Web Semantics: Science, Services and Agents on the World Wide Web 39 (2016) 15–24

Contents lists available at ScienceDirect

Web Semantics: Science, Services and Agents on the World Wide Web

journal homepage: www.elsevier.com/locate/websem

DL-Learner—A framework for inductive learning on the Semantic Web

Lorenz Bühmann [a,*], Jens Lehmann [b], Patrick Westphal [a]

[a] University of Leipzig, Institute of Computer Science, AKSW Group, Augustusplatz 10, D-04009 Leipzig, Germany

[b] University of Bonn, Institute of Computer Science, Römerstr. 164, D-53117 Bonn, Germany

ARTICLE INFO

Article history:
Received 28 July 2015
Received in revised form
25 May 2016
Accepted 19 June 2016
Available online 1 July 2016

Keywords:
System description
Machine learning
Supervised learning
Semantic Web
OWL
RDF

ABSTRACT

In this system paper, we describe the DL-Learner framework, which supports supervised machine learning using OWL and RDF for background knowledge representation. It can be beneficial in various data and schema analysis tasks with applications in different standard machine learning scenarios, e.g. in the life sciences, as well as Semantic Web specific applications such as ontology learning and enrichment. Since its creation in 2007, it has become the main OWL and RDF-based software framework for supervised structured machine learning and includes several algorithm implementations, usage examples and has applications building on top of the framework. The article gives an overview of the framework with a focus on algorithms and use cases.

1. Introduction

Over the past two decades, we have witnessed a transition from an industrial driven society to a data and knowledge driven society. This trend is accompanied by a significant increase in research interest in and importance of (large-scale) data processing methods. In this broad field, semantic technologies have emerged as a means to structure, publish and integrate data. In particular, the RDF and OWL knowledge representation W3C standards are used in thousands of knowledge bases containing billions of facts.[1]

A major challenge that research faces today is to analyse this growing amount of information to obtain insights into the underlying problems. In many cases, in particular in the life sciences, it is beneficial to employ methods that are able to use the complex structure of available background knowledge when learning hypotheses. DL-Learner is an open software framework, which contains several such methods. It has the primary goal to serve as a platform for facilitating the implementation and evaluation of supervised structured machine learning methods using semantic background knowledge.

The most common scenario we consider is to have a background knowledge base in OWL and be additionally provided with sets of individuals in our knowledge base which serve as positive and negative examples. The goal is to find a logical formula, e.g. an OWL *class expression*,[2] such that all/many of the positive examples are *instances* of this expression and none/few of the negative examples are instances of it. The primary purpose of learning is to find a class expression which can classify unseen individuals (i.e. not belonging to the examples) correctly. It is also important that the obtained class expression is easy to understand for a domain expert. We call these criteria *accuracy* and *readability*.

As an example, consider the problem to find out whether a chemical compound can cause cancer. In this case, the background knowledge contains information about chemical compounds in general and certain concrete compounds we are interested in. The positive examples are those compounds causing cancer, whereas the negative examples are those compounds not causing cancer. The prediction for those examples may have been obtained from experiments and expensive long-term research trials. A learning algorithm can now derive a hypothesis from examples and background knowledge, e.g. a learned class expression in natural language could be "chemical compounds containing more than three phosphorus atoms". (Of course, in practice the expression will be more complex to obtain a reasonable accuracy.) Using this class expression, we can now classify unseen chemical compounds.

* Corresponding author.
E-mail addresses: buehmann@informatik.uni-leipzig.de (L. Bühmann), jens.lehmann@cs.uni-bonn.de (J. Lehmann), westphal@informatik.uni-leipzig.de (P. Westphal).

[1] http://lodstats.aksw.org/.

[2] http://www.w3.org/TR/owl2-syntax/#Class_Expressions.

http://dx.doi.org/10.1016/j.websem.2016.06.001
1570-8268/

To solve this and similar problems, researchers need to overcome hurdles which are highly important for machine learning in general: Algorithms have to process complex background knowledge, possibly coming from several sources. Logical inference is needed during the learning process to drive the algorithms. During their runtime, algorithms frequently need to evaluate thousands or millions of hypotheses and use the results of those tests to determine their learning strategy. The expressions which are eventually learned, can often be arbitrarily nested and in some cases need to portray complex relationships while still being as easy as possible to understand for domain experts. Some of those machine learning challenges have been identified by leading researchers, e.g. in [1] and [2], and DL-Learner aims to provide a platform to facilitate researchers in their quest for solutions.

A previous system paper on DL-Learner appeared in 2009 in the Journal of Machine Learning Research [3]. Compared to this system description, the major changes are as follows:

- Framework design: The framework has been generalized from being focused on learning OWL class expressions using OWL ontologies as background knowledge to a more generic supervised structured machine learning framework. Components are integrated via Java Beans and the Java Spring framework, which allows more fine-grained and more flexible interaction between them.
- New algorithms for learning SPARQL queries (as feedback component in question answering), fuzzy description logic expressions, parallel OWL class expression learning, a special purpose algorithm for the $\mathcal{EL}$ description logic, two algorithms for knowledge base enrichment of almost all OWL 2 axioms from SPARQL endpoints as well as an algorithm combining inductive learning with natural language processing have been integrated.
- Scalability enhancements: There are now statistical sampling methods available for dealing with a large number of examples as well as different knowledge base fragment selection methods for handling large knowledge bases in general.
- Major engineering changes: In 2011, the whole framework was refactored to be more easily extensible. Moreover, algorithms are continuously extended with options based on received feature requests. In the same manner, new APIs and reasoners are continuously upgraded.
- Code repository statistics: About 3500 commits were made which led to more than 4,500,000 changed lines of code. 30 new contributors could be acquired, 52 bugs could be fixed and 27 feature requests could be realized.

The article is structured as follows: In Section 2, we give a description of the problems DL-Learner aims to solve. Subsequently, in Section 3, the software framework is described. We summarize core algorithms implemented in DL-Learner in Section 4. Implementation statistics and notes are given in Section 5. Use cases of DL-Learner in different problem areas are covered in Section 6. In Section 7, we describe related work and give an outlook in Section 8.

2. Learning problems

The process of learning in logics, i.e. trying to find high-level explanations for given data, is also called *inductive reasoning* as opposed to *inference* or *deductive reasoning*. Deductive reasoning is known as the process of deriving logically certain conclusions from a set of general statements that are known to be true, e.g. given a statement like "Every bird can fly", we can deductively derive that the bird "tweety" can fly. Contrarily, the concept of inductive reasoning is to construct general statements from a given set of examples. For instance, given ten ravens out of which nine have a black colour, one could inductively derive that "all ravens are black" even though it does not hold for all ravens and bears some degree of uncertainty. Learning problems which are similar to the one we will analyse, have been investigated in *Inductive Logic Programming* [4].

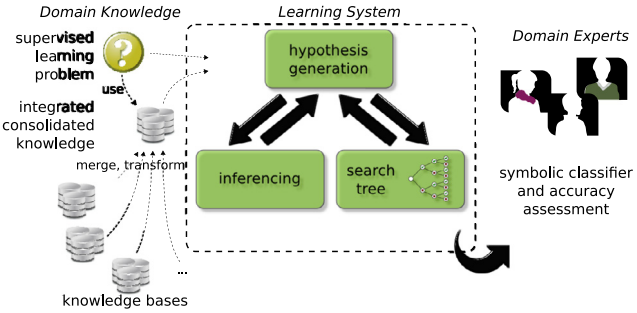

Fig. 1. General learning workflow.

The goal of DL-Learner is to provide a structural framework and reusable components for solving those induction problems. Fig. 1 depicts a typical workflow from a user's perspective. On the left hand side, there are several knowledge bases which together form the *background knowledge* for a given task. Within that background knowledge, some resources are selected as positive and some others as negative examples. In a medical setting, the resources could be patients reacting to a treatment (positive examples) and patients not reacting to a treatment (negative examples). Those are then processed by a supervised machine learning algorithm and return (in most cases in DL-Learner) a *symbolic classifier*. This classifier is human readable and expressed in a logical form, e.g. as a complex description logic concept or a SPARQL query. It serves two purposes: First, due to its logical representation it should give insights into the underlying problem, showing which concepts are relevant to distinguish positive and negative examples. Furthermore, the result can also be used to classify unseen resources, e.g. by checking whether they are an instance of the learned concept using an OWL reasoner, or whether they are returned by a SPARQL endpoint executing a learned query.

In DL-Learner, the following learning problems are relevant:

Standard supervised learning Let the name of the background ontology be $\mathcal{O}$. The goal in this learning problem is to find an OWL class expression C such that all/many positive examples are instances of C w.r.t. $\mathcal{O}$ and none/few negative examples are instances of C w.r.t. $\mathcal{O}$.

Positive only learning In case only positive examples are available, it is desirable to find a class expression that covers the positive examples while still generalizing sufficiently well (usually measured on unlabelled data).

Class learning In class learning, you are given an existing class A within an ontology $\mathcal{O}$ and want to describe it. This is similar to the previous problem in that you can use the instances of the class as positive examples. However, you can make use of existing knowledge about A in the ontology and (obviously) A itself should not be a solution.

In addition, there are different nuances of the above learning problems which depend on how negative knowledge should be treated (related to the open world assumption in description logics): The problems can be treated as a binary problem (hypotheses should cover positive examples and not cover negative examples) or a ternary problem (hypotheses should cover positive examples, cover the negation of negative examples and not cover all other resources). The preferable setting partially depends on the structure of the background knowledge. For ternary learning problems, more sophisticated schema axioms, e.g. containing negation, disjunction