An Automatic Workflow for the Formalization of Scholarly Articles' Structural and Semantic Elements

Bahar Sateli and René Witte

Semantic Software Lab
Department of Computer Science and Software Engineering
Concordia University, Montréal, Canada

Abstract. We present a workflow for the automatic transformation of scholarly literature to a Linked Open Data (LOD) compliant knowledge base to address Task 2 of the Semantic Publishing Challenge 2016. In this year's task, we aim to extract various contextual information from full-text papers using a text mining pipeline that integrates LOD-based Named Entity Recognition (NER) and *triplification* of the detected entities. In our proposed approach, we leverage an existing NER tool to ground named entities, such as geographical locations, to their LOD resources. Combined with a rule-based approach, we demonstrate how we can extract both the structural (e.g., floats and sections) and semantic elements (e.g., authors and their respective affiliations) of the provided dataset's documents. Finally, we integrate the LODeXporter, our flexible exporting module to represent the results as semantic triples in RDF format. As the result, we generate a scalable, TDB-based knowledge base that is interlinked with the LOD cloud, and a public SPARQL endpoint for the task's queries. Our submission won the second place at the SemPub2016 challenge Task 2 with an average 0.63 F-score.

1 Introduction

The *Semantic Publishing Challenge*, which started in 2014, is a recent series of competitive efforts to produce linked open datasets from multi-format and multi-source input documents. The 2016 edition of the challenge[1] pursues the goal of assessing the quality of scientific output through an automatic analysis of scholarly documents for fine-grained contextual information. The dataset under study consists of 85 papers (45 training and 40 evaluation) taken from computer science workshop proceedings, published by CEUR-WS.org. The goal is to extract authors, affiliations, funding bodies, as well as a document's sections and floating elements and ultimately, populate a knowledge base with the results. Such information, when extracted, can be used to automatically provide an overview of the authors' scientific output, the universities and research institutions involved, as well as the active funding bodies in a domain.

In this paper, we present our automatic workflow we developed to address Task 2 of the challenge that can extract structural and semantic elements from the full-text of a document and transform the detected entities into RDF triples, which are eventually made persistent in a scalable, TDB-based knowledge base with a public SPARQL endpoint.

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

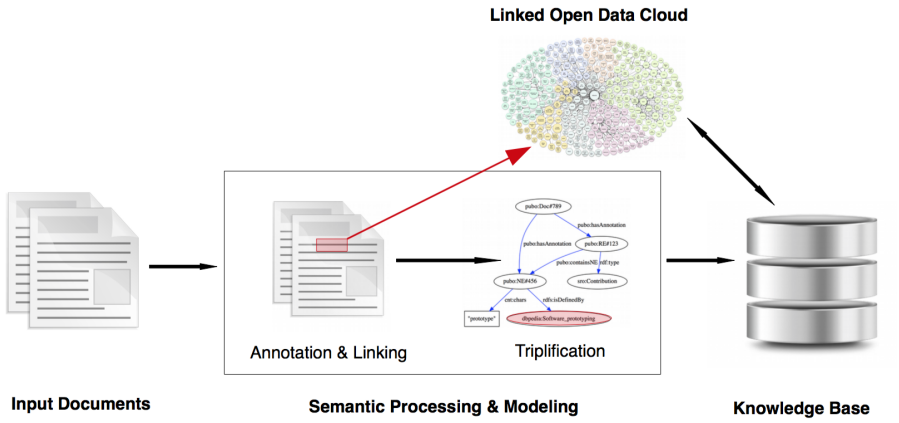


Fig. 1: Overall workflow of our text mining pipeline

The generated knowledge base is evaluated against a set of eight pre-defined queries for its correctness and completeness and exploited as a means of assessing the quality of scientific production in the respective workshops.
The challenge queries are concerned with searching for entities, categorized as follows:

- Authors, their Affiliations (**Q2.1**) and the country where the affiliation is located in (**Q2.2**);
- Supplementary Material (**Q2.3**), mentioned as part of the authors' contributions;
- Structural elements, in particular first-level Sections (**Q2.4**), and floating elements like Tables (**Q2.5**) and Figures (**Q2.6**); and
- Names of Funding Agencies (**Q2.7**) and European Projects (**Q2.8**) supporting the research presented in the paper.

Note that you can find supplementary material, such as the populated knowledge base and the text mining pipeline resources at http://www.semanticsoftware.info/sempub-challenge-2016.

2 Design

We designed an automatic workflow (Fig. 1) that starts from a set of documents (called a *corpus*), which go through multiple processing phases, and produces semantic triples as output. The *Syntactic Processing* phase breaks down the full-text of the documents into smaller segments and pre-processes the text for further semantic entities. The *Semantic Processing* phase takes the results of the syntactic analysis and attempts to annotate various entities in a text. Finally, each document's annotations are translated into semantic triples, according to a series of custom *mapping rules*, and stored in a knowledge base.

2.1 Syntactic Processing

In order to remove the various formatting and typesetting styles of the input documents, rather than working directly with the PDF files, we scrape the text from them and then process the extracted plain text. The full-text of a document is segmented into *tokens* – smaller, linguistically meaningful parts, like words, numbers and symbols. Subsequent syntactical processing components process the tokenized text into sentences and all sentence constituents are tagged with a Part-of-Speech category, like noun, verb or adjective. Finally, we lemmatize the tokens in a text to store their root format in order to ignore their morphological variations (e.g., "university" and "universities" will have the same canonical form).

2.2 Semantic Processing

The semantic analysis of the pre-processed text is conducted in an iterative fashion. In each iteration the semantic analysis method uses the output of the upstream processing steps in order to generate new *annotations* used incrementally to generate the desired entities.

Gazetteering. In text mining applications, gazetteers play the role of carefully curated, readily available knowledge resources. Essentially, gazetteers are dictionaries of entities with pre-defined semantics, e.g., the list of all countries or months of a year. In our approach, we use several gazetteers to annotate tokens, like "*Department*" or "*Institute*" that the downstream processing algorithms use, for example, to detect names of universities or research institutions in the authors' affiliations.

Named Entity Linking. In this year's challenge, we integrate a LOD-based Named Entity Recognition (NER) tool that allows us to ground various types of named entities in the documents to their corresponding resources on the LOD cloud, using a Universal Resource Identifier (URI). This approach not only alleviates the burden of manually updating and curating our gazetteers, e.g., of country names, but also by virtue of traversing the semantic links of a semantic named entity, we will are able to find additional, machine-readable information where needed. In our approach, we rely on an existing NER tool that can link the surface form of a document's terms to LOD resource, which we further filter to retain only nouns and noun phrases.

Rule-based Pattern Matching. We developed a set of hand-crafted rules that capture various syntactical (grammatical) structure of the entities of interest in Task 2. Most of the rules rely on the annotations generated from the pre-processing phase: the tokens, their root forms, their POS tag, as well as all the entities in a text that were matched against our gazetteers. Our rules look at pre-defined, hand-crafted sequences of annotations and when a match is found, generate an annotation with a semantic type, e.g., an Affiliation annotation.

Authors and Affiliations. We focused the authors and affiliations extraction to the region of text between a document's title and abstract section (which we call the metadata body), since both were present in all of the training set documents. The detection of authors largely reuses the rules from our last year's submission to the challenge [1]. We start from our gazetteers of common first names. For instance, all first name tokens followed by an upper initial token in the metadata body are annotated as Authors in a text.

For this year's challenge, we developed a new approach for detecting affiliations: Using a NER tool, we first annotate all tokens in the metadata body that can be mapped to a resource in the LOD cloud with either a location, city or country semantic type. Subsequently, we apply our pattern matching rules against the sequence of word and symbol tokens in text adjacent to the grounded location entities. The rules below show exemplary patterns, where the semantic types are shown in capital letters:

(1) (UPPER INITIAL) (University | Universität | Universidad) (of | de) (TOWN | CITY | COUNTRY)
↪ matched strings: "*Universidad de Chile*", "*Technische Universität Wien*"

(2) (COUNTRY ADJECTIVE) (ORGANIZATION) (of | de) (UPPER INITIAL)
↪ matched strings: "*Indian Institute of Technology*", "*National Institute of Aging*"

In addition to institutions names, our pipeline also detect various organizational units like research centres, departments and research group naming patterns. If an organizational unit is found in a text in adjacency of an institution name (e.g., a university name), then the longest span of text covering both entities will be annotated as the Affiliation annotation.

Affiliation Locations. This year, we employed a new approach in detecting where an affiliation is located. The goal is to find the country name of the affiliations in the metadata body. To this end, our new component applies a set of heuristics with a fallback strategy that looks at the Affiliation annotations and the country named entities generated by the NER tool. While investigating the training set documents, we found out that detecting such a relation is a complex task: First, we observed that the line-by-line scraping of the documents' text often mixes up the order of the metadata body entities. This way, e.g., in a two-column affiliation information, the university names are scraped in one line next to each other and the next line has both of their country information. Second, there exists a number of documents in the training set, which do not have the country name in the metadata body. Our *Affiliation–Location* Inferrer component, optimistically applies a set of rules on the metadata body:

- In the case that there is only one affiliation and one country entity in the metadata body, it matches the two entities together with a high confidence.
- If there is more than one annotation of each type (i.e., Affiliation and Location), it makes two lists from the annotations sorted by their start offset in a text. It then iterates through the lists and matches each affiliation with a location that has a greater offset, but is located in the shortest distance from the affiliation annotation.
- Finally, if no Location annotation is available in the metadata body, it tries to find the country name from the DBpedia ontology. To do so, it first performs a DBpedia Lookup[2] operation using the affiliation label in the document in order to find a

[2] DBpedia Lookup, https://github.com/dbpedia/lookup

DBpedia resource URI. Subsequently, it executes a federated query against the public DBpedia SPARQL endpoint,[3] looking for triples where the subject matches the affiliation URI, and the predicate is one of 'country' or 'state' properties from the DBpedia ontology. If such a triple is found, it returns the English label of the object (country) and infers a relation between the affiliation and the country name.

Sections and Floats. We curated a set of *trigger* words in the documents that are used in figures and tables' captions, such as "*Fig.*", "*Figure*" and "*Table*". The detection of sections and floats are dependant on whether the text scraping process was also successful in segmenting the document. In the case where the segmentation is properly done, we merely check for the existence of our trigger words in the candidate segments, i.e., text boundaries detected by the scraping tool. If the boundary truly represents a caption, we further analyze it for symbols and numbers and classify the caption as either a Figure or a Table. In case we cannot find a number in the caption, a counter incrementally numbers the generated annotations, ordered by their starting offset.

We use a similar approach in detecting document Sections: If the text scraping tool cannot successfully extract the section headers, we check them against a gazetteer of conventional section headers (e.g., "Introduction", "Results and Discussion", or "Conclusion") to find matching strings. If no match is found, the pipeline will simply skip the section detection phase.

Funding Agencies. The challenge in finding the names of funding agencies is to distinguish the agency or organization names that funded the work presented by the authors of a paper. We previously investigated in [2] that so-called *deictic* phrases are used by authors in the context of scholarly discourse to refer the readers to the document under study, such as "*In this work*" or "*The presented paper*". We use the approach described in [2] to find deictic phrases in a document and look at the tokens and annotations following such phrases to elect a candidate sentences for funding agency name detection. Similar to our approach from last year, the agency name is detected as either *(i)* one or more upper-initial word tokens, or *(ii)* an annotated organization name.

(3) (DEIXIS) (VERB) (funded) (by | within) (the) (UPPER INITIAL | ORGANIZATION)
↪ matched string: "*This research is funded by the DebugIT project.*"

We plan to further improve the detection by incorporating a dependency parser in our text mining pipeline so that the funding agency names can be extracted from the noun phrase following the "*funded by*" verb phrase in the sentence's dependency tree.

2.3 Knowledge Base Construction

Using the rule-based text mining approach described above, we can extract the entities we are interested in to answer the challenge queries. The next step is to transform the extracted annotations into semantic triples and store them in a LOD-compliant knowledge base.

[3] DBpedia SPARQL endpoint, http://dbpedia.org/sparql

Semantic Vocabularies. In our semantic model, we reuse the vocabularies from the Document Components Ontology (DoCO) [3] to describe the structural elements of documents, namely, the sentences, sections and floats. For the authors and affiliations we use the FOAF[4] and Bibliographic Ontology (BIBO).[5] All other semantic classes and relationships, such as modeling annotations in a document, embedded annotations and the relations between annotations are described using our PUBlication Ontology (PUBO).[6]

LODeXporter. We use our *LODeXporter*[7] component in our text mining workflow to populate a knowledge base. It accepts a set of custom mapping rules as input and transforms the designated document's annotations into their equivalent RDF triples. For each annotation type that is to be exported, the mapping rules have an entry that describes: *(i)* the annotation type in the document and its corresponding semantic type, *(ii)* the annotation's features and their corresponding semantic type, and *(iii)* the relations between exported triples and the type of their relation. Given the mapping rules, the mapper component then iterates over the document's entities and exports each designated annotation as the subject of a triple, with a custom predicate and its attributes, such as its features, as the object.

3 Implementation

We implemented our text mining pipeline described in Section 2 based on the *General Architecture for Text Engineering* (GATE) framework [4], shown in Fig. 2. The pipeline accepts scientific literature in PDF or XML format from local or remote URLs as input and stores the extracted entities in form of an RDF document in a knowledge base as output. In this section, we provide the details of our GATE pipeline's processing resources.

3.1 Text Pre-processing

Different from our last year's approach [1], we use PDFX[8] [5] to transform PDF articles to XML documents. The GATE framework itself relies on the Apache Tika library[9] for extracting the textual content of the XML files, while preserving the original XML elements. For the scope of this paper, we exclude a discussion on text extraction from PDF documents; a comprehensive overview can be found in [5]. We also re-use GATE's ANNIE and Tools plugins [6] to tokenize and lemmatize the text, detect sentence boundaries, and perform gazetteering on the text.

[4] FOAF, http://xmlns.com/foaf/spec/

[5] BIBO, http://purl.org/ontology/bibo/

[6] PUBO, http://lod.semanticsoftware.info/pubo/pubo.rdf

[7] LODeXporter is currently considered to be in pre-release and available at http://www.semanticsoftware.info/lodexporter.

[8] PDFX, http://pdfx.cs.man.ac.uk

[9] Apache Tika, https://tika.apache.org/

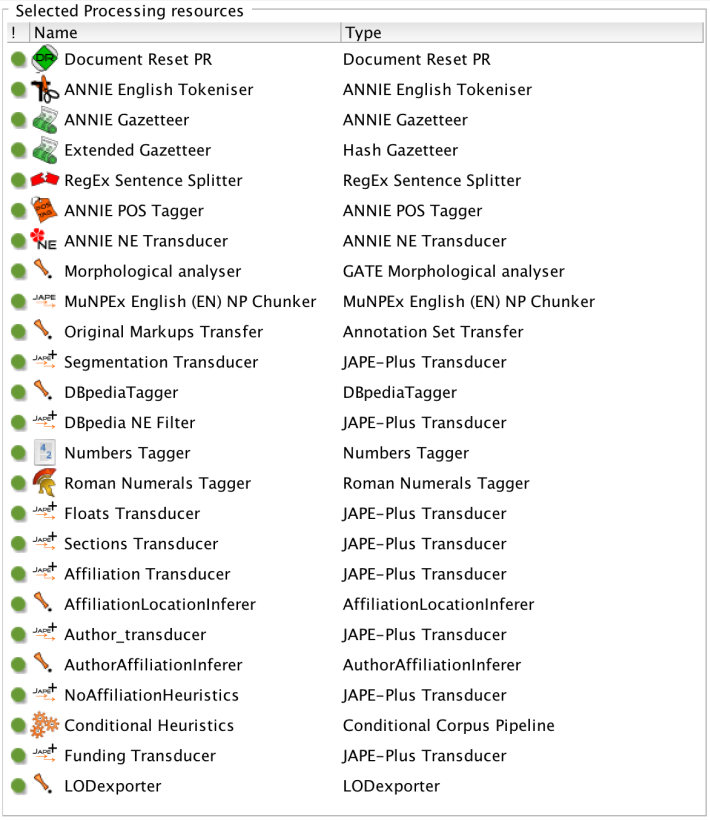


Fig. 2: The sequence of processing resources of our text mining pipeline

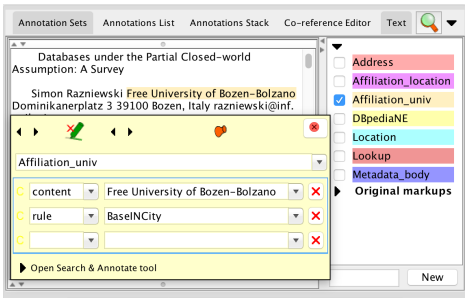
3.2 Named Entity Detection with Spotlight

To detect domain-specific entities in the documents, we rely on the LOD cloud, in particular DBpedia [7]. This provides for a rich, continuously-updated resource in a standard semantic format. By linking entities detected in documents to LOD URIs, we can semantically query a knowledge base for all papers on a specific entity (URI). For the actual entity tagging, we use an external tool, DBpedia Spotlight [8] version 0.7 with a statistical model for English. To integrate this web service into a GATE text mining pipeline, we use our *LODtagger* plugin.[10] This component sends the entire UTF-8 formatted text of a document as a RESTful POST request to a given Spotlight endpoint and receives the results in JSON format, which are subsequently parsed and transformed to GATE annotations. For the challenge queries, we annotate the metadata body of each document for city and country named entities, which are used in rules for affiliation detection in our pipeline.

[10] LODtagger, http://www.semanticsoftware.info/lodtagger

```
Rule: BaseINCity(
 ({Lookup.majorType=="org_pre"})?
 {Lookup.majorType == "org_base"}
 ({Token.orth == "upperInitial",
  !Lookup.majorType == "org_base"})?
 ({Lookup.majorType == "org_in"})+
 ({Affiliation_location.loc_type == "city"}
 | {Lookup.majorType == "location"})
):mention
-->
:mention.Affiliation_univ = {
   rule = "BaseINCity",
   content = :mention@cleanString }
```

(a) Example JAPE rule



(b) Detected annotation in GATE Developer

Fig. 3: Example JAPE rule (left) to extract an Affiliation entity and the generated annotation in GATE's graphical user interface

3.3 Rule-based Extraction of Contextual Entities

The rules described in Section 2.2 are implemented using GATE's JAPE language that provides for defining regular expressions over a document's annotations (by internally transforming them into finite-state transducers). Our text mining pipeline contains several *transducers* that apply our JAPE rules on the documents' text. Each JAPE rule has two main parts: a Left-Hand Side (LHS) pattern that essentially describes a regular expression over a document's annotations; When matched, the Right-Hand Side (RHS) of the rule adds a new semantic, typed annotation to the document. Additional information relevant to annotations are stored as their *features*. Fig. 3 shows an example JAPE rule used in our pipeline to extract an Affiliation annotation in a document. The rule shown matches a sequence of organizational base entries in our gazetteer (e.g., 'University') with an adverb (e.g., 'of'), followed by a city named entity.

Our submission this year also features a number of heuristics to automatically re-try detecting or correcting the authors and affiliations in a fuzzy manner, if the JAPE rules do not strictly match any patterns in a document. As shown in Fig. 2, if there is no Affiliation annotation is found in the document after all JAPE rules are executed, a set of conditional heuristics will then be applied on the document, for example, to blindly annotate spans of text between the last author name and the first mention of a country named entity, which may represent an Affiliation.

3.4 Knowledge Base Population

The mapping rules shown in Fig. 4 describe the specifications of exporting GATE annotations into several inter-connected semantic triples: Each Author annotation in the document should be exported with `<foaf:Person>` as its type, and its verbatim content in a text using the `<cnt:chars>` predicate. Similarly, Affiliation annotations are exported with their '*locatedIn*' feature describing the country they are located in from the GeoNames ontology (`<gn:locatedIn>`).[11] Subsequently, the value of the '*em-*

[11] GeoNames Ontology, http://www.geonames.org/ontology/documentation.html

```
@prefix map: <http://semanticsoftware.info/mapping#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix cnt: <http://www.w3.org/2011/content#> .
@prefix rel: <http://purl.org/vocab/relationship/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix gn: <http://www.geonames.org/ontology#> .

### Annotation Mapping ###
map:GATEAuthor a map:Mapping ;
       map:type   foaf:Person ;
       map:GATEtype  "Author" ;
       map:hasMapping map:GATEContentMapping .

map:GATEAffiliation a map:Mapping ;
       map:type   foaf:Organization ;
       map:GATEtype  "Affiliation" ;
       map:hasMapping map:GATEContentMapping ;
       map:hasMapping map:GATELocatedInFeatureMapping .

### Feature Mapping ###
map:GATEContentMapping a map:Mapping ;
       map:type   cnt:chars ;
       GATEattribute "content" .

map:GATELocatedInFeatureMapping a map:Mapping ;
       map:type   gn:LocatedIn ;
       GATEfeature "locatedIn" .

### Relation Mapping ###
map:AuthorAffiliationRelationMapping a map:Mapping ;
       map:type rel:employedBy ;
       map:domain map:GATEAuthor ;
       map:range map:GATEAffiliation ;
       GATEattribute "employedBy" .
```

Fig. 4: Excerpt of the mapping rules for exporting Authors, Affiliations and their relations

ployedBy' feature of each Author annotation is used to construct a `<rel:employedBy>` relation between an author instance and its corresponding affiliation instance in the knowledge base. We used vocabularies from our PUBO ontology wherever no equivalent term was available in existing Linked Open Vocabularies. For example, we use the `<pubo:containsNE>` property to build a relation between the metadata body and the entities that appear within its start and end offsets in a document. There exists several other mapping rules that we custom-tailored to model the extracted entities, so that the exported triples can be queried for the challenge Task 2. Ultimately, the LODeX-porter processing resource generates all of the desired RDF triples from the document's annotations, and stores them in a scalable, TDB-based[12] triplestore.

3.5 Query Results Export

We published the populated knowledge base described in the previous section through a Jena Fuseki[13] server. In order to conform to the output format required for the automatic

[12] Apache TDB, http://jena.apache.org/documentation/tdb/

[13] Jena Fuseki, https://jena.apache.org/documentation/serving_data/

Table 1: Quantitative analysis of Task 2 training and evaluation sets processing

Document Set	#Documents		#Pages			#Tokens			#Triples
	Full	Abstract	Min	Max	Avg	Min	Max	Avg	
Training	41	4	1	16	8.5	262	14401	3331	3880
Evaluation	38	2	1	15	6.9	293	13059	5306	3044

challenge evaluation tool,[14] we implemented a Java command-line tool that executes a set of hand-crafted queries against the Fuseki HTTP endpoint and transforms the results into Comma Separated Value (CSV) files. The command-line tool accepts a list of document URIs (e.g., `http://ceur-ws.org/Vol-1006/#paper2`) and a template for each of the challenge Task 2 queries. Each query template is designed to answer one of the Task 2 queries using same vocabularies explained in Section 3.4 and contains a placeholder for the document URI. Our query export tool parameterizes each query template with the correct URI form and performs a web service request against the knowledge base interface. Subsequently, the generated CSV files can be directly fed into the evaluator tool for generating our tool's performance report against the gold standard.

4 Results and Discussion

We analyzed the challenge training and evaluation sets of 45 and 40 documents, respectively, with our automatic workflow. The input documents are in PDF format and range from 3 to 16 pages for full-papers with various publisher-specific formatting. On average, each document contained 3331 and 5306 tokens in the training and evaluation sets, respectively. Table 1 shows the statistics of the document sets and the number of generated triples. The number of triples includes all the exported entities, their properties and inter-relationships, as well as the mapping rules themselves. On average, the workflow execution time is around 6 seconds per document (on a late 2013 MacBook Pro with 2.3GHz Intel Core i7 and 16 GB memory), where a majority of time is consumed by DBpedia Spotlight NER web service.

We evaluated the performance of our approach on the training set against the provided gold standard. Note that in our submission we only analyzed the documents to address queries Q2.1, Q2.2, and Q2.4–7. The SemPubEvaluator tool, however, computes the average metrics over all entries in the gold standard. On the training set, we obtained an average of 0.619, 0.598 and 0.605 for precision, recall and F-score, respectively. On the evaluation set, our approach achieved an average of 0.64, 0.629 and 0.632 for precision, recall and F-score, respectively, which placed us as the runner-up in the challenge.[15]

An error analysis of our results showed that many of Type I errors (i.e., false positives) are the results of fuzzy detection of affiliation names, where the span of annotated text

[14] SemPubEvaluator, https://github.com/angelobo/SemPubEvaluator

[15] The best performing tool reported 0.775, 0.778, and 0.771, for precision, recall and F-score, respectively. The complete scoreboard is available on https://github.com/ceurws/lod/wiki/SemPub2016.

also contained characters that were not in the gold standard, or when the boundary of the author names had mistakenly overlapped with the affiliations, e.g., in the case of Hasso Plattner Institute. Type II errors (i.e., false negatives), on the other hand, were mostly due to affiliation names in languages other than English, or when the organizational unit of the affiliation was missing from the annotation spans.

5 Conclusions

Semantic publishing research aims at making scientific publications readable and semantically understandable to computers. The long-term vision is to enable automated discovery and consumption of scholarly artifacts, like articles and datasets, by humans and machines alike. In this paper, we provided the details of our automatic workflow for the extraction of contextual information from full-text of computer science articles to address Task 2 of the Semantic Publishing Challenge 2016. We described our text mining pipeline which uses a rule-based pattern matching approach, combined with a named entity recognition tool to annotate the challenge datasets with various semantic entities. We then explained how we integrated our LODeXporter plugin to export the annotations to RDF triples, thereby populating a LOD-compliant knowledge base, in which the entities are inter-linked with resources on the linked open data cloud.

References

1. Sateli, B., Witte, R.: Automatic Construction of a Semantic Knowledge Base from CEUR Workshop Proceedings. In: The 12th Extended Semantic Web Conference (The Semantic Publishing Challenge 2015). Volume 548 of Semantic Web Evaluation Challenges: SemWebEval 2015 at ESWC 2015, Portorož, Slovenia, May 31 – June 4, 2015, Revised Selected Papers., Portoroz, Slovenia, Springer, Springer (06/2015 2015) 129–141
2. Sateli, B., Witte, R.: Semantic representation of scientific literature: bringing claims, contributions and named entities onto the Linked Open Data cloud. PeerJ Computer Science **1**(e37) (2015)
3. Shotton, D., Peroni, S.: DoCO, the Document Components Ontology (2011)
4. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V., Aswani, N., Roberts, I., Gorrell, G., Funk, A., Roberts, A., Damljanovic, D., Heitz, T., Greenwood, M.A., Saggion, H., Petrak, J., Li, Y., Peters, W.: Text Processing with GATE (Version 6). University of Sheffield, Department of Computer Science (2011)
5. Constantin, A., Pettifer, S., Voronkov, A.: PDFX: Fully-automated PDF-to-XML Conversion of Scientific Literature. In: Proceedings of the 2013 ACM Symposium on Document Engineering (DocEng '13), New York, NY, USA, ACM (2013) 177–180
6. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In: Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02). (2002)
7. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: Dbpedia: A nucleus for a web of open data. Springer (2007)
8. Mendes, P.N., Jakob, M., García-Silva, A., Bizer, C.: DBpedia Spotlight: Shedding Light on the Web of Documents. In: Proc. of the 7th International Conf. on Semantic Systems, ACM (2011) 1–8