2012 10th IAPR International Workshop on Document Analysis Systems

A modular metadata extraction system for born-digital articles

Dominika Tkaczyk, Łukasz Bolikowski, Artur Czeczko and Krzysztof Rusek
Interdisciplinary Centre for Mathematical and Computational Modelling, Univ. of Warsaw
ul. Prosta 69, 00-838 Warszawa, Poland
Email: {d.tkaczyk, l.bolikowski, a.czeczko}@icm.edu.pl, kr292291@students.mimuw.edu.pl

***Abstract*—We present a comprehensive system for extracting metadata from scholarly articles. In our approach the entire document is inspected, including headers and footers of all the pages as well as bibliographic references. The system is based on a modular workflow which allows for evaluation, unit testing and replacement of individual components. The workflow is optimized towards processing of born-digital documents, but may accept scanned document images as well. The machine-learning approaches we have chosen for solving individual tasks increase the ability to adapt to new document layouts and formats. The evaluation tests we have performed showed good results of the individual implementations and the entire metadata extraction process.**

***Keywords*-metadata extraction; page segmentation; content classification; bibliographic reference parsing.**

I. Introduction

Sometimes a digital library has to deal with documents without any metadata information included, or with metadata information that cannot be trusted due to its incorrectness or incompleteness. In such cases the library requires a reliable method of extracting metadata from documents at hand. To address these needs we defined and implemented a workflow for extracting metadata designed primarily for scientific articles in a digital form. Our main goal was to be able to extract as much information as possible, including title, authors, affiliations, abstract, parsed bibliographic references, etc.

The metadata extraction workflow we designed is flexible and easily applicable. The decomposition into clearly defined subtasks makes it easy to rewrite or replace the implementation of one process step without having to change other parts of the process. The implementations of key steps are based on machine-learning techniques, which increases the maintainability and the ability to conform to new document layouts and formats.

The first draft of the workflow was introduced in our previous paper [1]. In this article we present the first full implementation of the process, discuss the evaluation tests of individual tasks and the entire process and finally state the future work.

II. State of the art

The problem of metadata extraction is well-studied in the literature. Previous approaches are prepared for processing scanned documents and executing full digitisation from bitmap images. For example, the Medical Article Records System (MARS) [2] works on TIFF images containing the document's scanned pages. Also in the system presented by Flynn *et al.* [3] documents are first OCR-ed and converted to XML, then a rule-based approach is used to extract the metadata.

Nowadays, there are more and more born-digital documents, which do not require individual characters recognition. This difference affects both the workflow and the performance of the metadata extraction process. The approaches taken by various researchers differ in the scope of the solution, supported file formats and also methods, algorithms and improvements used. For example Giuffrida *et al.* [4] extract the content from PostScript files using a tool based on `pstotext`, and the metadata is extracted with the use of a set of rules and features computed for extracted text chunks. Esposito *et al.* [5] present a metadata extraction process for PDF/PS documents, in which page segmentation is done by a kernel-based method and zones are classified based on machine-learning approach. Marinai [6] extracts characters from PDF documents using JPedal package, performs rule-based page segmentation, and finally employs neural classifier for zone classification. Cui and Chen [7] use a Hidden Markov Model to extract metadata from PDF documents, while text extraction and page segmentation are done by `pdftohtml`, a third-party open-source tool. Rigamonti *et al.* [8] present a reverse engineering tool, that is able to process PDF documents in order to extract the physical layout structure along with the logical structures (text entities labels and hierarchical relationships between them). The system has been evaluated against a set of representative newspapers front pages.

III. Metadata extraction workflow

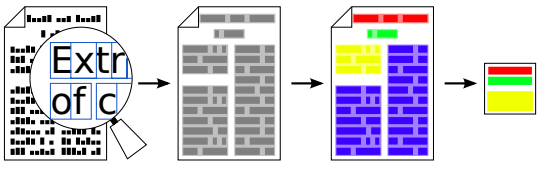

Figure 1. From a PDF document to a metadata record. After character extraction the document contains only individual characters. During page segmentation characters are gathered into words, lines and zones. Next, zones are classified and finally the metadata record is formed.

978-0-7695-4661-2/12 $26.00 © 2012 IEEE
DOI 10.1109/DAS.2012.4

The metadata extraction workflow we implemented is able to process documents in PDF format and the result is a record containing all metadata information that were extracted from the document. The metadata extraction process consists of three main stages and six process steps with carefully defined input and output:

1) building the tree structure holding the document's content, which includes character extraction and page segmentation steps,
2) analysing and enhancing the content based on the tree structure, which includes zone classification, bibliographic reference extraction and parsing steps,
3) extracting metadata information and forming the resulting metadata record.

As a result of such process decomposition it is easy to modify or replace the approach taken in a particular step without having to modify other parts of the process. In the following subsections we discuss definitions of individual steps, the approaches we have chosen, our implementations and finally the experiments and evaluation results.

A. Character extraction

The purpose of the character extraction step is to extract individual characters from the PDF stream along with their positions on the page, widths and heights. Those geometric parameters play important role in further steps, particularly page segmentation and bibliographic references extraction.

1) The implementation: Our implementation of character extraction is based on open-source iText library. We use iText to iterate over PDF's text-showing operators. During the iteration we extract text strings, their size and position on the page. Next, extracted strings are split into individual characters and their individual widths and positions are calculated. The result is an initial flat structure of the document, which consists only of pages and characters.

The widths and heights computed for individual characters are approximate and can slightly differ from the exact values depending on the font, style and characters used. Fortunately, those approximate values are sufficient for further steps.

B. Page segmentation

The goal of the page segmentation step is to create a tree structure storing the document's content. After page segmentation the document is represented by a list of pages, each page contains a set of zones, each zone contains a set of text lines, each line contains a set of words, and finally each word contains a set of individual characters. Each object in the structure has its content, position and dimensions. The tree structure is used in further steps, especially zone classification and bibliographic reference extraction.

1) The implementation: Our previous implementation of page segmentation was based on a top-down X-Y cut algorithm [9]. Lately we have replaced it with a bottom-up Docstrum algorithm [10]. In this approach, the nearest-neighbour pairs of individual characters are analyzed in order to estimate text line orientation, character and line spacing, which allows us to determine text lines and finally group lines into zones. In contrast to X-Y cut, Docstrum is independent from text orientation angle and in-line and between-line spacings used in the document.

We have added a few improvements to our Docstrum-based implementation of page segmentation:

- the distance between connected components, which is used for grouping components into lines, has been split into horizontal and vertical distance (based on estimated text orientation angle),
- fixed maximum distance between lines that belong to the same zone has been replaced with a value scaled relative to the line height,
- merging lines belonging to the same zone has been added,
- rectangular smoothing window has been replaced with Gaussian smoothing window,
- parallel and perpendicular distance proximity has been replaced with horizontal and vertical distance (based on estimated text orientation angle).

2) Evaluation: Our implementation of the page segmentation step has been tested on 112 documents from our test set (more information about the test set can be found in section IV). The goal was to check how many of the expected objects (words, lines or zones) are correctly detected by the page segmenter, how many of them are split and how many are merged. An object is considered to be correctly detected if the page segmenter generates an object containing exactly the same set of text characters. An object is split if after the segmentation its characters belong to more than one object. Finally, an object is considered merged if the set of its characters is a proper subset of a set of generated object's characters. Detailed results are shown in Figure 2. The page segmenter was able to correctly determine 90.74% of zones, 98.56% of lines and 99.49% of words.

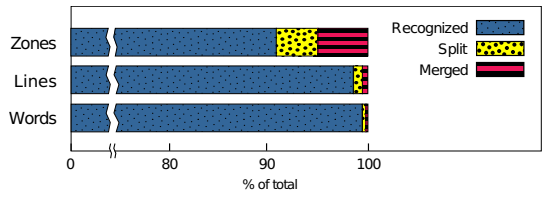

Figure 2. Page segmentation evaluation results. The diagram shows the percentage of objects correctly recognized, split and merged during page segmentation.

C. Zone classification

Zone classification is the key step in the metadata extraction process. The goal of this step is to identify the

role of every zone in the document based on its content, position and structure. The labels we use for classifying zones are: *abstract*, *affiliation*, *author*, *bibliographic* (for zones containing various bibliographic information, such as journal, volume, DOI, etc.), *body*, *copyright*, *contact*, *dates*, *editor*, *keywords*, *page number*, *references*, *title*, *type* (document's type, eg. "research", "case report"), *unknown*.

1) Previous work: Most approaches to content classification related tasks fall into two categories: rule or template-based approaches ([3], [4], [11]) and supervised machine-learning approaches ([12], [13], [14]).

An example of a rule-based approach is the system described by Mao *et al.* [11], in which content classification is based on a set of rules operating on automatically generated text features.

Machine learning-based approaches include among others Hidden Markov Models, Conditional Random Fields and Support Vector Machines. Seymore *et al.* [12] discuss how HMMs can be defined and trained for information extraction problems. Wang *et al.* [13] use HMMs and decision trees to classify zones in scanned documents. Han *et al.* [14] extract metadata from header of research papers by classifying lines of text using Support Vector Machine classifier.

2) The implementation: Our implementation of zone classifier is based on a Hidden Markov Model. HMM sequence is composed of the document's zones sorted accordingly to pages order and zones' positions on pages. Labels of zones are unknown states and vectors of features computed for zones are visible observations. The Viterbi algorithm is used to calculate the most probable zone labels based on initial, transition and emission probability obtained from a training set. We use a decision tree constructed from training elements' feature vectors to categorize feature vectors into a finite set of observation classes. More information about the model and calculations used can be found in [1].

To prevent the negative effects of a small training set we use a naïve smoothing technique for probabilities: zero probability values are replaced by a very small fixed value.

Currently we use 27 features to describe zones. The features are related to:

- zone's position and dimensions, eg. zone's relative height and width,
- zone's inner structure, eg. the presence of upper indexes, mean line height or text indentation,
- the text content of the zone, eg. the presence of digits, the presence of certain words (eg. "abstract", "references"), or words from a certain dictionary.

3) Evaluation: We used documents from our main test set for both training and tests. Our training set consisted of 20 carefully chosen documents with 215 pages and 2,893 zones. We tested our implementation on 92 documents with 807 pages and 16,346 zones. The test showed the accuracy rate 94.85% of correctly identified zone labels. The confusion matrix is shown in Table I.

D. Bibliographic reference extraction

During the zone classification step we identify zones containing bibliographic references. The goal of bibliographic reference extraction is to split the content of those zones into separate references, which can be parsed in the next step.

1) The implementation: Our first implementation of bibliographic reference extractor was based on a simple character frequency heuristic and processed only the text content of the document, not taking into account important text positioning parameters, such as between-line spacing, line length or line indentation. Since those parameters had no impact on the reference extraction results, in some cases the results were not satisfying.

Lately we have replaced the first version with a machine learning-based implementation, that takes into account not only the text content, but also text positioning parameters. In the new approach we process the sequences of text lines from references' zones. The goal is to classify lines as the first, last or inner line of the reference. Such classification allows us to group lines into separate references (every reference can be represented as a sequence of lines: *line_first (line_inner* line_last)?*).

The implementation is based on a Hidden Markov Model. The lines from references' zones form a HMM sequence, unknown line labels are hidden states and line feature vectors are visible observations. We use the Viterbi algorithm to determine the most probable labels of text lines. The details of the implementation are the same as in the case of the zone classifier.

Our feature vectors describing text lines contain 7 elements. The features are based on:

- the position of the line in the zone, eg. the space above and below the line, line indentation and length,
- the text of the line, eg. the presence of the year, if the line starts with a number/uppercase, or ends with a dot.

2) Evaluation: For training and tests we used documents from our main test set and a few additional documents with less common references layout. The training and the test sets consisted of 16 documents with 422 references and 122 documents with 4,199 references, respectively. The tests showed the accuracy rate of 96.71% of correctly extracted references. From 98 (80.33%) documents all references were extracted and 114 (93.44%) documents had at least 80% of correctly extracted references.

E. Bibliographic reference parsing

The goal of the bibliographic reference parsing step is to identify references' fragments containing meaningful pieces of information like author or title. The information we extract include: *author*, *title*, *journal*, *volume*, *issue*, *pages*, *publisher*, *location* and *year*. Parsed references play an important role in citation analysis that is outside of the scope of the metadata extraction workflow, such as references matching or citation networks.

	ABSTRACT	AFFILIATION	AUTHOR	BIBLIOGR.	BODY	COPYRIGHT	CONTACT	DATES	EDITOR	KEYWORDS	PAGE NR	REFERENCES	TITLE	TYPE	UNKNOWN
ABSTRACT	**163**			3	21										
AFFILIATION	4	**92**	1	1	15					1					8
AUTHOR			**83**		8		1							1	
BIBLIOGRAPHIC				**932**	158	8					8	2		1	14
BODY	2	14		124	**12546**	4					9	7			236
COPYRIGHT				6	4	**137**						1			
CONTACT	1	1			5		**35**								
DATES				1	3	1		**69**			1				2
EDITOR									**33**						
KEYWORDS				3	3	1				**12**					
PAGE NUMBER				7	11						**637**				1
REFERENCES				6	19							**336**			
TITLE				1	4	1							**85**		
TYPE				1	5									**78**	
UNKNOWN		2		4	92	1					2	1			**266**

Table I
THE CONFUSION MATRIX OF ZONE SEGMENTATION EVALUATION

1) Previous work: Two main approaches to reference parsing-related problems are: regular expressions and knowledge-based approaches ([15], [16]) and machine-learning techniques ([17], [18], [19]).

For example Day *et al.* [15] extract metadata from references using hierarchical template-based system and a knowledge database obtained from references strings in different formats. Gupta *et al.* [16] present a system for reference analysis that extracts metadata using a combination of regex-based heuristics and knowledge-based approach.

Connan and Omlin [17] employ a Hidden Markov Model with 32 reference token classes and the Viterbi algorithm to extract metadata from bibliographic references. Yin *et al.* [18] parse references with the aid of a bigram HMM, in which the emission probability is composed of "beginning" (emitted as first word) and "inner" (inner word) probability. Ojokoh *et al.* [19] propose a full second order HMM with modified Viterbi algorithm and a new smoothing technique for transition probabilities.

2) The implementation: The implementation of bibliographic reference parsing is based on a Hidden Markov Model. First the reference string is tokenized into substrings containing only letters and digits or another single character. Such sequence of tokens forms an HMM sequence with unknown token labels and calculated feature vectors treated as visible observations. The Viterbi algorithm is used to determine the most probable token labels. Finally, the tokens with the same labels are concatenated. The details of the implementation are the same as in the case of the zone classifier and the reference extractor.

We use a special label *text* for citation parts that do not belong to any significant metadata information (usually separators, parentheses, etc.). In HMMs the state depends only on the current observation and the previous state, so when two pieces of metadata are separated by a *text* fragment, the important information about the connection between those two pieces would not be taken into account and would have no impact on the results. To prevent this effect we use separate *text* tags for fragments appearing before various metadata fragments, eg. *text_before_title* or *text_before_volume*.

We use 33 features to describe references' tokens:

- features based on the presence of a special character class, eg. digits or lowercase/uppercase letters,
- features checking if the token is a certain character (eg. a square bracket, a comma or a dash), or a certain word,
- features checking whether the token appears in the dictionary built from the test set, eg. a dictionary of cities or words commonly appearing in the journal title.

3) Evaluation: We have used bibliographic references from the European Digital Mathematics Library for training and tests. The training and test sets contain 1,109 and 5,867 references respectively. The tests showed the accuracy rate 87.09% of correctly identified pieces of metadata information. More detailed results are shown in Figure 3.

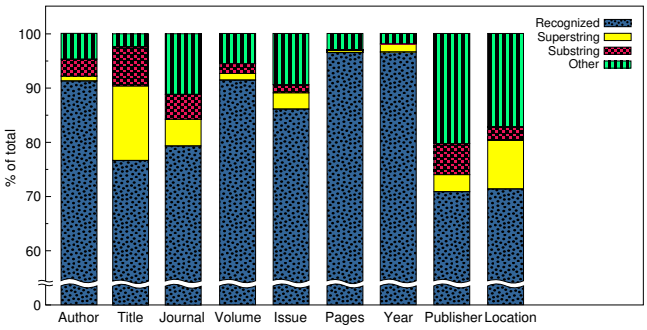

Figure 3. Bibliographic reference parsing evaluation results. The diagram shows the percentage of correctly identified reference fragments, fragments for which the superstring or substring was identified and other errors.

F. Metadata extraction

In the final step of the process the metadata is extracted from labelled zones and the resulting metadata record is formed. This includes several simple operations, eg. splitting

author and affiliation lists, associating authors with affiliations, extracting significant pieces of metadata from larger zones, filtering needless parts, eg. zone titles or separators.

In this step we attempt to extract: title, authors, authors' affiliations, contact emails, editors, document identifiers (DOI, ISSN, URN), abstract, keywords, dates (received, accepted, revised and published), journal name, volume, issue and pages range.

IV. Metadata extraction evaluation

We have tested the performance of the implementations of individual steps and the entire metadata extraction process. In the following subsections we briefly discuss the preparation of our test set and report the results of the metadata extraction process evaluation.

A. Preparation of the test set

First we compiled a list of journals published under CC-BY license from the Directory of Open Access Journals. A set of PDF documents was created by choosing one article from each four journals published by the same publisher. Next we used the first versions of our algorithms to generate two kinds of XML files: files containing the tree structure of the documents and files with metadata. Finally, we manually corrected the XML files. To make editing files containing the tree structure easier, we have developed a dedicated browsing and editing tool SegmEdit. The restriction to journals using CC-BY license allows us to freely distribute the entire test set (PDF documents + XML files as derivative works) under CC-BY license.

B. Metadata extraction results

During the tests on 113 documents the metadata extraction process was able to correctly determine 81,96% of metadata information fragments. More detailed results for various metadata types can be found in Table II.

C. Error analysis

Page segmentation and zone classification steps have the biggest impact on the results of the metadata extraction process. That is why the best effects were achieved in the case of metadata information that are usually contained by zones which are comparatively easy to classify, structure and isolate from surrounding objects. The examples are dates (received, accepted, revised and published) and bibliographic data (journal, publisher, volume, issue and pages).

Some of the metadata extraction mistakes were caused by errors occurring during page segmentation step. Some of the zones, especially those containing larger fonts, such as title or authors, were unnecessarily split by the segmenter, which affected the quality of extracted information. Also the presence of symbols such as upper or lower indexes, for example in abstract, caused in some cases line detection errors, and metadata extraction errors as a result.

	Title	**Author**	**Affiliation**	**Editor**	**Email**	**Abstract**	**Kwords**
Total	113	495	264	40	129	109	133
Extracted	95	434	169	39	94	73	65
Extracted %	**84.07**	**87.68**	**64.02**	**97.5**	**72.87**	**66.97**	**48.87**
	Journal	**Publisher**	**Volume**	**Issue**	**Pages**	**Dates**	**IDs**
Total	113	41	113	73	26	316	147
Extracted	105	40	107	67	25	296	122
Extracted %	**92.92**	**97.56**	**94.69**	**91.78**	**96.15**	**93.67**	**82.99**

Table II
THE RESULTS OF METADATA EXTRACTION PROCESS EVALUATION

Other metadata extraction errors occur as an effect of incorrect labels associated with zones during the zone classification step. This affects for example ids, affiliation (especially when the zone is placed at the end of the document), keywords or abstract (especially when abstract contains a few zones and not all of them are properly labeled).

V. Conclusions and future work

We have presented the workflow of extracting metadata from born-digital documents and its decomposition into subtasks. We have also discussed our implementations and reported the results of the evaluation tests.

Our plans for the future include:

- implementation of a better smoothing technique for HMM's probabilities,
- implementation of a better bibliographic reference parser, possibly based on CRFs,
- performing evaluation tests on a larger test set,
- adding a reading order resolver to the workflow.

Acknowledgments

The work is supported 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".

We would also like to thank the anonymous reviewers for their insightful comments.

References

[1] D. Tkaczyk and L. Bolikowski, "Workflow of Metadata Extraction from Retro-Born Digital Documents," in *Towards a Digital Mathematics Library*, P. Sojka and T. Bouche, Eds. Masaryk University Press, 2011, pp. 39–44.

[2] "Automating the production of bibliographic records for MEDLINE," Tech. Rep., 2001.

[3] P. Flynn, L. Zhou, K. Maly, S. Zeil, and M. Zubair, "Automated Template-Based Metadata Extraction Architecture," in *Proceedings of the 10th international conference on Asian digital libraries*, 2007, pp. 327–336.

[4] G. Giuffrida, E. Shek, and J. Yang, "Knowledge-based metadata extraction from PostScript files," in *Proc. of the fifth ACM conference on Digital libraries*, 2000, pp. 77–84.

[5] F. Esposito, S. Ferilli, T. M. A. Basile, and N. Di Mauro, "Machine Learning for Digital Document Processing: from Layout Analysis to Metadata Extraction," *Machine Learning in Document Analysis and Recognition*, vol. 138, pp. 105–138, 2008.

[6] S. Marinai, "Metadata Extraction from PDF Papers for Digital Library Ingest," in *Proc. of the 2009 10th International Conference on Document Analysis and Recognition*, 2009, pp. 251–255.

[7] B. Cui and X. Chen, "An improved hidden Markov model for literature metadata extraction," *Advanced Intelligent Computing Theories and Applications*, pp. 205–212, 2010.

[8] M. Rigamonti, J. Bloechle, K. Hadjar, D. Lalanne, and R. Ingold, "Towards a canonical and structured representation of PDF documents through reverse engineering," *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)*, pp. 1050–1054, 2005.

[9] G. Nagy, S. Seth, and M. Viswanathan, "A prototype document image analysis system for technical journals," *Computer*, vol. 25, no. 7, pp. 10–22, 1992.

[10] L. O'Gorman, "The document spectrum for page layout analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 11, pp. 1162–1173, 1993.

[11] S. Mao, J. W. Kim, and G. R. Thoma, "A Dynamic Feature Generation System for Automated Metadata Extraction in Preservation of Digital Materials," *Proceedings of the First International Workshop on Document Image Analysis for Libraries (DIAL'04)*, 2004.

[12] K. Seymore, A. McCallum, and R. Rosenfeld, "Learning Hidden Markov Model Structure for Information Extraction," in *AAAI 99 Workshop on Machine Learning for Information Extraction*, 1999.

[13] Y. Wang, I. Phillips, and R. Haralick, "Document zone content classification and its performance evaluation," *Pattern Recognition*, vol. 39, no. 1, pp. 57–73, 2006.

[14] H. Han, C. Giles, E. Manavoglu, H. Zha, Z. Zhang, and E. Fox, "Automatic document metadata extraction using support vector machines," in *Proc. of the 3rd ACM/IEEE-CS joint conference on Digital libraries*, 2003, pp. 37–48.

[15] M. Day, R. Tsai, C. Sung, C. Hsieh, C. Lee, S. Wu, K. Wu, C. Ong, and W. Hsu, "Reference metadata extraction using a hierarchical knowledge representation framework," *Decision Support Systems*, vol. 43, no. 1, pp. 152–167, Feb. 2007.

[16] D. Gupta, B. Morris, T. Catapano, and G. Sautter, "A New Approach towards Bibliographic Reference Identification, Parsing and Inline Citation Matching," *Communications in Computer and Information Science*, pp. 93–102, 2009.

[17] J. Connan and C. Omlin, "Bibliography extraction with hidden markov models," Tech. Rep., 2000.

[18] P. Yin, M. Zhang, Z. Deng, and D. Yang, "Metadata extraction from bibliographies using bigram HMM," *Digital Libraries: International Collaboration and Cross-Fertilization*, pp. 1–14, 2005.

[19] B. Ojokoh, M. Zhang, and J. Tang, "A trigram hidden Markov model for metadata extraction from heterogeneous references," *Information Sciences*, vol. 181, no. 9, pp. 1538–1551, May 2011.