Inductive Lexical Learning of Class Expressions

Lorenz Bühmann[1], Daniel Fleischhacker[2], Jens Lehmann[1], Andre Melo[2], and Johanna Völker[2]

[1] AKSW Research Group, University of Leipzig, Germany, {buehmann,lehmann}@informatik.uni-leipzig.de
[2] Data & Web Science Research Group, University of Mannheim, Germany, {daniel,andre,johanna}@informatik.uni-mannheim.de

Abstract. Despite an increase in the number of knowledge bases published according to Semantic Web W3C standards, many of those consist primarily of instance data and lack sophisticated schemata, although the availability of such schemata would allow more powerful querying, consistency checking and debugging as well as improved inference. One of the reasons why schemata are still rare is the effort required to create them. Consequently, numerous ontology learning approaches have been developed to simplify the creation of schemata. Those approaches usually either learn structures from text or existing RDF data. In this submission, we present the first approach combining both sources of evidence, in particular we combine an existing logical learning approach with statistical relevance measures applied on textual resources. We perform an experiment involving a manual evaluation on 100 classes of the DBpedia 3.9 dataset and show that the inclusion of relevance measures leads to a significant improvement of the accuracy over the baseline algorithm.

1 INTRODUCTION

There has recently been an increase in the number and size of RDF knowledge bases, in particular in the context of the Linked Open Data initiative. However, there is still a lack of knowledge bases that use expressive ontologies and instance data structured according to those ontologies. Many datasets focus on instance data and give less attention to the ontological layer. One of the reasons for this is the effort required to build up an ontology. To address this problem, a multitude of approaches have been devised using a plethora of methods [24]. In particular, there have been two main branches of research: On the one hand, lexical ontology learning approaches aim at constructing ontologies from textual input [25] and, on the other hand, logical learning approaches use existing RDF data as input to construct ontologies [15, 3]. In this work, we present the first algorithm, we are aware of, which combines lexical and logical ontology learning. This constitutes the first step on a larger research agenda aiming to improve ontology learning algorithms to a state in which they achieve sufficient precision and recall to be employed in practice. Previous studies have shown that current algorithms have not yet achieved this goal (see e.g. [20]) and ontology learning remains an extremely challenging problem.

Using a short example, we briefly want to illustrate how schemata improvements can enable more powerful reasoning, consistency checking, and improved querying possibilities. In particular, in this article we are concerned with learning $\mathcal{EL}$ description logic concepts for definitions.

Example 1. The following definition in description logic syntax was learned by our approach for the class `Astronaut`[3] in DBpedia [23].

$$\begin{aligned}\texttt{Astronaut} \equiv \texttt{Person} \sqcap \exists \texttt{mission.SpaceMission} \\ \sqcap \exists \texttt{timeInSpace.minute}\end{aligned}$$

The definition states that a person who was on a space mission and spent time in space is an astronaut and vice versa. Adding this definition to an ontology can have the following benefits: 1.) It can be used to detect inconsistencies and quality problems. For instance, when using the Pellet Constraint Validator[4] on a knowledge base with the above axiom, it would report astronauts without an associated space mission as violation.[5] 2.) Additional implicit information can be inferred, e.g., in the above example each person, who was on a space mission and spent time in space can be inferred to belong to the class `Astronaut`, which means that an explicit assignment to that class is no longer necessary. 3.) It can serve as documentation for the purpose and correct usage of schema elements. For instance, in the above example it can be argued that someone is an astronaut if he is trained for a space mission, whereas the definition requires to actually take part in such a mission. The definition clarifies the intended usage. Overall, we make the following contributions:

- first approach to combining logical and lexical ontology learning
- analysis of statistical relevance measures for learning class expressions
- a manual evaluation on a realistic large scale data set

The adapted algorithm is called *ELTL* ($\mathcal{EL}$ Tree Learner) and part of the open-source framework DL-Learner[6] [19] for concept learning in description logics (DLs). The remainder of the paper is structured as follows: In Section 2 we present related work. Section 3 covers preliminaries such as a definition of the learning problem in logical ontology learning and a description of the base algorithm we use. Subsequently, in Section 4, we describe how statistical relevance measures applied on textual resources can be integrated into the logical learning framework. Section 5 describes experiments and insights obtained from them and we conclude in Section 6.

2 RELATED WORK

Since we presented the first approach towards unifying logical (data-based) and lexical (text-based) ontology learning, we describe related work in both areas.

Ontology Learning from Structured Data. Early work on the application of machine learning to Description Logics (DLs) essentially focused on demonstrating the PAC-

[3] We omit the namespace `http://dbpedia.org/ontology/` for readability

[4] `http://clarkparsia.com/pellet/icv/`

[5] Under OWL semantics, this is not a violation, due to the Open World Assumption, unless we can infer from other knowledge that the person cannot have taken part in a mission

[6] `http://dl-learner.org`

learnability for various terminological languages derived from CLASSIC. In particular, Cohen and Hirsh investigate the CORECLASSIC DL proving that it is not PAC-learnable [5] as well as demonstrating the PAC-learnability of its sub-languages, such as C-CLASSIC [6], through the bottom-up LCSLEARN algorithm. These approaches tend to cast supervised concept learning to a structural generalizing operator working on equivalent graph representations of the concept descriptions. Recently, many approaches have been proposed that adopt the idea of *generalization as search* [26] performed through suitable operators that are specifically designed for DL languages [2, 15, 21] on the grounds of the previous experience in the context of ILP. There is a body of research around the analysis of such operators [22, 18] and studies on the practical scalability of algorithms using them [14]. Supervised learning systems, such as YINYANG [15] and DL-Learner [19] have been implemented and adoptions implemented for the ontology learning use case [20, 3]. Also techniques from the area of data mining have been used for unsupervised ontology learning [31]. As an alternative model, a new version of the FOIL algorithm [29] has been implemented, resulting in the DL-FOIL system [8]. The general framework has been extended to cope with logical representations designed for formal Web ontologies [9].
Ontology Learning from Text. Unlike logical approaches which have been developed to generate ontologies from structured data, lexical or NLP-based methods [32] draw upon the huge amounts of unstructured text available, e.g., on the web. Many of these methods combine lexico-syntactic patterns (e.g., Hearst patterns [13]) and linguistic resources like WordNet with machine learning techniques. While a growing number of ontology learning methods also leverages linked data or ontologies (e.g. FRED [28]), the results are mostly limited to atomic entities and simple axioms. An exception to this are pattern-based approaches to translating natural language definitions into class expressions such as LExO [30].

Altogether, we can see that attempts have been made to integrate semantic web data and logical inference into lexical approaches to ontology learning. However, there has been little if any work on integrating lexical evidence into logics-based ontology learning algorithms so far.

3 PRELIMINARIES

For an introduction to OWL and description logics, we refer to [1]. In this section, we focus on giving an overview of the base learning algorithm we draw on. The task we investigate resembles *Inductive Logic Programming* [27] using a description logic knowledge base as background knowledge and $\mathcal{EL}$ concepts as target language. In the ontology learning problem we consider, we learn a definition of a class A, which has (inferred or asserted) instances in the considered ontology. To define the class learning problem, we need the notion of a *retrieval* reasoner operation $R_{\mathcal{K}}(C)$, which returns the set of all instances of C in a knowledge base $\mathcal{K}$.

Definition 1 (class learning problem). *Let an existing named class A in a knowledge base $\mathcal{K}$ be given. Analogous to standard information retrieval, the F-Score of an $\mathcal{EL}$ concept C is computed based on precision on recall where the precision is defined as*

$\frac{|R_{\mathcal{K}}(C) \cap R_{\mathcal{K}}(A)|}{|R_{\mathcal{K}}(C)|}$ *and recall as* $\frac{|R_{\mathcal{K}}(C) \cap R_{\mathcal{K}}(A)|}{|R_{\mathcal{K}}(A)|}$. *The goal of the* class learning problem *is to maximize F-Score wrt. A.*

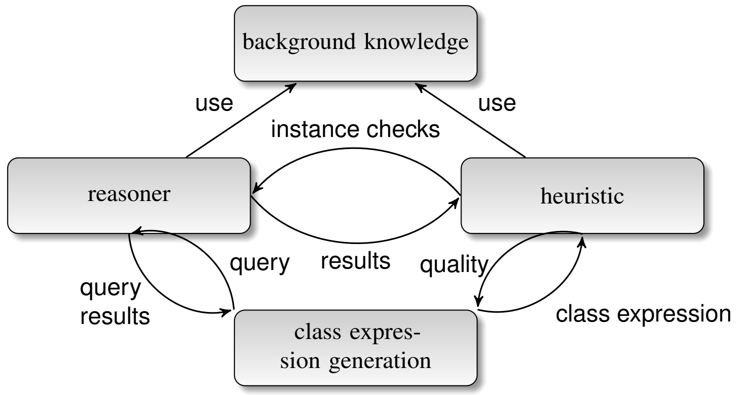

Fig. 1: Outline of the general learning approach in ELTL: Class expressions taking the available background knowledge into account are generated and evaluated in a heuristic with respect to the target learning problem. Figure adapted from [14].

Figure 1 gives a brief overview of our base algorithm *ELTL* ($\mathcal{EL}$ Tree Learner), which follows the common "generate and test" approach in ILP. This means that learning is seen as a search process and several class expressions are generated and tested against a background knowledge base. Each of those class expressions is evaluated using a heuristic, which we will analyze later in more detail.

Definition 2 (refinement operator). *A* quasi-ordering *is a reflexive and transitive relation. In a quasi-ordered space* $(S, \preceq)$ *a* downward (upward) refinement operator ρ *is a mapping from S to 2^S, such that for any $C \in S$ we have that $C' \in \rho(C)$ implies $C' \preceq C$ ($C \preceq C'$). C' is called a* specialization *(*generalization*) of C.*

Refinement operators can be used for searching in the space of expressions. As ordering we can use subsumption ($\sqsubseteq$), which is a quasi-ordering relation. If an expression C subsumes an expression D ($D \sqsubseteq C$), then C will cover all examples which are covered by D. This makes subsumption a suitable order for searching in expressions as it allows to prune parts of the search space without losing possible solutions.

The approach we used is a top-down algorithm based on refinement operators as illustrated in Figure 2. This means that the first class expression which will be tested is the most general expression ($\top$), which is then mapped to a set of more specific expressions by means of a downward refinement operator. Naturally, the refinement operator can be applied to the obtained expressions again, thereby spanning a *search tree*. The search tree can be pruned when an expression does not cover sufficiently many instances of the class A we want to describe.

The heart of such a learning strategy is to define a suitable refinement operator and an appropriate search heuristics for deciding which nodes in the search tree should be expanded. The refinement operator in the ELTL algorithm is defined and evaluated in [21] and has several beneficial theoretical properties not further detailed here.

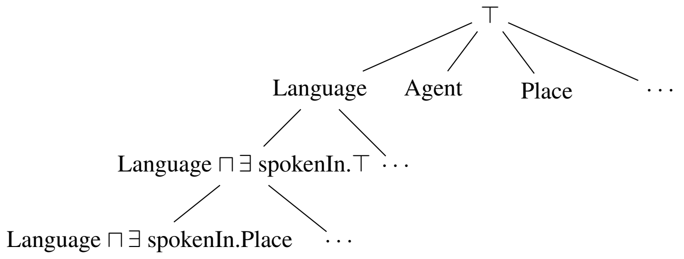

Fig. 2: Illustration of a search tree in ELTL.

4 APPROACH

The learning algorithms in the DL-Learner framework are designed by combining a refinement operator with a search heuristic. While the operator itself is *ideal* with respect to a set of beneficial theoretical properties as shown in [21], we are investigating and improving the learning algorithm by incorporating a more intelligent search heuristic.

Learning concepts in Description Logics is a search process. The refinement operator is used for building the search tree, while a heuristic decides which nodes to expand. This decision can be done based on different criteria like the number of positive and/or negative examples covered by the class expression or the length of the concept represented by the node. While the existing heuristics basically rely on scores based on metrics according to coverage of the examples obtained via logical inference, we have observed in previous experiments [20, 3] that best coverage does not always result in the most intuitive class expressions, and the search process can be improved by taking into account information contained in textual resources. Therefore, we extract statistical information out of given text, which might give some insights on the relevance of other ontology entities for the class we want to describe. This idea is substantiated by the assumption that words which are more related to each other tend to co-occur more often in texts as also expressed in the famous statement to "know a word by the company it keeps" [10] as frequently quoted throughout the linguistics community. This relevance score can then be combined with the other metrics in the heuristic and, thus, influence the navigation in the search tree, which in the end can result in better class descriptions.

Apart from the novel idea of including those relevance measures, one of our main goals was to evaluate which measure is suitable. In order to measure the relevance of entities for the definition of a given entity, we use popular co-occurrence based association measures [4]. The measures employed in this paper are Jaccard, Dice, Semi-conditional Information (SCI), Pointwise Mutual Information (PMI), Log Likelihood Ratio (LLR), Chi-Square (χ^2), T-test, and Significant PMI (sPMI) [7]. The first two measures have the advantage of being simple to compute and their values always fall into the $[0, 1]$ interval. The latter five measures incorporate some notion of statistical significance, by considering the ratio of observed ($f(x, y)$) and expected frequency assuming independence $Ef(x, y)$ of an entity pair x, y. PMI takes into account only the occurrence probabilities, ignoring the absolute frequency, which results in a tendency to yield high score values for low frequency pairs. sPMI solves this problem by incorporating corpus level significance, which considers the probability of observing a given deviation between $f(x, y)$ and its expected value $Ef(x, y)$. SCI multiplies PMI by the conditional probability $p(y|x)$, which tends to favor highly frequent pairs, therefore compensating the PMI's problem. LLR and χ^2 are the only ones which have the null addition property [4], which means that the measure is affected by the addition of data containing neither x not y.

Apart from influencing the search process in the learning algorithm, the textual evidence included by relevance measures can also influence the final ranking of class expressions. Actually, we have two possibilities of how the learning process can benefit from information contained in texts related to the knowledge base:

External Text Corpus The first option relies on an external text corpus $\mathcal{C}$. We treat each document d in the corpus as a separate context and get information about the occurrence of the class c we want to describe, the occurrences of each other schema entity (a class or property) e_i, as well as the joint occurrences of c with each e_i. The retrieved information is then processed by the chosen relevance measures. The computation of the relevance score for each e_i is done in advance before the learning algorithm starts, because we have to normalize the values (we're doing a min-max normalisation), especially when we're using more than one relevance measure. In the current approach, we check the occurrence of an entity in a text by just taking a human-readable name for the entity[7] and check if it occurs syntactically in the text, i.e., we do not perform any kind of disambiguation which is planned to be integrated in a future version.

Local Textual Information The second option uses textual resources which are contained in the underlying knowledge base, e.g., the individuals could be accompanied by textual descriptions summarizing important facts about them like the birth place of a person or that an astronaut participated in a particular space mission. This information can be used to get the relevance of the schema entities e_i by checking for the occurrences of its labels in the descriptions of instances of the class we want to describe.

5 EVALUATION

5.1 Experimental Setup

We performed our experiments on the English DBpedia data set in version 3.9 accessible via a local mirror. The DBpedia data set was extracted from Wikipedia and at its core consists of resources corresponding to Wikipedia articles and facts extracted from article pages. DBpedia provides a lightweight OWL ontology, which defines the different classes and properties used throughout the data set. The DBpedia ontology contained a total of 529 classes of which 423 are leaf classes, i.e., classes not having any non-trivial subclasses. Furthermore, the ontology contained 927 object and 1,406 datatype properties. The ontology is also used for extracting the data from Wikipedia using mappings between infobox templates and ontology classes as described in [23]. The extraction process based on this mapping results in the so-called mapping-based data set which we use in our experiments. Overall, our experiments data set contained about 63.5 million triples describing 3,243,481 instances.

On this data set, we used DL-Learner (specifically ELTL) enhanced with the relevance metrics as described above to learn class expressions for the classes contained in

[7] Usually we use `rdfs:label`, but it's of course possible to use any other property.

the DBpedia ontology. As a corpus for the relevance metrics the abstracts of all Wikipedia articles which described concepts modeled by an ontology class were crawled and then provided for retrieval using a SOLR[8] instance. This corresponds to the "External Text Corpus" scenario described in the Section 4. Though we also implemented the second scenario, we opted to only evaluate the first one which is more generally usable because it does not rely on greater amounts of textual descriptions in the data set itself.

We generated class expressions for all leaf classes of the DBpedia ontology which had at least 3 instances (288 out of 423). We then computed a sample set of at most 100 positive examples (instances belonging to the class) and 200 negative examples. For the negative examples, 100 instances belonging to the sibling classes and 100 instances of super classes which were not contained in the class to describe were randomly chosen. After applying DL-Learner, we performed a manual evaluation to find the combination of relatedness measures which resembles the human perception of intuitiveness of a class expression best. To do this, we randomly chose 100 of the classes with at least 10 alternative class expressions generated. For each class with more than 50 class expressions, we picked the top-50 expressions ranked by F-score. We handed the generated lists to four human annotators (two researchers not involved in the research presented in this submission from the Universities of Mannheim and Leipzig, respectively) along with the instruction to mark the class expressions which they consider most suitable for being added to the DBpedia ontology as definitions of the class. Additionally, we explicitly highlighted the possibility of marking multiple class expressions in cases where they are equally suitable or no expression if there was no expression close to an acceptable definition. The evaluation process took two hours per annotator on average. An example of evaluated class expressions for the class `Astronaut` is shown in Table 1.

In the second part of the evaluation, we applied several classification approaches to the F-score and relevance measures values to find a combination which is suited to reproduce the human assessment of intuitiveness. For this purpose, we employed the implementations provided by the Weka toolkit [12] in version 3.6.6.

5.2 Results

First, we computed the inter-rater agreement using the Fleiss' Kappa [11] statistical measure to get a score of how much homogeneity, or consensus, there is in the ratings given by judges. We evaluated the agreement on two different levels in terms of granularity. On the class level we expect to have an agreement if the evaluators selected at least one class expression to be useful as definition for the corresponding class. Here we got a Fleiss' Kappa value of 0.51 which can be interpreted as "moderate agreement" [17]. On the more fine-grained class expression level, we assume to have an agreement if the same class expression was selected as an appropriate class definition. The Fleiss' Kappa value was approximately 0.28 which can be seen as a "fair agreement". For the 288 classes processed by DL-Learner on average 51 class expressions have been generated. The average length of the expressions, which is defined in a straightforward way, namely as

[8] `https://lucene.apache.org/solr/`, version 4.1.0

#	class expression	F-score	PMI	sPMI
1	Person $\sqcap \exists$ selection.$\top$	0.977	0.662	0.529
3	Person $\sqcap \exists$ selection.$\top$ $\sqcap \exists$ birthPlace.PopulatedPlace	0.960	0.797	0.549
4	Person $\sqcap \exists$ selection.$\top$ $\sqcap \exists$ birthPlace.Place	0.960	0.716	0.518
5	Person $\sqcap \exists$ mission.SpaceMission	0.950	0.493	0.664
8	Person $\sqcap \exists$ selection.$\top$ $\sqcap \exists$ nationality.Country	0.947	0.707	0.498
12	Person $\sqcap \exists$ nationality.Country	0.937	0.697	0.489
13	Person $\sqcap \exists$ selection.$\top$ $\sqcap \exists$ occupation.PersonFunction	0.937	0.672	0.487
15	Person $\sqcap \exists$ timeInSpace.minute	0.933	0.771	0.571
17	Person $\sqcap \exists$ mission.SpaceMission $\sqcap \exists$ timeInSpace.minute	0.933	0.620	0.643
19	Person $\sqcap \exists$ selection.$\top$ $\sqcap \exists$ mission.SpaceMission	0.933	0.584	0.603
21	Person $\sqcap \exists$ mission.SpaceMission $\sqcap \exists$ birthPlace.Place	0.933	0.615	0.599
22	Person $\sqcap \exists$ selection.$\top$ $\sqcap \exists$ nationality.Country $\sqcap \exists$ birthPlace.Place	0.933	0.733	0.499
29	Person $\sqcap \exists$ selection.$\top$ $\sqcap \exists$ birthDate.date	0.923	0.553	0.466
30	Person $\sqcap \exists$ mission.SpaceMission $\sqcap \exists$ occupation.PersonFunction	0.923	0.571	0.568
31	Person $\sqcap \exists$ mission.SpaceMission $\sqcap \exists$ nationality.Country	0.923	0.605	0.579
41	Person $\sqcap \exists$ selection.$\top$ $\sqcap \exists$ birthPlace.PopulatedPlace $\sqcap \exists$ mission.SpaceMission	0.920	0.703	0.596
48	Person $\sqcap \exists$ selection.$\top$ $\sqcap \exists$ nationality.Country $\sqcap \exists$ occupation.PersonFunction	0.917	0.701	0.477

Table 1: Excerpt of the 50 class expressions that have been evaluated for the class `Astronaut`. The first column denotes the rank of the DL-Learner output without taking statistical measures into account. Only the class expressions have been shown in random order to the evaluators.

the sum of the numbers of concept, role, quantifier, and connective symbols occurring in the expression was ≈ 10.

Our experiments address the following research questions:

1.) Which relevance measures are particular suitable, and how should they be combined?
2.) Can a combination of statistical relevance measures improve the results of logical ontology learning?

In order to answer the first question, we cast this task as a supervised machine learning problem itself in which F-score and the presented relevance measures are features of a learned definition. A definition is then considered to be a positive example if an evaluator selected it in our experiment and negative otherwise. Since this leads to a skewed distribution with more negative examples, we applied random subsampling on the negative examples. This results in an equal distribution of 302 positive and negative examples. In a first step, we used these to obtain a suitable classifier. We ran different types of classifiers (see Table 2a), i.e., support vector machines, decision trees, rules and probabilistic classifiers, as implemented in the Weka toolkit[9], with their default settings and used 10-fold cross-validation. As a baseline, we used an optimal threshold for the F-Score, which was determined by the Weka threshold selector meta classifier. An interesting insight is that the inclusion of relevance measures indeed significantly improves the standard approach of computing F-Score on the underlying RDF data, which allows us to positively answer the first research question.

The C4.5 decision tree algorithm performed best, so we used it as a base for feature analysis. This analysis was performed by using standard wrappers for feature subset selection [16]. In this case, we could exhaustively run all combinations of features

[9] `http://www.cs.waikato.ac.nz/ml/weka/`

algorithm	accuracy	F-score	AUC
C4.5	77.5%	77.1%	79.6%
SVM	73.3%	74.6%	73.3%
Logistic Regression	72.8%	73.3%	79.9%
Conjunctive Rule	69.5%	67.9%	72.5%
Naive Bayes	64.1%	54.3%	75.6%
ELTL Baseline	59.4%	61.7%	63.8%

(a) Results of 10-fold cross-validation for different classifiers.

feature added	accuracy
T-test	61.3%
+ F-score	73.5%
+ LLR	77.3%
+ Jaccard	77.0%
+ PMI	78.0%
+ χ^2	78.1%
+ SCI	78.5%

(b) Accuracy gain for features using C4.5.

Table 2: Results of relevance measure analysis.

in C4.5 via 10-fold cross-validation and optimizing for predictive accuracy. The best performing feature subset is {F-score, PMI, χ^2, Jaccard, LLR, SCI, T-test}. We used this subset and iteratively removed the feature which caused the least loss in predictive accuracy. This allows us to observe the increase in accuracy obtained by adding features as shown in Table 2b:

The first three features led to significant improvements in the ability to detect promising definitions whereas the other features showed only small contributions (even negative in one case). We also analyzed the weights of normalized features in the SVM classifier:

$$\begin{array}{llll} 3.6477 \times \text{F-score} & -2.2027 \times \text{PMI} & -0.1476 \times \chi^2 & \\ +0.7601 \times \text{Dice} & +0.8325 \times \text{Jaccard} & -0.4517 \times \text{LLR} & \\ +0.2963 \times \text{SCI} & +3.7772 \times \text{sPMI} & +1.1387 \times \text{T-test} & -4.5916 \end{array}$$

It is notable here that PMI indeed has a negative weight. We believe this is due to high PMI values for low frequency entity pairs, so the negative weight along with the high positive weight of sPMI essentially acts as a noise filter. χ^2 and LLR also have negative weights, which might be related to their null addition property. We also noted that some metrics have very low values close to zero in the majority of cases and are essentially only used a tie breaker in the SVM classifier whereas the C4.5 decision tree can make better use of those values.

5.3 Discussion

During the manual annotation of the created class expressions, the annotators did not find suitable definitions for the classes in a number of cases. Based on the comments provided by the annotators and manual inspection of the affected classes, we were able to find patterns helping to categorize the problems. In the following, we describe the categories and give examples for each.

Limited Ontology Vocabulary This problem arises due to relying on the classes and properties defined in the DBpedia ontology. In these cases, the ontology does not provide any entities which could be used to describe the class both accurately and exhaustively. For example, the expressions generated for describing the class `Bodybuilder` contained properties as `height` together with the class `Athlete`. Obviously, this is a correct description of a bodybuilder but it also matches all other athletes since the "restrictions" are actually properties of the parent class. However, DL-Learner was unable to choose a better definition since the ontology did not contain single properties or combinations of these specifically able to describe a body builder. This type of problems could only be solved by manually adding more specific properties or classes.

Limited Usage of Vocabulary A related problem arose when the ontology contained an entity usable to describe a class fully which was not created by DL-Learner. For instance, when describing the concept `CanadianFootballLeague`, DL-Learner created definitions like $\texttt{SportsLeague} \sqcap \exists \texttt{team.SportsTeam}$ that describes a sports league but is not specific enough to exclude sport leagues other than Canadian football league. Replacing `SportsTeam` in the definition by `CanadianFootballTeam` would lead to a flawless definition but is not proposed by DL-Learner. This is because the positive examples do not contain a significant number of assignments of teams to a Canadian football league that are also asserted to be Canadian football teams.[10] Again, this problem is hardly solvable when learning expressions but only at the data level.

Superfluous Restrictions Some class expressions were also not chosen by the annotators because they contained superfluous restrictions. This is most often the case when defining subclasses of `Person` like `Writer` and including restrictions on, e.g., `birthPlace`. Clearly, this is not a restricting property for writers all being persons. Thus, some of these definitions were not chosen by the annotators who tried to choose the definitions as compact as possible. Most probably, this problem is also caused by the missing vocabulary to describe some classes suitably. We could try to prevent the generation of such definitions by considering the domain and ranges of properties and filtering restrictions if properties are defined for super classes of the currently considered class. However, then we would depend more strongly on the correctness of the schema whose quality showed to be doubtful in many cases throughout our experiments.

Another interesting example is the definition of an `Architect`, which uses the property `significantBuilding` though from the word meaning this would not be a definition covering all architects but only the more renowned ones which not only have regular buildings but also significant ones. DBpedia, as it is derived from Wikipedia, contains only few data on architects which are not famous for their buildings. A different but less general problem was discovered for classes belonging to the biological taxonomy in DBpedia. Here, some generated wrong definitions pointed to flaws in the usage of biology-specific properties like `kingdom`.

In summary, we discovered a combination of measures for generating more intuitive class expressions. From the inclusion of textual information, we were able to complement the purely logical information employed by DL-Learner with additional knowledge about how related specific properties are evaluated by humans. Most problems detected during the manual annotation can be traced back to missing or underspecified input data.

6 CONCLUSION

In this paper, we presented first steps towards combining the previously distinct logical and lexical ontology learning areas. By extending a formerly pure logic based approach

[10] Only 2 of 10 teams are assigned to be a Canadian football team.

with statistical methods which can be used on text corpora, we were able to foster the generation of more intuitive class expressions. An extensive manual evaluation with four human annotators showed that the integration of relevance measures can significantly improve results. Nevertheless, we also discovered and analyzed several problems for which we were partially able to trace them back to data quality issues.

In the near future work, we plan to closely integrate the output of the lexical analysis into the refinement process. This might positively influence the search in the hypotheses space, thus, might result in a faster generation of more intuitive solutions first. Additionally, we are going to extend our approach by more sophisticated word sense disambiguation techniques, which will help us to more accurately identify mentions of ontology entities in the text. We will also include WordNet and other lexical resources that can facilitate the detection of words which are synonymous to ontology entity labels. Furthermore, we are going to apply our novel combined logical and statistical approach on more datasets to examine its performance in other domains and use cases.

Acknowledgment: This research was partially funded by the DFG project GOLD (*Generating Ontologies from Linked Data*). Johanna Völker is supported by a Margarete-von-Wrangell scholarship of the European Social Fund (ESF) and the Ministry of Science, Research and the Arts Baden-Württemberg.

References

1. Franz Baader, Diego Calvanese, Deborah McGuinness, Daniel Nardi, and Peter Patel-Schneider, editors. *The Description Logic Handbook.* Cambridge University Press, 2003.
2. Liviu Badea and Shan hwei Nienhuys-Cheng. A refinement operator for description logics. In *Proc. of the Int. Conf. on Inductive Logic Programming (ILP)*, volume 1866 of *LNAI*, pages 40–59. Springer, 2000.
3. Lorenz Bühmann and Jens Lehmann. Pattern based knowledge base enrichment. In *12th International Semantic Web Conference, 21-25 October 2013, Sydney, Australia*, 2013.
4. Dipak Chaudhari, Om P. Damani, and Srivatsan Laxman. Lexical co-occurrence, statistical significance, and word association. In *EMNLP*, pages 1058–1068. ACL, 2011.
5. William W. Cohen and Haym Hirsh. Learnability of description logics. In *Proceedings of the Fourth Annual Workshop on Computational Learning Theory*. ACM Press, 1992.
6. William W. Cohen and Haym Hirsh. Learning the CLASSIC description logic. In *Proc. of the Int. Conf. on Principles of Knowledge Representation and Reasoning*, pages 121–133. Morgan Kaufmann, 1994.
7. Om P. Damani. Improving pointwise mutual information (pmi) by incorporating significant co-occurrence. *CoRR*, abs/1307.0596, 2013.
8. Nicola Fanizzi, Claudia d'Amato, and Floriana Esposito. DL-FOIL: Concept learning in description logics. In F. Zelezný and N. Lavrac, editors, *Proc. of the 18th Int. Conf. on Inductive Logic Programming (ILP)*, volume 5194 of *LNAI*, pages 107–121. Springer, 2008.
9. Nicola Fanizzi, Claudia d'Amato, and Floriana Esposito. Induction of concepts in web ontologies through terminological decision trees. In José L. Balcázar et al., editors, *Proceedings of ECML PKDD 2010, Part I*, volume 6321 of *LNCS/LNAI*, pages 442–457. Springer, 2010.
10. J.R. Firth. A synopsis of linguistic theory 1930-1955. *Studies in linguistic analysis*, pages 1–32, 1957.
11. J.L. Fleiss et al. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382, 1971.

12. Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA data mining software: an update. *SIGKDD Explorations*, 11(1):10–18, 2009.
13. Marti A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *COLING*, pages 539–545, 1992.
14. Sebastian Hellmann, Jens Lehmann, and Sören Auer. Learning of OWL class descriptions on very large knowledge bases. *International Journal on Semantic Web and Information Systems*, 5(2):25–48, 2009.
15. Luigi Iannone, Ignazio Palmisano, and Nicola Fanizzi. An algorithm based on counterfactuals for concept learning in the semantic web. *Applied Intelligence*, 26(2):139–159, 2007.
16. Ron Kohavi and George H John. Wrappers for feature subset selection. *Artificial intelligence*, 97(1):273–324, 1997.
17. J. Richard Landis and Gary G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):pp. 159–174, 1977.
18. Jens Lehmann. Hybrid learning of ontology classes. In *Proc. of the 5th Int. Conference on Machine Learning and Data Mining (MLDM)*, volume 4571 of *LNCS*, pages 883–898. Springer, 2007.
19. Jens Lehmann. DL-Learner: learning concepts in description logics. *Journal of Machine Learning Research (JMLR)*, 10:2639–2642, 2009.
20. Jens Lehmann, Sören Auer, Lorenz Bühmann, and Sebastian Tramp. Class expression learning for ontology engineering. *Journal of Web Semantics*, 9:71 – 81, 2011.
21. Jens Lehmann and Christoph Haase. Ideal downward refinement in the EL description logic. In *Proc. of the Int. Conf. on Inductive Logic Programming*, volume 5989 of *LNCS*, pages 73–87. Springer, 2009.
22. Jens Lehmann and Pascal Hitzler. Concept learning in description logics using refinement operators. *Machine Learning journal*, 78(1-2):203–250, 2010.
23. Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. DBpedia – A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web Journal*, 2014.
24. Jens Lehmann and Johanna Völker, editors. *Perspectives on Ontology Learning*. Studies on the Semantic Web. AKA Heidelberg, 2014.
25. Alexander Maedche and Steffen Staab. Ontology learning for the semantic web. *IEEE Intelligent systems*, 16(2):72–79, 2001.
26. Tom M. Mitchell. Generalization as search. *Artificial Intelligence*, 18(2):203–226, 1982.
27. Shan-Hwei Nienhuys-Cheng and Ronald De Wolf. *Foundations of inductive logic programming*, volume 1228. Springer, 1997.
28. Valentina Presutti, Francesco Draicchio, and Aldo Gangemi. Knowledge extraction based on discourse representation theory and linguistic frames. In *Knowledge Engineering and Knowledge Management*, volume 7603 of *LNCS*, pages 114–129. Springer, 2012.
29. J. Ross Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239–266, 1990.
30. Johanna Völker, Pascal Hitzler, and Philipp Cimiano. Acquisition of OWL DL axioms from lexical resources. In *ESWC*, pages 670–685, 2007.
31. Johanna Völker and Mathias Niepert. Statistical schema induction. In *Proc. of the Extended Semantic Web Conference (ESWC)*, pages 124–138, 2011.
32. Wilson Wong, Wei Liu, and Mohammed Bennamoun. Ontology learning from text: A look back and into the future. *ACM Comput. Surv.*, 44(4):20, 2012.