Ontology Learning and Reasoning — Dealing with Uncertainty and Inconsistency

Peter Haase, Johanna Völker

Institute AIFB, University of Karlsruhe, Germany
{pha,jvo}@aifb.uni-karlsruhe.de

Abstract. Ontology Learning from text aims at generating domain ontologies from textual resources by applying natural language processing and machine learning techniques. It is inherent in the ontology learning process that the acquired ontologies represent uncertain and possibly contradicting knowledge. From a logical perspective, the learned ontologies are potentially inconsistent knowledge bases that thus do not allow meaningful reasoning directly. In this paper we present an approach to generate consistent OWL ontologies from learned ontology models by taking the uncertainty of the knowledge into account. We further present evaluation results from experiments with ontologies learned from a Digital Library.

1 Introduction

Ontology Learning from text aims at generating domain ontologies from a given collection of textual resources by applying natural language processing and machine learning techniques. Due to an increasing demand for efficient support in knowledge acquisition, a number of tools for automatic or semi-automatic ontology learning have been developed during the last years. Common to all of them is the need for handling the uncertainty which is inherent in any kind of knowledge acquisition process. Moreover, ontology-based applications which rely on learned ontologies have to face the challenge of reasoning with large amounts of imperfect information resulting from automatic ontology generation systems.

Causes for the imperfection of information can be found thrice. According to [1] imperfection can be due to *imprecision*, *inconsistency* or *uncertainty*. Imprecision and inconsistency are properties of the information itself - either more than one world (in the case of ambiguous, vague or approximate information) or no world (if contradictory conclusions can be derived from the information) is compatible with the given information. Uncertainty means that an agent, i.e. a computer or a human, has only partial knowledge about the truth value of a given piece of information. One can distinguish between objective and subjective uncertainty. Whereas objective uncertainty relates to randomness referring to the propensity or disposition of something to be true, subjective uncertainty depends on an agent's opinion about the truth value of information. In particular, the agent can consider information as unreliable or irrelevant.

In ontology learning, (subjective) uncertainty is the most prominent form of imperfection. This is due to the fact that the results of the different algorithms have to be

considered as unreliable or irrelevant due to imprecision and errors introduced during the ontology generation process. There exist different approaches for the representation of uncertainty: Uncertainty can for example be represented as part of the learned ontologies, e.g. using probabilistic extensions to the target knowledge representation formalism, or at a meta-level as application-specific information associated with the learned structures.

In Text2Onto [7], a framework for ontology learning and data-driven ontology evolution, we follow a slightly different approach: In a first step, we apply ontology learning algorithms to generate ontologies based on a *Learned Ontology Model* (LOM), which is independent of a concrete ontology representation language. In the LOM, we represent uncertainty as annotations capturing the confidence about the correctness of the ontology elements. Most importantly, since the LOM does not have any logical semantics, in this step we do not have to consider logical inconsistencies which are often introduced during the ontology learning process. In a second step, we transform the LOM model to a standard logic-based ontology language, in order to be able to apply standard reasoning over the learned ontologies (e.g. for query answering). In our work we build on the OWL ontology language, as it is now the standard for representing ontologies on the web, and – with its grounding in Description Logics – reasoning with OWL ontologies is very well understood and tractable. Because of the uncertain and thus potentially contradicting information in the LOM models, a naive translation of the LOM model to OWL would result in highly inconsistent ontologies, which do not allow meaningful reasoning. We therefore make use of the confidence annotations of the LOM to guide the transformation process.

An obvious alternative approach to dealing with potential inconsistencies is to prohibit primitives that introduce inconsistencies in the first place (e.g. negation, disjointness). However, as shown in [21], semantically rich primitives such as disjointness of concepts can be used for effective semantic clarification in ontologies and thus enables to draw more meaningful conclusions.

As a main contribution of this work we present a transformation that results in an ontology that is (1) consistent and (2) "most likely correct", relying on the certainty information of the LOM model. The transformation is based on the notion of an evaluation function that measures the quality of ontologies with respect to given criteria, i.e. in our case consistency and certainty.

Application Scenario Intelligent search over document corpora in Digital Libraries is one application scenario that shows the immediate benefit of the ability to reason over ontologies automatically learned from text. While search in Digital Libraries nowadays is restricted to structured queries against the bibliographic metadata (author, title, etc.) and to unstructured keyword-based queries over the full text documents, complex queries that involve reasoning over the knowledge present in the documents are not possible. Ontology learning enables obtaining the required formal representations of the knowledge available in the corpus to be able to support such advanced types of search. This application scenario is the subject of a case study within the Digital Library of BT (British Telecom) as part of the SEKT[1] project. One of the key elements

[1] http://www.sekt-project.com/

of the case study is to automatically learn ontologies to enhance search and finally be able support queries of the kind "Find knowledge management applications that support Peer-to-Peer knowledge sharing." To validate the work the presented in this paper, we performed experiments with data from the BT Digital Library.

Overview of the paper The rest of the paper is organized as follows. In Section 2 we recapitulate the foundations of the OWL ontology language, query answering with OWL ontologies and the role of logical inconsistencies. In Section 3 we introduce the Learned Ontology Model (LOM). In Section 4 we discuss the transformation of LOM models to OWL ontologies. We discuss experimental results in Section 5 and present related work in Section 6 before we conclude in Section 7.

2 Reasoning with OWL

In this section we provide on overview of the OWL ontology language (specifically OWL-DL), typical reasoning tasks and show why standard reasoning with inconsistent ontologies does not yield meaningful results.

OWL-DL is a syntactic variant of the $\mathcal{SHOIN}(\mathbf{D})$ description logic [15]. Hence, although several syntaxes for OWL-DL exist, in this paper we use the traditional description logic notation since it is more compact.

Definition 1 (Ontology). *We use a datatype theory* $\mathbf{D}$*, a set of concept names* N_C*, sets of abstract and concrete individuals* N_{I_a} *and* N_{I_c}*, respectively, and sets of abstract and concrete role names* N_{R_a} *and* N_{R_c}*, respectively.*

The set of $\mathcal{SHOIN}(\mathbf{D})$ concepts *is defined by the following syntactic rules, where* A *is an atomic concept,* R *is an abstract role,* S *is an abstract simple role,* $T_{(i)}$ *are concrete roles,* d *is a concrete domain predicate,* a_i *and* c_i *are abstract and concrete individuals, respectively, and* n *is a non-negative integer:*

$$C \rightarrow A \mid \neg C \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2 \mid \exists R.C \mid \forall R.C \mid \geq n\, S \mid \leq n\, S \mid \{a_1, \ldots, a_n\} \mid \\ \mid \geq n\, T \mid \leq n\, T \mid \exists T_1, \ldots, T_n.D \mid \forall T_1, \ldots, T_n.D$$
$$D \rightarrow d \mid \{c_1, \ldots, c_n\}$$

A $\mathcal{SHOIN}(\mathbf{D})$ *ontology* O *is a finite set of axioms of the form concept inclusion axioms* $C \sqsubseteq D$*, for* C *and* D *concepts, transitivity axioms* $\mathsf{Trans}(R)$*, role inclusion axioms* $R \sqsubseteq S$ *and* $T \sqsubseteq U$*, concept assertions* $C(a)$*, role assertions* $R(a, b)$*, individual (in)equalities* $a \approx b$*, and* $a \not\approx b$*, respectively.*

The semantics of the $\mathcal{SHOIN}(\mathbf{D})$ description logic is defined via a model-theoretic semantics, which explicates the relationship between the language syntax and the model of a domain: An interpretation $I = (\triangle^I, \cdot^I)$ consists of a domain set $\triangle^I$, disjoint from the datatype domain $\triangle^I_{\mathbf{D}}$, and an interpretation function $\cdot^I$, which maps from individuals, concepts and roles to elements of the domain, subsets of the domain and binary relations on the domain, respectively[2]. An interpretation $\mathcal{I}$ satisfies an ontology O, if it

[2] For a complete definition of the interpretation, we refer the reader to [15].

satisfies each axiom in O. Axioms thus result in semantic conditions on the interpretations. Consequently, contradicting axioms will allow no possible interpretations. This leads us to the definition of a consistent ontology:

Definition 2 (Consistent Ontology). *An ontology O is consistent iff O is satisfiable, i.e. if O has a model.*

To be able to define queries against ontologies, we rely on the notion of entailment: We use $O \models \alpha$ to denote that the ontology O entails the axiom α (alternatively, we say that α is a consequence of the ontology O), iff α holds in any model in which O holds.

Definition 3 (Query and Query Answer). *A query with respect to an entailment relation $\models$ is a pair of an ontology O and an axiom α, written '$O \models \alpha?$'. An answer to a query '$O \models \alpha?$' is a value in the set $\{true, false\}$ as $O \models \alpha$ and $O \not\models \alpha$ respectively.*

Standard entailment as defined above is explosive, i.e. any axiom is a consequence of an inconsistent ontology. Namely, if an ontology O is not consistent, then for any axiom α, $O \models \alpha$. In other words, query answers for inconsistent ontologies are completely meaninginless, as for any query the query answer will be *true*. For a detailed discussion on inconsistencies in OWL ontologies, we refer the reader to [13].

3 LOM - A Learned Ontology Model

We believe, that linguistic evidence with respect to an ontology can be appropriately measured by ontology learning techniques which try to capture the ontological commitment in human language. Since ontology learning algorithms such as implemented in TextToOnto [7] consider the relation of individual ontology elements with the data the ontology has been engineered from, they allow to assess how well the ontology reflects the underlying corpus of data. This is especially relevant for an application scenario as introduced in Section 1, which involves question answering in the context of a Digital Library. In the following we describe the ontology model of Text2Onto and the ontology learning algorithms used in our approach.

A *Learned Ontology Model* (LOM) as used by Text2Onto is a collection of instantiated modeling primitives which are independent of a concrete ontology representation language. These primitives are defined in a declarative fashion which allows for translating the LOM into any knowledge representation language as long as the expressivity of the primitives does not exceed the expressivity of the target language. In Text2Onto we follow a translation-based approach to knowledge engineering. So called *ontology writers* are then responsible for translating instantiated modeling primitives into a specific target knowledge representation language. While a translation to various ontology languages is possible, in the scope of this paper, we focus on the translation to OWL ontologies. The modeling primitives we use in Text2Onto and their correspondences in the OWL ontology model are described by Table 1.

To capture contextual information about ontology elements, such as provenance and certainty in the learning process, we introduce the notion of *rating annotations*.

Modeling Primitive	Explanation	OWL
`concept`	A concept C. Example: *man, person*	C
`instance`	An instance a. Example: *John, Mary*	a
`subconcept-of`	Concept inheritance. Example: `subconcept-of`(*man,person*)	$C_1 \sqsubseteq C_2$
`instance-of`	Concept instantiation. Example: `instance-of`(*John,person*).	$C(a)$
`relation`	A relation R between C_1 and C_2. Example: `love`(*person,person*)	$C_1 \sqsubseteq \forall R.C_2$
`part-of`	Mereological part-whole relation between C_1 and C_2. Example: `part-of`(*wheel,car*)	*part-of*(C_1, C_2)
`equivalence`	Equivalence of concepts C_1 and C_2. Example: `equivalence`(*town,city*)	$C_1 \equiv C_2$
`equality`	Equality of instances a_1 and a_2. Example: `equality`(*UN,United Nations*)	$a_1 \approx a_2$
`disjointness`	Disjointness of concepts C_1 and C_2. Example: `disjointness`(*man,woman*)	$C_1 \sqsubseteq \neg C_2$

Table 1. LOM Modeling Primitives

Definition 4. *Let N denote the set of all possible ontology elements and $\mathcal{X}$ be a suitable representation of a context space, then an ontology rating annotation is a partial function $r : N \rightarrow \mathcal{X}$.*

In Text2Onto we use these rating annotations to model the certainty of the system about the correctness of a particular ontology element. In particular, we define a special ontology rating annotation

$$r_{conf} : N \rightarrow [0, 1]$$

to indicate how confident the system is about the correctness of an ontology element. The confidences are calculated based on different kinds of evidences provided by the ontology learning algorithms that indicate the correctness and the relevance of ontology elements for the domain in question. They can be considered as a corpus-based support for ontology elements.

Algorithms We now describe for each modeling primitive the algorithms used to learn corresponding instances thereof. In particular, we explain the way the confidence and relevance ratings for an instantiated modeling primitive are calculated.

Concepts and Instances Different term weighting measures are used to compute the relevance of a certain concept or instance with respect to the corpus: Relative Term Frequency (RTF), TFIDF, Entropy and the C-value/NC-value method in [17].

Subconcept-of Relations In order to learn subconcept-of relations, we have implemented a variety of different algorithms exploiting the hypernym structure of WordNet [11], matching Hearst patterns [14] in the corpus as well as in the WWW and applying linguistic heuristics mentioned in [24]. The resulting confidence values of these algorithms are then combined through combination strategies as described in [6].

Instance-of Relations In order to assign instances or named entities appearing in the corpus to a concept in the ontology Text2Onto relies on a similarity-based approach extracting context vectors for instances and concepts from the text collection and as-

signing instances to the concept corresponding to the vector with the highest similarity with respect to their own vector [8]. Alternatively, we also implemented a pattern-matching algorithm similar to the one used for discovering part-of relations.

General Relations To learn general relations, Text2Onto employs a shallow parsing strategy to extract subcategorization frames (e.g. `hit(subj,obj,pp(with))`, transitive + PP-complement) enriched with information about the frequency of the terms appearing as arguments [19]. These subcategorization frames are mapped to relations such as `hit`(*person*,*thing*) and `hit_with`(*person*,*object*). The confidence is estimated on the basis of the frequency of the subcategorization frame as well as of the frequency with which a certain term appears at the argument position. For the purpose of discovering **part-of relations** in the corpus, we developed regular expressions matching lexico-syntactic patterns as described in [5] and implemented an algorithm counting the occurrences of patterns indicating a part-of relation between two terms t_1 and t_2, i.e. part-of(t_1,t_2). The confidence is then calculated by dividing by the sum of occurrences of patterns in which t_1 appears as a part. The results are combined with confidences which can be acquired by consulting WordNet for mereological relations.

Equivalence and Equality Following the assumption that terms are similar to the extent to which they share similar syntactic contexts, we implemented algorithms calculating the similarity between terms on the basis of contextual features extracted from the corpus, whereby the context of a terms varies from simple word windows to linguistic features extracted with a shallow parser. This corpus-based similarity is then taken as the confidence for the equivalence of the corresponding concepts or instances.

Disjointness For the extraction of disjointness axioms we implemented a simple heuristic based on lexico-syntactic patterns. In particular, given an enumeration of noun phrases $NP_1, NP_2, ...(and|or)NP_n$ we conclude that the concepts $C_1, C_2, ...C_k$ denoted by these noun phrases are pairwise disjoint, where the confidence for the disjointness of two concepts is obtained from the number of evidences found for their disjointness in relation to the total number of evidences for the disjointness of these concepts with other concepts.

4 Transforming Learned Ontologies to OWL

In this section we discuss the transformation of learned ontologies as described in the previous section to OWL ontologies (c.f. Section 2). As mentioned before, a naive translation that simply disregards the certainty information (rating annotations) would result in a potentially highly inconsistent knowledge base that would not allow meaningful reasoning. The goal of the transformation therefore is to obtain an ontology that is (1) consistent (to allow meaningful reasoning), and (2) captures the most certain information while disregarding the potentially erronous information. In general, there may be many different consistent ontologies obtained from a LOM. The difficulty is to select the "best" ontology, i.e. the one that will result in most meaningful reasoning.

Evaluation Function In order to able to define what a "good" ontology for a particular context is, we need to be able to measure the quality of the ontology with respect to given set of criteria. We therefore define the notion of an *ontology evaluation function*.

Definition 5. *Let $\mathcal{O}$ be the set of possible ontologies, then an ontology evaluation function e is a function $e : \mathcal{O} \rightarrow [0, 1]$.*

Effectively, the evaluation function provides a total order over the space of possible ontologies and thus allows to compare given ontologies. Here it is important to note that the evaluation function can take the rating annotations into account and thus provides an evaluation measure for a given context. Using the evaluation function, we can define the problem of translating a given learned ontology LOM to a "discrete" and consistent OWL ontology as: $max_{O \subseteq LOM}\ e(O)$.
In other words, we try to find the best ontology O based on the knowledge in LOM that maximizes the evaluation function.

For our particular goal to obtain a consistent ontology capturing the most certain information, we can define an evaluation function as follows:

$$e_{certainty}(O) = \begin{cases} \max\left(\frac{\sum_{\alpha \in O}\ r_{conf}(\alpha) - t}{\|O\|}, 0\right) & \text{if } O \text{ is consistent} \\ 0 & \text{if } O \text{ is inconsistent} \end{cases} \quad (1)$$

Let us discuss the intuition behind this function. The basic idea is to maximize the certainty of the ontology based on the confidence of its individual axioms, as given by $r_{conf}(\alpha)$. The threshold t is introduced to "filter out" axioms with a confidence below a minimal value: Adding an axiom with a confidence below t will thus decrease the value of ontology. An inconsistent ontology is defined to have "no value".

In general, it will be hard to determine the optimal ontology that maximizes the evaluation function, as one theoretically would need to search entire space of possible consistent ontologies. However, in most cases it is not necessary to prove the optimality of an obtained solution, especially when considering that the rating annotations themselves are already somewhat imprecise. Instead it is possible to exploit heuristics to obtain a "fairly" optimal ontology.

We now outline an algorithm that exploits the behavior of the evaluation function and local characteristics of inconsistencies to maximize the value. It is based on the ideas of consistent ontology evolution as presented in [12]. Consistent ontology evolution ensures the consistency of ontologies when the ontology is changed by mapping consistency conditions that need to be satisfied to resolution functions that resolve introduced inconsistencies. The task of the resolution function consists of two main steps: (1) localizing the inconsistency and (2) generating additional changes that lead to another consistent state.

We treat the transformation of a LOM ontology to a consistent OWL ontology in a similar way as shown in Algorithm 1: Starting with an empty ontology O, we incrementally add all axioms from the learned ontology LOM whose confidence is equal to or greater than the threshold t. If adding the axioms leads to an inconsistent ontology, we localize the inconsistency by identifying a minimal inconsistent subontology. (For the details of this procedure, we refer the reader to [12]). An ontology O' is a minimal inconsistent subontology of O, if O' and every subontology of O' is consistent. Within this minimal inconsistent subontology we then identify the axiom that is most uncertain, i.e. has the lowest confidence value. This axiom will be removed from the ontology, thus resolving the inconsistency.

Algorithm 1 Algorithm for Transforming a LOM into a consistent OWL ontology

Require: A learned Ontology LOM

$O := \emptyset$
for all $\alpha \in LOM, r_{conf}(\alpha) \geq t$ **do**
  $O := O \cup \{\alpha\}$
  while O is inconsistent **do**
    $O' :=$ minimal_inconsistent_subontology(O, α)
    $\alpha^- := \alpha$
    for all $\alpha' \in O'$ **do**
      if $r_{conf}(\alpha') \leq r_{conf}(\alpha)$ **then**
        $\alpha^- := \alpha'$
      end if
    end for
    $O := O \setminus \{\alpha^-\}$
  end while
end for

5 Evaluation and Experimental Results

We have applied the approach presented in the previous chapter to ontologies learned from a corpus of 1700 abstracts (from documents about knowledge management) of the BT Digital Library. The learned ontology (LOM) consisted of 938 concepts and 125 instances. For the concepts, 406 subconcept-of relations and 2322 disjoint-concepts relations were identified. For the instances, 143 instance-of relations were obtained (as multiple instantiations is allowed).

For the transformation of the LOM ontology to a discrete OWL ontology, we applied the evaluation function and algorithms presented in the previous section. Here we performed an analysis of the influence of the threshold of uncertainty on the transformation. The results in Table 2 clearly show the connection between the level of uncertainty and inconsistency introduced:

Threshold t	# of Inconsistencies	# of Axioms in Result
0.1	40	1706
0.2	8	705
0.4	3	389
0.8	0	197

Table 2. Influence of certainty threshold t on transformation process

A low threshold t results in more uncertain information being allowed in the target ontology. As a result, the chances for inconsistencies increase. How to choose the "right" threshold t for the transformation process will very much depend on the application scenario, as it essentially means finding a trade-off between the amount of information learned and the confidence in the correctness of the learned information.

In the following we will discuss typical types of inconsistencies and present examples of such inconsistencies that were detected and resolved. The first type of inconsistency involves unsatisfiable concepts (often called incoherent concepts) in the $\mathcal{T}$-Box of

the ontology. This can for example happen if two concepts are identified to be disjoint, but at the same time these concepts are in a subconcept-relation (either explicitly asserted or inferred). Interestingly, this type of inconsistency often occurred for concepts for which even for a domain expert the correct relationship is hard to identify, as the following example shows:

Example 1. The relationship between the concepts *Data*, *Information*, and *Knowledge* is a very subtle (often philosophical) one, for which one will encounter different definitions depending on the context. The (inconsistent) definitions learned from our data set stated that $Data$ is a subconcept of both $Information$ and $Knowledge$, while $Information$ and $Knowledge$ are disjoint concepts:

Axiom t	Confidence
$Data \sqsubseteq Information$	1.0
$Data \sqsubseteq Knowledge$	1.0
$Information \sqsubseteq \neg Knowledge$	0.7

The inconsistency was resolved by removing the disjointness axiom, as its confidence value was lowest.

The second type of inconsistencies involves $\mathcal{A}$-Box assertions. Here, typically instances were asserted to be instances of two concepts that were identified to be disjoint. We again present an example:

Example 2. Here $KaViDo$ was identified to be both an instance of $Application$ and a $Tool$ (based on the abstract of [23]), however, $Application$ and $Tool$ were learned to be disjoint concepts:

Axiom t	Confidence
$Application(kavido)$	0.46
$Tool(kavido)$	0.46
$Tool \sqsubseteq \neg Application$	0.3

This inconsistency was again resolved by removing the disjointness axiom.

Other types of inconsistencies involving, for example, domain and range restrictions were not considered in our current experiments, thus being left for future work. Nevertheless, this evaluation showed that inconsistency is an important issue in ontology learning.

6 Related Work

Since building an ontology for a huge amount of data is a difficult and time consuming task a number of tools such as TextToOnto [20], the ASIUM system [10], the Mo'k Workbench [3], OntoLearn [24] or OntoLT [4] have been developed in order to support the user in constructing ontologies from a given set of (textual) data. So far, none of these tools explicitly addresses the problem of uncertainty. Text2Onto implements the first approach towards integrating uncertainty into ontology learning. Obviously, the LOM of Text2Onto is not probabilistic in a strict mathematical sense. Nevertheless,

several researchers have already addressed the issue of integrating and reasoning with probabilities in knowledge representation formalisms. [9] for example present a probabilistic extension of the Ontology Language OWL which relies on Bayesian Networks for reasoning. Other researchers have integrated probabilities into first-order logic [2] or description logics [18]. Fuzzy extensions of OWL have been proposed e.g. in [22].

The approach to dealing with inconsistencies presented in this work is based on the idea of obtaining a consistent ontology from a LOM to be then able to derive consistent query answers. A very related approach is that of reasoning with inconsistent ontologies. A typical technique is the selection of a consistent subontology for a given query, which yields a consistent query answer (c.f. [16]). The important question here is how to select the *right* subontology. While current techniques often rely on syntactic selection functions, it would also be possible to rely on the rating annotations available in the LOM to guide the selection function. Another related approach is that of diagnosis and repair of inconsistencies based on techniques such as pinpointing [21]. The pinpointing technique tries to identify and remove a *minimal* set of axioms (in terms of number of axioms) to obtain a consistent ontology, while we try to identify the *most certain* consistent ontology. As there are typically multiple possible pinpoints, a combination of pinpointing with the notion of certainty of our work is an interesting path to explore.

7 Conclusion and Future Work

Ontology learning is a promising technique for automated knowledge acquisition from text corpora. However, as we have shown, uncertainty and inconsistencies are issues that need to be dealt with in order to allow meaningful reasoning over the learned ontologies. In this paper we have presented how uncertainty can be represented in the Learned Ontology Model (LOM) and how such learned ontologies can be transformed to consistent OWL ontologies using the notion of an ontology evaluation function. Our experiments with ontologies learned from documents of a Digital Library show the feasibility and usefulness of the approach. An extensive evaluation will be performed as part of a case study within the SEKT project.

It is important to mention that confidence as generated by ontology learning algorithms represent a data-driven approach to the evaluation of ontologies. There are many other notions of ontology quality and consistency which could be used for the definition of an ontology evolution function. In particular, we will in the future integrate an automatic approach towards the formal evaluation of ontologies by means of the OntoClean methodology as presented in [25].

Acknowledgements Research reported in this paper has been financed by the EU in the IST project SEKT (IST-2003-506826) (`http://www.sekt-project.com/`).

References

1. P. Smets A. Motro. *Uncertainty Management In Information Systems*. Springer, 1997.
2. F. Bacchus. *Representing and Reasoning with Probabilistic Knowledge*. MIT Press, 1990.

3. G. Bisson, C. Nedellec, and L. Canamero. Designing clustering methods for ontology building - The Mo'K workbench. In *Proc. of the ECAI Ontology Learning WS*, 2000.
4. P. Buitelaar, D. Olejnik, and M. Sintek. OntoLT: A protégé plug-in for ontology extraction from text. In *Proceedings of the International Semantic Web Conference (ISWC)*, 2003.
5. E. Charniak and M. Berland. Finding parts in very large corpora. In *Proceedings of the 37th Annual Meeting of the ACL*, pages 57–64, 1999.
6. P. Cimiano, A. Pivk, L. Schmidt-Thieme, and S. Staab. Learning taxonomic relations from heterogeneous sources of evidence. In *Ontology Learning from Text: Methods, Applications and Evaluation*. IOS Press, 2005.
7. P. Cimiano and J. Völker. A framework for ontology learning and data-driven change discovery. In *Proc. of the NLDB'2005*, 2005.
8. P. Cimiano and J. Völker. Towards large-scale, open-domain and ontology-based named entity classification. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP'05)*, SEP 2005.
9. Z. Ding and Y. Peng. A probabilistic extension to ontology language OWL. In *Proceedings of the 37th Hawaii International Conference on System Sciences*, 2004.
10. D. Faure and C. Nedellec. A corpus-based conceptual clustering method for verb frames and ontology. In *Proceedings of the LREC Workshop on Adapting lexical and corpus resources to sublanguages and applications*, 1998.
11. C. Fellbaum. *WordNet, an electronic lexical database*. MIT Press, 1998.
12. P. Haase and L. Stojanovic. Consistent evolution of OWL ontologies. In *Proceedings of the Second European Semantic Web Conference, Heraklion, Greece, 2005*, MAY 2005.
13. P. Haase, F. van Harmelen, Z. Huang, H. Stuckenschmidt, and Y. Sure. A framework for handling inconsistency in changing ontologies. In *Proc. of the Fourth International Semantic Web Conference (ISWC'05)*, NOV 2005.
14. M.A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics*, pages 539–545, 1992.
15. I. Horrocks and P. F. Patel-Schneider. Reducing OWL Entailment to Description Logic Satisfiability. *Journal of Web Semantics*, 1(4), 2004.
16. Z. Huang, F. van Harmelen, and A. ten Teije. Reasoning with inconsistent ontologies. In *Proceedings of IJCAI'05*, August 2005.
17. J. Tsuji K. Frantzi, S. Ananiadou. The c-value/nc-value method of automatic recognition for multi -word terms. In *Proceedings of the ECDL*, pages 585–604, 1998.
18. D. Koller, A. Levy, and A. Pfeffer. P-classic: A tractable probabilistic description logic. In *Proceedings of AAAI-97*, pages 390–397, 1997.
19. A. Maedche and S. Staab. Discovering conceptual relations from text. In W. Horn, editor, *Proceedings of the 14th ECAI'2000*, 2000.
20. A. Maedche and S. Staab. Ontology learning. In S. Staab and R. Studer, editors, *Handbook on Ontologies*, pages 173–189. Springer, 2004.
21. S. Schlobach. Debugging and semantic clarification by pinpointing. In *Proceedings of the Second European Semantic Web Conference, Heraklion, Greece, 2005*, pages 226–240, 2005.
22. U. Straccia. Towards a fuzzy description logic for the semantic web (preliminary report). In *Proceedings of the Second European Semantic Web Conference, 2005*, pages 167–181, 2005.
23. O. Tamine and R. Dillmann. Kavido: a web-based system for collaborative research and development processes. *Computers in Industry*, 52(1):29–45, 2003.
24. P. Velardi, R. Navigli, A. Cuchiarelli, and F. Neri. Evaluation of ontolearn, a methodology for automatic population of domain ontologies. In *Ontology Learning from Text: Methods, Applications and Evaluation*. IOS Press, 2005.
25. J. Völker, D. Vrandecic, and Y. Sure. Automatic evaluation of ontologies (AEON). In *Proc. of the Fourth International Semantic Web Conference (ISWC'05)*, NOV 2005.