Entity extraction: From unstructured text to DBpedia RDF triples

Exner, Peter; Nugues, Pierre

Published in:
Proceedings of the Web of Linked Entities Workshop in conjuction with the 11th International Semantic Web Conference (ISWC 2012)

2012

Link to publication

Citation for published version (APA):
Exner, P., & Nugues, P. (2012). Entity extraction: From unstructured text to DBpedia RDF triples. In *Proceedings of the Web of Linked Entities Workshop in conjuction with the 11th International Semantic Web Conference (ISWC 2012)* (pp. 58-69). CEUR-WS. http://ceur-ws.org/Vol-906/paper7.pdf

Total number of authors:
2

Entity Extraction:
From Unstructured Text to DBpedia RDF Triples

Peter Exner and Pierre Nugues

Department of Computer science
Lund University
peter.exner@cs.lth.se
pierre.nugues@cs.lth.se

Abstract. In this paper, we describe an end-to-end system that automatically extracts RDF triples describing entity relations and properties from unstructured text. This system is based on a pipeline of text processing modules that includes a semantic parser and a coreference solver. By using coreference chains, we group entity actions and properties described in different sentences and convert them into entity triples. We applied our system to over 114,000 Wikipedia articles and we could extract more than 1,000,000 triples. Using an ontology-mapping system that we bootstrapped using existing DBpedia triples, we mapped 189,000 extracted triples onto the DBpedia namespace. These extracted entities are available online in the N-Triple format[1].

1 Introduction

By using the structured and semi-structured information from Wikipedia, DBpedia [1] has created very large amounts of linked data and is one the most significant achievements of the Semantic Web initiative. Datasets from DBpedia are used in a wide range of applications such as faceted search, model training for information extraction, etc.

DBpedia focuses on extracting structured information from Wikipedia articles, such as infobox templates and categorization information. However, the unstructured text of the articles is left unprocessed. Some recent projects have attempted to use this text content to extend the DBpedia triple base. Examples include iPopulator [2] that populates incomplete infoboxes with attribute values it identifies from the article text, while two recent systems, LODifier [3] and KnowledgeStore [4], extract semantic information from the text. LODifier creates RDF triples based on WordNet URIs while KnowledgeStore uses its own ontology. Nonetheless, these systems show limitations in the form of preexisting infobox templates or data structures that are not fully compliant with the DBpedia namespace.

In this paper, we introduce a framework to carry out an end-to-end extraction of DBpedia triples from unstructured text. Similarly to LODifier and KnowledgeStore, our framework is based on entities and identifies predicate–argument structures using a generic semantic processing pipeline. However, instead of recreating new semantic structures, we integrate the DBpedia property ontology and therefore make the reuse

[1] http://semantica.cs.lth.se/

and extension of the DBpedia dataset much easier. Starting from the DBpedia dataset, we link the triples we extract from the text to the existing DBpedia ontology, while going beyond the existing infobox templates. Applications already using DBpedia would then benefit from a richer triple store.

We applied our system to over 114,000 Wikipedia randomly selected articles and we could extract more than 1,000,000 triples. Using the ontology-mapping system that we bootstrapped using existing DBpedia triples, we mapped 189,000 extracted triples onto the DBpedia namespace. Interestingly, we could rediscover from the text 15,067 triples already existing in the DBpedia dataset. We evaluated our framework on a sample of 200 sentences and we report a F1 score of 66.3% on the mapped triples.

2 Related Work

The extraction of relational facts from plain text has long been of interest in information extraction research. The key issue in relation extraction is to balance the trade-off between high precision, recall, and scalability. With the emergence of the Semantic Web and numerous ontologies, data integration has become an additional challenge.

There has been a considerable amount of research on semi-supervised [5–7] methods using bootstrapping techniques together with initial seed relations to create extraction patterns. Unsupervised approaches [8, 9] have contributed further improvements by not requiring hand-labeled data. These approaches have successfully answered scalability and precision factors, when applied on web-scale corpora. The challenge of ontology and data integration has been addressed by [10].

Due to concerns on scaling, the use of syntactic or semantic relation extraction techniques in relation extraction has been relatively sparse. Few systems carry out a complete analysis of the source documents using coreference resolution or discourse analysis to extract all statements. Exceptions include LODifier [3] and KnowledgeStore [4], that have extracted semantic information and applied coreference resolution. However, the entities extracted by these systems have not been integrated to a single homogenous ontology.

In contrast to these approaches, we suggest an end-to-end system, that extracts all the entity relations from plain text and attempts to map the entities onto the DBpedia namespace. We balance precision and recall by employing a combination of NLP tools, including semantic parsing, coreference resolution, and named entity linking. Scalability issues are handled by parallelizing the tasks on a cluster of computers. Furthermore, we propose an ontology mapping method that bootstraps learning from existing triples from the DBpedia dataset.

3 System Architecture

The architecture of our system is a pipeline that takes the Wikipedia articles as input and produces entities in the form of DBpedia RDF triples. As main features, the system includes a generic semantic processing component based on a semantic role labeler (SRL) to discover relations in text, an automatic learning of ontology mappings to link the extracted triples to the DBpedia namespace, and an algorithm to rank named entity

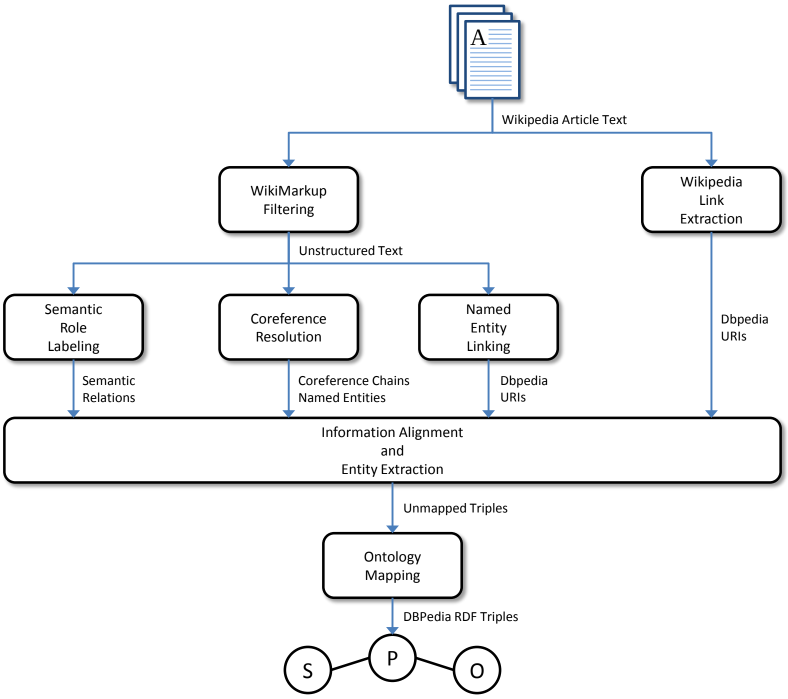

Fig. 1. Overview of the entity extraction pipeline.

links (NEL) found in coreference chains in order to discover representative mentions. In total, the end-to-end processing of Wikipedia article text consists of seven modules (Figure 1):

1. A *WikiMarkup filtering* module that removes the Wikimedia markup, providing the plain text of the articles to the subsequent modules;
2. A *Wikipedia link extractor* that extracts Wikipedia links from the articles;
3. A *semantic parsing* module, Athena [11], a framework for large-scale semantic parsing of text written in natural language;
4. A *coreference resolution* module that detects and links coreferring mentions in text;
5. A *mention-to-entity linking* module that links mentions to a corresponding DBpedia URI;
6. An *information aligning and entity extracting* module that aligns the output from top-level modules and extracted entities in the form of triples.
7. An *ontology mapping* module that carries out the final mapping of predicates from the Propbank nomenclature onto the DBpedia namespace.

4 Processing of Wikipedia Article Text

WikiMarkup Filtering. Prior to any analysis, the text must be filtered. This is an essential step that seeks to remove annotations and markups without affecting the running text. Without this step, subsequent modules would fail in their analysis and lead to erroneous extractions.

Wikipedia articles are composed of text written in natural language annotated with a special markup called wikitext or wiki markup. It is a simple markup language that allows among other things the annotation of categories, templates, and hyperlinking to other Wikipedia articles. Wikipedia also allows the use of common HTML tags.

By filtering Wikipedia text, we aim at removing all annotations, sections that contain only links and references, and keeping only the running text. This process is difficult since the HTML syntax is often invalid. The most common errors are tags that are left unclosed or are incorrectly nested.

Wikipedia Link Extraction. During the Wikipedia link extraction, we extract and preserve the original links along with their corresponding mentions in the article. In addition to extracting the links annotated by the article authors, we make the assumption that the first noun phrase in the first sentence corresponds to the article link. The rationale behind it is that the longest coreference chain in the article often starts with this first mention.

The direct correspondence between Wikipedia articles and DBpedia resources allows us to map Wikipedia links onto their corresponding DBpedia URI by simply adding the DBpedia namespace.

Semantic Parsing. Frame semantics [12] is a linguistic theory that assumes that the meaning of a sentence is represented by a set of predicates and arguments. The Proposition Bank [13] is a project that applied this theory to annotate corpora with predicate-argument structures. For each predicate, Propbank identifies up to six possible core arguments denoted *A0*, *A1*, ..., and *A5* that go beyond the traditional annotation of subjects and objects. Propbank also includes modifiers of predicates, such as temporal and location adjuncts. These roles are instrumental in performing the extraction of entities as they allow the identification of properties containing temporal and locational data with high precision.

We use the Athena framework created for parallel semantic parsing of unstructured text. At its core, the system uses a high-performance multilingual semantic role labeler that obtained top scores in the CONLL-2009 shared task [14, 15].

Coreference Resolution. A coreference resolver creates chains of coreferring mentions by discovering and linking anaphoric phrases to their antecedents. We used a coreference solver, included in the Stanford CoreNLP package [16, 17], to link mentions of entities in the different parts of text. This allows us to group entity actions and properties described in different sentences. CoreNLP uses a pipeline of tokenizers, part-of-speech tagger, named entity recognizer, syntactic parser, and coreference solver to annotate unstructured text. In addition to coreference annotation, we store the named entity classification created by the pipeline. The named entity classes are used to filter named entity links having a conflicting ontology classification.

Named Entity Linking. An important step in entity extraction is the grounding of named entities to unique identifiers. In most articles, only the first mention of a named entity is annotated with a corresponding Wikipedia link; subsequent mentions are often left unannotated. Wikifier [18] is a named entity linking system that annotates unstructured text with Wikipedia links. By applying Wikifier, we can link unannotated named entities in the Wikipedia articles to a corresponding DBpedia URI.

Ontology Mapping. During semantic parsing, the sentences are annotated with predicate–argument structures called rolesets. As dictionary, the parser uses PropBank that defines more than 7,000 rolesets. Propbank associates each predicate with a set of senses, for instance *bear* has six senses denoted *bear.01*, *bear.02*, ..., *bear.06*. Finally, each predicate-sense has a set of core arguments that differ with each roleset. For example, *bear.02* has two core arguments: *A0*, the mother, and *A1*, the child. Considering only the core roles, this amounts to more than 20,000 roles.

The objective of ontology mapping is to map the predicate and argument roles from PropBank onto DBpedia properties. We perform this final step to create the DBpedia RDF triples. Figure 2 shows an example of end-to-end processing to DBpedia RDF triples of the sentences: *Luc Besson (born 18 March 1959) is a French film director, writer and producer. Besson was born in Paris to parents who were both Club Med scuba diving instructors.*

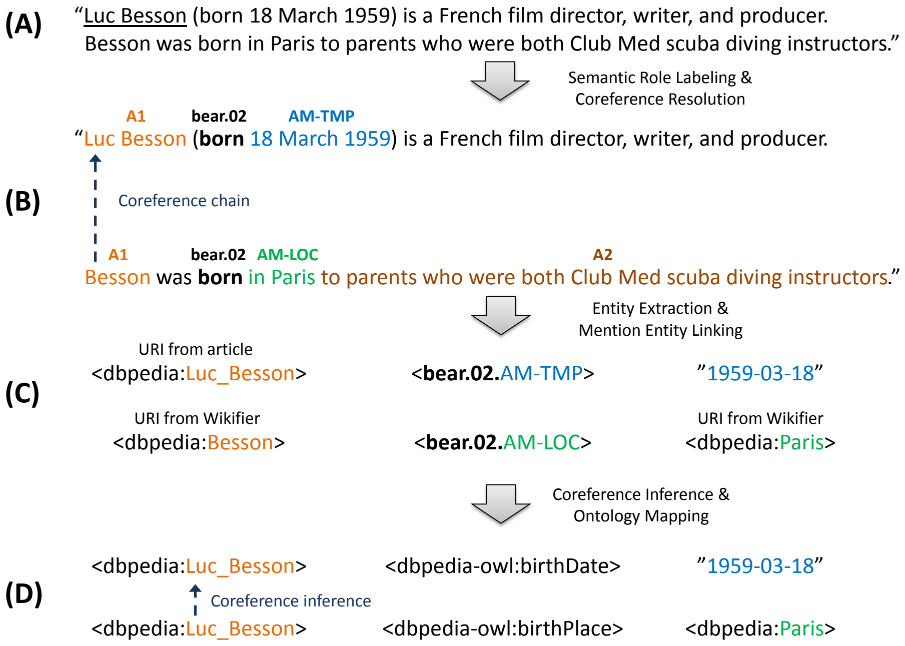

Fig. 2. An ideal conversion from text to the DBpedia RDF triples: (A) The input sentences. (B) The sentences after semantic parsing and coreference resolution. (C) Entity extraction. (D) Ontology mapping.

5 Entity Extraction

The arguments created during semantic parsing are searched in order to find named entity links corresponding to RDF subjects and objects. This process uses the mentions discovered by the coreference solver, Wikipedia links predicted by Wikifier, and

Wikipedia links extracted from the article. In order to keep the task tractable, we have limited the entities to those found in DBpedia and we do not introduce new named entities to the DBpedia ontology.

RDF Subjects. PropBank uses the *A0* label as the argument describing agents, causers, or experiencers, while arguments labeled as *A1* describe entities undergoing a state of change or being affected by an action. In both cases, arguments labeled *A0* or *A1* can be considered containing RDF subjects and are consequently searched for named entity links. Arguments labeled *A0* are searched first, arguments labeled *A1* are only searched if a named entity link wasn't discovered in the preceding arguments.

RDF Objects. Following the subject extraction, the remaining arguments are examined to discover potential objects. The core arguments and two auxiliary arguments, temporal *AM-TMP* and location *AM-LOC*, are searched. The extracted data types can be categorized as following: Named entity links expressed as DBpedia URIs, dates and years, integers, and strings. We search date expressions in the temporal arguments *AM-TMP* using regular expressions. By using seven common date patterns, we are able to extract a large amount of date and year expressions. We associate the location arguments *AM-LOC* to named entity links representing places. These links are extracted only if they are classified as *dbpedia-owl:Place* by the DBpedia ontology.

Named Entity Link Ranking and Selection. During the search of RDF subject and objects, we search and select candidate named entity links in the following order:

1. Wikipedia links, converted to DBpedia URIs. We consider named entity links extracted from the article as being most trustworthy.
2. Wikifier-predicted Wikipedia links, converted to DBpedia URIs, and having a DBpedia ontology class matching the predicted named entity class. A predicted named entity link is chosen only in the case when an extracted Wikipedia link isn't given. Furthermore, predicted links are pruned if their DBpedia ontology class doesn't match the named entity class predicted by the Stanford coreference solver.
3. Coreference mentions; the most representative named entity link (according to the score described in section Using Coreference Chains) in the coreference chain is selected. We consider named entities inferred through coreference chains as the least trustworthy and select them only if an extracted or predicted named entity link is not given. A mention placed in the wrong coreference chain will be considered as an incorrect named entity link; a situation which Wikifier can rectify with higher precision.

Using Coreference Chains. Coreference chains are used to propagate named entity links to arguments having neither an extracted nor a predicted named entity link. This situation arises most commonly for arguments consisting of a single pronoun. Before propagation takes place, we determine the most representative named entity link in the coreference chain using a ranking and scoring system:

- Extracted named entity links are always selected over predicted links.

- A score of +2 is given to a named entity link if it has a DBpedia ontology class matching the predicted named entity class.
- The score is increased by the number of tokens of the named entity minus 1.
- If a tie is given between equally scoring named entity links, the link closest to the top of the chain is selected.

We derived the set of scoring rules by performing an empirical examination of coreference chains. We observed that coreference chains representing people, often started with a mention containing the full name, followed by single-token mentions having only the first or last name. The named entity links of single-token mentions, as predicted by Wikifier, often incorrectly pointed to either a place or a family. By rewarding named entity links having multiple tokens and matching ontology classes, we filtered these incorrect links. Table 1 shows an example, where the mention *Robert Alton*, a person name, is given the highest score due to matching entity classes and token length. Although the mention *Alton* refers to the same entity and belongs to the coreference chain, an incorrect named entity link to a city (Alton, Illinois) has been predicted. Given our previous rules, the predicted named entity link is discarded due to a mismatch with the predicted named entity class. The correct named entity link is thus resolved by propagating the link through the coreference chain.

Unmapped Triple Generation. Given a set of extracted RDF subjects and objects, we create binary relations from n-ary predicate–argument relations by a combinatorial generation. We discover negative relations by searching the argument roles for *AM-NEG*; these are then discarded.

Mention	NE class	NE link	DBpedia ontology class	Score
Alton	Person	dbpedia:Alton,_Illinois	dbpedia-owl:Place	0
Robert Alton	Person	dbpedia:Robert_Alton	dbpedia-owl:Person	3
He				0

Table 1. Example showing how scoring resolves the most representative predicted named entity link in a coreference chain. NE stands for named entity. The NE class is obtained from the NE recognition module in the Stanford CoreNLP package. The NE link is predicted by Wikifier.

6 Ontology Mapping

The final step in extracting DBpedia RDF triples from text concerns the mapping of predicates onto the DBpedia namespace. The unmapped extracted triples have predicates described using the Propbank dictionary. The predicates together with their sense and unique set of argument roles comprise more than 20,000 different roles. With ontology mapping, our goal is to map the resulting triples onto a more general roleset described by 1,650 DBpedia properties.

Since the manual mapping of such a large amount of roles is a requiring task, we wished to perform the it automatically. Our approach was to bootstrap the learning of ontology mappings by matching the subject and object of the extracted triples onto existing triples in the DBpedia dataset. Generic mappings are created and reused by generalizing DBpedia URIs, found in subjects and objects, to 43 top-level DBpedia ontology classes. The learning process consists of the following steps (Figure 3):

1. The subject and object of the extracted triples are matched exactly to existing triples in the DBpedia dataset.
2. From the matching set of triples, links between Propbank roles and DBpedia properties are created. The mappings with the highest count are stored.
3. The subject and object of the extracted triples that contain DBpedia URIs are generalized to 43 top-level DBpedia ontology classes. Objects containing strings, dates and numbers are generalized to the categories: String, Date, and Number respectively.

As an example, consider the following sentence:

Besson married Milla Jovovich on 14 December 1997.

We extract the triple:

<dbpedia:Luc_Besson> <marry.01.A1> <dbpedia:Milla_Jovovich>

and match to the existing triple in the DBpedia dataset:

<dbpedia:Luc_Besson> <dbpedia-owl:spouse> <dbpedia:Milla_Jovovich>

and finally generalize the subjects and objects to create the mapping:

<dbpedia-owl:Person> <marry.01.A1> <dbpedia-owl:Person>
maps to:
<dbpedia-owl:spouse>

Table 2 shows five of the most frequent mappings learned during the bootstrapping process. Most systems express mappings as alignments between single entities belonging to different ontologies. In addition, we also retain a generalized form of the related subject and object entities in such alignments. Together with the predicate sense and object argument role, we use them to express a more detailed mapping. By including the generalized object and its argument role in our mappings, we can differentiate between different domains for a certain predicate, such as between a birth place and a birth date.

When a new sentence, describing the same relation, is encountered for which there is no existing DBpedia triple we can perform our ontology mapping. Mappings learned from one entity can thus be used on other entities as more descriptive entities create mappings that are reused on entities with fewer properties. As an example, consider the following sentence:

On April 30, 2008, Carey married Cannon at her private estate...

with the extracted triple:

<dbpedia:Mariah_Carey> <marry.01.A1> <dbpedia:Nick_Cannon>

generalized to:

<dbpedia-owl:Person> <marry.01.A1> <dbpedia-owl:Person>

we can apply our previously learned mapping and infer that:

<dbpedia:Mariah_Carey> <dbpedia-owl:spouse> <dbpedia:Nick_Cannon>

that is not present in DBpedia.

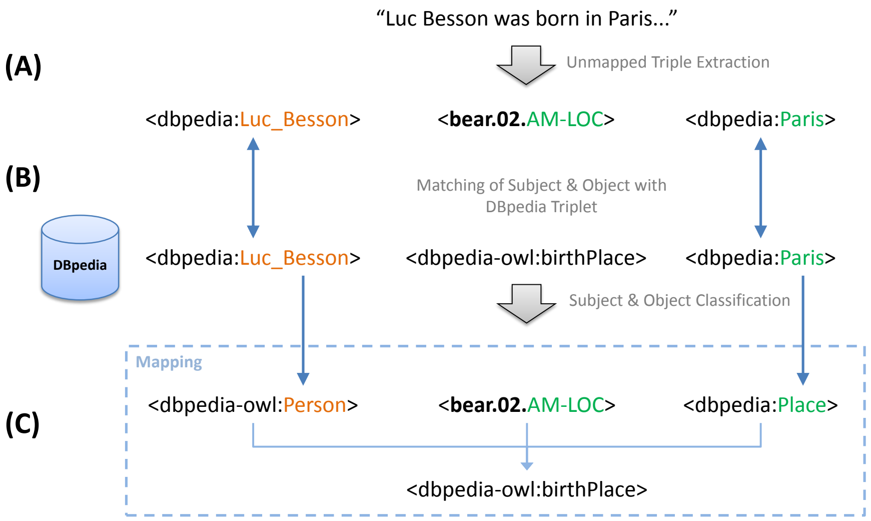

Fig. 3. (A) An unmapped triple is extracted from the sentence. (B) The extracted triple is matched to an existing DBpedia triple. (C) A mapping is created by linking the predicates between the two different namespaces and generalizing the subject and object.

Subject	Predicate	Object	Mapping
dbpedia-owl:Person	bear.02.AM-LOC	dbpedia-owl:Place	dbpedia-owl:birthPlace
dbpedia-owl:Person	bear.02.AM-TMP	xsd:date	dbpedia-owl:birthDate
dbpedia-owl:Person	marry.01.A1	dbpedia-owl:Person	dbpedia-owl:spouse
dbpedia-owl:Organisation	locate.01.AM-LOC	dbpedia-owl:Place	dbpedia-owl:city
dbpedia-owl:Organisation	establish.01.AM-TMP	xsd:integer	dbpedia-owl:foundingYear

Table 2. Five of the most frequent ontology mappings learned through bootstrapping.

7 Experimental Results

The aim of the evaluation is to answer the question of how much information in the form of entity relation triples can be extracted from sentences. We also wish to evaluate the quality of the extracted triples. Since there is no gold standard annotation of entities found in the main text of Wikipedia articles, we performed the evaluation by manually analyzing 200 randomly sampled sentences from different articles. Sampled sentences are examined for relevant subject-predicate-object triples and compared to the corresponding retrieved triples. We computed the precision, recall, and F1 scores, and in the occurrence of an extraction error, we made a note of the originating source.

We evaluated the attributes of each triple in a strict sense: Each extracted attribute must exactly match the corresponding attribute in the sentence. For instance, in evaluating the birthplace of a person, if a sentence states a city as the location, we only consider an extracted DBpedia link to the city as correct. In contrast, if the extracted link refers to a more generalized toponym, such as region or country, we mark the extracted object as erroneous.

In total, we processed 114,895 randomly selected articles amounting to 2,156,574 sentences. The articles were processed in approximately 5 days on a cluster of 10 machines. Table 3, left, shows the number of processed articles categorized by DBpedia ontology classes. From the processed articles, we extracted a total of 1,023,316 triples, of which 189,610 triples were mapped to the DBpedia ontology. The unmapped triples differ in having the predicate localized to the Propbank namespace. In Table 3, right, we can see that from the 189,610 extracted triples, 15,067 triples already exist in the DBpedia dataset. This means that our framework introduced 174,543 new triples to the DBpedia namespace. Almost 3% of the extracted triples are duplicates, the majority of these are triples repeated only once. Since a statement with the same meaning can occur in more than one article, we consider these occurrences natural. In comparing the number of extracted triples to the number of processed sentences, we find that roughly every second sentence yields one extracted triple. In comparison to the number of processed articles, we extracted nearly 9 triples per article.

The extracted mapped triples reached a F1 score of 66.3%, a precision of 74.3%, and a recall of 59.9%. The largest source of errors came from predicates, 46.4%, followed by subjects, 27.2%, and objects, 26.4%.

Based on post-mortem analysis of the evaluated triples, we find that reasons for the extraction errors can be attributed to the following causes:

- An incorrect mapping from the Propbank predicate-argument roles to the DBpedia ontology properties.
- A new entity is detected, that has previously not been introduced to the DBpedia datasets and therefore lacks a corresponding DBpedia URI.
- The wrong URI is predicted for an entity and cannot be resolved or corrected by the scoring algorithm.
- A mention is placed in the incorrect coreference chain by the coreference solver.

The majority of errors stem from erroneous ontology mappings. We believe that ontology mapping can be improved by using a more fine grained approach to the subject-object generalization. Currently, we categorize subjects and objects to 43 top-level DBpedia ontology classes out of 320 possible classes. In addition, increasing the amount of bootstrapping data used during learning can be done by utilizing links to other datasets, such as LinkedMDB[2]. We also believe that the current rule-based mapping system could be replaced by a more capable system based on machine learning.

The linking of mentions to DBpedia URIs was also found to be a major source for errors. We believe that retraining Wikifier using a more current Wikipedia dump may improve named entity linking.

8 Conclusions and Future Work

In this paper, we described an end-to-end framework for extracting DBpedia RDF triples from unstructured text. Using this framework, we processed more than 114,000

[2] http://www.linkedmdb.org/

English Wikipedia	
Persons	32,423
Places	63,503
Organisations	18,969
Total Articles	114,895
Sentences	2,156,574

Type	Count
DBpedia Mapped Triples	189,610
(of which 15,067 already exist in DBpedia)	
Unmapped Triples	833,706
Total	1,023,316

Table 3. Left table: An overview of entity extraction statistics. **Right table**: The number of extracted triples grouped by triple type.

articles from the English edition of Wikipedia. We extracted over 1,000,000 triples that we stored as N-Triples. We explored a method for creating ontology mappings of predicates between the Proposition Bank and the DBpedia namespace. By bootstrapping the learning of mappings through the alignment of extracted triples with triples in the DBpedia dataset, we mapped 189,000 triples to the DBpedia namespace. We evaluated the mapped triples on a randomly selected sample of 200 sentences and we report a F1 score of 66.3%.

The largest source of errors stemmed from incorrect mappings. For instance, a mapping describing a person receiving a thing corresponding to an award property, requires a more detailed analysis since the thing received may represent items other than awards. We believe this can be significantly improved by applying a more fine-grained approach during the generation of mappings. We also believe that retraining the named entity linker and improving filtering of erroneous coreference mentions may increase the results.

The resulting database may be used to populate Wikipedia articles with lacking or sparse infoboxes and to aid article authors. We also believe that a future application of the framework might be to fully create Wikipedias from unstructured text.

By using the framework, we wish to bridge the gap between unstructured and annotated text, in essence, creating training material for machine learning. One possible application that we wish to investigate is the creation of parallel corpora, by means of entity linking.

An archive of extracted triples is available for download in N-Triple format[3].

Acknowledgments. This research was supported by Vetenskapsrådet, the Swedish research council, under grant 621-2010-4800 and has received funding from the European Union's seventh framework program (FP7/2007-2013) under grant agreement 230902.

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: DBpedia: A nucleus for a web of open data. In: The Semantic Web. Volume 4825 of LNCS. Springer Berlin (2007) 722–735

[3] `http://semantica.cs.lth.se/`

2. Lange, D., Böhm, C., Naumann, F.: Extracting structured information from Wikipedia articles to populate infoboxes. In: Proceedings of the 19th CIKM, Toronto (2010) 1661–1664
3. Augenstein, I., Padó, S., Rudolph, S.: LODifier: Generating linked data from unstructured text. In: The Semantic Web: Research and Applications. Volume 7295 of LNCS. Springer Berlin (2012) 210–224
4. Cattoni, R., Corcoglioniti, F., Girardi, C., Magnini, B., Serafini, L., Zanoli, R.: The KnowledgeStore: an entity-based storage system. In: Proceedings of LREC 2012, Istanbul (2012)
5. Agichtein, E., Gravano, L.: Snowball: extracting relations from large plain-text collections. In: Proceedings of DL '00, New York, ACM (2000) 85–94
6. Etzioni, O., Cafarella, M., Downey, D., Kok, S., Popescu, A.M., Shaked, T., Soderland, S., Weld, D.S., Yates, A.: Web-scale information extraction in knowitall. In: Proceedings of WWW '04, New York, ACM (2004) 100–110
7. Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka, E.R., Mitchell, T.M.: Toward an architecture for never-ending language learning. In: Proceedings of AAAI-10. (2010) 1306–1313
8. Banko, M., Etzioni, O.: Strategies for lifelong knowledge extraction from the web. In: Proceedings of K-CAP '07, New York, ACM (2007) 95–102
9. Fader, A., Soderland, S., Etzioni, O.: Identifying relations for open information extraction. In: Proc. of EMNLP '11. (2011) 1535–1545
10. Suchanek, F.M., Sozio, M., Weikum, G.: Sofie: A self-organizing framework for information extraction. In: Proceedings of WWW '2009, New York (2009) 631–640
11. Exner, P., Nugues, P.: Constructing large proposition databases. In: Proc. of LREC'12, Istanbul (2012)
12. Fillmore, C.J.: Frame semantics and the nature of language. Annals of the New York Academy of Sciences: Conference on the Origin and Development of Language and Speech **280** (1976) 20–32
13. Palmer, M., Gildea, D., Kingsbury, P.: The Proposition Bank: an annotated corpus of semantic roles. Computational Linguistics **31** (2005) 71–105
14. Björkelund, A., Hafdell, L., Nugues, P.: Multilingual semantic role labeling. In: Proceedings of CoNLL-2009, Boulder (2009) 43–48
15. Bohnet, B.: Top accuracy and fast dependency parsing is not a contradiction. In: Proceedings of COLING-2010, Beijing (2010) 89–97
16. Raghunathan, K., Lee, H., Rangarajan, S., Chambers, N., Surdeanu, M., Jurafsky, D., Manning, C.: A multi-pass sieve for coreference resolution. In: Proc. of EMNLP-2010, Boston (2010) 492–501
17. Lee, H., Peirsman, Y., Chang, A., Chambers, N., Surdeanu, M., Jurafsky, D.: Stanford's multi-pass sieve coreference resolution system at the CoNLL-2011 shared task. In: Proceedings of the CoNLL-2011 Shared Task, Boulder (2011) 28–34
18. Ratinov, L., Roth, D., Downey, D., Anderson, M.: Local and global algorithms for disambiguation to Wikipedia. In: Proceedings of the 49th Annual Meeting of the ACL, Portland (2011) 1375–1384