A Formalization of Ontology Learning From Text

Michael Sintek[1], Paul Buitelaar[2], and Daniel Olejnik[2]

[1] DFKI GmbH
Knowledge Management Department
Erwin-Schrödinger-Str., Bldg. 57
D-67663 Kaiserslautern, Germany
sintek@dfki.uni-kl.de
[2] DFKI GmbH
Language Technology Department
Stuhlsatzenhausweg 3
D-66123 Saarbrücken, Germany
{paulb,olejnik}@dfki.de

Abstract. Recent developments towards knowledge-based applications in general and Semantic Web applications in particular are leading to an increased interest in ontologies and in dynamic methods for developing and maintaining them. As human language is a primary mode of knowledge transfer, ontology learning from relevant text collections has been among the most successful strategies in this work. Such methods mostly combine a certain level of linguistic analysis with statistical and/or machine learning approaches to find potentially interesting concepts and relations between them. Here, we discuss a formalization of this process (in the specific context of the OntoLT tool for ontology learning from text) in order to arrive at a better definition of this task, which we hope to be of use in a more principled comparison of different approaches. As ontology representation formalisms we will consider those that have a model-theoretic semantics, with OWL (and subsets of OWL) being appropriate candidates.

Keywords: ontologies, ontology learning from text, linguistic analysis, OWL, formalization, ontology entailment, model-theoretic semantics

1 Motivation

Recent developments towards knowledge-based applications such as Intelligent Question-Answering, Semantic Web Services and Semantic-Level Multimedia Search are also leading to an increased interest in ontologies. Additionally, as ontologies are domain descriptions that tend to evolve rapidly over time and between different applications, there has been an increased interest also towards developing and maintaining ontologies dynamically (see also [1]).

As human language is a primary mode of knowledge transfer, ontology learning from relevant text collections has been among the most successful strategies

in this work. See, for instance, the overview of ontology learning systems and approaches as discussed by the OntoWeb deliverable 1.5 [2]. Some recent examples of systems for ontology learning from text are: ASIUM [3], TextToOnto [4], Ontolearn [5], OntoLT [6]. All of these combine a certain level of linguistic analysis with statistical and/or machine learning approaches to find potentially interesting concepts and relations between them.

In order to allow a principled comparison of these approaches and to define evaluation environments, we propose a formalization of ontology (and knowledge base) learning from text. Since the systems we want to concentrate on mainly use ontology representation languages that are frame systems or (subsets of) description logics like OWL [7], we developed a formalization approach which is suited for this class of ontology languages, namely those that have a model-theoretic semantics which we use as a basis for operations on ontologies. The OntoLT [6] tool for ontology learning from text will be used to clarify the various details of this formalization.

2 Ontology Learning From Text

A typical approach in ontology learning from text first involves term extraction from a domain-specific corpus through a statistical process that determines their relevance for the domain corpus at hand. These are then clustered into groups with the purpose of identifying a taxonomy of potential classes. Subsequently, relations can be identified by computing a statistical measure of "connectedness" between identified clusters.

In the context of this paper we assume a similar approach, but we additionally aim at a more direct connection between ontology learning and linguistic analysis. Through such an approach, relations may be identified additionally on the basis of linguistic analysis of the "dependency structure" between terms and connecting or modifying words (i.e., verbs, prepositions, adjectives) in their context.

2.1 OntoLT

The OntoLT plug-in for Protégé implements this approach through the definition of mapping rules with which classes and properties can be extracted automatically from linguistically annotated text collections. Through the use of such rules, linguistic knowledge (context words, morphological and syntactic structure, etc.) remains associated with the constructed ontology and may be used subsequently in its application and maintenance, e.g., in knowledge markup, ontology mapping, and ontology evolution.

The ontology extraction process is implemented as follows. OntoLT provides a precondition language, with which the user can define mapping rules. Preconditions are implemented as XPath [8] expressions over XML-based linguistic annotation of relevant text collections (see Section 2.2 below). If all constraints

are satisfied, the mapping rule activates one or more operators that describe in which way the ontology should be extended if a candidate is found.

Predefined preconditions select for instance the predicate of a sentence, its linguistic subject or direct object. Preconditions can also be used to check certain conditions on these linguistic entities, for instance if the subject in a sentence corresponds to a particular lemma (the morphological stem of a word).

Selected linguistic entities may be used in constructing or extending an ontology. For this purpose, OntoLT provides operators to create classes, slots and instances. According to which preconditions are satisfied, corresponding operators will be activated to create a set of candidate classes and slots that are to be validated by the user (see Fig. 1). Validated candidates are then integrated into a new or existing ontology.

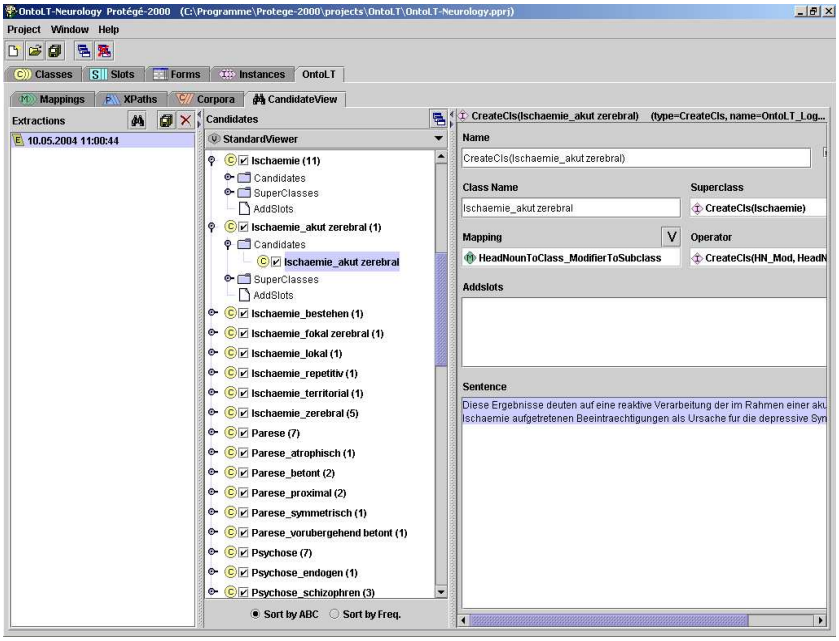

Fig. 1. OntoLT Screenshot

For example, OntoLT includes the following two mapping rules:

- HeadNounToClass_ModToSubClass: maps a head-noun to a class and in combination with its modifier(s) to one or more sub-class(es)
- SubjToClass_PredToSlot_DObjToRange: maps a linguistic subject to a class, its predicate to a corresponding slot for this class and the direct object to the range of this slot

Consider the following sentence to which these rules can be applied:

This disease is characterized primarily by impaired mental function caused by damage to the brain.

Result for the HeadNounToClass_ModToSubClass rule:
Class('impaired mental function') subClassOf Class('function')
Result for the SubjToClass_PredToSlot_DObjToRange rule:
Property('characterize') domain Class('function') range Class('disease')

2.2 Linguistic Analysis and Annotation

We consider linguistically annotated text corpora, using an XML-based annotation format, which integrates multiple levels of linguistic and semantic analysis in a multi-layered DTD with each analysis level (e.g., morphological, syntactic and dependency structure) organized as a separate track with options of reference between them via indices.

Linguistic annotation is currently provided by SCHUG, a rule-based system for German and English analysis [9] that implements a cascade of increasingly complex linguistic fragment recognition processes. SCHUG provides annotation of part-of-speech (through integration of TnT [10]), morphological inflection and decomposition (based on Mmorph [11]), phrase and dependency structure (head-complement, head-modifier and grammatical functions).

In Fig. 2, we present a tree representation of the linguistic annotation for (part of) the following sentence (German with translation in English):

An 40 Kniegelenkpräparaten wurden mittlere Patellarsehnendrittel mit einer neuen Knochenverblockungstechnik in einem zweistufigen Bohrkanal bzw. mit konventioneller Interferenzschraubentechnik femoral fixiert.

(In 40 human cadaver knees, either a mid patellar ligament third with a trapezoid bone block on one side was fixed on the femoral side in a 2-diameter drill hole, or a conventional interference screw fixation was applied.)

The linguistic annotation for this sentence consists of part-of-speech and lemmatization information in the `<text>` level, phrase structure (including head-modifier analysis) in the `<phrases>` level and grammatical function analysis in the `<clauses>` level (in this sentence there is only one clause, but more than one clause per sentence is possible).

Part-of-speech information consists of the correct syntactic class (e.g., noun, verb) for a particular word given its current context. For instance, the word *works* will be either a verb (working the whole day) or a noun (all his works have been sold).

Morphological information consists of inflectional, derivational or compound information of a word. In many languages other than English the morphological system is very rich and enables the construction of semantically complex compound words. For instance the German word *Kreuzbandverletzung* corresponds in

English with three words: *cruciate ligament injury*. Phrase structure information consists of an analysis of the syntactic structure of a sentence into constituents that are headed by an adjective, a noun or a preposition. Additionally, the internal structure of the phrase will be analyzed and represented, which includes information on modifiers that further specify the head. For instance, in the nominal phrase *neue Technik* (*new technology*) the modifier *neu* further specifies the head *Technik*.

Clause structure information consists of an analysis of the core semantic units (clauses) in a sentence with each clause consisting of a predicate (mostly a verb) with its arguments and adjuncts. Arguments are expressed by grammatical functions such as the subject or direct object of a verb. Adjuncts are mostly prepositional phrases, which further specify the clause. For instance, in *John played football in the garden*, the prepositional phrase in the garden further specifies the clause "play (John, football)."

All such information is provided by the annotation format that is illustrated in Fig. 2. For instance, the direct object (`DOBJ`) in the sentence above covers the nominal phrase `p2`, which in turn corresponds to tokens `t5` to `t10` (*mittlere Patellarsehnendrittel mit einer neuen Knochenverblockungstechnik*). As token `t6` is a German compound word, a morphological analysis is included that corresponds to lemmas `t6.l1`, `t6.l2`, `t6.l3`.

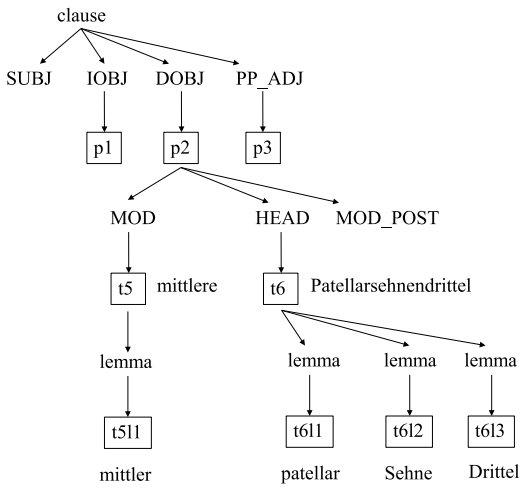

Fig. 2. Linguistic Annotation Example

3 Formalizing Ontology Learning From Text

In this section, we will describe our general approach for formalizing ontology learning from text and instantiate this for the OntoLT system. The general

approach and the OntoLT system do not only apply to ontologies, but also to knowledge bases (i.e., ontologies together with instances), but we will mainly concentrate on ontology learning in the following.

For describing the problem of ontology learning from text, we consider a suggestion function σ which maps a text corpus C, an ontology O, and (background) knowledge K to suggestions S:

$$\sigma : \ C \times O \times K \ \rightarrow \ S$$

The background knowledge will often contain mapping specifications that map a part of the corpus (e.g., one sentence) to some suggestions. It might also contain results of statistical analyses of the text corpus (like the distribution of nouns), thesaurus information (like WordNet), etc.

The suggestions will usually be a (possibly weighted) set of operations changing the ontology. They will be presented to the user who can select, possibly correct, and then apply them to the old (possibly empty) ontology. This process can be repeated as new documents come in, or as new background knowledge has been defined.

3.1 Operations on Ontologies

In order to specify suggestions, we will consider a minimal set of operations on ontologies, namely + and −:

$$+ : \ O \times O \ \rightarrow O$$

$$- : \ O \times O \ \rightarrow O$$

Their semantics will be defined without referring to concrete ontology languages, which allows our formalization to be applied to a wide range of systems. The only thing we require is that the ontology language has a model-theoretic semantics. This approach is radically different from approaches like ontology algebrae (e.g., [12]) which seem not to be applicable to such a wide range of ontology languages as our approach.

We define the semantics of + and − with the help of ontology *entailment*. An ontology O_1 is said to *entail* an ontology O_2, written $O_1 \models O_2$, if every model for O_1 is also a model for O_2 (cf. [13]).

Two ontologies O_1 and O_2 are said to be semantically equal ($O_1 \doteq O_2$) iff $O_1 \models O_2$ and $O_2 \models O_1$.

An ontology O is said to be a *most general* ontology for a condition C if O fulfills C and there exists no other ontology $O' \not\doteq O$ which fulfills C and for which $O \models O'$ holds. Note that, in general, more than one most general ontology for a condition exists. A *least general* ontology for a given condition is specified analogously.

We now define + and − as follows:

$O_1 + O_2$ is a most general ontology O with $O \models O_1 \ \wedge O \models O_2$.
$O_1 - O_2$ is a least general ontology O with $O_1 \models O \ \wedge O \not\models O_2$.

Note that, in general, the result of $O_1 + O_2$ and $O_1 - O_2$ is not well-defined, depending on the choice of the ontology language. Furthermore, + and − are not symmetric: + adds *all* of O_2 to O_1, while − removes only a minimal portion of O_2 from O_1. While + is usually well-defined since most ontology languages allow the statements that are used to define ontologies to be joined in some simple way, − is usually not well-defined, thus leaving several choices to the user. This does, however, not cause any problems in our scenario since the user has to interact with the suggestions anyway.

3.2 Ontology Language

As ontology language, we consider in OntoLT a simple subset of standard description logics where we only allow "subclass of" axioms $C \sqsubseteq CE$ where C is a class name and CE a class expression that uses only class names, intersection ($\sqcap$), and range restrictions ($\forall P.C$). $\top$ is the superclass of all classes.

Because of $\{C \sqsubseteq C_1 \sqcap C_2\} \doteq \{C \sqsubseteq C_1,\ C \sqsubseteq C_2\}$ we will in the following only consider axioms without intersection, i.e., we allow only axioms of the form $C_1 \sqsubseteq C_2$ and $C_1 \sqsubseteq \forall P.C_2$, where C_i are class names and P is a property name.

This results in a description logics that is the subset of OWL Lite [7] with the expressiveness of frame systems like RDF Schema [14] and Protégé [15]

In the OntoLT system, we also take instances into account. The extensions necessary for this are straight-forward and will not be further described here.

Note that the proposed formalization allows the use of any other ontology (or knowledge-base) language as long as it has a model-theoretic semantics,

3.3 Suggestions

For suggestions, we consider sequences of ontology operations, written as $\pm\{a_1, a_2, \ldots\}, \{b_1, b_2, \ldots\}, \ldots$ where $\pm$ is either + or − and a_i and b_i are axioms as described above. For $\pm\{a\}$, we also write $\pm a$.

The current implementation of OntoLT only handles suggestions for ontology *extensions*, so we only need the + operation. For ontology languages based on description logics, where an ontology is a set of axioms, + directly maps to set union $\cup$:

$$O_1 + O_2 \ \doteq O_1 \cup O_2$$

Note that even in the case of our simple ontology language, the − operator is not well-defined. Nevertheless, this would not be a big problem in a concrete tool, as the user could then be presented with several choices of how to remove a certain axiom from the ontology, as he will have to choose which suggestions to apply anyway.

3.4 Mapping Rules

OntoLT uses a set of mapping rules $R(\Sigma)$ that maps a *single* sentence (which is the parameter Σ for the rule set) from the text corpus to a set of suggestions. These rules are of the following form:

$$P \mapsto S$$

P is the precondition and expressed as a formula in FOL. S is a sequence of ontology operations as defined above, or, to allow a shorthand for alternatives, an expression of the form $S_1 \mid S_2 \mid \ldots$ where S_i is a sequence of operations. S will usually share some variables with P; these variables must be introduced with a common *forall* quantifier:

$$\forall V_1, \ldots, V_n \; P(V_1, \ldots, V_n) \mapsto S(V_1, \ldots, V_n)$$

We also allow the specification of Horn rules [16] (with FOL syntax) $H \leftarrow B$ that can be accessed in the precondition.

The predicate $\text{xpath}(X, E, V)$ applies an XPath [8] expression E to an XML fragment X and enumerates the results in V. This predicate is used to extract information from the corpus (by applying it on Σ).

The preconditions of the mapping rules and the bodies of the Horn rules may also access the ontology. For this, DL axioms ($C_1 \sqsubseteq C_2$, $C_1 \sqsubseteq \forall P.C_2$) can be used as literals.

In OntoLT, we also allow access to thesauri, in particular WordNet, via additional builtin predicates.

3.5 Enactment

For the enactment of rules, all mapping rules are transformed into Horn rules, as shown in the following table:

$P \mapsto S$	$\text{sugg}(S') \leftarrow P$
$P \mapsto S_1 \mid S_2 \mid \ldots$	$\text{sugg}(S'_1) \leftarrow P$ $\text{sugg}(S'_2) \leftarrow P$ $\ldots$

Here, S' is a representation of an ontology operation sequence S with FOL function symbols. Quantifiers and other syntactic constructs which are not allowed in normal Horn rules are removed with the Lloyd-Topor transformation [17], just as has been done in SiLRI [18] and TRIPLE [19], allowing the resulting rules to be enacted by a standard PROLOG engine (with some additional builtins).

A set of mapping rules $R(\Sigma)$ can now be enacted for a sentence s by executing the query $\forall S \leftarrow \text{sugg}(S)$ for the set of Horn rules obtained from the above transformation (and after replacing Σ with s in the rule set).

To evaluate the mapping rules for the whole text corpus C, we simply have to evaluate $R(\Sigma)$ for each sentence s from the text corpus, thus obtaining a set of suggestions that will be presented to the user.

3.6 Example

The following example are the rules for turning the subject of a sentence to a class and the predicate to a slot with the direct object as range:

$$
\begin{aligned}
&\forall S\ \text{subject}(S) \leftarrow \\
&\quad \text{xpath}(\Sigma, \text{"}\texttt{.//phrase[@id=...[@type="SUBJ"]/@phrase]/head}\text{"}, S) \\
&\quad \wedge\ \ldots\ . \\
&\forall P\ \text{predicate}(P) \leftarrow \\
&\quad \text{xpath}(\Sigma, \text{"}\texttt{.//phrase[@id=./../..//clause/@pred]}\text{"}, P) \\
&\quad \wedge\ \ldots\ . \\
&\forall O\ \text{directObject}(O) \leftarrow \\
&\quad \text{xpath}(\Sigma, \text{"}\texttt{.//phrase[@id=...[@type="DOBJ"]/@phrase]/head}\text{"}, O) \\
&\quad \wedge\ \ldots\ . \\
&\forall S, P, O\ \text{subject}(S) \wedge \text{predicate}(P) \wedge \text{directObject}(O) \mapsto \\
&\quad +\{S \sqsubseteq \top, O \sqsubseteq \top, S \sqsubseteq \forall P.O\}.
\end{aligned}
$$

The first three rules simply define how to find subjects, predicates, and direct objects in sentences with the help of some XPath expressions. The fourth rule generates the suggestion that S and O should become classes ($S \sqsubseteq \top, O \sqsubseteq \top$) with P a property on S with range O ($S \sqsubseteq \forall P.O\}$).

4 Conclusions

Automatic methods for text-based ontology learning have developed over recent years, see, e.g., the proceedings of the ECAI-2000[3], IJCAI-2001[4] and ECAI-2002[5] workshops on Ontology Learning. Still, a remaining challenge is to evaluate how useful or accurate the extracted ontologies are. In fact, we believe this to be a central issue as it is currently very hard to compare methods and approaches, due to the lack of a shared and formal understanding of the task at hand. By the work described in this paper, we hope to have contributed to such a shared understanding by providing a formal definition of the OntoLT approach to ontology learning from text. We believe that such a formal definition will allow for a better comparison with similar approaches, not so much on the level of specific methods, but rather on the level of preconditions, inputs, and results.

5 Acknowledgements

This research has in part been supported by EC grants IST-2000-29243 for the OntoWeb project and IST-2000-25045 for the MEMPHIS project, and by bmb+f grant 01 IMD01 A for the SmartWeb project.

[3] `http://ol2000.aifb.uni-karlsruhe.de/`
[4] `http://ol2001.aifb.uni-karlsruhe.de/home.html`
[5] `http://www-sop.inria.fr/acacia/WORKSHOPS/ECAI2002-OLT/`

References

1. Maedche, A.: Ontology Learning for the Semantic Web. Volume 665 of International Series in Engineering and Computer Science. Kluwer (2003)
2. Gomez-Perez, A., Manzano-Macho, D.: A survey of ontology learning methods and techniques. Technical Report Deliverable 1.5, OntoWeb Project (2003) Available from `http://ontoweb.aifb.uni-karlsruhe.de/`.
3. Faure, D., Nédellec, C., Rouveirol, C.: Acquisition of semantic knowledge using machine learning methods: The system ASIUM. Technical Report ICS-TR-88-16 (1998)
4. Maedche, A., Staab, S.: Semi-automatic engineering of ontologies from text. In: Proceedings of the 12th International Conference on Software Engineering and Knowledge Engineering. (2000)
5. Navigli, R., Velardi, P., Gangemi, A.: Ontology learning and its application to automated terminology translation. IEEE Intelligent Systems (2003)
6. Buitelaar, P., Olejnik, D., Sintek, M.: A Protégé plug-in for ontology extraction from text based on linguistic analysis. In: Proceedings of the European Semantic Web Symposium (ESWS). (2004)
7. Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinnes, D., Patel-Schneider, P., Stein, L.: OWL web ontology language reference (2004) `http://www.w3.org/TR/owl-ref/`.
8. Clark, J., DeRose, S.: XML path language (XPath) (1999) `http://www.w3.org/TR/xpath`.
9. Declerck, T.: A set of tools for integrating linguistic and non-linguistic information. In: Proceedings of the SAAKM workshop at ECAI. (2002)
10. Brants, T.: TnT — a statistical part-of-speech tagger. In: Proceedings of the 6th ANLP Conference. (2000)
11. Petitpierre, D., Russell, G.: MMORPH — the Multext morphology program. Technical Report Deliverable for task 2.3.1, Multext Project (1995) Available from ISSCO, University of Geneva.
12. Mitra, P., Wiederhold, G.: An ontology-composition algebra. Springer Series: International Handbooks on Information Systems (2004) 93–113
13. Horrocks, I., Patel-Schneider, P.: Reducing OWL entailment to description logic satisfiability. In: Proceedings of the International Semantic Web Conference. (2003)
14. Brickley, D., Guha, R.: RDF vocabulary description language 1.0: RDF Schema (2004) `http://www.w3.org/TR/rdf-schema/`.
15. Knublauch, H.: An AI tool for the real world: Knowledge modeling with Protégé. JavaWorld (2003) Protégé is available from `http://protege.stanford.edu/`.
16. Lloyd, J.: Foundations of logic programming. Springer-Verlag New York, Inc. (1984)
17. Lloyd, J., Topor, R.: Making Prolog More Expressive. Journal of Logic Programming **3** (1984) 225–240
18. Decker, S., Brickley, D., Saarela, J., Angele, J.: A query and inference service for RDF. In: QL'98 — The Query Languages Workshop, Boston, USA, World-WideWeb Consortium (W3C) (1998)
19. Sintek, M., Decker, S.: TRIPLE—A query, inference, and transformation language for the Semantic Web. In: 1st International Semantic Web Conference (ISWC2002). (2002)