics collected in the second step.
6. **Fixing encoding** - In this step we try to fix problems with unknown character encoding. Common example is converting ligatures (e.g., **fi** in "classification") back to their original (textual) form. These problems are partly resolved with use of the English dictionary which makes it possible to guess the right encoding of some characters by determining whether words created after substitution of missing characters were proper English terms.

4 Clustering Documents by Style

Discovering rules identifying interesting elements of a document is easier for a set of similar (in a sense of similar layout) documents than for the whole corpus. Therefore clustering of the documents based on their styles is an important preprocessing step. The aim is to divide a given set of articles into disjoint groups representing different styles. Even if clustering method is not perfect and as a result we obtain a few styles in one cluster it is still much better than dealing with all documents. Number of rules describing position of particular elements for such cluster should be much smaller than for the whole documents set.

We have developed and tested our methods of clustering mostly on the basis of a small part of BazTech database [BazTech] - over 600 digital PDF documents divided a priori into 6 directories, each representing different style. We had to manually filter out the outliers. Based on this set we could easily check performance of the clustering methods by comparing result clusters with directories representing perfect style clustering. Number of articles in these 6 directories is respectively 88, 116, 72, 87, 6 and 269, so totally we have 638 articles.

Original PDF format is not very convenient to make an analysis of the content, in this case with purpose of clustering documents. Therefore we have decided to use a Python tool PDFMiner [PDFMiner]. We use it to transform PDF documents into an XML files which are the input for our clustering methods.

XML files produced by PDFMiner contain page characters together with their position in document and type of font grouped into text lines. Furthermore text lines are grouped into text boxes. Each text box is visually coherent and distinct from the rest of the document. PDFMiner obtains text boxes using heuristic, which of course does not work perfectly. However, it is still very good comparing to other similar tools which we have tested.

We have considered and tested two main ideas for clustering. The difference between them lies in a way of document representation (inferred from XML format of PDFMiner output).

The first idea is to represent a document by the first few text boxes, describing each of them as a sequence of numbers - sizes of fonts in the text box, written in order of appearance (change of a font is not considered inside a text line - each text line is represented by the most frequent size of a font). Using this representation we have considered several distance functions and performed for each of them typical clustering methods, like k-medoids and hierarchical clusterization, we have also done visualization with CMS (Classical Multidimensional Scaling) and PCA (Principal Component Analysis). The results were not optimistic - some styles have big variance of their representation, so big that we were not able to clearly distinguish them from the others.

The second idea seems to be more promising. In contrast to the first approach, we try to find more general features of documents sufficient to recognize styles without looking at details which, as we have seen, are sometimes not regular. So far we have realized this idea in a very simple version, where document is represented by 6 attributes, describing: size of a page (width - attribute W, and height - attribute H) and position of a text on a page (coordinates of minimal rectangle covering the whole text on a page - attributes X0, Y0, X1 and Y1).

Figure 1 presents visualization of these attributes. Each picture corresponds to one attribute written below. X-axis represents sequence of all documents. They are written in order: documents from the first directory (first style), documents from the second directory (second style) and so on. Y-axis represents values of considered attribute for documents. Each point on these pictures is a document where X coordinate is a document ID and Y coordinate is an attribute value for this document. Additionally, color of a point corresponds to the directory to which it belongs. From these pictures we can see that chosen attributes should allow us to perform style clusterization of considered documents with a high accuracy.

For comparison, Figure 2 presents one more visualization for attribute expressing size of the most frequent font on a page - which we name F. Here we can see a reason why the first idea for clusterization was not successful. Indeed, it has turned out that for considered set of articles font attributes are significantly less useful (in the clustering process) than position attributes.

For each 'good' attribute: W, H, X0, Y0, X1, Y1 we performed average-linkage agglomerative clustering, where the number of clusters was chosen using elbow criterion. Final clustering of documents was obtained by intersection of clustering results for each attribute. As the result there are 18 clusters, but 7 biggest represent perfectly desired partition with one exception - directory 1 is represented by 2 clusters (it is also important that remaining 11 clusters represent only 19 articles). Details of obtained partition are following: we have two clusters of size 29 and 56 for directory 1 and then the rest directories from 2 to 6 are represented by single clusters of size respectively 114, 60, 87, 6 and 267.

The results have turned out to be not only much better than for the first representation, but generally very close to the desired partition. However this method should be examined on a bigger set of documents. We expect that proposed document representation will be not sufficient. Therefore we will work on its improvement. For example, one of natural extensions may be considering page geometry in more details including not only general position of the whole text but also positions of text boxes.

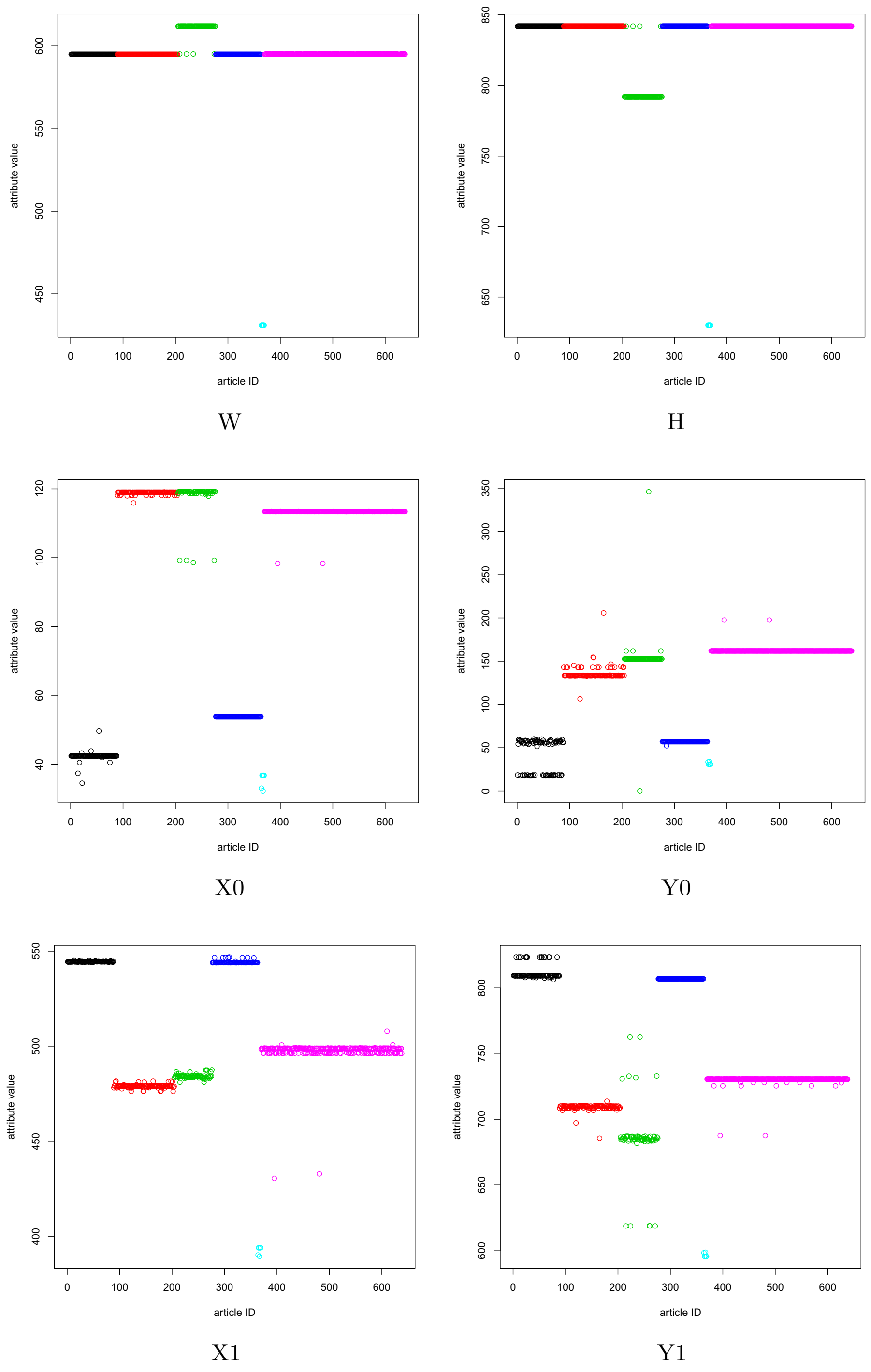

Fig. 1. Visualization of attributes used in clustering procedure

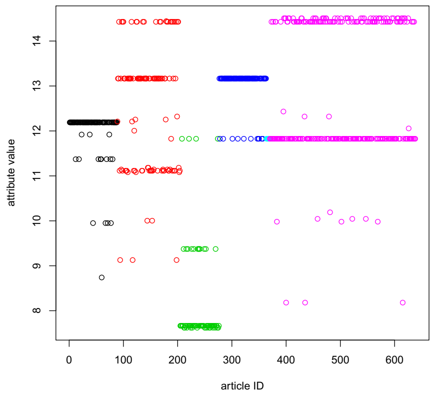

Fig. 2. Attribute F visualization

4.1 Usage of a Domain Knowledge

The extraction of content units from electronic documents can be facilitated by application of a domain knowledge. Many publications were formatted using professional systems such as TeX or other DTP packages. Most of such systems allow "styles" that help keeping publications by the same publisher in a common, uniform layout. We attempt to elicit external knowledge from these styles, whenever possible, in order to improve the performance of the content extraction process.

Typically, predefined styles would provide formatting data in the form of rules, e.g.

$$[attribute_1 = v_1] \wedge \ldots \wedge [attribute_k = v_k] \Longrightarrow [content = c],$$

for instance

$$[font_{size} = 14] \wedge [font_{bold} = TRUE] \wedge [centering = TRUE]$$
$$\Longrightarrow [content = TITLE].$$

By analyzing the available style files, we can build a collection of such rules to be used later by the extraction/recognition process. Since documents of the same publisher tend to conform to a set of pre-defined styles, we can expect to attain higher accuracy with a larger set of documents with the application of such a priori rules. Such set of rules could significantly improve performance of the algorithm retrieving information about the document described in the previous chapter.

5 Conclusions and Further Work

Experiments conducted by other teams mentioned in Section 2.3 clearly show usefulness of the chosen document representation. It is not a simple task to convert a scientific article into NXML and the algorithm presented here is just a proof of concept. In the future we plan to extend retrieving system to extract more information about the document and its content.

Experiment with clustering documents by style described in Section 4 gave optimistic results. In a long run we plan to add a rule-based subsytem which will facilitate recognition process. Rules will be manually created and adjusted for the extracted clusters.

There are many possible ways of development of the content retrieving system. First of all we need to create an evaluation procedure. So far results were inspected only manually. PubMed Central [PubMed] is a large database of the medical documents already available in the NXML format. Therefore it is a good choice for a test dataset. Another possible way to improve the system is to use domain knowledge in detecting NXML notions. For this purpose we can use for example a database of authors or scientific institutions. Layout extraction algorithm implemented in PDFMiner is not optimal and in some cases it could be adjusted to our needs.

References

[BazTech] Consortium BazTech: BazTech - Database of the Polish Technical Journal Contents (2011), http://baztech.icm.edu.pl/

[DBPedia] The DBPedia Community: The DBPedia Knowledge Base (2011), http://DBpedia.org

[PubMed] PubMed Central, http://www.ncbi.nlm.nih.gov/pmc/

[ISMIS] S. Hoa Nguyen, Świeboda, W., Jaśkiewicz, G.: Extended document representation for search result clustering. In: Bembenik, R., Skonieczny, Ł., Rybiński, H., Niezgódka, M. (eds.) To be published in: Intelligent Tools for Building a Scientific Information Platform (2011)

[NXML] Mulberry Technologies, Inc.: Journal Archiving and Interchange Tag Set Tag Library version 3.0 (2008), http://dtd.nlm.nih.gov/archiving/tag-library

[PDFMiner] Shinyama, Y.: PDFMiner: Python PDF parser and analyzer (2010), http://www.unixuser.org/~euske/python/pdfminer/

[RSKT] Szczuka, M., Janusz, A., Herba, K.: Clustering of rough set related documents with use of knowledge from dBpedia. In: Yao, J. (ed.) RSKT 2011. LNCS, vol. 6954, pp. 394–403. Springer, Heidelberg (2011)