Learning of OWL Class Descriptions on Very Large Knowledge Bases

Sebastian Hellmann, Jens Lehmann, Sören Auer
Universität Leipzig, Department of Computer Science
Johannisgasse 26, D-04103 Leipzig, Germany
{auer,hellmann,lehmann}@informatik.uni-leipzig.de

ABSTRACT

The vision of the Semantic Web is to make use of semantic representations on the largest possible scale - the Web. Large knowledge bases such as DBpedia, OpenCyc, GovTrack, and others are emerging and are freely available as Linked Data and SPARQL endpoints. Exploring and analyzing such knowledge bases is a significant hurdle for Semantic Web research and practice. As one possible direction for tackling this problem, we present an approach for obtaining complex class descriptions from objects in knowledge bases by using Machine Learning techniques. We describe how we leverage existing techniques to achieve scalability on large knowledge bases available as SPARQL endpoints or Linked Data. Our algorithms are made available in the open source DL-Learner project and can be used in real-life scenarios by Semantic Web applications.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning—*Concept Learning*; F.4.3 [**Mathematical Logic and Formal Languages**]: Formal Languages; H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Algorithms, Performance

Keywords

Large Knowledge Bases, Machine Learning, Class Description, SPARQL, OWL

1. INTRODUCTION

The vision of the Semantic Web is to make use of semantic representations on the largest possible scale - the Web. We currently experience that Semantic Web technologies are gaining momentum and large knowledge bases such as DBpedia[1], OpenCyc[6], GovTrack[1] and others are freely available. These knowledge bases are based on semantic knowledge representation standards like RDF and OWL. They describe millions of concepts, contain hundred thousands of properties as well as classes and an even larger number of facts and relationships. These knowledge bases and many more[2] are available as Linked Data [2] or SPARQL endpoints.

Due to their sheer size, precise retrieval of information remains a burden for users. Domain experts often have a very precise imagination what kind of results they would like to retrieve, but might not be able to express their queries in a structured form at all. A historian, for example, searching for ancient Greek law philosophers influenced by Plato in DBpedia can easily name some examples and if presented a selection of prospective results he will be able to quickly identify false results. However, he might not be able to efficiently construct a formal query adhering to the large DBpedia knowledge base a priori.

2. THE METHOD

The basic problem we want to solve is common in Machine Learning: Given a set of positive and negative examples, find a definition entailing all positive examples and none of the negatives. For instance, given a set of substances knowing to cause cancer and other substances, which do not cause cancer, we can try to learn a description of the substances causing the disease. In our context, examples are OWL individuals and the learned descriptions are OWL class descriptions[3]. Section 3 gives a concrete example learning problem.

One of the challenges of such methods is that they heavily rely on reasoning to learn class descriptions. Their scalability is thus limited by that of the underlying reasoning algorithm. Although advancements have been made in approximate reasoning for OWL, it is not feasible to load very large knowledge bases like DBpedia, OpenCyc, and others into a reasoner. Furthermore, not all of the knowledge may be locally available, i.e. large volumes of data may need to be transferred over a network to obtain the background knowledge for a given learning problem.

For this reason, we propose a knowledge fragment selection approach. Figure 1 gives a general overview. A number of example instances is used as a starting point. The fragment selection method gathers knowledge about these instances using SPARQL queries or available Linked Data. This knowledge forms a small fragment of the underlying large knowledge bases and can be consumed efficiently by a reasoner. Finally, this reasoner can be used as a backend for learning algorithms.

The fragment selection approach is, in principle, independent of the used learning algorithm. We implemented our approach in the DL-Learner framework and refer to the lit-

[1]`http://www.govtrack.us`
[2]`http://esw.w3.org/topic/SparqlEndpoints` and `http://linkeddata.org`
[3]`http://www.w3.org/TR/owl2-syntax/#Descriptions`

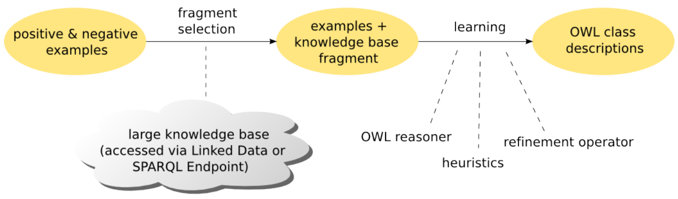
positive & negative examples
fragment selection
examples + knowledge base fragment
learning
OWL class descriptions
large knowledge base (accessed via Linked Data or SPARQL Endpoint)
OWL reasoner
heuristics
refinement operator

Figure 1: Process Illustration: In a first step, a fragment is selected based on instances from a knowledge source and in a second step the learning process is started on this fragment and the given examples.

erature[3, 4, 5] for information about some of the algorithms implemented within this framework. The main goal of the fragment selection procedure is to select relevant knowledge. For learning OWL class descriptions, this includes related individuals up to a given recursion depth as well as the subsumption hierarchy of concepts and properties of those individuals. Therefore, the algorithm explores the RDF graph recursively and extracts information that provides a sufficient description of the individuals and corresponding concepts to apply the learning algorithm. The extraction of resource descriptions is inspired by Concise Bounded Descriptions (CBD[4]), but resources are additionally filtered (to omit irrelevant facts) and corrected in a way that they comply with OWL DL, normally a requirement for reasoners. Figure 2 depicts the basic process: Starting from a set of instances, neighbours of those are explored and in a final step schema information is obtained for those instances. For brevity, we omit the technical details.

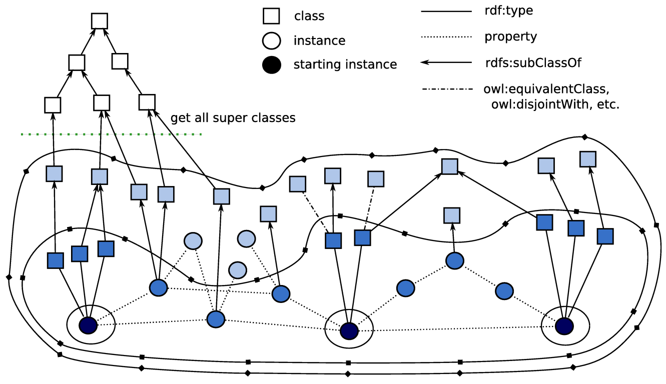

Figure 2: Extraction with three starting instances. The inner line represents recursion depth 1 and the outer recursion depth 2.

We evaluated the method on the DBpedia and YAGO knowledge bases. For our experiments we choose a typical ontology enrichment task, i.e. learn class definitions. We randomly selected YAGO classes and retrieved DBpedia instances that belong to the class as positive examples and then selected the same number of negative examples from superclasses. We performed an extraction with varying recursion depth, which is the most important factor influencing performance, and measured 1) number of triples extracted, 2) time needed for extraction and 3) total time needed for extraction and learning. To summarize the results, we found out that all the number of extracted triples grows exponentially with recursion depth and that all three curves have a similar shape. This means that extraction and learning performance are largely determined by the knowledge fragment size, which can in turn be regulated by the recursion depth.

[4]http://www.w3.org/Submission/CBD

3. USAGE SCENARIOS

Learned class descriptions are useful in numerous scenarios, because they represent a concise theory about the example instances. This opens the field for uses such as:

Discovery/Recommendation: Find instances via retrieval which are similar to the positive examples, but not similar to the negative examples.

Navigation: Generate navigation suggestion based on previously visited sites, articles etc.

Ontology Engineering: Enrich very large knowledge bases by learning the definition of a class using existing instances of this class.

Instance data analysis - The learning algorithm is biased towards short human-readable class descriptions, which give, like short profiles, insight into the data.

Classification: Learn descriptions to classify unseen instances, e.g. detecting which substances may cause cancer based on a learned description.

In continuation of the evaluation, we used the method to learn class descriptions in DBpedia with the goal of finding missing instances, i.e. instances which belong to the class, but where Wikipedia authors forgot to make such assertions. Furthermore, we used the AudioScrobbler RDF Service[5] to obtain a list of artists, which a user from Last.fm listened to. Because they are connected to MusicBrainz [6], we were able to create short profiles in form of a class description (e.g. $\texttt{UK-Artist} \sqcup (\texttt{Rock-Genre} \sqcap \exists \texttt{bioEvent.Death})$) for the artists most recently listened to by a user (in this case "Genesis", "Children of Stun", "Robbie Williams", and "Dusty Springfield" as positive examples).

4. REFERENCES

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. G. Ives. DBpedia: A nucleus for a web of open data. In *ISWC/ASWC (2007)*, LNCS (4825), pages 722–735. Springer, 2007.

[2] T. Berners-Lee. Linked data, 2006. `http://www.w3.org/DesignIssues/LinkedData.html`.

[3] J. Lehmann. Hybrid learning of ontology classes. In *MLDM 2007*, volume 4571 of *LNCS*, pages 883–898. Springer, 2007.

[4] J. Lehmann and P. Hitzler. Foundations of refinement operators for description logics. In *ILP 2007, Revised Selected Papers*, volume 4894 of *LNCS*, pages 161–174. Springer, 2008.

[5] J. Lehmann and P. Hitzler. A refinement operator based learning algorithm for the alc description logic. In *ILP 2007, Revised Selected Papers*, volume 4894 of *LNCS*, pages 147–160. Springer, 2008.

[6] D. Lenat. CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38, 1995.

[5]http://dbtune.org/last-fm
[6]http://musicbrainz.org