2014 11th IAPR International Workshop on Document Analysis Systems

CERMINE — automatic extraction of metadata and references from scientific literature

Dominika Tkaczyk, Paweł Szostek, Piotr Jan Dendek, Mateusz Fedoryszak and Łukasz Bolikowski
Interdisciplinary Centre for Mathematical and Computational Modelling, Univ. of Warsaw
ul. Prosta 69, 00-838 Warszawa, Poland
Email: {d.tkaczyk, p.dendek, m.fedoryszak, l.bolikowski}@icm.edu.pl, pawel.szostek@gmail.com

***Abstract*—CERMINE is a comprehensive open source system for extracting metadata and parsed bibliographic references from scientific articles in born-digital form. The system is based on a modular workflow, whose architecture allows for single step training and evaluation, enables effortless modifications and replacements of individual components and simplifies further architecture expanding. The implementations of most steps are based on supervised and unsupervised machine-learning techniques, which simplifies the process of adjusting the system to new document layouts. The paper describes the overall workflow architecture, provides details about individual implementations and reports evaluation methodology and results. CERMINE service is available at http://cermine.ceon.pl.**

Keywords*—*document analysis, metadata extraction, bibliographic references extraction, PDF processing, zone classification

I. Introduction

The amount of literature stored in digital libraries nowadays is huge and constantly growing. A fully functional, modern digital library system in order to provide high quality services requires an access not only to the sources of stored documents, but also to their metadata including information such as title, authors, keywords, abstract or bibliographic references. This requirement cannot be emphasized enough. Reliable metadata information is crucial for data organizing, providing search tools, finding similar and related documents, building citation networks, and many more tasks.

Unfortunately, sometimes a digital library lacks the luxury of having an access to rich and trustworthy metadata. In such cases the library needs a reliable method to extract metadata and references from documents at hand. In this paper we present CERMINE — a comprehensive system for automatic metadata and parsed bibliographic references extraction from born-digital scientific literature.

CERMINE is based on a modular workflow composed of three paths and a number of steps with carefully defined input and output. By virtue of such workflow architecture individual steps can be maintained separately. As a result it is easy to perform evaluation or training, improve or replace one step implementation without changing other parts of the workflow. For most implementations we used supervised and unsupervised machine-learning techniques, which increased the maintainability of the system, as well as its ability to adapt to new document layouts. The evaluation we conducted showed good performance of key process steps and the entire metadata extraction path.

The first version of the system was presented in [1]. Since then we introduced the following improvements:

- Workflow architecture was reorganized. The new version contains two parallel paths: metadata extraction path and parsed references extraction path.
- New reading order resolving step was added. In this step we compute the order in which the elements of the document should be read.
- The implementations of many workflow steps were improved or replaced, including zone classification, references extraction and parsing.
- We introduced new classification models based on documents from PubMed [2].
- We performed the evaluation of key workflow steps and the whole metadata extraction path using a large dataset composed of documents from PubMed [2].

CERMINE web service, as well as the source code, can be now accessed online at http://cermine.ceon.pl.

II. State of the art

Extracting metadata from articles and other documents is a well-studied problem. Older approaches expected scanned documents on the input and were prepared for executing full digitization from bitmap images. Nowadays we have to deal with growing amount of born-digital documents which do not require individual character recognition. The approaches to the problem differ in the scope of the solution, supported file formats and methods and algorithms used.

Most approaches focus on extracting only article's metadata and usually do not process the entire input document. Proposed solutions are usually based on classification techniques, such as SVM [3], neural classifiers [4] or HMM [5]. Machine learning techniques have also been successfully applied to extracting metadata from article's references [6], [7], [8].

Some extraction systems are available for use online. For example the system based on TeamBeam algorithm proposed by Kern *et al.* [9] is able to extract a basic set of metadata from PDF documents using an enhanced Maximum Entropy classifier. PDFX [10] is a non open source rule-based system for processing scholarly articles. PDFX extracts metadata, body (sections, tables, etc.), and unparsed reference strings.

Unlike the examples above, CERMINE is a holistic system able to extract not only the document's metadata, but also bibliographic references along with their metadata and structured body content of the document (experimental) directly from a PDF file. CERMINE is open source and available for use as a web service. The flexibility of the system is granted by its modular architecture and the use of machine-learning techniques.

III. CERMINE EXTRACTION WORKFLOW

CERMINE is able to process documents in PDF format. The workflow does not include any OCR phase, it analyses only the PDF text stream found in the input document. The workflow inspects the entire content of the document and produces two kinds of output in NLM format [11]: the document's metadata and parsed bibliographic references.

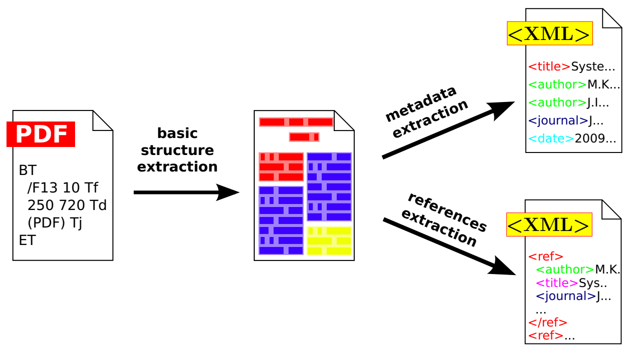

Fig. 1. Workflow architecture. At the beginning the basic structure is extracted from the PDF file. Then metadata and bibliographic references are extracted in two parallel paths.

CERMINE workflow is composed of three paths (Figure 1):

(A) Basic structure extraction path takes a PDF file on the input and produces a geometric hierarchical structure representing the document in TrueViz format [12]. The structure is composed of pages, zones, lines, words and characters, along with their coordinates and dimensions. The reading order of all elements is computed. Every zone is labelled with one of four general categories: *metadata*, *references*, *body* and *other*.
(B) Metadata extraction path analyses *metadata* parts of the geometric hierarchical structure and extracts a rich set of document's metadata from them.
(C) References extraction path analyses parts of the structure labelled as *references*. The result is a list of document's parsed bibliographic references.

One of the most important changes since the first version of the workflow [1] is introducing two parallel executions paths. Once the geometric hierarchical structure is extracted from the input file and general categories are assigned to all zones, further processing can be executed in parallel.

IV. TOOLS AND ALGORITHMS

Table I shows the decomposition of the extraction workflow into paths and steps and provides information about tools and algorithms used. This section describes in details steps that changed since the previous version of the workflow [1].

A. Reading order resolver

A PDF file contains by design a stream of strings that undergoes extraction and segmentation process. As a result we obtain pages containing characters grouped into zones, lines and words, all of which have a form of unsorted bag of items. The aim of setting the reading order is to determine the order in which all the structure elements should be read. This information is used in zone classifiers and also allows to extract the full text of the document.

In general the algorithm is based on a bottom-up strategy: firstly characters are sorted within words and words within lines horizontally, then lines are sorted vertically within zones, and finally we sort zones. The fundamental principle for sorting zones was taken from [13]. We make use of an observation that the natural reading order descends from top to bottom, if successive zones are aligned vertically, otherwise it traverses from left to right. First we calculate the distances between all zones using the value of cosine of slope of the vector connecting zones to make sure that zones aligned vertically are in general closer than those aligned horizontally. Then a binary tree with zones stored in leaves is built from the bottom by repeatedly joining the closest zones and groups of zones. Finally, an in order tree traversal gives the desired zones order.

B. Zone classification

The system contains two zone classifiers: initial classifier and metadata classifier. The task of the initial classification is to label each zone with one of four general classes: *metadata*, *references*, *body* (publication's text, equations, figures and tables) and *other* (acknowledgments, conflicts of interests statements, page numbers, etc.). The goal of metadata zone classification is to classify all *metadata* zones into specific metadata classes: *abstract*, *bib_info*, *type*, *title*, *affiliation*, *author*, *correspondence*, *dates*, *editor* and *keywords*. Both classifiers use Support Vector Machines and their implementation is based on LibSVM library [14].

In order to perform zone classification, each zone is transformed into a vector of 78 feature values:

- geometrical — based on attributes such as zone's dimensions and coordinates, distance to the nearest zone, free space below and above the zone, etc.,
- lexical — based upon keywords characteristic for different parts of narration, that is: affiliation, acknowledgment, abstract, references, article type etc.,
- sequential — based on sequence-related information, eg. class of the previous zone, presence of the same text blocks on the surrounding pages, etc.,
- formatting — eg. font size in the current and adjacent zones, amount of blank space inside zones etc.,
- heuristics — eg. uppercase word count, percentage of numbers in a text block, if each line starts with enumeration-like tokens, etc.

SVM parameters were found with the use of a set of 1,000 documents from our dataset (Section V-A). The feature vectors were scaled linearly to interval [0;1] according to bounds found in the learning samples at the learning stage.

TABLE I. THE DECOMPOSITION OF THE WORKFLOW INTO INDEPENDENT PROCESSING PATHS AND STEPS.

Path	Step	Goal	Implementation
A. **Basic structure extraction**	A1. **Character extraction**	Extracting individual characters along with their page coordinates and dimensions from the input PDF file.	**iText library**
	A2. **Page segmentation**	Constructing the document's geometric hierarchical structure containing (from the top level) pages, zones, lines, words and characters, along with their page coordinates and dimensions.	**enhanced Docstrum**
	A3. **Reading order resolving**	Determining the reading order for all structure elements.	**bottom-up heuristic-based**
	A4. **Initial zone classification**	Classifying the document's zones into four main categories: *metadata*, *body*, *references* and *other*.	**SVM**
B. **Metadata extraction**	B1. **Metadata zone classification**	Classifying the document's zones into specific metadata classes.	**SVM**
	B2. **Metadata extraction**	Extracting atomic metadata information from labelled zones.	**simple rule-based**
C. **Parsed references extraction**	C1. **Reference strings extraction**	Dividing the content of *references* zones into individual reference strings.	**KMeans clustering**
	C2. **Reference parsing**	Extracting metadata information from references strings.	**CRF**

TABLE II. THE RESULTS OF PARAMETER SEARCH FOR ZONE CLASSIFIERS. THE TABLE SHOWS THE BEST MEAN ACCURACIES FOR 10-FOLD CROSS-VALIDATION FOR PARTICULAR KERNEL FUNCTION TYPES, AS WELL AS ASSOCIATED VALUES OF BINARY LOGARITHMS OF C AND γ.

Initial zone classification				
kernel	**linear**	**4th poly.**	**RBF**	**sigmoid**
$log_2(C)$, $log_2(\gamma)$	8, -3	6, -5	8, -1	8, -10
mean accuracy	81.03	86.09	84.14	73.65

Metadata zone classification				
kernel	**linear**	**4th poly.**	**RBF**	**sigmoid**
$log_2(C)$, $log_2(\gamma)$	8, -6	5, -2	8, -1	8, -9
mean accuracy	89.76	99.14	99.14	86.20

In order to find the best parameters for the classifiers we performed a grid-search over 3-dimensional space of kernel function types (linear, fourth degree polynomial, radial-basis functions and sigmoid functions) and C (penalty parameter) and γ coefficients, whereas binary logarithm of γ varied in discrete interval [2;8] and binary logarithm of C ranged in discrete interval [-10, -1]. At every grid point we performed a 10-fold cross-validation maximizing mean accuracy. The majority classes were undersampled to ensure that all classes are equally represented. Parameters for the best obtained results are presented in Table II. For both classifiers we chose fourth degree polynomial kernel function. For initial classification chosen values of C and γ parameters are 2^6 and 2^{-5}, respectively. For metadata classification chosen values of C and γ parameters are 2^5 and 2^{-2}, respectively.

C. Extracting reference strings

References zones contain a list of reference strings, each of which can span over one or more text lines. The goal of reference strings extraction is to split the content of those zones into individual reference strings. Previous implementation of this step was based on a supervised lines classifier. New version utilizes unsupervised machine-learning techniques, which allows to omit time-consuming training set preparation and learning phases, while achieving similar extraction results.

Each text line in a reference zone belongs to exactly one reference string, some of them are first lines of their reference strings, some are the last ones. In order to group lines into consecutive references, first we determine which lines are first lines of their references. To achieve this, we transform all lines to feature vectors and cluster them into two sets. We make use of a simple observation that the first line from all references blocks is also the first line of its reference. Thus the cluster containing this first line is assumed to contain all first lines. After recognizing all first lines it is easy to concatenate lines to form consecutive reference strings.

For clustering lines we use KMeans algorithm with Euclidean distance metric. As initial centroids we use the first line's feature vector and a vector with the largest distance to the first one. We use 5 features based on line relative length, line indentation, space between the line and the previous one, and the text content of the line (if the line starts with an enumeration pattern, if the previous line ends with a dot).

D. Reference strings parsing

Reference strings extracted from references zones contain important reference metadata. In this step metadata is extracted from reference strings and the result is the list of document's parsed bibliographic references. The information we extract from the strings include: *author*, *title*, *journal name*, *volume*, *issue*, *pages*, *publisher*, *location* and *year*.

First the reference strings are tokenized. The tokens are transformed into vectors of features and labels are assigned to them. Finally, the neighbouring tokens with the same label are concatenated and the resulting reference metadata record is formed. The heart of the implementation is the token classifier, which employs Conditional Random Fields and is built on top of GRMM and MALLET packages [15].

We use 42 features to describe the tokens. Some of them are based on the presence of a particular character class, eg. digits or lowercase/uppercase letters. Others check whether the token is a particular character (eg. a dot, a square bracket, a comma or a dash), or a particular word. Finally, we use features checking if the token is contained by the dictionary built from the dataset, eg. a dictionary of cities or words commonly appearing in the journal title. It is worth to notice that the token's label depends not only on its feature vector, but also on surrounding tokens. To reflect this in the classifier, the token's feature vector contains not only features of the token itself, but also features of two preceding and two following tokens.

V. EVALUATION

The previous version of the system was evaluated against a small set of documents prepared semi-automatically [16]. The new workflow and the key steps were evaluated with the use of a much larger dataset based on PubMed [2].

TABLE III. CONFUSION MATRIX FOR INITIAL ZONE CLASSIFICATION FOR 5-FOLD CROSS-VALIDATION. ROWS AND COLUMNS REPRESENT DESIRED AND OBTAINED CLASSIFICATION DECISION, RESPECTIVELY.

	META.	BODY	REFER.	OTHER	precision	recall
META.	**20,836**	1,665	29	210	90.81	91.63
BODY	1,338	**107,815**	278	973	95.61	97.65
REFER.	172	670	**10,155**	254	96.62	90.26
OTHER	599	2,611	48	**7,491**	83.90	69.69

TABLE V. THE RESULTS OF REFERENCE PARSING EVALUATION. THE TABLE SHOWS THE NUMBERS OF DESIRED, EXTRACTED BY THE PARSER AND SUCCESSFULLY EXTRACTED METADATA FRAGMENTS, AND ALSO PRECISION AND RECALL FOR FIVE METADATA TYPES.

	Author	Title	Journal	Pages	Year
Desired	4,161	3,400	3,252	1,974	3,222
Extracted	4,089	3,391	3,220	1,967	3,205
Success	3,230	3,028	2,947	1,877	3,066
Precision	**78.99**	**89.30**	**91.52**	**95.42**	**95.66**
Recall	**77.62**	**89.06**	**90.62**	**95.09**	**95.16**

A. Datasets preparation

The dataset used for evaluating zone classifiers and the entire system was built using documents from PubMed repository. First we used our segmentation algorithm to recognize the zones in a group of 124,678 PDF files. Then for every document the text content of the zones was matched with the sections of the corresponding NLM file, which allowed us to label the zones and generate ground-truth files in TrueViz format. In some cases the resulting files contained sparsely labelled zones as a result of poor quality NLM files containing no valuable information. For the final dataset we selected only the documents, for which at least 90% of zones were successfully labelled. Te final dataset is composed of 15,894 documents (12.75% of the original dataset) with 145,734 pages and 2,446,332 zones, which gives the average of 9.17 pages per document and 16.79 zones per page.

For reference parser evaluation we used CiteSeer [17] and Cora-ref [18] datasets joined together into a set containing 3,438 references in total. Labels from original datasets were mapped in the following way: *author*, *title*, *journal*, *pages* and *year* remained the same; *booktitle*, *tech* and *type* were mapped to *journal*; *date* was mapped to *year*; all remaining tokens were labelled as *text* (a label used for separators and other parts without a specific metadata label assigned).

B. Zone classification

Initial zone classifier was evaluated by performing 5-fold cross-validation on a subset of our documents dataset containing 964 documents and 155,144 zones in total. We obtained the mean precision and recall 91.74% and 87.31%, respectively. More detailed results can be found in table III.

Metadata zone classifier was also evaluated by performing 5-fold cross-validation. The subset used in this case contained 1,934 documents and 45,035 metadata zones. We obtained the mean precision and recall 92.49% and 93.83%, respectively. More detailed results can be found in table IV.

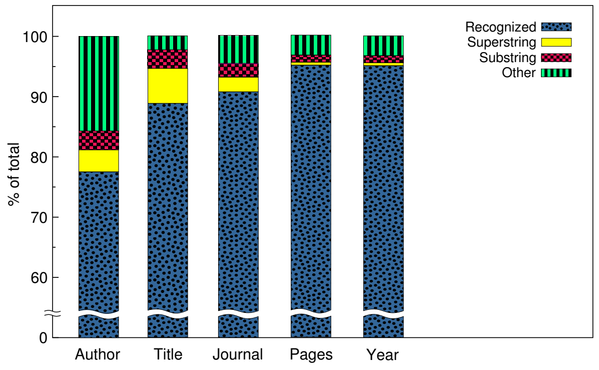

Fig. 2. The results of reference parsing evaluation. The diagram shows the percentage of correctly recognized reference metadata, metadata for which the superstring or substring were identified and other errors.

C. Reference parsing

To evaluate the reference parser we performed 5-fold cross-validation on the references dataset. For all five metadata classes we compared strings from original references ("expected" strings) with strings generated by the parser ("extracted" strings). We consider a metadata fragment of a certain class correctly extracted if the extracted string is identical with the expected string. We also checked for how many metadata fragments the parser extracted more than expected (expected string is a substring of extracted string) and for how many fragments the parser extracted a little less than expected (extracted string is a substring of expected string). The results are shown in Table V and Figure 2.

D. System evaluation

The evaluation of metadata extraction was performed with the use of our main dataset. The classifiers were trained on a subset containing 1,934 documents used previously for metadata classifier evaluation. CERMINE was evaluated on the remaining 13,960 files. The PDF files were processed by the workflow and the resulting metadata was compared with metadata stored in NLM files.

For each type of metadata we used different measures of correctness. Some information, such as article's volume, issue, DOI, dates, pages and journal's ISSN were considered correct if exactly equal to NLM data. As the journal name is often abbreviated, we considered it correct if it was a subsequence of the real journal name. Article's title and abstract were tokenized and the Smith-Waterman distance was applied, which resulted in the average string alignment. The expected and obtained lists of authors, keywords and affiliations were compared for every document giving individual precision and recall, and finally the mean precision and recall for the entire set were computed. The results are shown in Table VI. CERMINE achieved good results for the most important metadata types. Poor results in some cases, eg. pages or ISSN were caused by the lack of relevant information in many PDF files.

VI. CONCLUSIONS AND FUTURE WORK

The article presents CERMINE — a system for extracting both metadata and parsed bibliographic references from born-digital scientific articles. The system is open source and

TABLE IV. CONFUSION MATRIX FOR THE METADATA CLASSIFICATION FOR 5-FOLD CROSS-VALIDATION. ROWS AND COLUMNS REPRESENT DESIRED AND OBTAINED CLASSIFICATION DECISION, RESPECTIVELY.

	ABSTRACT	**AFFIL.**	**AUTHOR**	**BIB_INFO**	**CORRESP.**	**DATES**	**EDITOR**	**KEYW.**	**TITLE**	**TYPE**	**COPYR.**	**precision**	**recall**
ABSTRACT	**5,851**	31	13	20	10	24	0	48	202	10	10	97.78	94.08
AFFIL.	15	**2,417**	37	18	80	67	0	47	27	4	13	91.07	88.70
AUTHOR	3	19	**1,920**	5	15	0	0	3	15	7	8	93.02	96.24
BIB_INFO	22	23	11	**17,256**	15	223	0	17	36	25	138	98.66	97.13
CORRESP.	3	78	20	5	**1,006**	24	0	1	2	0	1	86.80	88.25
DATES	9	26	2	90	20	**4,584**	0	3	4	1	89	92.27	94.95
EDITOR	0	0	0	0	0	0	**3**	0	0	0	0	100.00	100.00
KEYW.	34	21	11	7	5	3	0	**956**	43	7	5	79.80	87.55
TITLE	32	24	25	36	4	6	0	99	**4,078**	32	12	91.74	93.79
TYPE	3	3	7	13	0	0	0	10	20	**1,676**	3	94.74	96.60
COPYR.	12	12	18	41	4	37	0	14	18	7	**3,021**	91.55	94.88

TABLE VI. THE RESULTS OF METADATA EXTRACTION EVALUATION.

	precision	**recall**
journal title	68.68%	49.23%
volume	97.57%	78.57%
issue	52.50%	56.64%
pages	51.37%	34.71%
year	98.79%	89.18%
DOI	93.60%	57.46%
ISSN	44.29%	3.01%

	avg. string adjustment
article title	95.03%
abstract	91.43%

	avg. precision	**avg. recall**
authors	87.19%	82.07%
affiliations	70.13%	59.44%
keywords	61.11%	68.37%

available online. The modular architecture and the use of machine-learning techniques make CERMINE flexible and easy to adapt to new document layouts. The evaluation against a large diverse dataset shows good results for the key individual steps and the entire metadata extraction path.

Our future plans include:

- expanding the workflow architecture by adding a process path for extracting structured full text containing sections and subsections, headers and paragraphs,
- performing the evaluation of the whole references extraction path using the PubMed-based dataset,
- the evaluation of other similar systems using the same dataset and comparing the extraction results.

ACKNOWLEDGMENTS

The work is partly 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".

REFERENCES

[1] D. Tkaczyk, L. Bolikowski, A. Czeczko, and K. Rusek, "A modular metadata extraction system for born-digital articles," in *10th IAPR International Workshop on Document Analysis Systems*, 2012, pp. 11–16.

[2] "PubMed," http://www.ncbi.nlm.nih.gov/pubmed.

[3] H. Han, C. L. Giles, E. Manavoglu, H. Zha, Z. Zhang, and E. A. Fox, "Automatic document metadata extraction using support vector machines," in *3rd ACM/IEEE-CS Joint Conference on Digital Libraries*, 2003, pp. 37–48.

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

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

[6] E. Cortez, A. S. da Silva, M. A. Gonçalves, F. Mesquita, and E. S. de Moura, "Flux-cim: flexible unsupervised extraction of citation metadata," in *7th ACM/IEEE-CS Joint Conference on Digital Libraries*, 2007, pp. 215–224.

[7] E. Hetzner, "A simple method for citation metadata extraction using hidden markov models," in *8th ACM/IEEE-CS Joint Conference on Digital Libraries*, 2008, pp. 280–284.

[8] L. Gao, Z. Tang, and X. Lin, "Cebbip: a parser of bibliographic information in chinese electronic books," in *9th ACM/IEEE-CS Joint Conference on Digital Libraries*, 2009, pp. 73–76.

[9] R. Kern, K. Jack, and M. Hristakeva, "TeamBeam - Meta-Data Extraction from Scientific Literature," *D-Lib Magazine*, vol. 18, 2012.

[10] A. Constantin, S. Pettifer, and A. Voronkov, "Pdfx: fully-automated pdf-to-xml conversion of scientific literature," in *2013 ACM Symposium on Document Engineering*, 2013, pp. 177–180.

[11] "NLM," http://dtd.nlm.nih.gov/archiving/.

[12] C. H. Lee and T. Kanungo, "The architecture of TrueViz: a groundTRUth/metadata editing and VIsualiZing ToolKit," *Pattern Recognition*, vol. 15, 2002.

[13] "PdfMiner," http://www.unixuser.org/ euske/python/pdfminer/.

[14] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011.

[15] A. K. McCallum, "MALLET: A Machine Learning for Language Toolkit," 2002.

[16] D. Tkaczyk, A. Czeczko, K. Rusek, L. Bolikowski, and R. Bogacewicz, "Grotoap: ground truth for open access publications," in *12th ACM/IEEE-CS Joint Conference on Digital Libraries*, 2012, pp. 381–382.

[17] C. L. Giles, K. D. Bollacker, and S. Lawrence, "CiteSeer: An automatic citation indexing system," in *3rd ACM Conference on Digital Libraries*. ACM, 1998, pp. 89–98.

[18] A. McCallum, K. Nigam, and J. Rennie, "Automating the construction of internet portals with machine learning," *Information Retrieval*, pp. 127–163, 2000.