Mapping text to ontology with DBpedia Lemon and BOA

Denis Lukovnikov, Sebastian Hellmann, Daniel Gerber, Christina Unger

1 Introduction

With the huge amount of knowledge available as linked data, there is a growing need to make this data accessible for humans in an easy and intuitive way, preferably by means of natural language. One of the main challenges in the development of NLP systems over linked data is to determine the exact meaning of a natural language expression. More specifically, there is the problem of entity linking – finding the named entity a mention in a text refers to, as well as the more challenging problem of ontology linking – finding what vocabulary elements from an OWL ontology is the target of referral. This work focuses on the second problem. We attempt to automatically extend lexical mappings for an OWL ontology from a high-quality seed lexicon. This work is explored in the context of keyword-based question answering in English and the QALD challenge[1]. In the first section, we describe the background, i.e. (1) the existing *lemon* lexicon for DBpedia [5] and (2) the BOA project [2]. Then, our approach – the use of BOA extraction patterns for ontology lexicon construction – is discussed in more detail. Finally, we present some preliminary evaluation results, give a conclusion and propose future steps to improve the approach described here.

2 Background and motivation

2.1 Lexicon and ontology linking

lemon[2] (LExicon Model for ONtologies) is a vocabulary for extending an ontology with lexical information describing (among other things) how the elements of the ontology are verbalized in natural language. This information is represented as RDF data, thus making the lexical information a part of the respective ontology and the linked data cloud.
The English *lemon* lexicon for DBpedia 3.8[3] was constructed manually, the first version covering about 98% of the ontology classes and 20% of the properties. However, the DBpedia Lemon lexicon is already useful for NLP application evaluation since it covers roughly 95% of URIs used in the QALD-3 dataset for question answering.

An interesting use case of the DBpedia lexicon is its possible use in keyword-based question answering systems. Whereas entity linking is a straightforward problem in NLP, ontology linking poses more challenges. One possible approach for ontology linking is using text similarity with the label of an ontology element to estimate the probability that a word is referring to this ontology element. But such approaches cannot account for synonyms very well. For example, the word "spouse" can easily be matched to the property `dbpedia-owl:spouse`, since the label of the property is "spouse". But "wife", "husband" and "married to" also refer to this property and similarity-based approaches often fail to find those lexicalizations.

A better approach for ontology linking is using a lexicalization dictionary, such as the DBpedia lemon lexicon. It is easy to generate mappings from surface forms to ontology elements from such a lexicon. However, there are several issues for widespread usage of the first version of the DBpedia lexicon for ontology linking: (1) it only covers 20% of the properties, (2) it does not cover properties that are not in the DBpedia ontology but can be found in DBpedia (especially OWL properties in the `http://dbpedia.org/property/`

[1] `http://www.sc.cit-ec.uni-bielefeld.de/qald`
[2] `http://lemon-model.net/`
[3] `http://lemon-model.net/lexica/dbpedia_en/`

namespace) and (3) there are no statistics about ontology verbalization, which can be helpful to improve ontology linking.

2.2 BOA patterns

The BOA project (BOotstrapping linked datA) follows an approach similar to NELL [1] (Never-Ending Language Learner) to learn extraction patterns and use them to learn new knowledge and use the new knowledge to learn new extraction patterns. In contrast to NELL, the BOA project uses the DBpedia knowledge as an initial seed to bootstrap the loop where NELL started out with a much lower number of initial seed knowledge. An example of a BOA extraction pattern (referred to as BOA pattern from now on) is:

"?R is the wife of ?D → <D> <`http://dbpedia.org/ontology/spouse`> <R> . "

where <D> and <R> are valid URIs from DBpedia, thus producing a valid RDF statement.

This pattern would allow to extract new instances of the `dbo:spouse`[4] relation from natural text. And the newly extracted instances of `dbo:spouse` can be represented in a different way in other pieces of text, allowing to find new extraction patterns, for example "?D's wife, ?R".

Iterative approaches like NELL and BOA are mainly aimed at extracting new knowledge. However, once acquired, BOA patterns have been proven useful in several applications such as template-based question answering (AutoSPARQL-TBSL, [4]) and fact validation (Defacto, [3]). Employing BOA patterns directly proved unsuitable for answering keyword-like search questions since such questions generally will not fit a BOA pattern. Still, BOA patterns can be used for ontology linking in keyword-based question answering, by searching the patterns through an index using a keyterm.

3 Using *lemon* seeds in BOA

BOA patterns can be employed to automatically expand the coverage of the DBpedia-lemon lexicon to verbalize more `dbpedia-owl` properties. Additionally, the patterns can be used to generate a lexicon for ontology elements found on DBpedia (and BOA patterns) that are not in the `dbpedia-owl` ontology. The statistical information associated with BOA patterns can be used to add statistical information to lexicon elements.

The input of our method is a set of BOA patterns as well as the initial and manually constructed DBpedia lemon lexicon. We then use the lexicon entries to find mapping extraction patterns which will then be used to extract new lexical information.

3.1 Finding mapping extraction patterns

Consider the *lemon* lexicon for DBpedia as a set of mappings from text to DBpedia ontology properties that can be represented as follows:

"wife" → `dbo:spouse`

We call the set of lexical entries L.
Consider the set P of BOA patterns, represented as follows:

?R is the wife of ?D → <D> <`dbo:spouse`> <R>

We construct two indices, $I[L]$ for L and $I[P]$ for P. Having these indices, we iterate over all mappings $l \in L$ and for each l, we search $I[P]$ to find all patterns matching the mapping l, giving us $I[P](l)$.
The matching function currently used is simply checking whether the surface form of the mapping ("wife") occurs in the BOA pattern string ("?R is the wife of ?D") and whether the property provided by the mapping

[4] `dbo:` is the prefix for `http://dbpedia.org/ontology/`

is the same as the property of the BOA pattern.
For each BOA pattern in $I[P](l)$, we extract the meta-pattern by substituting matched surface form from l in the pattern by a property variable (??P??). Then, if not already present, each meta-pattern is added to the set of meta-patterns MP. If the meta-pattern is already in MP, we update the support of the meta-pattern. An example meta-pattern that could be extracted from the above example lexicalization of the `dbo:spouse` property is

?R is the ??P?? of ?D

where ?D and ?R are entity placeholders[5] and ??P?? is a property variable.

3.2 Extracting new lexical mappings

In this phase, we want to extract new lexical mappings and score the existing ones. Here, we take MP and P and search for new elements for L.

To this end, for every BOA pattern $p \in P$ we find the matching meta-patterns using simple pattern matching with regular expressions. From the possibly applicable meta-patterns, the best meta-pattern is selected and used to extract the lexical mapping. If the mapping is already present in L, we update the scores, otherwise the mapping is added to L.

Selection of the best meta-pattern from all meta-patterns applicable to some BOA pattern is done by choosing the most surface form-specific meta-pattern. We define the most specific pattern to be the pattern producing the shortest surface form string in the lexical mapping.

As an example illustrating the generation of new mappings in this phase, consider the BOA pattern

?R is the consort of ?D → <D> <`dbo:spouse`> <R>

and the meta-pattern extracted from the wife example in the previous subsection:

?R is the ??P?? of ?D

However, another (more general) meta-pattern could have been extracted in the first phase, for example:

?R ??P?? ?D

Both meta-patterns could match the BOA pattern, but the first one is more specific because it produces a shorter surface form ("consort") than the second meta-pattern would produce ("is the consort of"). Therefore, the first meta-pattern is applied to the BOA pattern to produce the mapping

"consort" → `dbo:spouse`

Notice that the second (more general) meta-pattern would produce the mapping

"is the consort of" → `dbo:spouse`

which is worse than the mapping produced by the more specific meta-pattern, if not wrong for the purposes of lexicon construction.

4 Evaluation

The methodology described above was tested using a sample dataset containing BOA patterns of 10 properties (spouse, award, team, author, subsidiary, starring, deathPlace, birthPlace, leaderName and foundationPlace). Only 7 of the 10 properties in the sample BOA dataset had verbalizations in the DBpedia lexicon. In total, the lexicon contains 17 relevant verbalizations. Below, the preliminary evaluation results of the prototype implementation are described.

[5] ?D stands for the *rdfs:domain* of a given property and ?R for *rdfs:range* respectively.

Applied to the given data, our method found 119 meta-patterns and 5,274 lexical mappings in one iteration. This initial set of found lexical mappings contains a lot of noisy mappings. A fraction of the noisy mappings could still be useful in NLP applications (for example "goalkeeper"→ `dbo:team`), but it would be wrong to include them in the lexicon. In the future, we hope to filter out the noisy mappings.

We perform a preliminary evaluation of the method by looking at how many of the verbalizations present in the DBpedia lexicon were found by our method. We used the sample BOA dataset with only 10 properties which give us 17 mappings from the DBpedia lexicon, as discussed in the first paragraph of this section. With stemming on the surface forms to counter minor variations (verb conjugations, multiplicity), 8 of the 17 mappings were also found by our method. Thus, we can state the recall of the prototype implementation applied to BOA patterns of 10 properties is 47%.

5 Conclusion and future work

The method proposed is an interesting use case of BOA patterns. The mappings extracted from the manual DBpedia lexicon already can be useful for NLP purposes. The approach for automatic generation of new mappings extends the coverage of the DBpedia lexicon. The proposed method for automatic extraction of mappings can be extended to a method for automatic lexicon generation.

Adding the scores derived from BOA and our method improves the usability of the extended lexical mappings for NLP purposes. For example, the commonness score of a mapping tells how often the surface form is used to refer to the given ontology element, w.r.t. other ontology elements that can be referred to by the same surface form.

Preliminary evaluation of the proposed method using a sample BOA pattern dataset indicates acceptable recall. However, it should be noted that the preliminary evaluation was done on a very small dataset. The method should perform better on bigger datasets.

Here, the prototype implementation and its preliminary evaluation are presented. Future work includes improving the scoring system for better estimation of the relevance of (meta-)patterns and mappings and improvement of the specificity criterium for meta-pattern selection in the second phase. Using POS tags in the pattern matching process should reduce the number of generated noisy mappings dramatically. The ultimate goal is a method for automated expansion of ontology lexica.

References

[1] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka Jr, and T. M. Mitchell. Toward an Architecture for Never-Ending Language Learning. In *AAAI*, 2010.

[2] D. Gerber and A.-C. Ngonga Ngomo. Extracting Multilingual Natural-Language Patterns for RDF Predicates. In *Proceedings of the 18th International Conference on Knowledge Engineering and Knowledge Management*, 2012.

[3] J. Lehmann, D. Gerber, M. Morsey, and A.-C. Ngonga Ngomo. Defacto - deep fact validation. In *Proceedings of the International Semantic Web Conference*, 2012.

[4] C. Unger, L. Bühmann, J. Lehmann, A.-C. Ngonga Ngomo, D. Gerber, and P. Cimiano. Template-based question answering over rdf data. In *Proceedings of the 21st international conference on World Wide Web*, pages 639–648, 2012.

[5] C. Unger, J. Mccrae, S. Walter, S. Winter, and P. Cimiano. lemon lexicon for DBpedia. In *Proceedings of 1st International Workshop on NLP and DBpedia, October 21-25, Sydney, Australia*, volume 1064 of *NLP & DBpedia 2013*, Sydney, Australia, October 2013. CEUR Workshop Proceedings.