CERMINE - extracting metadata and references from scientific publications

Paweł Szostek
p.szostek@icm.edu.pl

Dominika Tkaczyk
d.tkaczyk@icm.edu.pl

Łukasz Bolikowski
l.bolikowski@icm.edu.pl

Centre for Open Science, Interdisciplinary Centre for Mathematical and Computational Modelling, Univ. of Warsaw
ul. Prosta 69, 00-838 Warszawa, Poland

ABSTRACT

We present 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.

Categories and Subject Descriptors

I.7.5 [**Document and Text Processing**]: Document Capture—*Document analysis*

General Terms

Algorithms, Design

Keywords

document analysis, metadata extraction, bibliographic references extraction, zone classification

1. INTRODUCTION

The amount of resources 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 records including information such as title, authors, keywords, abstract or bibliographic references. This requirement cannot be emphasized enough. Reliable metadata information is needed for organizing data, searching, building citation networks, computing impact factors, and many more tasks.

Unfortunately, sometimes a digital library has to deal with resources without metadata. Sometimes the metadata information is not trustworthy enough due to its incorrectness or incompleteness. In such cases the library needs a reliable method to extract metadata and references from documents at hand. In this paper we present OurSystem — a comprehensive system for automatic metadata and parsed bibliographic references extraction from born-digital scientific publications.

OurSystem is based on a modular workflow composed of three paths and a number of steps with carefully defined input and output. Thanks to such workflow architecture individual steps can be maintained separately. As a result it is easy to perform evaluation, training, improve or replace one step implementation without changing other parts of the workflow. For most implementations we used supervised and unsupervised machine-leaning 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 complete version of the system was presented in [6]. Since then we introduced the following changes and 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 performed the evaluation of key workflow steps and the whole metadata extraction path. For this purpose we used a large dataset composed of documents from PubMed [4].

A demonstration of OurSystem, as well as the source code, can be accessed online (LINK).

In the following sections we discuss the state of the art (section 2), describe in details the workflow architecture and

steps implementations (section 3) and finally report evaluation methodology and results along with basic error analysis (section 4).

2. 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.

For example, the Medical Article Records System (MARS) [5] is able to process document's scanned pages in TIFF format. Also Flynn *et al.* [11] present a system, in which documents are first OCR-ed and converted to XML, and metadata is extracted using a rule-based approach.

Nowadays we have to deal with growing amount of born-digital documents which do not require individual character recognition. This difference affects both the workflow architecture and the performance of the metadata extraction system. The approaches to this problem differ in the scope of the solution, supported file formats and methods and algorithms used.

For example Giuffrida *et al.* [13] describe a metadata extraction system, which processes PostScript files using a tool based on `pstotext`, and metadata is extracted by a set of rules and features computed for extracted text chunks.

Rigamonti *et al.* [21] present a reverse engineering tool processing PDF documents in order to extract the physical layout structure along with the logical structures. The system has been evaluated against a set of representative newspapers front pages.

Metadata extraction process presented by Esposito *et al.* [10] is able to process both PS and PDF formats. In this approach page segmentation is done by a kernel-based method and zones are classified based on machine-learning approach.

Marinai [17] uses JPedal package to extract characters from PDF documents, page segmentation is done using rule-based approach, and finally a neural classifier is used for zone classification.

In the approach proposed by Cui and Chen [9] text extraction and page segmentation are done by `pdftohtml`, a third-party open-source tool, and metadata is extracted using Hidden Markov Models.

The TeamBeam algorithm proposed by Kern *et al.* [15] extracts a basic set of metadata from PDF documents. BDF-Box library is used to process PDF content and key classification step is done by enhanced Maximum Entropy classifier.

3. METADATA AND REFERENCES EXTRACTION WORKFLOW AND ALGORITHMS

OurSystem is able to process documents in PDF format. Currently the workflow does not include any OCR phase, it analyses only the PDF text stream found in the input document. As a result, PDF documents containing only the images of pages will not be properly processed. The workflow inspects the entire content of the document and produces two kinds of output: the document's metadata and parsed bibliographic references.

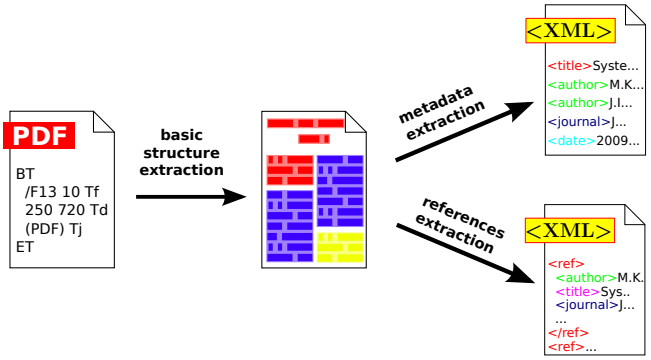

Figure 1: Workflow architecture. At the beginning the basic structure is extracted from the PDF file. Then metadata and bibliographic references are extracted in two parallel paths.

There are three main parts of the workflow (Figure 1):

- At the beginning the basic structure is extracted from the file. This part is common for both workflow paths. The output is a geometric hierarchical structure holding the document's text content. The structure is composed of pages, zones, lines, words and characters; every zone is labelled with one of five general categories: METADATA, REFERENCES, BODY, OTHER and UNKNOWN. More details about the meaning of general categories can be found in section 3.1.5.
- Metadata extraction path takes as input parts of the geometric hierarchical structure labelled as METADATA and extracts document's metadata from it.
- References extraction path analyses parts of the geometric hierarchical structure labelled as REFERENCES and the result is a list of document's parsed bibliographic references.

One of the most important changes since the first version of the workflow [6] is introducing the 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 two separate parallel paths. In the future we plan to extend the architecture by adding a third path in a similar way. Its goal will be the extraction of a structured document content composed of sections, subsections, headers and paragraphs.

3.1 Basic structure extraction

Basic structure extraction is the first part of the workflow, common for all extraction paths. The goal of this part is to process the input PDF document and extract its content in the form of geometric hierarchical structure. In this representation the document is composed of pages, each page is composed of zones, each zone is composed of lines, each line is composed of words and finally each word is composed of

individual characters. Each element of the structure stores the text content and coordinates on the document's page. The reading order is defined for all the elements of every structure level. Additionally, all zones are labelled with one of the five main categories. The output of basic structure extraction is document's hierarchical structure in TrueViz format [16].

Basic structure extraction is composed of four steps:

1. First individual characters along with their page coordinates are extracted directly from the input PDF file.
2. After that we group characters into words, words into lines and lines into zones, building the geometric hierarchical structure holding the document's text content.
3. Next, the reading order is computed for elements of all structure levels.
4. Finally, the document's zones are classified into five main categories: METADATA, BODY, REFERENCE, OTHER and UNKNOWN. This allows to split the document's content and redirect to specialized extraction paths.

3.1.1 *Character extraction*

Character extraction is the first step of the entire workflow. During this step, the PDF stream found in the input document is analysed and individual characters along with their page coordinates are extracted.

The implementation based on iText library [1] has not been changed since the first version of the process. More details can be found in [6].

3.1.2 *Page segmentation*

During the page segmentation step individual characters extracted previously are grouped into larger elements: words, lines, and zones. As a result the document's text content is represented by a hierarchical geometric structure composed of lists of pages, zones, lines, words and characters on consecutive levels.

The implementation is based on modified bottom-up Docstrum algorithm [20]. No changes were introduced since the first version of the process. More details can be found in [6].

3.1.3 *Reading order resolving*

The aim of setting the reading order is to determine the order, in which the document's elements should be read, by transforming a 2-dimension geometrical structure of text zones on a plane into a 1-dimensional stream of text zones. Reading order resolution is one of the major changes since the previous version. We have found it crucial to find out in which order blocks of text are intended to be read.

A sequence of text zones has an impact on zones' classification as there is an obvious correlation between classes of a zone and its predecessor. For instance, zones containing metadata appear usually at the beginning of the document and are preceded by other metadata zones. On the contrary,

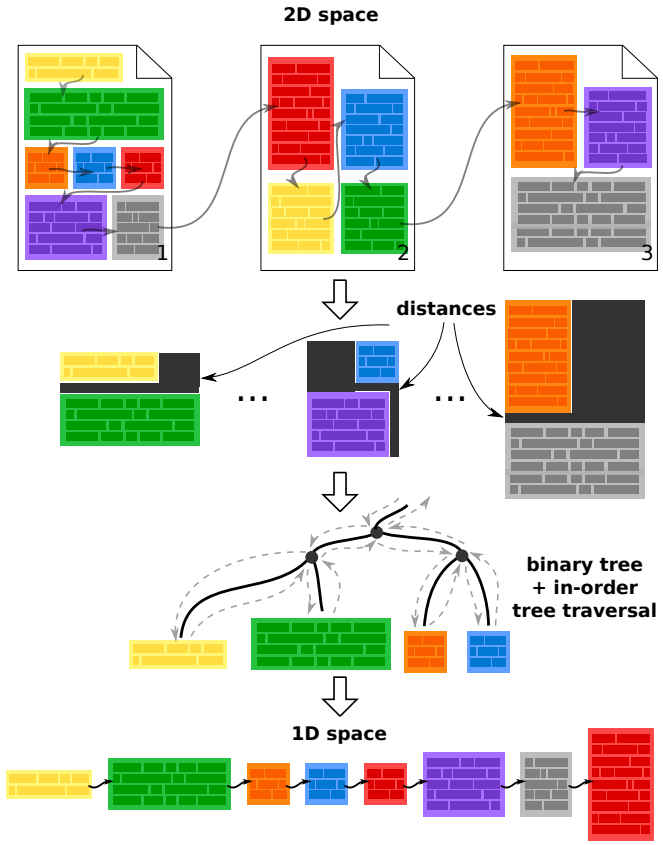

Figure 2: An outline of the idea of reading order calculation. Firstly distances between zones are calculated, whereas vertical alignments are prioritized. Then zones are clustered hierarchically into a binary tree. Finally, in-order tree traversal algorithm is applied to the tree.

references usually occur at the end of the document after a body or acknowledgment zone. Body zones almost never lie behind a reference zone, etc.

A PDF file contains by design a stream of strings that undergoes extraction and segmentation process. As a result, pages containing characters grouped into zones, lines and words are obtained, all of which have a form of unsorted bag of items. These elements have to be put in the order in which they appeared in the source input file. To this end, a bottom-up strategy is applied: firstly characters are sorted within words and words within lines in ascending order according to X coordinate value. Afterwards, lines are sorted with Y coordinate as the sorting key. Below we describe a heuristic responsible for setting order of text zones whose fundamental principle was taken from [3]. Its schematic diagram can be found in Figure 2.

Let first define directions: north, south, east and west as respectively top, bottom, right and left side of the page as seen by the reader. We say that two zones are vertically aligned if their vertical bounds share X coordinate. Similarly, we say that two zones are horizontally aligned if their horizontal

bounds share Y coordinate. In the presented algorithm we make use of a simple observation that the natural reading order descends from top to bottom, if successive zones are aligned vertically. Otherwise it traverses from left to right, if possible.

1. Calculate distance between each pair of zones on a particular page. This is done by finding the smallest rectangle covering both zones. From the area of yielded rectangle summed area of two zone rectangles is subtracted. Multiply this value by the absolute value of cosine of slope of the vector connecting their middle points. This is to ensure that zones aligned vertically are less distant then those aligned horizontally.
2. Put tuples containing pairs of zones together with the associated distances in a list. Sort this list by the distance and process it in FIFO order starting from the lowest distance.
3. Take an element from the list. If there is a zone between two unvestigated zones, move them to the end of the list and get the first element from the list.
4. Merge these two zones into one group of zones. Remove pairs containing merged zones from the list and calculate distance between the new object and existing objects. Add them to the list and sort. In this way we create hierarchy of objects in a form of a binary tree.
5. Say that if a node has two children, one of them is left child and one of them is one child. For every node swap its children so that each right node lies east, north-east, south-east or south from the left child.
6. Apply depth-first search algorithm on the tree. At each node select left node first. When reaching a leaf (which is always a zone, not a cluster of zones) yield it as a next zone of a sequence.

3.1.4 Classification tasks

In our system we employed two Support Vector Machines classifiers, each of which assigns a class from a set of classes specified for particular classifier. We made use of LibSVM library [8] being a robust and widely used implementation.

In order to perform zone classification, each zone is transformed into a vector of feature values. We restricted feature types to boolean, integer and float. To make them compliant with LibSVM they are all converted to floating point values and scaled linearly to the [0;1] interval according to bounds found in the learning samples at the learning stage. SVM classifiers do not take by design information about the sequence into account. In order to incorporate this information into classification process, we developed sequence-aware features such as classification decision of preceding zone or content of surrounding zones.

Implemented features are to a great extent common for both classification stages. We used 78 features that can be divided into several categories:

- geometrical — they are based on geometrical zones attributes such as: zone's width and height, X and Y coordinates on the page, distance to the nearest zone, zone's width to height ratio, amount of free space below and above a zone, etc.,
- lexical — they are based upon keywords characteristic for different parts of narration, this is: affiliation, acknowledgment, abstract, references, article type etc. A vast majority of publications follows the same structural guidelines naming particular parts of articles in the same way or using the same keywords,
- sequential — based on the sequence-related information, eg. class of the previous zone, presence of the same text blocks on the surrounding pages, etc.,
- formatting — eg. font size used in the current block, comparison of the font with adjacent zones, amount of blank space inside zones etc.,
- heuristics — eg. uppercase word count, percentage of numbers in a text block, if each line is started with enumeration-like tokens, etc.

Employed classification algorithm ensures that meaningless or noisy features are ignored, but can cause training and classification time penalty. As this is not the issue in OutProduct, we did not optimize this part.

We followed a model of classifier training problem as presented in [14]. 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]. For this task we prepared a constant set of documents drawn randomly from our dataset, which is widely described in section 4.1. At every grid point we performed a 10-fold cross-validation. In order to perform the optimization in reasonable time, we were forced to limit size of the dataset to 1,000 TrueViz files.

As this dataset is strongly imbalanced and we chose mean accuracy as the classification's quality indicator, training was susceptible to biasing towards the most frequently represented class. To counteract this problem, we undersampled majority classes to ensure that all classes are equally represented.

Parameters for the best obtained results are presented in Table 1. For the final evaluation, as well as for the production classifier in our service, we chose fourth degree polynomial kernel function and C and γ equal respectively to 2^6 and 2^{-5}.

3.1.5 Initial zone classification

The task of the initial classification step is to classify each zone into one of five general classes. These are:

- BODY — this class incorporates the proper publication's text, equations, equation labels, figures, figure captions, tables' content, table captions,

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 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

Table 1: The results of parameter search for initial zone classifier and metadata classifiation. The table shows the best mean accuracies over all classes for ten-fold cross-validation for particular kernel function types, as well as associated values of binary logarithms of C and γ.

- METADATA — this class includes text zones regarding article's abstract, authors' affiliations, article's bibliographic information, authors' correspondence, dates (e.g. revision, submission and publication date), article's academic editor, keywords, title and type (e.g. research paper, case study, technical report, etc.),
- OTHER — embraces copyright information, acknowledgments, conflicts of interests statements, page headers and footers, page numbers,
- REFERENCES — involves exclusively article's references at any place in the article,
- UNKNOWN — text zone having indeterminate class for the labelling person or application.

The label UNKNOWN is a result of non-ideal dataset that was tagged only to some extent. OurDataset dataset [7] contains 3.4% of text zones whose class was unknown to the human labelers. Similarly, Pubmed-based dataset contains zones that were not matched by our in-house automatic matcher with the content of provided NLM file. These are mainly zones that do not contain metadata nor proper article content. In our opinion it makes less harm to mark this potentially uninteresting zones as UNKNOWN, than to try to assign a class possibly making them pollute classes of bigger interest. We describe briefly the process of preparation of the dataset in section 4.1.

3.2 Metadata extraction path

The goal of metadata extraction path is to extract metadata information from the input document. This includes: title, authors, authors' affiliations, contact emails, editors, identifiers (DOI, ISSN, URN), abstract, keywords, dates (received, accepted, revised and published), journal name, volume, issue and pages range.

The output is the metadata record in NLM format [2] containing all extracted information.

Metadata extraction path consists of two steps:

1. Firstly, document's zones labeled as METADATA by the initial classifier are classified into specific metadata classes.
2. Secondly, atomic metadata information is extracted from labelled zones and resulting metadata record is formed.

3.2.1 Metadata zone classification

The goal of metadata zone classification is to classify all METADATA zone into specific metadata classes: ABSTRACT, BIB_INFO, TYPE, TITLE, AFFILIATION, AUTHOR, CORRESPONDENCE, DATES, EDITOR and KEYWORDS.

The implementation is based on an SVM classifier learnt using OurDataset and Pubmed-based data. In contrast to the initial classification, there is no UNKNOWN label as the training set for this algorithm was made by gathering zones labeled with mentioned labels in the used document dataset. All zones of imprecisely recognizable content was filtered out be the first step of classification. Thus, at this step the classifier is learnt to determine meaningful class for all test zones.

3.2.2 Metadata extraction

The objective of the final step is to extract atomic metadata information from labelled zones and to build the resulting metadata record. This involves a few simple, arbitrary selected operations, varying by the class of the zone, such as splitting authors or keywords list, merging zones containing abstract, cleaning up ligatures and hyphenated words, extracting atomic pieces of data like days, months and years from date zones, etc.

3.3 References extraction path

References extraction path is responsible for analyzing the document's zones labelled as REFERENCES in order to extract bibliographic references along with their metadata information. The result is the list of input document's parsed bibliographic references in NLM format [2].

References extraction path is composed of two steps:

1. First the content of REFERENCES zones are split into strings containing one bibliographic reference each.
2. Finally, individual reference strings are parsed and their metadata is extracted.

3.3.1 Extracting reference strings

In the geometric hierarchical structure extracted during the initial part of the process there are usually a few zones of category REFERENCES. Those zones contain a list of document's reference strings, each of which can be composed of one or more text lines. The goal of reference strings extracting step is to split the content of REFERENCES zones into individual reference strings by grouping text lines according to the document's layout.

Previous implementation of this step was based on Hidden Markov Models classifier and required a training phase. New version utilizes unsupervised machine-learning techniques,

which allows to omit time-consuming training set preparation and training phases, while achieving comparable extraction results.

In order to extract individual reference strings, first we recognize all lines, which are first lines of consecutive references. To achieve this, we represent all lines as vectors of features and cluster them into two sets. Cluster containing the first line of all references lines is assumed to contain all first references lines. After recognizing all first lines it is easy to concatenate lines to form individual reference strings.

Clustering method we use is complete linkage clustering. The distance between feature vectors is calculated using Euclidean metric. We also make use of a simple observation that first line in references zones is always the first line of the first reference. Before performing the actual clustering, we analyse the list of distances between the first feature vector and all other feature vectors in order to approximate a good maximum distance used during clustering.

Feature vectors we use contain five features. The features are based on line length, indentation, spaces between lines and text content of the line.

3.3.2 Reference strings parsing

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

First the reference strings are tokenized. After that the tokens are represented as vectors of features and labels are assigned to tokens. 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. The previous version of the classifier was based on Hidden Markov Models. New classifier employs Conditional Random Fields and is built on top of GRMM and MALLET packages [19].

We use 42 features to describe the tokens:

- features based on the presence of a particular character class, eg. digits or lowercase/uppercase letters,
- features checking if the token is a particular character (eg. a dot, square bracket, a comma or a dash),
- features checking if the token is a particular word,
- features checking whether 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.

4. EVALUATION

We performed separate evaluation of each step of metadata extraction. Extensive description of evaluation of initial zone classification, metadata zone classification and metadata extraction can be found below in sections 4.5, 4.6 and 4.7, respectively.

4.1 Dataset preparation

Dataset used for training of classifiers consists of two subsets. The smaller part is OurDataset containing 113 PDF documents in PDF and corresponding ground truth documents in TrueViz format, selected to contain possibly many different layouts. This dataset was tagged semi-automatically — the results of imperfect extraction algorithms were corrected by trained human experts. Thus, it is considered to be a highly reliable source of data. More details about the dataset and its preparation can be found in [7].

Data in NLM files vary greatly in quality from perfectly labelled down to containing no valuable information. Poor quality NLM results in sparsely tagged zones in TrueViz files. Hence, it was necessary to filter documents whose zones are classified in satisfying measure. If poor quality metadata was associated with particular publishers, choosing only highly covered documents could result in eliminating particular publishers and their layouts, which was to be avoided. Figure 3 shows a histogram of documents with specified percentage of zones labeled with classes. One can see that there are many documents having more than 80% of zones tagged. Following this fact, we randomly selected over 8,000 TrueViz files that have at least 80% labeled zones. Each zone was transformed into a training sample consisting of vector of feature values and classification decision.

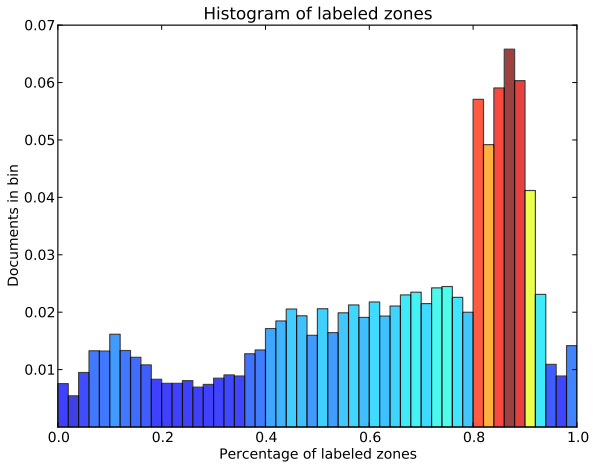

Figure 3: Histogram of documents in the Pubmed-based dataset having given percentage of zones with an assigned class value

Tables 2 and 3 present respectively number and percent of total of initial training samples and metadata training samples in both subsets. As can be noticed, in Pubmed-based dataset much more zones were unmatched and therefore tagged as UNKNOWN. Due to its size it was also not examined by an expert. Therefore we consider this dataset to

	Author	Title	Journal	Pages	Year
Original	4161	3400	3252	1974	3222
Extracted	4089	3391	3220	1967	3205
Success	3230	3028	2947	1877	3066
Precision	**78.99%**	**89.30%**	**91.52%**	**95.42%**	**95.66%**
Recall	**77.62%**	**89.06%**	**90.62%**	**95.09%**	**95.16%**

Table 6: The results of reference parsing evaluation. The table shows the numbers of original, extracted by the parser and successfully extracted metadata fragments. Additionally precision and recall for successfully extracted fragments are shown.

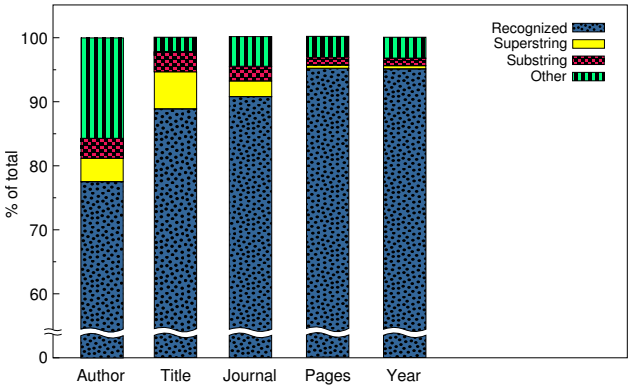

Figure 4: The results of reference parsing evaluation. The diagram shows the percentage of correctly recognized reference metadata, metadata for which the superstring or substring were identified and other errors.

be of worse quality in terms of classification accuracy, however covering more layouts seen in the wild. It our solution we join these two datasets trying to obtain the best of the two worlds. In the sections 4.5 and 4.6 we will discuss the results of training of classifiers using combination of the two datasets.

We made no assumptions about layouts supplied to our production system. Hence, we decided to include as many different layouts as possible in the training set to make OurSystem perform well on wide range of input file. We did not restrict publication's year nor language as this could eliminate some layouts making them poorly recognizable in the production system. We wanted to make our results fully reproducible. To achieve this, we used in evaluation exclusively open access articles and made available all used datasets.

4.2 Reference extraction evaluation

For evaluation purposes we used documents from OurDataset dataset [7] and a few manually chosen documents with less common references layout. The whole dataset contains 138 documents with 4,619 references in total. Clustering-based references extractor extracted 4,789 reference strings, out of which 4,400 references were extracted correctly, achieving precision 91.88% and recall 95.26%. From 109 (78.99%) documents all references were correctly extracted and 124 (89.86%) documents had at least 80% of correctly extracted references.

4.3 Reference parsing evaluation

For reference parser evaluation we used datasets CiteSeer [12] and Cora-ref [18]. The datasets were 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). The resulting set contains the following metadata labels: *author*, *title* and *journal*, *pages* and *year* and a special label *text*.

We performed 5-fold cross-validation on the resulting dataset. For all five metadata labels (author, title, journal, pages, year) we compared strings from original references ("expected" strings) with strings generated by the parser ("extracted" strings). We consider a metadata fragment of a certain label correctly extracted if the extracted string is identical as 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 metadata fragments the parser extracted a little less than expected (extracted string is a substring of expected string).

The results are shown in Table 6 and Figure 4. Table 6 shows the number of original, extracted by the parser and successfully extracted metadata fragments of all labels, as well as their precision (the ratio of successes number to extracted number) and recall (the ratio of successes number to expected number). Figure 4 shows what fraction of expected fragments were successfully extracted fragments, superstrings or substrings of expected strings, and other errors.

4.4 Combining OurDataset and Pubmed

We performed a 5-fold cross-validation on different combinations of OurDataset and Pubmed-based datasets. Each combination has different ratio of OurDataset to Pubmed training samples, from 1:0 up to 1:20 and from 0:05 up to 0:10.

We intended to check if enhancing moderate quality data with refined data has positive impact on the process of zone classification and what is the best proportion of such a combination.

In figures 5 6 one can see variation of precision and recall against the OurDataset-PubMed combination ratio for initial zone classification and metadata classification respectively. Results are sorted against the total size of dataset.

As a result of a human-based review of OurDataset, the best results are clearly obtained for 1:0 ratio. This outcome is hard to met both by combined and purely PubMed-based dataset, however it is easy to observe that the bigger a dataset is, the better results are yielded. Hence, the improvement in results depends exclusively on dataset size.

4.5 Initial zone classification evaluation

	BODY	META	REFERENCES	OTHER	UNKNOWN	total
OurDataset	15,623 (77.7%)	2,332 (11.6%)	431 (2.1%)	1024 (5.1%)	693 (3.4%)	20,103
Pubmed	345,053 (44.0%)	181,241 (23,0%)	64,888 (8.2%)	92,511 (11.8%)	101,122 (12.9%)	784,815

Table 2: Number of training samples per class per subset for initial zone classification

	ABSTRACT	AFFIL.	AUTHOR	BIB_INFO	CORRESP.	total
OurDataset	227 (9.7%)	149 (6.3%)	114 (4.9%)	1419 (60.8%)	50 (2.1%)	2,333
Pubmed	22,006 (12.1%)	13,699 (7.6%)	8,982 (5.0%)	55,352 (30.54%)	4,713 (2.6%)	181,241
	DATES	**EDITOR**	**KEYWORDS**	**TITLE**	**TYPE**	**total**
OurDataset	92 (3.9%)	39 (1.7%)	27 (1.2%)	113 (4.9%)	103(4.4%)	”
Pubmed	49,425 (27.3%)	3 (0.0%)	2,821 (1.6%)	19,716 (10.9%)	4524 (2.5%)	”

Table 3: Number of training samples per class per subset for metadata classification

For the purpose of initial zone classification evaluation the whole prepared dataset was exported to LibSVM format. This resulted in 768,836 zones, out of which 20,121 came from OurDataset. We did a 5-fold cross-validation with random undersampling of training zones in terms of their classes. Confusion matrix together with precision and recall values is to be found in table 4.

One can see that that this part of classification gives satisfactory results. Classes BODY, METADATA nad REFERENCES were frequently recognized, whereas class OTHER got worse results. It might be caused by unprecise definition of what belongs to the class OTHER - it encapsulates both page numbers and acknowledgments blocks. The latter might be not sufficiently different from BODY blocks in terms of developed features. This might be fixed by incorporating acknowledgment/conflict statement block into BODY, implementing better features allowing to better distinguish between switched classes or elemininating OTHER class and treating everything besides metadata and references as BODY.

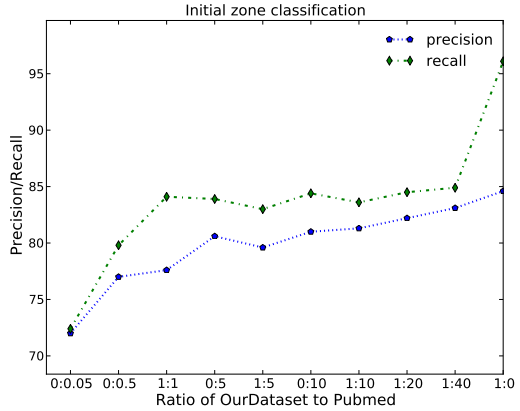

Figure 5: Precision and recall for initial zone classification for different datasets. The x-axis represent combination ratio between OurDataset and Pubmed, where 1 unit equals 2000 text zones.

4.6 Metadata classification evaluation

Similarly to initial zone classification, in this step we exported all metadata zones as training samples to LibSVM format. We conducted five-fold cross-validation for set of training samples built from OurDataset and Pubmed combined in ratio 1:20. Its results are presented as confusion matrix in table 4.1. We see that these results are better than for initial classification, with precisions and recalls being equal or greater than 90%. One od the exceptions is TITLE class, where 666 abstract zones are recognized as titles and 102 titles were recognized as abstracts. This might be an effect of poor page segmentation, where for non-standard layouts two blocks can converge into, whose class can't be correctly recognized. There are 136 BIB_INFO zones recognized as title. This is in turn caused by poor Pubmed-based dataset, where a title taken from NLM file was a sign to a de facto bibliographic information zone, containing for instance journal name and title of the paper. There are 125 abstract rezognized as keywords, what makes KEYWORD's precision fall to 80.73%. This again caused by wrong segmentation effecting in zones cotaining keywords and abstracts being melted into one piece.

4.7 Metadata extraction evaluation

The evaluation of metadata extraction was performed on Pubmed-based data set. We took PDF files as an input and conducted the whole process of metadata extraction. Eventually the output metadata was compared with metadata from NLM files. For each type of metadata we used different measures of correctness. Article's title, volume, issue, ISSN, DOI, URN, dates and pages were considered correct if extactly equal to Pubmed NLM data. Comparison of dates was divided into two categories: year and entire date. Publisher name, abstract, and journal title were tokenized and the Smith-Waterman distance with no penalty for injection of tokens nor misalignments was applied. This allows some flexibility as not all tokens of compared data have to equal and their sequence doesn't have to be equal.

Evaluation documents were drawn randomly from the subset of Pubmed-based dataset whereas minimum and maximum fraction of labeled zones were equal respectively 0.7 and 0.8 Evaluation set didn't include training documents. We com-

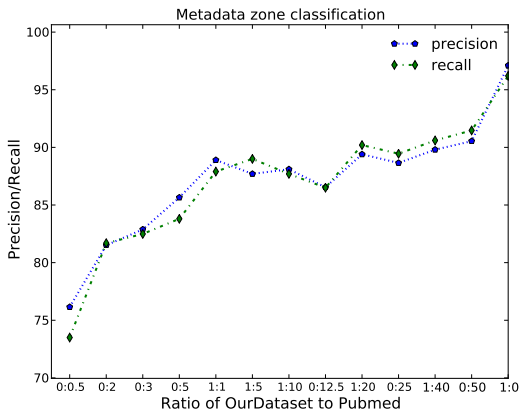

Figure 6: Precision and recall for metadata classification for different datasets. The x-axis represent combination ratio between OurDataset and Pubmed, where 1 unit equals 2000 text zones

pared each piece of metadata from Pubmed NLM files with the output produced by OutProduct. The results are shown in the table 7.

5. CONCLUSIONS AND FUTURE WORK

We presented a system for extracting metadata and parsed bibliographic references from scientific articles in born-digital form. We discussed in details the architecture of the extraction workflow and the implementations of all workflow steps. We highlighted the changes introduced since the previous version of the system and reported evaluation methodology and results.

We performed extensive evaluation of particular components of the process and obtained results. We prepared a prominent dataset of ground-truth, PDF and metadata files that we made available to the public.

Our future plans include:

- expanding the workflow architecture by adding another process path for extracting structured full text containing sections and subsections, headers and paragraphs,
- improvement of Pubmed-based dataset in terms of accuracy of automatic zone labeling,
- improvement of text processing heuristics,
- better keywords and author zones handling,
- performing the evaluation of references extraction treated as a whole using the dataset composed of documents from MEDLINE repository.

6. REFERENCES

[1] iText. http://itextpdf.com/.

[2] NLM. http://dtd.nlm.nih.gov/archiving/.

[3] PdfMiner. http://www.unixuser.org/ euske/python/pdfminer/.

[4] PubMed. http://www.ncbi.nlm.nih.gov/pubmed.

[5] Automating the production of bibliographic records for MEDLINE. Technical report, 2001.

[6] Author1, Author2, Author3, and Author4. Metadata extraction system. In *Conference*, 2012.

[7] Author1, Author2, Author3, Author4, and Author5. Ourdataset. In *Conference*, 2012.

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

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

[10] 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*, 138:105–138, 2008.

[11] 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*, pages 327–336, 2007.

[12] C. L. Giles, K. D. Bollacker, and S. Lawrence. CiteSeer: An automatic citation indexing system. In *Proceedings of the third ACM conference on Digital libraries*, pages 89–98. ACM, 1998.

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

[14] C. Hsu, C. Chang, and C. Lin. A practical guide to support vector classification. 2003.

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

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

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

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

[19] A. K. McCallum. MALLET: A Machine Learning for Language Toolkit. 2002.

[20] L. O'Gorman. The document spectrum for page layout analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1162–1173, 1993.

[21] 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)*, pages 1050–1054, 2005.

	BODY	META.	REFER.	OTHER	UNKNOWN	precision	recall
BODY	**50999**	1288	642	4515	2148	96.16	85.58
META.	435	**24247**	171	1324	661	87.54	89.97
REFER.	246	281	**7461**	303	522	86.07	84.66
OTHER	837	904	236	**1169**	788	58.06	79.01
UNKNOWN	1078	836	236	1169	**10356**	71.54	75.73

Table 4: Confusion matrix for the initial zone classification for 5-fold cross-validation. Rows represent desired classification decision, columns represent obtained decisions.

	AUTHOR	TITLE	EDITOR	DATES	AFFIL.	ABSTRACT	TYPE	KEYWORDS	BIB_INFO	CORRESP.	precision	recall
AUTHOR	**2860**	21	1	2	19	15	17	2	16	39	92.98%	95.59%
TITLE	56	**6180**	0	8	8	102	34	43	56	2	86.99%	95.24%
EDITOR	1	0	**13**	0	2	0	0	0	0	0	92.86%	81.25%
DATES	7	5	0	**16012**	159	44	10	1	496	8	93.70%	95.64%
AFFILIATION	38	19	0	31	**4323**	56	11	11	32	87	91.92%	92.92%
ABSTRACT	15	666	0	41	59	**6507**	14	125	23	10	95.13%	87.23%
TYPE	17	44	0	4	2	15	**1372**	3	27	1	90.32%	92.39%
KEYWORDS	6	30	0	1	4	50	2	**838**	3	3	80.73%	89.43%
BIB_INFO	48	136	0	986	49	40	58	13	**17554**	10	95.38%	92.91%
CORRESP.	28	3	0	4	78	11	1	2	6	**14351**	89.87%	91.52%

Table 5: Confusion matrix for the metadata classification for 5-fold cross-validation. Rows represent desired classification decision, columns represent obtained decisions.

	recall
title	60.8%
date full	54.9%
date year	96.1%
volume	65.2%
DOI	60.8%

	avg. string adjustment
abstract	88.1%
journal title	91.36%

	precision	recall
authors	40.5%	37.5%
keywords	42.9%	49.6%

Table 7: Quality measures for metadata extraction.