Automatically Identify and Label Sections in Scientific Journals Using Conditional Random Fields

Sree Harsha Ramesh[1(✉)], Arnab Dhar[1], Raveena R. Kumar[1], Anjaly V.[1], Sarath K.S.[1], Jason Pearce[2], and Krishna R. Sundaresan[1]

[1] Surukam Analytics, Chennai, Tamil Nadu, India
{harsha,arnab,raveena,anjaly,sarath,krishna}@surukam.com
[2] Newgen KnowledgeWorks, Chennai, Tamil Nadu, India
jason@newgen.co

Abstract. In this paper, we describe a pipeline that automatically converts a journal article in the PDF format to an XML which conforms to NLM JATS DTD. First, the text and typographical features are extracted from the document using character level information. Then, we use a trickle down multi-level conditional random fields based classifier where at each level the pre-trained CRF model classifies a given line of text into one of the tags of DTD at a particular depth and feeds the resulting tag into the next level model as a feature. After identifying tags upto level three, we make use of separate supervised models for parsing authors, affiliations, references and citations. We employ heuristic based methods for matching affiliation to authors, and citation to references. The JATS XML thus generated, is converted into an RDF document. SPARQL queries are run on the RDF, to address the queries of Task 2 of the Semantic Publishing Challenge.

Keywords: Multi-level CRF · BIO encoding · NLM JATS · JATS2RDF

1 Introduction

Scientific journals have been typically published in the PDF format over the years and continue to do so ever-increasingly. However, the Portable Document Format (PDF) is optimized for presentation – it was created to be independent of application, hardware, and operating system [1] – but lacks structural information about sections within the documents and their labels and other such metadata, thus making indexing of documents for search and retrieval, very difficult. This necessitates systems which automatically parse journal content, annotate different sections and extract metadata.

Semantic Publishing Challenge (SemPub)[1] is a series of shared tasks, organised alongside the annual academic conference - Extended Semantic Web Conference (ESWC). The current edition of 2016 has three related tasks, with the

[1] Semantic Publishing Challenge 2016 - https://github.com/ceurws/lod/wiki/SemPub2016.

H. Sack et al. (Eds.): SemWebEval 2016, CCIS 641, pp. 255–266, 2016.
DOI: 10.1007/978-3-319-46565-4_20

overarching theme of "annotating a set of multi-format input documents and to produce a Linked Open Data that fully describes these documents, their context, and relevant parts of their content".

This paper describes our submission to the Task 2 of Semantic Publishing Challenge 2016. As is the motivation behind this task, the approach described in this paper helps "provide a deeper understanding of the context in which a paper was written", by extracting contextual information from full text PDFs. The entities required to be extracted in this edition of the challenge were affiliations, countries in affiliations, supplementary material, titles of first-level sections, table and figure captions, funding agencies and EU projects that supported the research.

To this effect, we have created a system that predominantly employs Conditional Random Fields (CRF) [2]. CRF belongs to a class of probabilistic graphic models and is especially popular in sequence labelling, because of the context-aware predictions it can be trained to make, unlike ordinary classifiers. Each CRF module works on the output of the previous models and also includes a post-processing phase where we use heuristics to handle sparse edge cases which could not be learned by the algorithm. In Sect. 3, we describe the architecture of our system along with the tools which were used while implementing it.

2 Related Work

A rule-based approach to extract heading-based sectional hierarchies of HTML documents is proposed in [5]. HTML DOM (Document Object Model) tree analysis was used to identify the section and subsections of the documents. Some heuristics employed in this paper are that the section headings usually don't begin with punctuation symbols, that they are enclosed within line breaks, and that they are typographically[2] distinct from rest of the document. The DOM tree is processed to obtain blocks of elements in the document and it identifies the possible headings in the document. The obtained hierarchy is then rearranged based on the identified headings.

In [6] a method is described to identify section headers in Legal Briefs. To identify the section and type of section different machine learning approaches such as – naive Bayes, logistic regression, decision trees, support vector machines and neural networks were used. Informations about italics, bold are ignored and considering only analysis of the word and character sequences. Similar to [5], a header block of text is identified heuristically and is labelled using multi class classifiers.

The "Author and affiliation extraction and matching system" described in [7] consists of two steps – an extraction step and a matching step. Extraction step employs CRF, in which each token in the author and affiliation string is classified into one of the three classes, name (author name or affiliation tokens), symbol (correspondence marker) and separator (any separating token). Two models were

[2] Typographical features include information about typefaces, point size and line length.

learned for name and affiliation, using same feature set but, different training data. The feature set contains content features (related to textual content) and layout features (related to text layout). The second step, relational classification matches affiliations to authors using SVM.

The challenge of annotating content from PDF documents has been an open problem, with concerted efforts put forth by ESWC in solving it. SemPub has led to the development of tools [10,11] that automatically annotate elements within PDFs,thereby generating linked open data out of academic publications. The following papers, which were among the best submissions at the SemPub, define the state-of-the-art.

A processing pipeline was developed in [4] to analyze the structure of a PDF document incorporating machine learning techniques. The extraction of contiguous text blocks from raw data employs an unsupervised machine learning method. A supervised machine learning mechanism, combination of Maximum Entropy and Beam Search is employed for the extraction of metadata. The reference section was identified by using a set of heuristics and, reference string parsing process used a supervised machine learning approach based on CRF.

CERMINE [3], the state-of-the-art system for extraction of structured affiliations from scientific articles in PDF format, follows a modular approach towards metadata extraction. Document structure extraction and content classification are done using SVM, so as to decipher the reading order of a document in PDF. Author and affiliation extraction were done using heuristics and affiliation and reference parsing were done using CRF.

3 The Pipeline

This section explains each phase of the proposed architecture shown in Fig. 1. In each sub-section dedicated for explaining a particular phase of extraction, we also list the tools and resources used.

The first phase of the pipeline is the PDF extraction phase which extracts line and character level information from the journal articles in the PDF format. This serve as the basis for the subsequent models. An exhaustive list of features are extracted from this information. The feature file is fed into a sequence of three CRF models, where the model at each level classifies a given line of text into one of its respective categories and feeds the result into the next level as a feature.

We have followed the NLM Journal Archiving Tag Set (JATS) 3.0 DTD[3] specifications for generating an XML out of the text extracted from the PDF Extraction phase, and also defining the classes at each level of the CRF algorithms. Having identified the NLM tag three levels deep, there is an independent fine-grained classification phase for entities like affiliation, author, references and citation identification and mapping of citation to reference and mapping of

[3] NLM JATS DTD. http://dtd.nlm.nih.gov/archiving/tag-library/3.0/index.html.

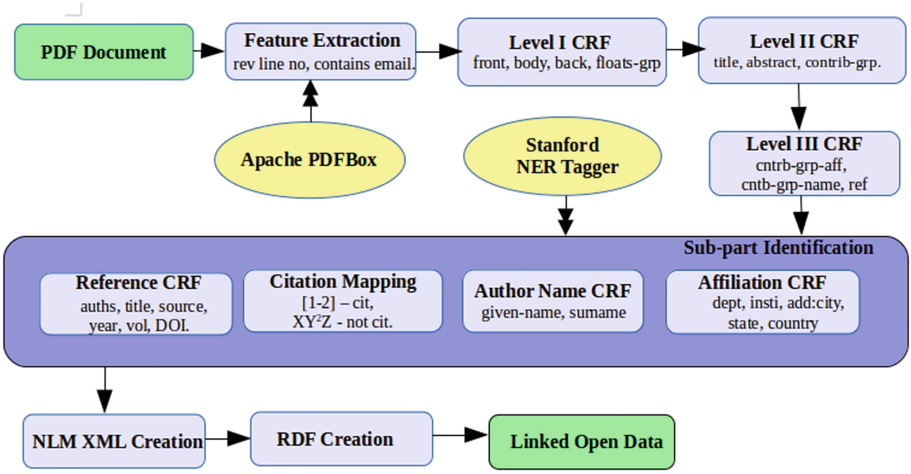

Fig. 1. A schematic diagram depicting the overall architecture of our system to automatically extract semantic information from PDF documents and convert it to NLM JATS XML and further convert it to RDF triples.

affiliation to author. At the end of this stage we have an XML which conforms to NLM JATS and then we generate RDF[4] using JATS2RDF XSLT transform.

3.1 PDF Extraction

We have used Apache PDFBox[5] library to extract the requisite information from the PDFs. PDFBox provides a stream of characters and their bounding boxes. A bounding box consists of four coordinates: the X and Y of the lower left corner, and the X and Y of the upper right corner of the character, where X is the width measured from the left margin of the page and Y is the height measured from the top of the page. It also provides information about the font of the character, such as the font family and font weight. We use the following heuristics to extract some metrics which will be further used in the feature extraction phase.

1. All the characters which lie at the same height belong to the same line of text.
2. Absence of any characters in particular region, across a span of rows indicates the presence of multiple columns.
3. If a character is of a smaller size than its predecessor but is roughly half its size above or below the Y coordinate of the lower left corner of the characters in the same line, indicates that it is superscript or subscript respectively.

[4] Resource Description Framework (RDF), http://www.w3.org/RDF/.
[5] Apache PDFBox. https://pdfbox.apache.org/.

3.2 Feature Extraction

The following features were generated from the raw features extracted in the PDF Extraction, phase and are subsequently fed into the Level I CRF, Level II CRF and Level III CRF models. In hindsight, we found that Klampfl and Kern [4] used a similar list of features and we have followed their method of categorizing them as follows:

Formatting Features. *fontSize* and *lineLength* are set[6] if the calculated value for a given line of text lies within 1 Standard Deviation from the feature value averaged over the entire list of sentences. *isCommonFontFamily* is set if the font family name of the current line is the same as the most occurring one in the document. Typical font family names are Times Roman and Helvetica. *lineNo* is the line number of the given line within the entire document. *fwdLineNoWithinPage* and *revLineNoWithinPage* are the forward line number and reverse line number within a given page respectively. *isBold* is set if the given line of text consists only of *bold* characters. As mentioned in PDF Extraction, PDFBox provides us with the font weight value of a given character – a value of 400 indicates normal weight whereas 700 indicates that it is bold.

Vocabulary Features. *containsIntroduction*, *containsAbstract*, *containsKeywords*, *containsReference*, *containsAcknowledgment*, *containsTOC*, *containsAssociatedContent*, *containsSupportingInformation*, *containsSynopsis*, *containsAuthorInfo*, *containsSection*, *containsFigure* and *containsTable* are set based on the presence of the respective word or its common misspellings, in the given line of text.

Heuristic Features. *containsTelephone*, *containsFax* and *containsEmail* are set based on respective regular expression matches. *containsSuperscript* is set, if the PDF Extraction phase found any superscript in the given line. *punctuationRatio* and *capsRatio* are encoded as categorical features based on the bucket – the bucketing ranges are heuristically determined – the punctuation count belongs to.

Language Modelling Features. We used Stanford 3-class NERTagger[7] [8] to identify the words which are Location, Person or Organization in a given sentence. Then we calculated features such as *departmentRatio* and *nameRatio* as the ratio of the length of the words tagged as "ORGANIZATION" or "PERSON" respectively, to the length of the given sentence.

[6] If it is a binary feature, then set would mean setting the value to 1. If it is a multi-categorical feature, then the values are discrete integers ranging from 0 to (number of buckets - 1).

[7] Stanford NER Tagger: http://nlp.stanford.edu/software/CRF-NER.shtml.

3.3 Level I CRF

To use the CRF algorithm described in [2], we have used the open-source implementation – CRF++[8]. The inputs to this tool are a feature file which has a whitespace-separated (spaces or tabular characters) list of features and a feature template file. The feature template file describes the semantics of these columns and also, enables the creation of the context windows. These windows allow the CRF++ package to look at the previous and next tokens to infer about the current state and decipher the context.

The first level CRF model is trained by feeding the feature file in CoNLL[9] format, generated in Sect. 3.1 along with the target classes such as front, body, back and floats-group. These classes correspond to the first level of the JATS XML.

1. *front*: It is a container element for article metadata.
2. *body*: It contains the main body of the article.
3. *back*: It contains references.
4. *floats-group*: It contains figures, tables and their captions and titles.

3.4 Level II CRF

The second level CRF model uses the same feature file as compiled in the Feature Extraction phase — Sect. 3.2, with one additional feature being the first level class predicted by the Level I CRF. The target classes for this model are *document-title, abstract-group, contrib-group, kwd-group, sec, ack, ref-list, fig, table-wrap*. These classes generally correspond to the nodes at a depth of 2 in the JATS XML.

1. *document-title*: It is a container element for article name.
2. *contrib-group*: It contains author details and affiliation information.
3. *abstract-group*: It is a container element for abstract, synopsis dek, and table of contents.
4. *kwd-group*: It contains a list – usually comma separated — of keywords that is associated with the whole document.
5. *sec*: A child of body element in Level I, this contains one or more paragraphs of main text and their section headers.
6. *ack*: This element contains acknowledgement.
7. *ref-list*: This element contains a list of references which are usually found at the *back* part of the document.
8. *fig*: This element consists of figure label and caption.
9. *table-wrap*: Within this tag we find table rows and columns and also table captions and footnotes.

[8] CRF++: https://taku910.github.io/crfpp/.
[9] CoNLL: http://www.cnts.ua.ac.be/conll2000/chunking/.

3.5 Level III CRF

The feature file used by the third level classifier is compiled by adding the second level class prediction as a new feature to the feature file used by the Level II CRF — Sect. 3.4. The target classes are mostly the same as those for the second level classifier, except *contrib-group* which is further classified into *contrib-group-contrib-name* and *contrib-group-aff*. These classes are not in the JATS DTD, but were chosen for our convenience. To identify first level section headings of a paper, i.e., Q4 of Task 2 - we have also added *p* and *head* classes to this model.

1. *contrib-group-contrib-name*: It is a container element for author name - given name and surname. The lines tagged as *contrib-group-contrib-name* are fed as inputs to the Author Name CRF in Sect. 3.6.
2. *contrib-group-aff*: It contains affiliation parts, namely, department name, institute name, street address, city, postal code, state and country. The lines tagged as *contrib-group-aff* are fed as inputs to the Affiliation CRF in Sect. 3.7.

3.6 Author Name CRF

This module uses CRF to identify given-name and surname regions of a line predicted as *contrib-group-contrib-name* by Level III CRF.

Here, we have assumed that these lines have names only, and that they do not have any affiliations. This is a heuristic we deduced from the CEUR dataset[10] released for the challenge.

Sentences are split on the tokens — comma and "and" — into individual names. Every name is assigned a unique author id, so as to map it to the corresponding affiliation, in a later step. The Author Name CRF model is trained on typographical features like *characterCase*, *tokenLength*, *isSingleCapitalLetter* and a keyword feature — *tokenAsFeature*, where the full token is fed back as a feature so as to help the model to memorize names.

BIO Tagging. The target class names, such as *given-name*, *surname* are encoded using BIO tagging scheme [9]. BIO encoding divides the tokens belonging to a certain tag, as either being begin-of-entity (B_X) or continuation-of-entity (I_X).

For e.g., *Sree Harsha Ramesh* where *Sree Harsha* is the GIVEN_NAME and *Ramesh* is the SURNAME could be tokenized and tagged as follows: *Sree* - B_GIVEN_NAME; *Harsha* - I_GIVEN_NAME; *Ramesh* - B_SURNAME. Once the text line containing author-name is tokenized and the textual features of each individual token are identified, it is passed through the learner. The learner will classify each token as either B_GIVEN_NAME, I_GIVEN_NAME, B_SURNAME or I_SURNAME.

[10] A subset of scientific journals published CEUR-WS.org - https://github.com/ceurws/lod/wiki/SemPub16_Task2#training-dataset-td2.

After classification, the tokens with consecutive B_X and I_X of the same tag are merged so as to form the complete token. For example, if we have a sequence of token-prediction tuples like (*Jean*, B_GIVEN), (-, I_GIVEN), (*Baptiste*, I_GIVEN), the tokens belonging to this list are merged to form the complete given name - *Jean-Baptiste*.

3.7 Affiliation CRF

This module uses CRF to identify entities such as department, institution, street-address, postal-code, city, state and country in lines predicted to be *contrib-group-aff* by Level III CRF.

In each sentence, superscripts and affiliation markers like *, †, ‡ and § are tagged as affiliation labels.

The Affiliation CRF model is trained on language features such as part-of-speech(POS) tag identified using Stanford Log-linear Part-Of-Speech[11] tagger and NER tag identified using Stanford Named Entity Recognizer. It also uses the following keyword-based features:

1. *isDepartmentMultiLang,isUniversityMultiLang*: simple heuristics to check if a token belongs to a list of words denoting university or department in languages like German, French, Italian, Spanish and English.
2. *isCity,isCountry*: The token is checked against a database of city-names[12] and Gazetteer list of countries.
3. *hasAddress*: The token is searched for words like 'road', 'street' and postal code patterns which denote address.

The target classes are BIO encoded. After decoding the tags like in Sect. 3.6, we obtain full classes such as department, institution, street-address, postal-code, city, state and country.

Also, affiliation markers[13] found in the Author Name CRF — Sect. 3.6, are mapped to their corresponding aff-ids identified in this section.

3.8 Citation Mapping

Citations are alphanumeric strings embedded in the document body that denote an entry in the bibliographic references section of a research article. Among citation systems, Vancouver system[14] uses sequential numbers either bracketed or superscript or both. The numbers refer to either footnotes, endnotes or references. Parenthetical referencing also known as Harvard referencing[15] has in-text

[11] Stanford Log-linear Part-Of-Speech Tagger - http://nlp.stanford.edu/software/tagger.shtml.

[12] Maxmind Free World Cities Database - https://www.maxmind.com/en/free-world-cities-database.

[13] Symbols like *, †, ‡ and §, or numbers 0–9.

[14] Vancouver System of Referencing - https://en.wikipedia.org/wiki/Vancouver_system.

[15] Harvard Referencing - https://en.wikipedia.org/wiki/Parenthetical_referencing.

citations enclosed within parentheses. Our model is trained to identify citations that follow the Vancouver system, which is predominantly the style followed by the papers published by CEUR-WS.org.

From the Level III CRF — Sect. 3.5, sentences which are classified as *sec* are tokenized and passed through the model created for citation identification.

Citation identification model is trained on POS tag, NER tag, punctuation features like *containsParentheses*, keyword features like *containsEtAl*, typographical features like *isBold* and *isSuperscript*. The target classes — CITATION and NOT_CITATION, are BIO Encoded.

3.9 Reference Parsing

A bibliographic list item usually contains one or more citations describing a referenced work. This module works best for atomic references, which contain a single citation.

From the Level III CRF — Sect. 3.5, text lines which are classified as *ref-list* are tokenized and passed through the model created for reference parsing. Each reference string is tokenized by splitting at the occurrence of every whitespace character. Punctuation symbols such as *comma*(,), *semicolon*(;), and *hyphen*(-) are considered to be individual tokens, because these symbols have an inherent meaning in the particular reference style used in a document.

To train a CRF model, each token in the tokenized ref line, is tagged with the textual and layout features such as *leadingCaps*, *hasCaps*, *allCaps*, *allNums*, *isYear*, *trailingDot*, *isHyphen* and *digitRatio*. The target classes are BIO encoded.

After resolving the BIO encoded tags like in Sect. 3.6, we obtain full classes such as Author, Article Title, Source, Year, Volume, Issue, DOI, First Page, Last Page, Note and URI.

3.10 NLM XML and RDF Creation

In the previous sections, author names and their corresponding affiliations are resolved into one xml node called *contrib-group*, and similarly the references module generates a separate xml node called *ref-list*. Likewise, the remaining annotations are compiled into a DOM conforming to the NLM JATS standard.

To generate an RDF from the consolidated XML, we have used an XSLT transform called JATS2RDF [12] that automates the creation of RDF metadata from a JATS-marked up document. The mappings defined in this transform function are based on various SPAR[16] ontologies and also other vocabularies such as the Dublin Core Metadata Initiative (DCMI) Metadata Terms and the Friend of a Friend (FOAF) Vocabulary.

Table 1 shows example RDF mappings for JATS elements such as country and affiliation. For the JATS elements not mapped in the JATS2RDF transform

[16] SPAR - the Semantic Publishing and Referencing Ontologies is an integrated ecosystem of various ontologies like DoCO and CiTO.

Table 1. JATS to RDF mapping

Element/attribute name	XML example	RDF translation
country (child of address and aff)	<aff> <country> XXX < /country> < /aff>	:this-agent-contact-info vcard:address [a vcard: Address; vcard:country-name "XXX"].
aff	<contrib> .... <aff>....< /aff> .... <contrib>	:this-agent pro:holdsRoleInTime [pro:withRole scoro:affilate ; pro:relatesToOrganization :this-organization pro:relatesToDocument :conceptual-work]

The first column gives the JATS element or attribute name, the second shows an example XML usage from the JATS specification, and the third describes the mapping of to RDF.

function, but which are relevant to this challenge, such as *fig* and *table-wrap*, we have defined mappings based on the DoCO[17] ontology.

4 Results and Evaluation

We trained our system on the 146 workshop papers published by CEUR-WS.org that were released as training datasets for 2015[18] and 2016[19] challenges. The results shown here would be updated with the performance scores on the evaluation dataset which is yet to be released.

Table 2 shows the training level precision, recall and f-scores for individual tags described in the Level I, Level II and Level III CRF modules. The values seen in the table were computed using the perl script - conlleval.pl[20].

The consolidated NLM XMLs were generated and transformed into RDF documents as explained in Sect. 3.10. Then we ran the SPARQL queries written specifically to address the aforementioned eight queries of Task 2. When the 320 queries (40 different queries for each of the challenge's 8 queries) were posed against the populated knowledge base, our system yielded an average F-measure of 0.612 (Precision: 0.629, Recall: 0.62), when evaluated against the gold data, as calculated by the tool - SemPubEvaluator[21] released for this challenge. These figures are different from the reported F-measure of 0.389 (Precision: 0.393, Recall: 0.428)[22] due to the presence of empty lines in the output files, which were misconstrued for query output.

[17] Document Components Ontology (DoCO), http://purl.org/spar/doco.
[18] https://github.com/ceurws/lod/wiki/SemPub15_Task2#training-dataset-td2.
[19] https://github.com/ceurws/lod/wiki/SemPub16_Task2#training-dataset-td2.
[20] http://www.cnts.ua.ac.be/conll2000/chunking/conlleval.txt.
[21] https://github.com/angelobo/SemPubEvaluator.
[22] https://github.com/ceurws/lod/wiki/SemPub2016#winners.

Table 2. CRF training accuracy for sentence level annotation

	Tags	Precision	Recall	F score
First level	front	93.61	80.38	86.49
	body	96.71	98.09	97.39
	back	91.9	96	93.91
	floats-group	58.06	30	39.56
Second level	abstract-group	95.24	99.55	97.35
	ack	93.10	48.21	63.53
	contrib-group	98.94	96.88	97.89
	document-title	96.77	96.77	96.77
	fig	93.62	100.00	96.7
	sec	98.35	99.11	98.73
	kwd-group	100.00	80.00	88.89
	ref-list	96.47	99.75	98.08
	table-wrap	100.00	81.25	89.66
Third level	contrib-group aff	94.23	97.35	95.77
	contrib-group contrib name	91.43	78.05	84.21
	p	98.51	99.21	98.86
	head	69.08	54.12	60.69

In every row, the tag and its corresponding precision, recall and F score values are shown.

5 Conclusion and Future Work

In this paper, we proposed an approach to automatically identify sections and their labels by using a sequence of multiple CRF models where each model builds on the predictions of the preceding models. A JATS XML was then generated which was transformed into an RDF document, thereby contributing towards building a knowledge base. As part of SemPub-2016, we have addressed all of the eight queries of Task 2. We have shown that our post-challenge evaluation scores are comparable against those of the pattern matching approaches presented at SemPub-2016, thereby underscoring the value of integrating machine learning and natural language processing techniques into information retrieval systems. Future work would focus on improving the overall accuracy of the meta-data extraction module of our system by training on a bigger, annotated dataset.

References

1. Rosenthol, L.: Developing with PDF: Dive Into the Portable Document Format. O'Reilly Media Inc., Sebastopol (2013)
2. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of ICML, pp. 282–289 (2001)

3. Tkaczyk, D., Szostek, P., Fedoryszak, M., Dendek, P.J., Bolikowski, Ł.: CERMINE: automatic extraction of structured metadata from scientific literature. Int. J. Doc. Anal. Recogn. (IJDAR) **18**, 317–335 (2015). Springer
4. Klampfl, S., Kern, R.: Machine learning techniques for automatically extracting contextual information from Scientific Publications. In: Gandon, F., Cabrio, E., Stankovic, M., Zimmermann, A. (eds.) SemWebEval 2015. CCIS, vol. 548, pp. 105–116. Springer, Heidelberg (2015). doi:10.1007/978-3-319-25518-7_9
5. Pembe, F.C., Güngör, T.: Heading-based sectional hierarchy identification for HTML documents. In: 22nd International Symposium on Computer and Information Sciences, ISCIS, pp. 1–6. IEEE (2007)
6. Vanderbeck, S., Bockhorst, J., Oldfather, C.: A machine learning approach to identifying sections in legal briefs. In: MAICS, pp. 16–22 (2011)
7. Do, H.H.N., Chandrasekaran, M.K., Cho, P.S., Kan, M.Y.: Extracting and matching authors and affiliations in scholarly documents. In: Proceedings of the 13th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 219–228. ACM (2013)
8. Finkel, J.R., Grenager, T., Manning, C.: Incorporating non-local information into information extraction systems by Gibbs sampling. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics (2005)
9. Ramshaw, L.A., Mitchell, P.M.: Text chunking using transformation-based learning (1995). arXiv preprint: arXiv:cmp-lg/9505040
10. Iorio, A.D., Lange, C., Dimou, A., Vahdati, S.: Semantic publishing challenge – assessing the quality of scientific output by information extraction and interlinking. In: Gandon, F., Cabrio, E., Stankovic, M., Zimmermann, A. (eds.) SemWebEval 2015. CCIS, vol. 548, pp. 65–80. Springer, Heidelberg (2015). doi:10.1007/978-3-319-25518-7_6
11. Lange, C., Di Iorio, A.: Semantic publishing challenge – assessing the quality of scientific output. In: Presutti, V., et al. (eds.) SemWebEval 2014. CCIS, vol. 475, pp. 61–76. Springer, Heidelberg (2014)
12. Peroni, S., Lapeyre, D.A., Shotton, D.: From markup to linked data: mapping NISO JATS v1.0 to RDF using the SPAR (Semantic Publishing and Referencing) ontologies. In: Journal Article Tag Suite Conference (JATS-Con) Proceedings 2012 [Internet]. National Center for Biotechnology Information (US), Bethesda (MD) (2012). http://www.ncbi.nlm.nih.gov/books/NBK100491/