Ontology-Driven Information Extraction from Research Publications

Vayianos Pertsas[1(✉)] and Panos Constantopoulos[1,2]

[1] Department of Informatics, Athens University of Economics and Business, Athens, Greece
{vpertsas, panosc}@aueb.gr
[2] Digital Curation Unit, Athena Research Centre, Athens, Greece

Abstract. Extraction of information from a research article, association with other sources and inference of new knowledge is a challenging task that has not yet been entirely addressed. We present Research Spotlight, a system that leverages existing information from DBpedia, retrieves articles from repositories, extracts and interrelates various kinds of named and non-named entities by exploiting article metadata, the structure of text as well as syntactic, lexical and semantic constraints, and populates a knowledge base in the form of RDF triples. An ontology designed to represent scholarly practices is driving the whole process. The system is evaluated through two experiments that measure the overall accuracy in terms of token- and entity- based precision, recall and F1 scores, as well as entity boundary detection, with promising results.

Keywords: Information extraction from text · Ontology population
Linked data · Knowledge base creation

1 Introduction

Extracting and encoding the knowledge contained in a research article is a multi-dimensional challenge. For instance, detecting who has done what, their interests and goals, affiliations, etc., requires extracting, analyzing and mapping onto an appropriate schema information from the metadata of the article. Also, several kinds of named entities need to be recognized (e.g. method employed in an experiment) and linked to other relevant information. Established named entity recognizers offer pre-trained models that support "common" types of named entities such as: Location, Person, Organization, Money, Events, and 'miscellaneous' [1]. For "non-common" types of named entities (e.g. 'tools', 'methods'), a classifier needs to be trained using annotated corpora, specifically created by human annotators, an expensive, time consuming process. Furthermore, to capture the information contained in a publication about a scholarly activity, its context and outcomes, entities and relations of many different types have to be extracted, which differ considerably from named entities in that they extend over widely variable lengths of text, even in more than one sentences. Every possible aspect of context needs to be exploited: from surface/lexical form, to part of speech and deep syntactic role of each token in a sentence; and from discourse structure in sections and paragraphs, to the role and position of each sub-sentence in a sentence.

E. Méndez et al. (Eds.): TPDL 2018, LNCS 11057, pp. 241–253, 2018.
https://doi.org/10.1007/978-3-030-00066-0_21

Finally, the output of the above tasks needs to be aligned in a semantic framework for comparison or integration with other existing knowledge published as linked data.

In this paper we present Research Spotlight (RS), a system that extracts information from research articles, enriches it with relevant information from other Web sources, organizes it according to the Scholarly Ontology (SO) [2], and republishes it in the form of linked data. Existing information is leveraged by accessing SPARQL endpoints, scraping Web pages or through APIs. Harvested information is further used as background knowledge for training classifiers or for extracting information from semi-structured or unstructured texts. So, RS generates linked, contextualized, structured data describing research activities and their outcomes, thus addressing the growing need for integrated access to information scattered in different publications.

Knowledge bases created using RS can support researchers in finding details of relevant work without reading the articles; discovering uses of resources, processes and methods in particular contexts; promoting communities of interests; formulating future directions and project proposals. Besides, funders and research councils can get a "bird's eye view" of scholarly work useful for planning and evaluation.

2 Related Work

To the best of our knowledge the exact task of extracting information from scientific article and republishing it as Linked Data, as prescribed in this paper, has not been addressed yet. That said however, several past efforts aimed at extracting information from text based on an existing ontology: In [3] RDF triples are extracted from RSS feeds and published as Linked Open Data using mappings to DBpedia entities. The focus is on statistical methods and rules based on lexical form. Syntactic dependencies of tokens in the sentence that could allow better context understanding are not exploited. The DBpedia project itself [4] is a huge operation to automatically extract knowledge from Wikipedia pages and info-boxes involving various NLP and feature-matching extractors that create RDF triples as instances of the DBpedia ontology. Here predefined rules are based on the DBpedia schema, metadata mappings, statistics of page links or word counts, and a number of feature extractors that exploit xml/html tags. However, the lexical, syntactic or structural analysis of raw text is not supported. In [5] an ontology is used to guide the automatic creation of RDF triples from facts previously extracted from various Web pages and to publish linked data in a SESAME triple store. Here too, the methods employed exploit string features of noun phrases, distributions of text found around those phrases in other Web pages, and the HTML structure of the Web pages containing the noun phrases. In [6] a knowledge base is created with information extracted from French news wires by linking extracted entities to the instances of an ontology that unifies the models of GeoNames and Wikipedia and contains entities of type Person, Organization or Location retrieved from these sources. Common types of named entities are recognized and aligned with an existing database. In domain-specific endeavors, such as [7], an ontology is defined from fragments of CIDOC-CRM in order to describe the domain of Arts, on the basis of which knowledge is extracted from various Web pages in order to create personalized biographies of artists. Only common types of named entities are supported. In [8], a knowledge base is

constructed by semi-automatic extraction of relations, based on the PRIMA ontology for risk management and a combination of machine learning techniques and predefined handcrafted rules. Syntactic dependencies that could yield patterns exploiting the deeper syntactic structure of sentences are not considered. In [9], an event ontology is used in order to guide NLP modules in extracting instances from unstructured texts in a semi-supervised manner based on shallow syntactic parsing. Finally, in [10] an ontology-based information extractor employs handcrafted rules in order to extract soccer-related entities from various Web sources and map them onto soccer-specific semantic structures. The recognition of named entities is based solely on named entity lists, thus not supporting recognition of entities that are absent from the lists.

In this context, the main contributions of RS are:

- An end-to-end solution for understanding "who has done what, how, why and with what results" from the text of research articles.
- A domain-independent procedure that automatically creates annotated corpora for training named entity recognizers, especially useful for entities of "non-common" type.
- A system that leverages semantic information, surface form as well as deep syntactic and structural text analysis in order to extract information using both machine learning modules and handcrafted rules.
- A workflow that combines information from metadata and linked data with knowledge extracted from text and republishes it as a knowledge base, adhering to linked data standards.

3 Conceptual Framework: The Scholarly Ontology

The conceptual model underlying RS is based on the Scholarly Ontology (SO) [2], a domain-independent framework for modeling scholarly activities and practices. The rationale behind SO is to support answering questions of the form "*who* does *what*, *when*, and *how*" in a given scholarly domain, so the ontology is built around the central notion of *activity* and combines three perspectives: the *agency* perspective, concerning actors and intentionality; the *procedure* perspective, concerning the intellectual framework and organization of work; and the *resource* perspective, concerning the material and immaterial objects consumed, used or produced in the course of activities. We here briefly review a subset of core SO concepts that constitute the RS schema guiding the extraction as well as the structuring of information (see Fig. 1).

Activity (e.g. an evaluation, a survey, an archeological excavation, a biological experiment, etc.) represents real events that have occurred in the form of intentional acts carried out by actors. Sequence of activities and composition from sub-activities are represented by the *follows* and *partOf* relations. The instances of the Activity class are real processes with specific results, as opposed to those of the *Method* class, which are specifications, or procedures for carrying out activities to address specific goals. *Actor* instances are entities capable of performing intentional acts they can be accounted or referenced for. Actors can participate in activities, actively or passively, in one or more roles. Subclasses of *Actor* are the classes *Person* and *Group* representing

individual persons and collective entities respectively. Further specializations of *Group* are the classes *Organization* and *Research Team*. *Content Item* comprises information resources, regardless of their physical carrier, in human readable form (e.g. images, tables, texts, mathematical expressions, etc.). *Proposition* comprises assertions in affirmative or negative form, resulting from activities and *supportedBy* evidence provided by content items. Finally, the class *Topic* comprises thematic keywords expressing the subject of methods, the topic of content items, the research interests of actors, etc.

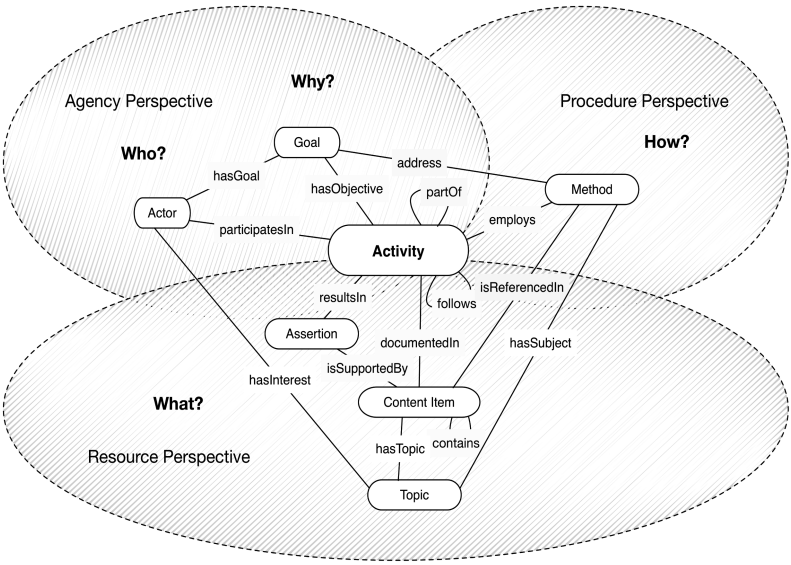

Fig. 1. The Scholarly Ontology core

4 Knowledge Base Creation

4.1 Process Overview

An overview of the knowledge creation process is given in Fig. 2. The input comprises published -open access- research articles retrieved from repositories or Web pages in the preferred html/xml format. The format is exploited in extracting the metadata of an article, such as authors' information, references and their mentions in text, legends of figures, tables etc. Entities, such as activities, methods, goals, propositions, etc., are extracted from the text of the article. These are associated in the relation extraction step, through various relations, e.g. *follows, hasPart, hasObjective, resultsIn, hasParticipant, hasTopic, has Affiliation,* etc. Encoded as RDF triples, these are published as linked data, using additional "meta properties", such as *owl:sameAs, owl:equivalentProperty, rdfs:Label, skos:altLabel,* where appropriate.

The entities targeted for extraction can be categorized into: (i) *named entities*, i.e. entities that have a proper name [1], such as instances of the SO classes: *ContentItem, Person, Organization, Method* and *Topic*; and (ii) *nameless,* or *non-named entities*, identified by their own description but not given a proper name, such as instances of SO classes *Activity*, *Goal* and *Proposition*.

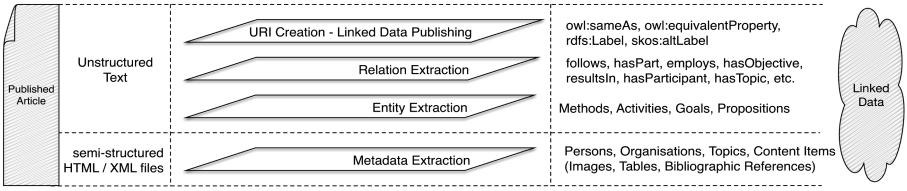

Fig. 2. Knowledge base creation. Left to right: input, processes, extracted entities and relations

Different modules handle entities of each category. Figure 3 shows the architecture of RS implementing the above process. The inputs of the system are:

(i) SPARQL endpoints of various Web sources for creating Named Entities (NE) lists;
(ii) user search keywords indicating the type of named entity to be recognized; and
(iii) URLs (e.g. journal Web pages that can be scraped) or publishers' APIs.

The main output of the system is the knowledge base published as linked data. The knowledge base creation process consists of two phases: (1) Preprocessing, for creating named entities lists and training the NER classifier and (2) Main Processing for the actual information extraction and publishing.

In *Preprocessing*, information is retrieved from sources such as DBpedia in order to build lists of named entities through the *NE List Creation* module. Specific queries using these entities are then submitted to the sources via the *API Querying* module. Retrieved articles are processed by the *Text Cleaning* module and the raw text at the output is added to a training corpus through the *Automatic Annotation* module that uses the entries of *NE list* to spot named entities in the text. The annotated texts are used to train a classifier to recognize the desired type of named entities. For details regarding the pre-processing phase, see Sect. 4.2.

Main Processing begins with harvesting research articles from Web sources, either using their APIs or by scraping publication Web sites. The articles are scanned for metadata which are mapped to SO instances according to a set of rules. In addition, specific html/xml tags inside the articles indicating images, tables and references are extracted and associated with appropriate entities according to SO, while the rest of the unstructured, "raw" text is cleaned and segmented into sentences by the *Text Cleaning & Segmentation* module (Sect. 4.3). The unstructured, "raw" text of the article is then input into the *Named Entity Recognition* module, where named entities of specific types are recognized. The segmented text is also inserted into a dependency parser using the *Syntactic Analysis* module. The output consists of annotated text -in the form of dependency trees based on the internal syntax of each sentence- which is further

processed by the *Non-Named Entities Extraction* module, so that text segments that contain other entities (such as Activities, Goals or Propositions) can be extracted (Sect. 4.4). The output of the above steps (named entities, non-named entities and metadata) is fed into the *Relation Extraction* module that uses four kinds of rules (Sect. 4.5): (i) syntactic patterns based on outputs of the dependency parser; (ii) surface form of words and POS tagging; (iii) semantic rules derived from SO; (iv) proximity constraints capturing structural idiosyncrasies of texts. Finally, based on the information extracted in the previous steps, URIs for the SO namespace are generated, and linked -when possible- to other strong URIs (such as the DBpedia entities stored in the named entities lists) in order to be published as linked data through a SPARQL endpoint.

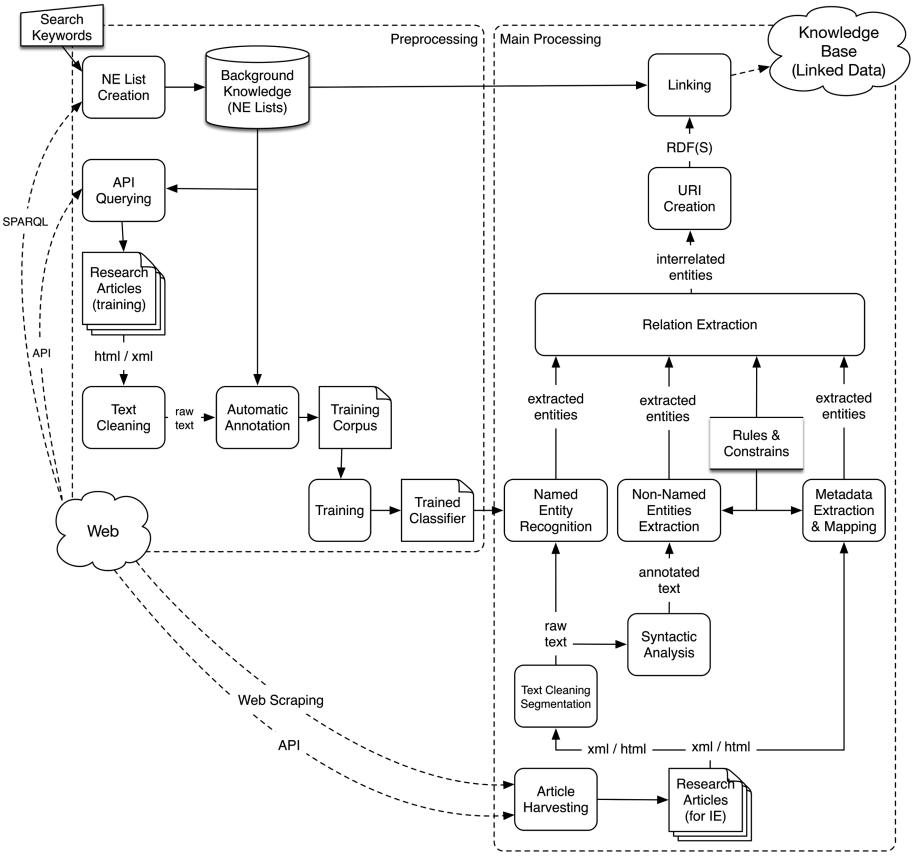

Fig. 3. Research Spotlight - system architecture

4.2 Preprocessing

In the Preprocessing phase (see Fig. 3), information is gathered from external sources in order to create a substantial amount of "background" knowledge. We currently use DBpedia, but other sources can be used as well. The use of background knowledge is twofold: (i) provide instances of the *Method* and *Topic* classes; and (ii) distant supervision for the creation of training data for named entity recognition (NER).

By querying DBpedia we create two NE lists, one for *Topics* and one for *Methods*. Research Methods is a named entity type not supported by existing NER models (they usually support common types of named entities such as: persons, organizations, locations, events and "miscellaneous"). We use the entries from the Methods List, for distant supervision so that training data from retrieved research articles can be created automatically. The benefits of this process are multiple: (1) being entirely automatic, it can help create a very large (noisy) training data set; (2) it can reduce the work of human annotators to correcting the already automatically annotated corpus. Here we employ the latter approach, in order to generate a dataset for the recognition of NEs of type *Method,* but in the same context, datasets of other types of NEs (such as *Tools*, *Persons*, *Locations* etc.) could be generated.

Along this line, the Methods List is used to generate training data for NER to recognize entities of type *Method*. Through the APIs of sources such as Springer and Elsevier we retrieve research articles that have Methods List entries as topic keywords, thus maximizing the likelihood of finding named entities of those specific types in the texts. Articles are segmented into sentences, which are scanned for entities from the Methods List and annotated by the automatic annotation module, using regular expressions for the name (and variants) of each entity in the NE list.

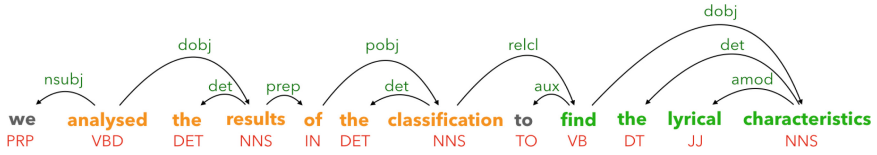

Fig. 4. Dependency tree

4.3 Metadata Extraction

By "metadata extraction" we mean the acquisition of all the structured information encoded in the article, either delivered in a separate format, such as Json, or embedded in the html/xml encoding of the document. Retrieved articles are parsed and entities of type *Person, Organization, Article* (subclass of Content Item) and *Topic* extracted from the xml tags. ORCID[1] is integrated through its API, so it can be used for duplicate

[1] https://orcid.org/.

detection and additional information. The html/xml encoding of the article is parsed using Beautiful Soup[2] to extract information about figures, tables and references. After extracting information from the html/xml encoding of the article, the *Text Cleaning* module is used to remove all the html/xml tags and the raw text is segmented into sentences and stored along with paragraph and section indicators.

4.4 Entity Extraction

Apart from "named entities" that can be identified using a NER (i.e. instances of *Method* class), we also need to extract "non-named" entities of highly variable length. Textual chunks indicating *Activities*, *Goals* and *Propositions* are detected using syntactic analysis in conjunction with rules that exploit lexico-syntactic patterns derived from the reasoning frame of SO [2]. A dependency tree containing POS tags and syntactic dependencies for each word in a sentence is obtained using Spacy[3]. Each sentence is further analyzed using the semantic definitions of SO classes, the surface form of words, their POS tag and their syntactic dependencies.

A sentence with verb in past or past/present perfect tense -in active or passive voice- containing no markers such as 'if' or 'that', quite likely describes an *Activity*, assuming the subject has the correct surface form ('we' or 'I' depending on the number of authors for active voice, no personal pronouns or determiners -to exclude vague subjects- for passive voice). Besides, 'that' following a verb can introduce a sub-sentence classified as *Proposition*, while a verb with dependent nodes with surface form 'to' or 'in order to' can introduce a sub-sentence classified as *Goal*. For example, consider the sentence

"We analyzed the results of the classification to find the lyrical characteristics". The syntactic analysis would yield the dependency tree of Fig. 4, from which "analysed the results of the classification" would be classified as an *Activity* and "find the lyrical characteristics" as a *Goal*. RS can detect multiple instances of *Activity*, *Goal*, or *Proposition* in the same sentence using the same rules with the addition of conjunction indicators. A detailed analysis of the employed algorithms can be found in [11].

4.5 Relation Extraction

The last step of information extraction involves detecting relations between previously extracted entities. SO semantics are employed for identifying the proper relation based on its domain and range. The organization of the text in sections and paragraphs induces proximity constraints enabling the inference of more complex, possibly inter-sentence, relations such as parthood and sequence of activities. The constraints used to identify relations are listed in Table 1. Relations marked with * are inherited from entity super-classes (Image, Table, Bib. Reference, Article from ContentItem; Person from Actor). The constraints for the *partOf* and *follows* relations (marked with **) can be relaxed in the presence of certain special indicators in the text (see Table 2).

[2] https://www.crummy.com/software/BeautifulSoup/.

[3] https://spacy.io/.

Parthood or sequence relations are assigned between the current and the last extracted activity either when a parthood or sequence indicator is detected, or by virtue of the relevant constraint. Figure 5 illustrates the extraction of sequence and parthood relations.

Table 1. Types of constraints per relation type

Relation Type	**Semantic Constrains (derived from SO)**	**Proximity Constraints - *P_C()* (from text structure)**
*isSupportedBy**	Domain: Proposition Range: Image|Table|Bibl. Reference	XML/HTML pointers inside the Proposition chunk
*partOf***	Domain: Activity Range: Activity	Co-occurrence with parent Activity in the same paragraph
*follows***	Domain: Activity Range: Activity	Co-occurrence with last Activity in the same paragraph
*contains**	Domain: Article Range: Image|Table| Bibl. Reference	Co-occurrence in the same Article
*participatesIn**	Domain: Person Range: Activity	Co-occurrence in the same Article
employs	Domain: Activity Range: Method	Co-occurrence in the same sentence
resultsIn	Domain: Activity Range: Proposition	Co-occurrence in the same paragraph
hasObjective	Domain: Activity Range: Goal	Co-occurrence in the same sentence
addresses	Domain: Method Range: Goal	Co-occurrence in the same sentence
*hasTopic**	Domain: Article Range: Topic	Co-occurrence in the same Article
hasSubject	Domain: Method Range: Topic	Co-occurrence in the same Article
*hasInterest**	Domain: Person Range: Topic	Co-occurrence in the same Article
*hasGoal**	Domain: Person Range: Goal	Co-occurrence in the same Article
*isDocumentedIn**	Domain: Activity Range: Article	Co-occurrence in the same Article
*isReferencedIn**	Domain: Method Range: Article	Co-occurrence in the same Article

Table 2. Sequence and Parthood indicators along with their surface forms

Sequence and parthood indicators	Surface forms
beginning_of_sequence	‘first’, ‘initially’, ‘starting’
middle_of_sequence	‘second’, ‘third’, ‘forth’, ‘fifth’, ‘sixth’, ‘then’, ‘afterwards’, ‘later’, ‘moreover’, ‘additionally’, ‘next’
end_of_sequence	‘finally’, ‘concluding’, ‘lastly’, ‘last’
parthood_indicators	‘specifically’, ‘first’, ‘concretely’, ‘individually’, ‘characteristically’, ‘explicitly’, ‘indicatively’, ‘analytically’

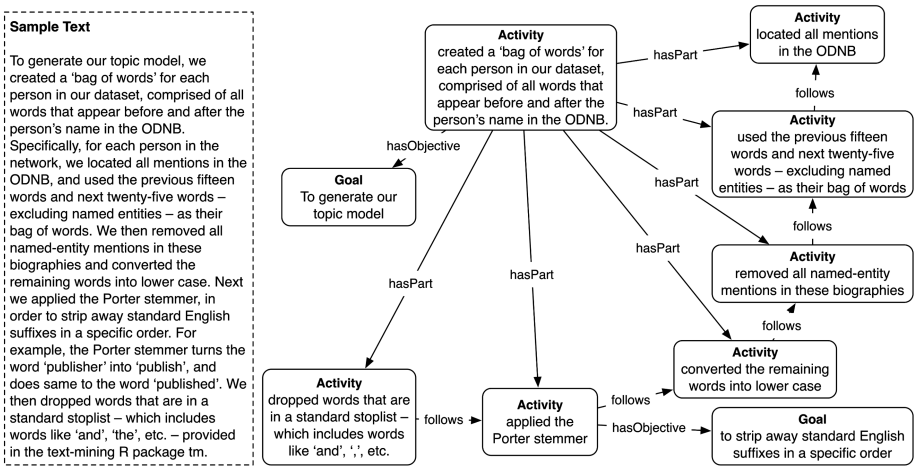

Fig. 5. Parthood and sequence relations

5 Evaluation

Metadata association exhibited very good performance since it relies solely on pre-constructed mappings between fixed schemas. Few isolated incidents (less than 1%) of improper association were due to errors in xml/html tags. The Information Extraction Modules of RS were evaluated by comparing their output with a "gold standard" produced by human annotators. According to established practice [12–14], we generated the confusion matrices by comparing the output of the system with that of the human annotators and, using micro and macro-averaging, we calculated the precision, recall and F1 scores. We conducted two evaluation experiments: one "strict" and one "lenient", in which the confusion matrices were created based on "per-entity" and "per-token" calculations respectively.

5.1 Evaluation Experiments

Regarding **non-named entities** and their relations, our "gold standard" consisted of corpora produced from 50 articles annotated by two researchers. We drew from 29 different journals from various research areas (Digital Humanities, Geology, Medicine, Bioinformatics, Biology, Computer Science, Sociology and Anthropology) to try our system with multiple writing styles. The non-named entities extracted belong to the classes *Activity*, *Goal* and *Proposition,* along with their relations *follows(act1, act2), hasPart(act1, act2), hasObjective(act, goal), resultsIn(act, prop)*. The manual annotation process took about 3.5–4 h per article on average. Inter-annotator agreement was 83% (kappa-statistic) based on corpora of 5 articles annotated by both annotators. Annotation produced about 1700 *Activities*, 300 *Goals*, 700 *Propositions*, 1000 *follows ()*, 100 *hasPart()*, 250 *hasObjective()*, 200 *resultsIn()*.

Regarding **named entities** (instances of *Method* class) and the *employs(act, meth)* relation, the dataset was created in the pre-processing phase. A list of 12,000 methods

was populated by the NE list creation module. After cleaning, the list was reduced to 7,000 method names and used to retrieve 210 articles, which were automatically annotated and then manually curated by two doctoral students. Inter-annotator agreement was 81% (kappa-statistic) based on corpora of 5 articles annotated by both annotators. Annotation gave about 3,800 *Methods* and 400 *employs()* relations. For the experiments we used the Stanford NE[4] recognizer, trained/evaluated in the above dataset.

Adopting the framework of [13], we designed two experiments yielding two different confusion matrices in order to conduct one token-based evaluation – where token is defined as any non-empty sequence of characters -, and one entity-based evaluation – where entity is defined as any non-empty sequence of tokens. To be correct, the prediction of the system must exactly equal the answer produced by humans in entity-based evaluation, whereas in token-based evaluation it should only overlap to at least a certain extent. The purpose of the second evaluation is to measure the performance of boundary detection of the system. In our experiment, after testing, the threshold for overlap was set to 86%, a difference of 1–5 tokens in most cases. In order to avoid promoting large entities in token-based evaluation, scores were calculated based on the relative distance to the perfect match, while penalties for remaining extra and missing tokens were assigned proportionally. The micro- and macro-averaged precision, recall and F1 scores based on confusion matrices from entity and token-based evaluation experiments, and individual scores for each type of entities and relations, are displayed in Tables 3, 4 and 5 respectively.

Table 3. Macro & micro averaging scores

	Macro-averaging			Micro-averaging		
	Precision	Recall	F1	Precision	Recall	F1
Entity-based	0.67	0.68	0.68	0.70	0.74	0.72
Token-based	0.87	0.77	0.81	0.84	0.83	0.83

Table 4. Entity extraction

Entity type	Entity-based			Token-based		
	P	R	F1	P	R	F1
Activity	0.70	0.75	0.72	0.79	0.85	0.81
Goal	0.74	0.78	0.76	0.86	0.74	0.80
Proposition	0.76	0.78	0.76	0.82	0.84	0.82
Method	0.80	0.69	0.74	0.75	0.72	0.73

Table 5. Relation extraction

Relation type	P	R	FI
follows	0.69	0.72	0.71
hasPart	0.57	0.54	0.55
hasObjective	0.79	0.78	0.78
resultsIn	0.54	0.58	0.56
employs	0.87	0.92	0.90

[4] https://nlp.stanford.edu/software/CRF-NER.html.

6 Discussion

The system performs adequately with F1 scores between 0.68 (lowest overall performance) and 0.83 (highest overall performance). Differences between the micro- and macro-averaged values are expected due to the way these measures are calculated. Because the F1 measure is mostly determined by the number of true positives, micro-averaging exposes effects related to large classes *(Activity, Method, Proposition, follows, employs*), while macro-averaging does so for small ones (*Goals, hasPart, hasObjective, resultsIn*). Token-based evaluation gives better scores since it is based on per-token comparison of system-extracted entities with human-extracted ones, thus constitutes a lenient but "closer to the real case" comparison.

The high increase in score values of the *Activity* class (increase of 11%) suggests average boundary detection. Analysis showed this to depend strongly on the complexity of the sentence. Regarding the *Proposition* class, a major source of errors was the rendering of a proposition as a statement in a separate phrase without introduction from an adverbial modifier (e.g. using "that…"). Regarding the NER module for the recognition of *Methods*, analysis showed that the majority of errors were caused by NEs with surface form that contains more than two words or punctuation marks. Regarding the extracted relations, the big difference in performance between the *employs()* relation and the rest can be attributed to the fact that the first involves mainly entities in the same sentence. When the domain and range of a relation were in different sentences or even paragraphs (e.g. *hasPart()*) relation extraction did poorly.

Regarding the entire RS workflow of KB creation, based on our measurements, information extracted from 50 articles –according to the semantics of SO, described in this paper- translates roughly to 100.000 triples, this of course being highly dependent on the writing style and the discipline. Indicative running times (intel i7, 16 GB RAM) for the entire process are approx. 120 secs/paper.

7 Conclusion

RS leverages an ontology of research practices and deep syntactic analysis to extract information from articles and populate a knowledge base published as linked data. RS acquires information from the Web in several ways (API integration, scrapping). Classifiers are automatically trained to recognize named entities of "non-common" type (e.g. research methods) not supported by current serialized models. Using these together with the knowledge captured in the Scholarly Ontology and deep syntactic text analysis, the system achieves extracting entities and relations representing research processes at a level of detail and complexity not addressed before. Future work includes improving recall addressing other types of entities (e.g. tools used in activities), and other types of rhetorical arguments stated in the text, thus improving overall coverage.

References

1. Jurafsky, D., Martin, J.H.: Speech and language processing - an introduction to natural language processing, computational linguistics, and speech recognition (2017)
2. Pertsas, V., Constantopoulos, P.: Scholarly ontology: modelling scholarly practices. Int. J. Digit. Libr. **18**, 173–190 (2017). https://doi.org/10.1007/s00799-016-0169-3
3. Gerber, D., Hellmann, S., Bühmann, L., Soru, T., Usbeck, R., Ngonga Ngomo, A.-C.: Real-time RDF extraction from unstructured data streams. In: Alani, H., et al. (eds.) ISWC 2013. LNCS, vol. 8218, pp. 135–150. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-41335-3_9
4. Lehmann, J., et al.: DBpedia - a large-scale, multilingual knowledge base extracted from Wikipedia. Semant. Web **6**, 167–195 (2015). https://doi.org/10.3233/SW-140134
5. Zimmermann, A., Gravier, C., Subercaze, J., Cruzille, Q.: Nell2RDF: read the web, and turn it into RDF. In: CEUR Workshop Proceedings, pp. 1–7 (2013)
6. Stern, R., Sagot, B.: Population of a knowledge base for news metadata from unstructured text and web data. In: AKBC-WEKEX 2012, pp. 35–40, Montreal, Canada (2012)
7. Alani, H., et al.: Automatic ontology-based knowledge extraction from web documents. IEEE Intell. Syst. **18**, 14–21 (2003)
8. Makki, J., Alquier, A.-M., Prince, V.: Ontology population via NLP techniques in risk management. Int. J. Humanit. Soc. Sci. **3**, 212–217 (2008)
9. Celjuska, D., Vargas-Vera, M.: Ontosophie: a semi-automatic system for ontology population from text. In: ICON 2004 (2004)
10. Buitelaar, P., Cimiano, P., Frank, A., Hartung, M., Racioppa, S.: Ontology-based information extraction and integration from heterogeneous data sources. Int. J. Hum.-Comput. Stud. **66**, 759–788 (2008). https://doi.org/10.1016/j.ijhcs.2008.07.007
11. Pertsas, V.: Modeling and extracting research processes. Athens University of Economics and Business, Athens (2018)
12. Manning, C.D., Raghavan, P., Schutze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
13. De Sitter, A., Calders, T., Daelemans, W.: A formal framework for evaluation of information extraction, University of Antwerp (2004)
14. Maynard, D., Peters, W., Li, Y.: Metrics for evaluation of ontology based information extraction. In: WWW 2006 Workshop on Evaluation of Ontologies for the Web (2006)