Automatic acquisition of adjective lexicalizations of restriction classes

Sebastian Walter, Christina Unger, Philipp Cimiano, and Bettina Lanser

Semantic Computing Group, CITEC, Bielefeld University

Abstract. Lexical knowledge plays a vital role for systems translating between natural language and structured data, and an important part of such lexical knowledge are adjectives. In this paper we introduce a low-cost method for automatically acquiring adjective lexicalizations of restriction classes from a knowledge base by inspecting the range of properties. The resulting lexicalizations can then, for example, be added to the existing manual DBpedia lexicon, achieving a significant increase in coverage.

1 Introduction

There is an increasing interest in providing common web users with access to structured knowledge bases such as DBpedia, e.g. by means of question answering systems. An essential task of such systems is translating between natural language and structured data. To this end, they require knowledge about how the vocabulary elements used in the available ontologies and datasets are verbalized in natural language, covering different verbalization variants, possibly in multiple languages.

An important part of such lexical knowledge are adjectives. For example, the 250 training and test questions of the QALD-4 benchmark[1] for question answering over DBpedia contain 76 adjectives. Most of these adjectives are gradable (e.g. high) or intersective (e.g. Australian). While the former cannot straightforwardly modeled in OWL (see [3]), the latter denote simple restriction classes that are not explicitly named in DBpedia. For example, Danish denotes the class $\exists$ `country.Denmark`, female denotes the class $\exists$ `gender.Female`, and Catholic denotes the class $\exists$ `religion.Catholic_Church` $\sqcup$ `Catholicism`. Knowledge about such adjectives is essential, for instance, when translating natural language questions such 1 into SPARQL queries such as 2.

1. Which female Danish politicians are catholic?
2.
```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX res: <http://dbpedia.org/resource/>
SELECT DISTINCT ?x WHERE {
    ?x rdf:type dbo:Politician .
    ?x dbo:country res:Denmark .
```

[1] `http://www.sc.cit-ec.uni-bielefeld.de/qald/`

```
    ?x dbo:gender res:Female .
  { ?x dbo:religion res:Catholic_Church . }
  UNION
  { ?x dbo:religion res:Catholicism .}
}
```

State-of-the-art approaches to learning lexicalizations, such as [9], [1] and [8], do not yet include methods for learning adjective lexicalizations. Therefore the generated lexica are necessarily incomplete and do not provide support when interpreting questions as the one above. In this paper we propose a lightweight approach for filling this gap, automatically acquiring adjective lexicalizations of restriction classes from a knowledge base.

2 Methodology

The goal is to learn adjective lexicalizations w.r.t. a knowledge base by inspecting the range of properties. The property `gender` in DBpedia, for example, occurs very often with objects `Male` and `Female`, thus male and female are obvious lexicalizations of the restriction classes $\exists$`gender.Male` and $\exists$`gender.Female`, respectively. Similarly, the property `country` occurs with objects like `Denmark`, `Germany`, etc., which have related adjective forms Danish and German, respectively.

Algorithm 1 gives an overview of our proposed approach. Input is a knowledge base, in our case DBpedia, e.g. accessed through a SPARQL endpoint; output is an ontology lexicon in *lemon* [4] format containing adjective lexicalizations of all those property that occur with a sufficient number of objects that contain either adjective forms or noun forms that have related adjective forms.

The first step consists of retrieving a list $O(p)$ of all object literals or labels for this property, and a list $O'(p)$ of pairs (a, o) if o contains as substring an adjective a or a noun that has a related adjective form a. To this end, we use WordNet [6] and DBnary [2]. If the overall ratio of both is greater than some threshold θ_1, i.e. if sufficiently many object strings contain (or are related to) adjective forms, p is considered a candidate for lexicalization. Next, we group all pairs (a, o) by the adjective form a. If there is more than one object containing the same adjective string and if the ratio of objects with this adjective string and objects in general is greater than some threshold θ_2, i.e. if an adjective form is contained by a sufficient number of objects, then we generate lexical entries for the adjective a. If p is a datatype property, the entry looks as in 3, if p is an object property, the entry looks as in 4. These are ontology lexicon macros [5].

3. `IntersectiveDataPropertyAdjective`(a, p, o)
4. `IntersectiveObjectPropertyAdjective`(a, p, o)

As an example, consider the property `gender`: 98 % of the objects are or contain an adjective (e.g. female, male, mixed-sex), which leads to a ratio $|O'(p)|/|O(p)|$

```
for each property p ∈ DBpedia do
    O(p) = [ o s.t. (s,p,o) ∈ DBpedia and o is a literal, or label(o) s.t.
    (s,p,o) ∈ DBpedia and o is a URI ];
    O'(p) = [ (a,o) ∈ O(p) s.t. o contains an adjective a or a noun for which
    there is a related adjective form a ];
    /* Collect all lexicalizations of p in L(p)                    */
    L(p) = { }
    if |O'(p)| / |O(p)| > θ1 then
        /* p is a candidate for lexicalization                     */
        lex(a) = { (a,o) s.t. (a,o) ∈ O'(p) };
        if |lex(a)| / |O'(p)| > θ2 then
            for each (a,o) ∈ lex(a) do
                /* Generate lemon entry                            */
                if p is datatype property then
                    L(p)+=IntersectiveDataPropertyAdjective(a,p,o);
                end
                if p is object property then
                    L(p)+=IntersectiveObjectPropertyAdjective(a,p,o);
                end
            end
        end
    end
end
```

Algorithm 1: *Algorithm for generating adjective entries.*

of 98 %. Similarly, 81 % of the objects in the range of the property `status` contain an adjective (e.g. such operational, active, retired). This leads to resulting lexical entries such as the following ones:

5. `IntersectiveDataPropertyAdjective("active",dbo:status,"active")`
6. `IntersectiveObjectPropertyAdjective("female",dbo:gender,res:Female)`

The algorithm uses two thresholds: θ_1 determines whether the objects of some property contain enough adjective forms, θ_2 determines whether a particular adjective form is contained in enough objects to be considered as lexicalization. Depending on how loose or strict these thresholds are set, more or less lexicalizations are considered. In the following section we run our algorithm on the DBpedia 3.9 properties and report on the number and quality of lexicalizations for different thresholds.

3 Proof of concept and discussion

As proof of concept, we run our proposed algorithm on all 1,371 properties from the DBpedia 3.9 ontology, of which 923 occur with at least one object that contains an adjective form. Table 1 lists the number of properties that are considered for lexicalization depending on the threshold θ_1, i.e. depending on how many of

the objects contain adjective forms. 26 properties occur only with objects containing adjective forms. This is in many cases meaningful, for example `hairColor` occurs with objects like `Black_hair` and `Blonde`, but in other cases accidental, for example `fundedBy` occurs only with objects like `European_Commission`.

θ_1 (in %)	# candidate properties
100	26
80	111
60	194
40	326
20	546
10	726
0	923

Table 1: *Number of properties considered for lexicalization depending on* θ_1.

Considering a threshold θ_1 of 80 % and θ_2 of 1 % (in order to avoid too much noise, e.g. adjective forms that occur only once or twice in a large number of objects), the average number of generated lexicalizations per property is 147, with 2,412 the maximum number, and 1 the minimum number of lexicalizations per property.

In order to analyze the quality of the lexicalizations, we have randomly chosen five properties that pass a threshold θ_1 of 80 %: `architecturalStyle`, `colour`, `geologicPeriod`, `militaryBranch`, and `party`. For these properties we manually evaluated all generated adjective lexicalizations by sorting them into four categories:

- The proposed entry is a *direct lexicalization* of the property, i.e. it is correct. An example is the adjective `blue` as lexicalization of the restriction class $\exists$ `colour.Navy_blue`.
- The proposed entry is a *related lexicalization*, i.e. it is not a direct lexicalization of the property but is semantically related to it. An example is the noun **mayor** as lexicalization of the property `leader`: although every mayor counts as a leader, not every leader is a mayor. We found that these cases are very rare with adjectives.
- The proposed entry is *not a valid lexicalization* of the property, i.e. it is wrong. An example is the adjective `new` as lexicalization of the restriction class $\exists$ `party.New_Zealand_Liberal_Party`.
- The proposed entry has the *wrong part of speech*, i.e. is not an adjective. These cases are due to errors in the lexical resources and could be avoided by using a stricter condition, e.g. that a word has to be tagged as an adjective by all resources instead of just by one of them.

The results for the mentioned five properties are given in Table 2, where *all* specifies the number of all proposed adjective lexicalizations, and *entries*

specifies the number of *lemon* entries resulting from the direct lexicalizations. Usually there are several entries for one adjective form, in particular one entry per restriction class. For example the lexicalization liberal results in 39 entries, denoting 39 restriction classes, including ∃ `party.Liberal_Party_of_Australia` and ∃ `party.Liberal_Party_of_Cuba`.

The results show that for the properties `architecturalStyle`, `colour`, `geologicPeriod`, and `party`, mainly correct lexicalizations were found. Here are a few examples:

- Gothic (∃ `architecturalStyle.Gothic_architecture`)
- red (∃ `colour.Red`)
- blue (∃ `colour.Midnight_blue`)
- Cambrian (∃ `geologicalPeriod."Cambrian"`)
- democratic (∃ `party.Democratic_Labor_Party`)
- communist (∃ `party.Communist_Party_of_Ireland`)

The property `militaryBranch`, however, resulted in almost only wrong lexicalizations (with the exception of confederate). This is due to many objects containing the adjective form united (as in United States Army, armed (as in Austrian Armed Forces), or forms such as British (as in British Army). In these cases, the occurring adjective forms are generally not an appropriate lexicalization of the restriction classes in question. This shows a limitation of our approach.

Property	all	direct	related	not valid	wrong POS	entries
`architecturalStyle`	23	20	0	3	0	48
`colour`	13	12	0	0	1	25
`geologicPeriod`	17	14	0	0	3	18
`militaryBranch`	11	1	0	9	1	3
`party`	17	9	0	5	3	468

Table 2: *Evaluation of the generated lexicalizations for five randomly chosen properties.*

Additionally, we checked how many of the correctly generated lexicalizations were not yet captured in the hand-crafted DBpedia lexicon[2] [7]. This, in fact, amounts to all of them, as the lexicon contains only one (non-adjective) entry for `colour`, `geologicalPeriod`, and `militaryBranch` each, and no entries for `architecturalStyle` and `party`. Furthermore, currently included adjective entries, e.g. for nationalities (such as Chinese for ∃ `nationality.China`) and religions (such as Buddhist for ∃ `religion.Buddhism`), are incomplete, usually only covering a few common cases. Thus, the DBpedia lexicon in its current state would greatly benefit from a semi-automatic process in which adjective lexicalizations for a large number of properties are generated automatically and then checked and corrected by a lexicon engineer.

[2] `https://github.com/cunger/lemon.dbpedia`

4 Conclusion

We presented a methodology for inducing an adjective lexicon for DBpedia by analyzing the adjectives mentioned in the range of properties. We show that depending on values chosen for two thresholds, results of reasonable accuracy can be obtained. We thus introduce a relatively low-cost and accurate method for adding adjective lexicalizations to the manual DBpedia lexicon presented earlier, achieving an increase by around the factor 20 in the number of adjective lexicalizations.

Acknowledgment

This work was supported by the Cluster of Excellence Cognitive Interaction Technology *CITEC* (EXC 277) at Bielefeld University, which is funded by the German Research Foundation (DFG).

References

1. Daniel Gerber and A-C Ngonga Ngomo. Bootstrapping the linked data web. *1st Workshop on Web Scale Knowledge Extraction, workshop co-located with the 10th International Semantic Web Conference (ISWC 2011)*, 2011.
2. Sérasset Gilles. DBnary: Wiktionary as a lemon-based multilingual lexical resource in RDF. *Semantic Web Journal (special issue on Multilingual Linked Open Data)*, to appear.
3. John McCrae, Francesca Quattri, Christina Unger, and Philipp Cimiano. Modelling the semantics of adjectives in the ontology-lexicon interface. In *Proceedings of Cognitive Aspects of the Lexicon (CogAlex), workshop co-located with the 25th International Conference on Computational Linguistics (COLING 2014)*, 2014.
4. John McCrae, Dennis Spohr, and Philipp Cimiano. Linking lexical resources and ontologies on the semantic web with lemon. In *The Semantic Web: Research and Applications*, pages 245–259. Springer, 2011.
5. John McCrae and Christina Unger. Design patterns for engineering the ontology-lexicon interface. In Paul Buitelaar and Philipp Cimiano, editors, *Towards the Multilingual Semantic Web*. Springer, 2014.
6. George A. Miller. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995.
7. Christina Unger, John McCrae, Sebastian Walter, Sara Winter, and Philipp Cimiano. A lemon lexicon for DBpedia. In *Proceedings of 1st International Workshop on NLP and DBpedia, co-located with the 12th International Semantic Web Conference (ISWC 2013), October 21-25, Sydney, Australia*, 2013.
8. Marta Vila, Horacio Rodríguez, and M Antònia Martí. WRPA: A system for relational paraphrase acquisition from Wikipedia. *Procesamiento del lenguaje natural*, 45:11–19, 2010.
9. Sebastian Walter, Christina Unger, and Philipp Cimiano. M-ATOLL: A framework for the lexicalization of ontologies in multiple languages. In *Proceedings of the 13th International Semantic Web Conference (ISWC 2014)*, 2014.