EU-IST Project IST-2003-506826 SEKT

SEKT: Semantically Enabled Knowledge Technologies

D1.6.1 Ontology Evaluation

Janez Brank
Marko Grobelnik, Dunja Mladenić
(Jožef Stefan Institute)

Abstract

This report presents an overview of topics dealing with the evaluation of ontologies. Ontology evaluation is the problem of assessing the quality of a given ontology, either to aid in the selection of an ontology for the needs of a particular task or organization, or to evaluate or guide an ontology construction effort (either manual or partially/fully automated). An ontology is commonly evaluated by comparing it to a "golden standard", or by testing how well it fits a domain-specific corpus of documents, or by using it in an application and evaluating the output of the application, or by using some other set of criteria. We present a software component for evaluating an ontology by comparing it to a golden standard. Both ontologies involved in the evaluation (the golden standard one and the one under evaluation) are assumed to be trees of concepts built on the same set of instances, but with different arrangement of instances into concepts and concepts into a hierarchy.

Keyword list: Semantic Web, ontologies, hierarchies, ontology evaluation, algorithms, methodologies

WP1
Prototype/Report PU
Contractual date of delivery: 31.6.2005
Actual date of delivery: 5.8.2005

SEKT Consortium

This document is part of a research project partially funded by the IST Programme of the Commission of the European Communities as project number IST-2003-506826.

British Telecommunications plc.
Orion 5/12, Adastral Park
Ipswich IP5 3RE
UK
Tel: +44 1473 609583, Fax: +44 1473 609832
Contact person: John Davies
E-mail: john.nj.davies@bt.com

Empolis GmbH
Europaallee 10
67657 Kaiserslautern
Germany
Tel: +49 631 303 5540
Fax: +49 631 303 5507
Contact person: Ralph Traphöner
E-mail: ralph.traphoener@empolis.com

Jozef Stefan Institute
Jamova 39
1000 Ljubljana
Slovenia
Tel: +386 1 4773 778, Fax: +386 1 4251 038
Contact person: Marko Grobelnik
E-mail: marko.grobelnik@ijs.si

University of Karlsruhe, Institute AIFB
Englerstr. 28
D-76128 Karlsruhe
Germany
Tel: +49 721 608 6592
Fax: +49 721 608 6580
Contact person: York Sure
E-mail: sure@aifb.uni-karlsruhe.de

University of Sheffield
Department of Computer Science
Regent Court, 211 Portobello St.
Sheffield S1 4DP
UK
Tel: +44 114 222 1891
Fax: +44 114 222 1810
Contact person: Hamish Cunningham
E-mail: hamish@dcs.shef.ac.uk

University of Innsbruck
Institute of Computer Science
Techikerstraße 13
6020 Innsbruck
Austria
Tel: +43 512 507 6475
Fax: +43 512 507 9872
Contact person: Jos de Bruijn
E-mail: jos.de-bruijn@deri.ie

Intelligent Software Components S.A.
Pedro de Valdivia, 10
28006
Madrid
Spain
Tel: +34 913 349 797
Fax: +49 34 913 349 799
Contact person: Richard Benjamins
E-mail: rbenjamins@isoco.com

Kea-pro GmbH
Tal
6464 Springen
Switzerland
Tel: +41 41 879 00
Fax: 41 41 879 00 13
Contact person: Tom Bösser
E-mail: tb@keapro.net

Ontoprise GmbH
Amalienbadstr. 36
76227 Karlsruhe
Germany
Tel: +49 721 50980912
Fax: +49 721 50980911
Contact person: Hans-Peter Schnurr
E-mail: schnurr@ontoprise.de

Sirma AI EAD, Ontotext Lab
135 Tsarigradsko Shose
Sofia 1784
Bulgaria
Tel: +359 2 9768 303, Fax: +359 2 9768 311
Contact person: Atanas Kiryakov
E-mail: naso@sirma.bg

Vrije Universiteit Amsterdam (VUA)
Department of Computer Sciences
De Boelelaan 1081a
1081 HV Amsterdam
The Netherlands
Tel: +31 20 444 7731, Fax: +31 84 221 4294
Contact person: Frank van Harmelen
E-mail: frank.van.harmelen@cs.vu.nl

Universitat Autonoma de Barcelona
Edifici B, Campus de la UAB
08193 Bellaterra (Cerdanyola del Vall` es)
Barcelona
Spain
Tel: +34 93 581 22 35, Fax: +34 93 581 29 88
Contact person: Pompeu Casanovas Romeu
E-mail: pompeu.casanovas@uab.es

Executive Summary

This report presents an overview of topics dealing with the evaluation of ontologies. Ontology evaluation is important in several contexts. A user may be wondering which ontology in a given library of ontologies is the most likely to be suitable of his or her requirements; or one may want to understand how good an ontology has been produced by some particular ontology construction effort (either manual or automated); additionally, evaluation can be a component in automated ontology learning approaches, to be used for model selection, tuning of parameters, or guiding the exploration of a search space.

Ontology evaluation can be approached from various perspectives, either by comparing the ontology to a "golden standard", or by testing how well the ontology fits a corpus of documents about the problem domain, or by using the ontology in a specific task or application and evaluating the output of the application, or by using some other set of criteria, scoring the ontology with respect to each criterion and then computing a combined weighted score.

Various techniques are used for ontology evaluation depending on which aspect or level of ontology we wish to evaluate: the selection of concepts; the terminology or vocabulary used; the hierarchical (is-a) relationships between concepts; other semantic relations; the interaction of the ontology with other ontologies, the functioning of the ontology within an application; syntactic, structural, and architectural aspects of the ontology.

We present a software component for evaluating an ontology by comparing it to a golden standard. The approach used assumes that both ontologies involved in the evaluation (the golden standard one and the one under evaluation) are hierarchical trees of concepts built on the same set of instances; however, the concepts and their hierarchical relationships may be different in each ontology. The software component processes descriptions of both ontologies and outputs a similarity measure. This approach is designed to fit the requirements of a concrete ontology evaluation task that is defined in the forthcoming PASCAL NoE ontology learning challenge. Evaluation of this type would be particularly suitable for evaluating approaches for automatic construction of ontologies from data. We also briefly discuss some approaches for ontology evaluation not involving a golden standard, which could be included in an automated ontology construction approach for the purpose of guiding the learning process.

Contents

SEKT Consortium **2**
Executive Summary **3**
Contents **4**
1. Introduction **5**
2. Related Work **5**
2.1 Ontology evaluation at different levels 6
2.2 Evaluation on the lexical/vocabulary and concept/data level 7
2.3 Evaluation of taxonomic and other semantic relations 8
2.4 Context-level evaluation 10
2.5 Application-based evaluation 11
2.6 Data-driven evaluation 11
2.7 Multiple-criteria approaches 12
3. Theoretical Framework for Ontology Evaluation **13**
4. Architecture and Approach **15**
4.1 Task description 15
4.2 Similarity measures on partitions 16
4.3 A similarity measure for ontologies 16
4.4 Approximation algorithms 18
4.5 Input and output files 19
5. Future work **20**
5.1 Evaluation without a golden standard 21
6. Relation to SEKT activities **22**
6.1 Relation to SEKT technical workpackages 22
6.2 Relation to SEKT case studies tasks 23
Appendix - User Guide **24**
Bibliography and References **24**

1. Introduction

We can observe that the focus of modern information systems is moving from “data processing” towards “concept processing”, meaning that the basic unit of processing is less and less an atomic piece of data and is becoming more a semantic concept which caries an interpretation and exists in a context with other concepts. Ontology is commonly used as a structure capturing knowledge about a certain area via providing relevant concepts and relations between them.

A key factor which makes a particular discipline or approach scientific is the ability to evaluate and compare the ideas within the area. The same holds also for Semantic Web research area when dealing with abstractions in the form of ontologies. Ontologies are a fundamental data structure for conceptualizing knowledge which is in most practical cases soft and non-uniquely expressable. As a consequence, we are in general able to build many different ontologies conceptualizing the same body of knowledge and we should be able to say which of these ontologies serves better some predefined criterion.

Thus, ontology evaluation is an important issue that must be addressed if ontologies are to be widely adopted in the semantic web and other semantics-aware applications. Users facing a multitude of ontologies need to have a way of assessing them and deciding which one best fits their requirements the best. Likewise, people constructing an ontology need a way to evaluate the resulting ontology and possibly to guide the construction process and any refinement steps. Automated or semi-automated ontology learning techniques also require effective evaluation measures, which can be used to select the “best” ontology out of many candidates, to select values of tunable parameters of the learning algorithm, or to direct the learning process itself (if the latter is formulated as a path through a search space).

The remainder of this report is structured as follows. In section 2, we present an overview of related work on ontology evaluation. We describe the main groups of ontology evaluation approaches, and show different techniques that are used to evaluate different aspects or levels of an ontology. In section 3, we refer to a formal framework for defining an ontology and show how the various aspects of evaluation can be incorporated in such a framework. In section 4, we present our software component for evaluating a hierarchic ontology by comparing it to a “golden standard”. In section 5, we present some guidelines for future work.

2. Related Work

Various approaches to the evaluation of ontologies have been considered in the literature, depending on what kind of ontologies are being evaluated and for what purpose. Broadly speaking, most evaluation approaches fall into one of the following categories:

- those based on comparing the ontology to a “golden standard” (which may itself be an ontology; e.g. MAEDCHE AND STAAB, 2002);
- those based on using the ontology in an application and evaluating the results (e.g. PORZEL AND MALAKA, 2004);

- those involving comparisons with a source of data (e.g. a collection of documents) about the domain that is to be covered by the ontology (e.g. BREWSTER *et al.*, 2004);
- those where evaluation is done by humans who try to assess how well the ontology meets a set of predefined criteria, standards, requirements, etc. (e.g. LOZANO-TELLO AND GÓMEZ-PÉREZ, 2004).

In addition to the above categories of evaluation, we can group the ontology evaluation approaches based on the level of evaluation, as described in the following subsections.

2.1 Ontology evaluation at different levels

An ontology is a fairly complex structure and it is often more practical to focus on the evaluation of different levels of the ontology separately rather than trying to directly evaluate the ontology as a whole. This is particularly true if the emphasis is on having the evaluation proceed automatically rather than being entirely carried out by human users/experts. Another reason for the level-based approach is that when automatic learning techniques have been used in the construction of the ontology, the techniques involved are substantially different for the different levels. The individual levels have been defined variously by different authors (e.g. GÓMEZ-PÉREZ, 1994; GÓMEZ-PÉREZ, 1996; BURTON-JONES *et al.*, 2004; PORZEL AND MALAKA, 2004; EHRIG *et al.*, 2005), but these various definitions tend to be broadly similar and usually involve the following levels:

- *Lexical, vocabulary, or data layer.* Here the focus is on which concepts, instances, facts, etc. have been included in the ontology, and the vocabulary used to represent or identify these concepts. Evaluation on this level tends to involve comparisons with various sources of data concerning the problem domain (e.g. domain-specific text corpora), as well as techniques such as string similarity measures (e.g. edit distance).
- *Hierarchy or taxonomy.* An ontology typically includes a hierarchical is-a or subsumption relation between concepts. Although various other relations between concepts may be also defined, the is-a relationship is often particularly important and may be the focus of specific evaluation efforts.
- *Other semantic relations.* The ontology may contain other relations besides is-a, and these relations may be evaluated separately. This typically includes measures such as precision and recall.
- *Context or application level.* An ontology may be part of a larger collection of ontologies, and may reference or be referenced by various definitions in these other ontologies. In this case it may be important to take this context into account when evaluating it. Another form of context is the application where the ontology is to be used; basically, rather than evaluate the ontology per se, it may be more practical to evaluate it within the context of a particular application, and to see how the results of the application are affected by the use of the ontology in question. Instead of focusing on an individual application, one may also focus on evaluation from the point of view of the individual users or the organization (e.g. company) that will use the ontology (FOX *et al.*, 1998).
- *Syntactic level.* Evaluation on this level may be of particular interest for ontologies that have been mostly constructed manually. The ontology is usually described in a particular formal language and must match the syntactic

requirements of that language (use of the correct keywords, etc.). Various other syntactic considerations, such as the presence of natural-language documentation, avoiding loops between definitions, etc., may also be considered (GÓMEZ-PÉREZ, 1994). Of all aspects of ontology evaluation, this is probably the one that lends itself the most easily to automated processing.

- *Structure, architecture, design.* This is primarily of interest in manually constructed ontologies. Assuming that some kind of design principles or criteria have been agreed upon prior to constructing the ontology, evaluation on this level means checking to what extent the resulting ontology matches those criteria. Structural concerns involve the organization of the ontology and its suitability for further development (e.g. addition of new concepts, modification or removal of old ones) (GÓMEZ-PÉREZ, 1994, 1996). For some applications, it is also important that the formal definitions and statements of the ontology are accompanied by appropriate natural-language documentation, which must be meaningful, coherent, up-to-date and consistent with the formal definitions, sufficiently detailed, etc. Evaluation of these qualities on this level must usually be done largely or even entirely manually by people such as ontological engineers and domain experts.

The following table summarizes which approaches from the list at the beginning of section 2 are commonly used for which of the levels discussed in this subsection.

Table 1. An overview of approaches to ontology evaluation on different levels.

	Approach to evaluation			
Level	**Golden standard**	**Application-based**	**Data-driven**	**Assessment by humans**
Lexical, vocabulary, concept, data	x	x	x	x
Hierarchy, taxonomy	x	x	x	x
Other semantic relations	x	x	x	x
Context, application		x		x
Syntactic	x[1]			x
Structure, architecture, design				x

[1] "Golden standard" in the sense of comparing the syntax in the ontology definition with the syntax specification of the formal language in which the ontology is written (e.g. RDF, OWL, etc.).

The next few subsections will present more details about the various approaches and the levels of evaluation.

2.2 Evaluation on the lexical/vocabulary and concept/data level

An example of an approach that can be used for the evaluation of a lexical/vocabulary level of an ontology is the one proposed by MAEDCHE AND STAAB (2002). Similarity between two strings is measured based on the Levenshtein edit distance, normalized to produce scores in the range [0, 1]. (Sometimes background knowledge about the domain can be used to introduce an improved domain-specific definition of the edit distance; for example, when comparing names of persons, one might take into account the fact that first names are often abbreviated; EHRIG *et al.*, 2005.) A *string matching* measure between two sets of strings is then defined by taking each string of the first set, finding its similarity to the most similar string in the second set, and averaging this over all strings of the first set. One may take the set of all strings used as concept identifiers in the ontology being evaluated, and compare it to a "golden standard" set of strings that are considered a good representation of the concepts of the problem domain under consideration. The golden standard could be in fact another ontology (as in Maedche and Staab's work), or it could be taken statistically from a corpus of documents (see section 2.6), or prepared by domain experts.

The lexical content of an ontology can also be evaluated using the concepts of precision and recall, as known in information retrieval. In this context, precision would be the percentage of the ontology lexical entries (strings used as concept identifiers) that also appear in the golden standard, relative to the total number of ontology words. Recall is the percentage of the golden standard lexical entries that also appear as concept identifiers in the ontology, relative to the total number of golden standard lexical entries. A downside of the precision and recall measures defined in this way is that they do not allow for minor differences in spelling (e.g. use of hyphens in multi-word phrases, etc.). Another way to achieve more tolerant matching criteria (BREWSTER *et al.*, 2004) is to augment each lexical entry with its hypernyms from WordNet or some similar resource; then, instead of testing for equality of two lexical entries, one can test for overlap between their corresponding sets of words (each set containing an entry with its hypernyms).

The same approaches could also be used to evaluate the lexical content of an ontology on other levels, e.g. the strings used to identify relations, instances, etc.

VELARDI *et al.* (2005) describe an approach for the evaluation of an ontology learning system which takes a body of natural-language text and tries to extract from it relevant domain-specific concepts (terms and phrases), and then find definitions for them (using web searches and WordNet entries) and connect some of the concepts by is-a relations. Part of their evaluation approach is to generate natural-language glosses for multiple-word terms. The glosses are of the form "x y = a kind of y, ⟨definition of y⟩, related to the x, ⟨definition of x⟩", where y is typically a noun and x is a modifier such as an adjective. A gloss like this would then be shown to human domain experts, who would evaluate it to see if the word sense disambiguation algorithm selected the correct definitions of x and y. An advantage of this kind of approach is that domain experts might be unfamiliar with formal languages in which ontologies are commonly described, and thus it might be easier for them to evaluate the natural-language glosses. Of course, the downside of this approach is that it nevertheless requires a lot of work on part of the domain experts.

2.3 Evaluation of taxonomic and other semantic relations

BREWSTER *et al.* (2004) suggested using a data-driven approach to evaluate the degree of structural fit between an ontology and a corpus of documents. (1) Given a corpus of documents from the domain of interest, a clustering algorithm based on expectation maximization is used to determine, in an unsupervised way, a probabilistic mixture model of hidden "topics" such that each document can be modeled as having been generated by a mixture of topics. (2) Each concept c of the ontology is represented by a set of terms including its name in the ontology and the hypernyms of this name, taken from WordNet. (3) The probabilistic models obtained during clustering can be used to measure, for each topic identified by the clustering algorithm, how well the concept *c* fits that topic. (4) At this point, if we require that each concept fits at least some topic reasonably well, we obtain a technique for lexical-level evaluation of the ontology. Alternatively, we may require that concepts associated with the same topic should be closely related in the ontology (via is-a and possibly other relations). This would indicate that the structure of the ontology is reasonably well aligned with the hidden structure of topics in the domain-specific corpus of documents. A drawback of this method as an approach for evaluating relations is that it is difficult to take the directionality of relations into account. For example, given concepts c_1 and c_2, the probabilistic models obtained during clustering in step (1) may be enough to infer that they should be related, but they are not really sufficient to infer whether e.g. c_1 is-a c_2, or c_2 is-a c_1, or if they should in fact be connected by some other relation rather than is-a.

Given a golden standard, evaluation of an ontology on the relational level can also be based on precision and recall measures. SPYNS (2005) discusses an approach for automatically extracting a set of lexons, i.e. triples of the form ⟨term$_1$, role, term$_2$⟩, from natural-language text. The result can be interpreted as an ontology, with terms corresponding to concepts and roles corresponding to (non-hierarchical) relations between concepts. Evaluation was based on precision and recall, comparing the ontology either with a human-provided golden standard, or with a list of statistically relevant terms. The downside of this approach is again the need for a lot of manual human work involved in preparing the golden standard.

A somewhat different aspect of ontology evaluation has been discussed by GUARINO AND WELTY (2002). They point out several philosophical notions (essentiality, rigidity, unity, etc.) that can be used to better understand the nature of various kinds of semantic relationships that commonly appear in ontologies, and to discover possible problematic decisions in the structure of an ontology. For example, a property is said to be *essential* to an entity if it necessarily holds for that entity. A property that is essential for all entities having this property is called *rigid* (e.g. "being a person": there is no entity that could be a person but isn't; everything that is a person is necessarily always a person); a property that cannot be essential to an entity is called *anti-rigid* (e.g. "being a student": any entity that is a student could also not be a student). A class defined by a rigid property cannot be the subclass of a class defined by an anti-rigid property. This observation allows us to conclude, if we see an ontology in which "person" is a subclass of "student", that this relationship is wrong. Various other kinds of misuse of the is-a relationship can also be detected in a similar way (for example, is-a is sometimes used to express meta-level characteristics of some class, or is used instead of is-a-part-of, or is used to indicate that a term may have multiple meanings). A downside of this approach is that it requires manual intervention by a trained human expert familiar with the above-mentioned notions such as rigidity; at the very least, the expert should annotate the concepts of the ontology with appropriate metadata tags, whereupon checks for certain kinds of errors can be made automatically. As pointed out e.g. in HARTMANN *et al.* (2005), applications where evaluation of this sort is truly important (and justifies the costs) are probably relatively rare. However, VÖLKER *et al.* (2005) recently proposed an approach to aid in the automatic assignment of these metadata tags.

MAEDCHE AND STAAB (2002) propose several measures for comparing the relational aspects of two ontologies. If one of the ontologies is a golden standard, these measures can also be used for ontology evaluation. Although this is in a way a drawback of this method, an important positive aspect is that once the golden standard is defined, comparison of two ontologies can proceed entirely automatically. The *semantic cotopy* of a term c in a given hierarchy is the set of all its super- and sub-concepts. Given two hierarchies H_1, H_2, a term t might represent some concept c_1 in H_1 and a concept c_2 in H_2. One can then compute the set of terms which represent concepts from the cotopy of c_1 in H_2, and the set of terms representing concepts from the cotopy of c_2; the overlap of these two sets can be used as a measure of how similar a role the term t has in the two hierarchies H_1 and H_2. An average of this may then be computed over all the terms occurring in the two hierarchies; this is a measure of similarity between H_1 and H_2.

Similar ideas can also be used to compare other relations besides is-a. Let R_1 be a binary relation in the first ontology, with a domain $d(R_1)$ and a range $r(R_1)$. Analogously, let R_2 be a binary relation in the second ontology. We can consider the relations to be similar if $d(R_1)$ is similar to $d(R_2)$ and $r(R_1)$ is similar to $r(R_2)$. Since $d(R_1)$ and $d(R_2)$ are simply two sets of concepts, they can be compared similarly as in the preceding paragraph: determine the set of terms that occur as names of any concept of $d(R_1)$ or any of its hypernyms; in analogous way, determine the set of terms for $d(R_2)$; then compute the overlap of these two sets. If there are several such pairs of relations, the similarity can be computed for each pair and then averaged to obtain an indicator of relational-level similarity between the two ontologies as a whole.

2.4 Context-level evaluation

Sometimes the ontology is a part of a larger collection of ontologies that may reference one another (e.g. one ontology may use a class or concept declared in another ontology), for example on the web or within some institutional library of ontologies. This context can be used for evaluation of an ontology in various ways. For example, the Swoogle search engine of DING *et al.* (2004) uses cross-references between semantic-web documents to define a graph and then compute a score for each ontology in a manner analogous to PageRank used by the Google web search engine. The resulting “ontology rank” is used by Swoogle to rank its query results. A similar approach has been used in the OntoKhoj portal of PATEL *et al.* (2003). In both cases an important difference in comparison to PageRank is that not all “links” or references between ontologies are treated the same. For example, if one ontology defines a subclass of a class from another ontology, this reference might be considered more important than if one ontology only uses a class from another as the domain or range of some relation.

Alternatively, the context for evaluation may be provided by human experts; for example, SUPEKAR (2005) proposes that an ontology be enhanced with metadata such as its design policy, how it is being used by others, as well as “peer reviews” provided by users of this ontology. A suitable search engine could then be used to perform queries on this metadata and would aid the user in deciding which of the many ontologies in a repository to use. The downside of this approach is that it relies almost entirely on manual human effort to both provide annotations and to use them in evaluating and selecting an ontology.

2.5 Application-based evaluation

Typically, the ontology will be used in some kind of application or task. The outputs of the application, or its performance on the given task, might be better or worse depending partly on the ontology used in it. Thus one might argue that a good ontology is one which helps the application in question produce good results on the given task. Ontologies may therefore be evaluated simply by plugging them into an application and evaluating the results of the application. This is elegant in the sense that the output of the application might be something for which a relatively straightforward and non-problematic evaluation approach already exists. For example, PORZEL AND MALAKA (2004) describe a scenario where the ontology, with its relations (both is-a and others) is used primarily to determine how closely related the meaning of two concepts is. The task is a speech recognition problem, where there may be several hypotheses about what a particular word in the sentence really means; a hypotheses should be coherent, which means that the interpretations of individual words should be concepts that are relatively closely related to each other. Thus the ontology is used to measure distance between concepts and thereby to assess the coherence of hypotheses (and choose the most coherent one). Evaluation of the final output of the task is relatively straightforward, and requires simply that the proposed interpretations of the sentences are compared with the gold standard provided by humans.

An approach like this can elegantly side-step the various complications of ontology evaluation and translate them to the problem of evaluating the application output, which is often simpler. However, this approach to ontology evaluation also has several drawbacks: (1) it allows one to argue that the ontology is good or bad when used in a particular way for a particular task, but it's difficult to generalize this observation (what if the ontology is used for a different task, or differently for the same task?); (2) the evaluation may be sensitive in the sense that the ontology could be only a small component of the application and its effect on the outcome may be relatively small (or depend considerably on the behavior of the other components); (3) if evaluating a large number of ontologies, they must be sufficiently compatible that the application can use them all (or the application must be sufficiently flexible), e.g. as regarding the format in which the ontology is described, the presence and names of semantic relations, etc. If it is necessary to adapt the application somewhat for each ontology that is to be evaluated, this approach to evaluation can quickly become very costly.

2.6 Data-driven evaluation

An ontology may also be evaluated by comparing it to existing data about the problem domain to which the ontology refers. This is usually a collection of textual documents. For example, PATEL *et al.* (2003) proposed an approach to determine if the ontology refers to a particular topic, and to classify the ontology into a directory of topics: one can extract textual data from the ontology (such as names of concepts and relations, and any other suitable natural-language strings) and use this as the input to a text classification model. The model itself can be trained by some of the standard machine learning algorithms from the area of text classification; a corpus of documents on a given subject can be used as the input to the learning algorithm.

Another data-driven approach has been proposed by BREWSTER *et al.* (2004). First, a set of relevant domain-specific terms are extracted from the corpus of documents, for example using latent semantic analysis. The amount of overlap between the domain-specific terms and the terms appearing in the ontology (e.g. as names of concepts) can then be used to measure the fit between the ontology and the corpus. Measures such as precision or recall could also be used in this context.

In the case of more extensive and sophisticated ontologies that incorporate a lot of factual information (such as Cyc, see e.g. *www.cyc.com*), the corpus of documents could also be used as a source of "facts" about the external world, and the evaluation measure is the percentage of these facts that can also be derived from information in the ontology.

2.7 Multiple-criteria approaches

Another family of approaches to ontology evaluation deals with the problem of selecting a good ontology (or a small short-list of promising ontologies) from a given set of ontologies, and treats this problem as essentially a decision-making problem. Therefore, techniques familiar from the area of decision support systems can be used to help us evaluate the ontologies and choose one of them. Usually, these approaches are based on defining several decision criteria or attributes; for each criterion, the ontology is evaluated and given a numerical score. Additionally a weight is also assigned (in advance) to each criterion, and an overall score for the ontology is then computed as a weighted sum of its per-criterion scores. This approach is analogous to the strategies used in many other contexts to select the best candidate out of many (e.g. tenders, grant applications, etc.). It could be particularly useful in situations where we are faced with a considerable number of ontologies roughly relevant to our domain in interest and wish to select the best ontology (or a few good ones). However, this type of approaches may still have difficulties such as the need for much manual involvement by human experts, for the presence of a golden standard ontology, etc. In effect, the general problem of ontology evaluation has been deferred or relegated to the question of how to evaluate the ontology with respect to the individual evaluation criteria.

BURTON-JONES *et al.* (2004) propose an approach of this type, with ten simple criteria such as syntactical correctness, clarity of vocabulary, etc. (a brief description of the way used to compute a numeric score for each attribute is included in parentheses):

- lawfulness (i.e. frequency of syntactical errors),
- richness (how many of the syntactic features available in the formal language are actually used by the ontology),
- interpretability (do the terms used in the ontology also appear in WordNet?),
- consistency (how many concepts in the ontology are involved in inconsistencies),
- clarity (do the terms used in the ontology have many senses in WordNet?),
- comprehensiveness (number of concepts in the ontology, relative to the average for the entire library of ontologies),
- accuracy (percentage of false statements in the ontology),

- relevance (number of statements that involve syntactic features marked as useful or acceptable to the user/agent),
- authority (how many other ontologies use concepts from this ontology),
- history (how many accesses to this ontology have been made, relative to other ontologies in the library/repository).

As can be seen from this list, this methodology involves criteria from most of the levels discussed in section 2.1. A downside of this approach is that there is little in it to help us ascertain to what extent the ontology matches the real-world state of the problem domain to which is refers (or indeed if it really deals with the domain we are interested in; it could be about some entirely unrelated subject; but this problem can be at least partially addressed by text categorization techniques, as used e.g. in PATEL *et al.*, 2004). The accuracy criterion in the list above provides a way to take the accuracy into account when computing the overall ontology score, but it's usually difficult to compute the percentage of false statements otherwise than by examining them all manually. On the positive side, the other criteria listed above can be computed automatically (although some of them assume that the ontology under consideration belongs to a larger library or repository of ontologies, and that metadata such as access history is available for the repository).

FOX *et al.* (1998) propose another set of criteria, which is however geared more towards manual assessment and evaluation of ontologies. Their criteria involve: functional completeness (does the ontology contain enough information for the application at hand?), generality (is it general enough to be shared by multiple users, departments, etc.?), efficiency (does the ontology support efficient reasoning?), perspicuity (is it understandable to the users?), precision/granularity (does it support multiple levels of abstraction/detail?), minimality (does it contain only as many concepts as necessary?).

LOZANO-TELLO AND GÓMEZ-PÉREZ (2004) define an even more detailed set of 117 criteria, organized in a three-level framework. The criteria cover various aspects of the formal language used to describe the ontology, the contents of the ontology (concepts, relations, taxonomy, axioms), the methodology used to construct the ontology, the costs (hardware, software, licensing, etc.) of using the ontology, and the tools available for working with the ontology. Many of the criteria are simple enough that the score of an ontology with respect to these criteria could be computed automatically or at least without much human involvement. The authors also cite several earlier works in the same area, with a more moderate number of criteria.

3. Theoretical Framework for Ontology Evaluation

In this section we present a formal definition of ontologies, provide examples of how various kinds of ontologies may be captured in the context of this formalization, and discuss how evaluation fits into this formal framework.

A reasonable and well thought-out formal definition of ontologies has been described recently in the work of EHRIG *et al.* (2005). In this formalization, the ontology (with datatypes) is defined as a structure $O = (C, T, R, A, I, V, \leq_C, \leq_T, \sigma_R, \sigma_A, \iota_C, \iota_T, \iota_R, \iota_A)$. It consists of (disjoint) sets of concepts (C), types (T), relations (R), attributes (A), instances (I) and values (V). The partial orders $\leq_C$ (on C) and $\leq_T$ (on T) define a concept hierarchy and a type hierarchy. The function σ_R: $R \rightarrow C^2$ provides relation signatures (i.e. for each relation, the function specifies which concepts may be linked by this relation), while σ_A: $A \rightarrow C \times T$ provides attribute signatures (for each attribute, the function specifies to which concept the attribute belongs and what is its datatype). Finally, there are partial instantiation functions ι_C: $C \rightarrow 2^I$ (the assignment of instances to concepts), ι_T: $T \rightarrow 2^V$ (the assignment of values to types), ι_R: $R \rightarrow 2^{I \times I}$ (which instances are related by a particular relation), and ι_A: $A \rightarrow 2^{I \times V}$ (what is the value of each attribute for each instance). (Another formalization of ontologies, based on similar principles, has also been described by BLOEHDORN *et al.* (2005) for the SEKT Deliverable 6.6.1.)

For some types of ontologies, this framework can be further extended, particularly with "concept attributes" in addition to the "instance attributes" mentioned above. The concept attributes would be a set A', with a signature function $\sigma_{A'}$: $A' \rightarrow T$ and an instantiation function $\iota_{A'}$: $A \rightarrow 2^{C \times V}$. The value of such an attribute would not be associated to a particular instance of a concept, but would apply to the concept as such. This extension will be useful for some of the evaluation scenarios considered later in this section. Other possible extensions, such as relations between concepts (as opposed to between instances), the introduction of metaclasses, or the introduction of relations with arity greater than 2, are probably of less practical interest.

A flexible formal network like this can accommodate various commonly-used kinds of ontologies:

- *Terminological ontologies* where concepts are word senses and instances are words. The WordNet ontology (http://www.cogsci.princeton.edu/~wn/) is an example of this. Attributes include things like natural-language descriptions of word senses (for concepts) and string representations of words (for instances).
- *Topic ontologies* where concepts are topics and instances are documents. Familiar examples include the Open Directory at http://www.dmoz.org/ or the Yahoo! directory at http://dir.yahoo.com/. Concept attributes typically consist of a name and a short description of each topic, and instance attributes consist of a document title, description, URL, and the main block of the text (for practical purposes, such text is often represented as a vector using e.g. the TF-IDF weighting under the vector space model of text representation).
- *Data-model ontologies* where concepts are tables in a data base and instances are data records (such as in a database schema). In this setting, datatypes and attributes in the above-mentioned formal definition of an ontology are straightforward analogies to the types and attributes (a.k.a. fields or columns) in a data base management system.

Evaluation can be incorporated in this theoretical framework as a function that maps the ontology O to a real number, e.g. in the range [0, 1]. However, as has been seen in section 2, a more practical approach is to focus the evaluation on individual components of the ontology O (which correspond roughly to the levels of ontology evaluation discussed in section 2). Results of the evaluation of individual components can later be aggregated into a combined ontology evaluation score (EHRIG *et al.*, 2005).

- The datatypes and their values (i.e. T, V, $\leq_T$, and ι_T) would typically not be evaluated; they are merely the groundwork on which the rest of the structure can stand.
- A lexical- or concept-level evaluation can focus on C, I, ι_C, and possibly some instance attributes from ι_A.
- Evaluation of the concept hierarchy (the is-a relationship) would focus on the $\leq_C$ partial order.
- Evaluation of other semantic relations would focus on R, ι_R, and the concept and instance attributes.
- One could also envision evaluation focusing on particular attributes; for example, whether a suitable natural-language name has been chosen for each concept. This kind of evaluation would take ι_C and the attributes as input and assess whether the concept attributes are suitable given ι_C and the instance attributes.
- Application- or task-based evaluation could be formalized by defining the application as a function $A(D, O)$ which produces some output given its input data D and the ontology O. By fixing the input data D, any evaluation function defined on the outputs of A becomes de facto an evaluation function on O. However, the practical applicability of such a formalization is debatable.
- Evaluation based on comparison to a golden standard can be incorporated into this theoretical framework as a function defined on a pair of ontologies (effectively a kind of similarity measure, or a distance function between ontologies). Similarly, data-driven evaluation can be seen as a function of the ontology and the domain-specific data corpus D, and could even be formulated probabilistically as $P(O|D)$.

4. Architecture and Approach

We have developed an approach for ontology evaluation primarily geared for the forthcoming concrete task defined in the ontology learning challenge organized in the context of the PASCAL network of excellence. The approach is based on the golden standard paradigm and its main focus is to compare how well the given ontology resembles the golden standard in the arrangement of instances into concepts and the hierarchical arrangement of the concepts themselves.

4.1 Task description

The ontologies that will be evaluated in the Pascal ontology learning challenge are based on the instances from the "Science" subtree of the dmoz.org internet directory. The dmoz directory is a topic ontology structured as a hierarchy of topics, and each topic may contain (besides subtopics) zero or more links to external web pages. Each link includes a title and a short description of the external web page. In the context of the ontology learning challenge, each link to an external web page represents an instance. The challenge is, given the set of all instances from the "Science" subtree of dmoz.org (a total of approx. 100000 instances), to arrange the instances into a hierarchy of concepts. In effect, this is similar to an unsupervised hierarchical clustering problem. The resulting hierarchy of concepts (with each instance attached to one of the concepts) is in effect a simple ontology (the hierarchical relationship between concepts can be approximately interpreted as an "is-a" relation). To evaluate these ontologies, they will be compared to the “Science” subtree of the real dmoz.org directory, which will thus assume the role of a golden standard.

In this evaluation task, each instance is represented by a short document of natural-language text (i.e. the title and description of the external page, as it appears in the dmoz.org directory). The concepts of the learned ontologies, however, are not explicitly represented by any terms, phrases, or similar textual descriptions. (The question of how to select a good short textual representation, or perhaps a set of keywords, for a particular learned concept could in itself be a separate task of the challenge, but is not part of the ontology learning task whose evaluation is being discussed here.) Additionally, since the number of instances (as well as concepts) is fairly large, the evaluation must be reasonably fast and completely automated.

4.2 Similarity measures on partitions

Our approach to evaluation is based on the analogies between this ontology learning task and traditional unsupervised clustering. In clustering, the task is to partition a set of instances into a family of disjoint subsets. Here, the ontology can be seen as a hierarchical way of partitioning the set of instances. The clustering community has proposed various techniques for comparing two partitions of the same set of instances, which can be used to compare the output of an automated clustering method with a golden-standard partition. If these distance measures on traditional “flat” partitions can be extended to hierarchical partitions, they can be used to compare a learned ontology to the golden-standard ontology (since both will be, in the context of this ontology learning task, two hierarchical partitions of the same set of instances).

One popular measure of agreement between two flat partitions is the Rand index (RAND, 1971). Assume that there is a set of instances $O = \{o_1, ..., o_n\}$,[1] with two partitions of O into a family of disjoint subsets, $U = \{U_1, ..., U_m\}$ and $V = \{V_1, ..., V_k\}$, where $\cup_{i=1..m} U_i = O$, $\cup_{j=1..k} V_j = O$, $U_i \cap U_{i'} = \{\}$ for each $1 \le i < i' \le m$, and $U_j \cap U_{j'} = \{\}$ for each $1 \le j < j' \le k$. Then one way to compare the partitions U and V is to count the agreements and disagreements in the placement of instances into clusters. If two items $o_i, o_j \in O$ belong to the same cluster of U but to two separate clusters of V, or vice versa, this is considered a disagreement. On the other hand, if they belong to the same cluster in both partitions, or to separate clusters in both partitions, this is considered an agreement between partitions. The Rand index between U and V is the number of agreements relative to the total number of pairs of instances (i.e. to $n(n–1)/2$).

4.3 A similarity measure for ontologies

We can elegantly formulate the Rand index as follows. Let us denote by $U(o)$ the cluster of U that contains the instance $o \in O$, and similarly by $V(o)$ the cluster of V that contains the instance $o \in O$. Let $\delta_U(U_i, U_j) = 1$ if $U_i = U_j$, and $\delta_U(U_i, U_j) = 0$ otherwise. If we define δ_V as well in an analogous manner, we can express the Rand index by the formula:

$$RandIdx(U, V) = 1 - [\Sigma_{1\le i<j\le n} |\delta_U(U(o_i), U(o_j)) - \delta_V(V(o_i), V(o_j))|] / [n(n-1)/2]. \qquad (1)$$

[1] In this section, O stands only for the set of instances, not for an entire ontology as in sec. 3. We use O instead of I for the set of instances to prevent confusion with the use of i as an index in subscripts.

That is, the term bracketed by |...| equals 1 if there is a disagreement between *U* and *V* concerning the placement of the pair of instances o_i and o_j. The sum over all *i* and *j* therefore counts the number of pairs where a disagreement occurs.

If we try to apply this measure for the purpose of comparing ontologies, we must take the hierarchical arrangement of concepts into account. In the original Rand index, what matters for a particular pair of instances is simply if they belong to the same cluster or not. However, when concepts or clusters are organized hierarchically, not any two different clusters are equally different. For example, two concepts with a common parent in the tree are likely to be quite similar even though they are not exactly the same; on the other hand, two concepts that do not have any common ancestor except the root of the tree are probably highly unrelated. Thus, if one ontology places a pair of instances in the same concept while the other ontology places this pair of instances in two different concepts with a common parent, this is a disagreement, but not a very strong disagreement; on the other hand, if the second ontology places the two instances into two completely unrelated concepts, this would be a large disagreement. We can still use the formula for *RandIndex*(*U*, *V*) given above, as long as we modify the functions δ_U and δ_V to take this intuition into account. That is, rather than returning merely 1 or 0 depending on whether the given two clusters are the same or not, the functions δ_U and δ_V should return a real number from the range [0, 1], expressing a measure of how closely related the two clusters are.

By plugging in various definitions of the functions δ_U and δ_V, we can obtain a family of similarity measures for ontologies, suitable for comparing an ontology with the golden standard in the context of the task that has been discussed at the beginning of this section. We propose two concrete families of δ_U and δ_V. Since the definitions of δ_U and δ_V will always be analogous to each other and differ only in the fact that each applies to a different ontology, we refer only to the δ_U function in the following discussion.

One possibility is inspired by the approach that is sometimes used to evaluate the performance of classification models for classification in hierarchies (see e.g. MLADENIĆ, 1998), and that could incidentally also be useful in the context of e.g. evaluating an automatic ontology population system. Given a concept U_i in the ontology *U*, let $A(U_i)$ be the set of all ancestors of this concept, i.e. all concepts on the path from the root to U_i (including U_i itself). If two concepts U_i and U_j have a common parent, the sets $A(U_i)$ and $A(U_j)$ will have a large intersection; on the other hand, if they have no common parent except the root, the intersection of $A(U_i)$ and $A(U_j)$ will contain only the root concept. Thus the size of the intersection can be taken as a measure of how closely related the two concepts are.

$$\delta_U(U_i, U_j) = |A(U_i) \cap A(U_j)| \,/\, |A(U_i) \cup A(U_j)|. \qquad (2)$$

This measure has the additional nice characteristic that it can be extended to cases where *U* is not a tree but an arbitrary directed acyclic graph. If the arrows in this graph point from parents to children, the set $A(U_i)$ is simply the set of all nodes from which *U* is reachable.

An alternative way to define a suitable function δ_U would be to work directly with the distances between U_i and U_j in the tree *U*. In this case, let *l* be the distance between U_i and U_j in the tree (length of the path from U_i to the common ancestor of U_i and U_j, and thence down to U_j), and *h* be the depth of the deepest common ancestor of U_i and

U_j. If l is large, this is a sign that U_i and U_j are not very closely related; similarly, if h is small, this is a sign that U_i and U_j don't have any common ancestors except very general concepts close to the root, and therefore U_i and U_j aren't very closely related. There are various ways of taking these intuitions into account in a formula for δ_U as a function of l and h. For example, RADA *et al.* (1989) have proposed a distance measure of the form:

$$\delta(l, h) = e^{-\alpha l} \operatorname{th}(\beta h) \qquad (3)$$

Here, α and β are nonnegative constants, and th is the hyperbolic tangent

$$\operatorname{th}(x) = (e^x - e^{-x}) / (e^x + e^{-x}) = 1 - 2/(1 + e^{2x}).$$

Thus, if h is small, $\operatorname{th}(\beta h)$ is close to 0, whereas for a large h it becomes close to 1. It is reasonable to treat the case when the two concepts are the same, i.e. when $U_i = U_j$ and thus $l = 0$, as a special case, and define $\delta(0, h) = 1$ in that case, to prevent $\delta_U(U_i, U_i)$ from being dependent on the depth of the concept U_i.

In fact the overlap-based version of d_U from eq. (2) can also be defined in terms of h and l. If the root is taken to be at depth 0, then the intersection of $A(U_i)$ and $A(U_j)$ contains $h + 1$ concepts, and the union of $A(U_i)$ and $A(U_j)$ contains $h + l - 1$ concepts. Thus, we see that eq. (2) is equivalent to defining

$$\delta(l, h) = (h + 1) / (h + l + 1). \qquad (4)$$

Note that the main part of the Rand index formula, as defined in equation (1), i.e. the sum $\Sigma_{1 \le i < j \le n} |\delta_U(U(o_i), U(o_j)) - \delta_V(V(o_i), V(o_j))|$, can also be interpreted as a Manhattan (L_1-norm) distance between two vectors of $n(n-1)/2$ components, one depending on the ontology U and the other depending only on the ontology V. Thus, in effect, we have represented an ontology U by a "feature vector" in which the (i, j)-th component has the value $\delta_U(U(o_i), U(o_j))$ describing how closely the instances o_i and o_j have been placed in that ontology. This interpretation opens the possibility of various further generalizations, such as using Euclidean distance instead of Manhattan distance, or even using kernel methods (cf. HAUSSLER, 1999). However, we leave such extensions for further work.

4.4 Approximation algorithms

As can be seen from eq. (1), the computation of our ontology similarity measure involves a sum over all pairs of documents, (i, j) for $1 \le i < j \le n$. This quadratic time complexity can be problematic when comparing ontologies with a fairly large number of instances (e.g. on the order of 100000, as in the case of the dmoz.org "Science" subtree mentioned in section 4.1).

One way to speed up the computation of the similarity measure and obtain an approximate result is to use a randomly sampled subset of pairs rather than all possible pairs of documents. That is, eq. (1) would then contain the average value of $|\delta_U(U(o_i), U(o_j)) - \delta_V(V(o_i), V(o_j))|$ over some subset of pairs instead of over all pairs.

Another way towards approximate computation of the similarity measure is to try to identify pairs (i, j) for which the difference $|\delta_U(U(o_i), U(o_j)) - \delta_V(V(o_i), V(o_j))|$ is not close to 0. If both ontologies classify the instances o_i and o_j into highly unrelated clusters, the values $\delta_U(U(o_i), U(o_j))$ and $\delta_V(V(o_i), V(o_j))$ will both be close to 0 and their difference will also be close to 0 and will not have a large effect on the sum. (In a typical dmoz-like hierarchy we can expect that a large proportion of pairs of instances will fall unto such relatively unrelated clustesr.) Thus it would be reasonable to try identifying pairs (i, j) for which o_i and o_j are in closely related clusters in at least one of the two ontologies, and computing the exact sum for these pairs, while disregarding the remaining pairs (or processing them using the subsampling technique from the previous paragraph). For example, suppose that δ_U is defined by eq. (4) as $\delta(l, h) = (h + 1) / (h + l + 1)$. Thus, we need to find pairs of concepts for which $(h + 1) / (h + l + 1)$ is greater than some threshold ε. (Then we will know that detailed processing is advisable for pairs of instances which fall into one of these pairs of concepts.) The condition $(h + 1) / (h + l + 1) > \varepsilon$ can be rewritten as $l < (h + 1)(1/\varepsilon - 1)$. Thus, suitable pairs of concepts could be identified by the following algorithm:

```
Initialize $P := \{\}$.
For each concept $c$:
    Let $h$ be the depth of $c$, and let $L = \lfloor (h + 1)(1/\varepsilon - 1) \rfloor$.
    Denote the children of $c$ (its immediate subconcepts) by $c_1, \ldots, c_r$.
    For each $l$ from 1 do $L$, for each $i$ from 1 to $r$, let $S_{l,i}$ be the set of
        those subconcepts of $c$ that are also subconcepts of $c_i$
        and are $l$ levels below $c$ in the tree.
    For each $l$ from 1 to $L$, for each $i$ from 1 to $r$,
        add to $P$ all the pairs from $S_{l,i} \times (\cup_{l' \leq L-1} \cup_{i' \neq i} S_{l',i'})$.
```

In each iteration of the outermost loop, the algorithm processes a concept c and discovers all pairs of concepts c', c'' such that c is the deepest common ancestor of c' and c'' and $\delta_U(c', c'') > \varepsilon$. For more efficient maintenance of the $S_{l,i}$ sets, it might be advisable to process the concepts c in a bottom-up manner, since the sets for a parent concept can be obtained by merging appropriate sets of its children.

For the time being, our software component supports random sampling of pairs as outlined at the beginning of this subsection. Separate treatment of pairs with $(h + 1) / (h + l + 1) > \varepsilon$ will be the topic of future work.

4.5 Input and output files

Both ontologies should be described in RDF format compatible with the one used for `structure.rdf` and `content.rdf` by the dmoz.org web directory (see http://rdf.dmoz.org/). The structure file contains information about concepts and the hierarchical relationship between them, while the content file contains information indicating how instances have been assigned to concepts.

The current version of our software component uses only a subset of the features available in the format of the RDF files currently used at dmoz.org.

The following example shows the syntax of a minimal `content.rdf` file:

```
<?xml version='1.0' encoding='UTF-8' ?>
<RDF xmlns:r="http://www.w3.org/TR/RDF/"
     xmlns:d="http://purl.org/dc/elements/1.0/"
     xmlns="http://dmoz.org/rdf">
<Topic r:id="Top/Arts/Movies/Titles/1/10_Rillington_Place">
  <link r:resource="http://www.britishhorrorfilms.co.uk/rillington.shtml"/>
  <link r:resource="http://www.shoestring.org/mmi_revs/10-rillington-place.html"/>
  ...
</Topic>
<Topic r:id="Top/Arts/Movies/Titles/1/1984_-_1984">
  <link r:resource="http://www.geocities.com/aaronbcaldwell/1984.html"/>
</Topic>
...
</RDF>
```

That is, the main `<RDF>` element contains a list of `<Topic>` subelements, and each `<Topic>` element contains zero or more `<link>` subelements. The value of the `r:id` attribute should be a unique identifier of the concept represented by that `<Topic>` element. Similarly, the `r:resource` attribute of a `<link>` should be the unique identifier of an instance. (In the original dmoz.org data, topic identifiers are hierarchical natural-language strings such as `Top/Arts/Movies`, and link identifiers are URLs of the external web pages, but our software component does not require that these assumptions hold for its input files.)

The following example shows the syntax of a minimal `structure.rdf` file:

```
<?xml version='1.0' encoding='UTF-8' ?>
<RDF xmlns:r="http://www.w3.org/TR/RDF/"
     xmlns:d="http://purl.org/dc/elements/1.0/"
     xmlns="http://dmoz.org/rdf">
<Topic r:id="Top/Arts">
  <narrow r:resource="Top/Arts/Movies"/>
  <narrow r:resource="Top/Arts/Classical_Studies"/>
  ...
</Topic>
<Topic r:id="Top/Arts/Movies">
  <narrow r:resource="Top/Arts/Movies/Characters"/>
  <narrow r:resource="Top/Arts/Movies/News_and_Media"/>
  <narrow r:resource="Top/Arts/Movies/Education"/>
  ...
</Topic>
...
</RDF>
```

The main `<RDF>` element contains a list of `<Topic>` subelements, each of which contains zero or more `<narrow>` elements (for reasons of compatibility with the dmoz.org files, `<narrow1>` and `<narrow2>` are also allowed and are treated as synonyms of the plain `<narrow>`) which enumerate the subtopics of that topic. The `r:resource` attribute of each `<narrow>` element must be the identifier of some topic (i.e. equal to the `r:id` attribute of some `<Topic>` element). The topic identifiers must be the same as those that appear in `content.rdf` (although their order may be different).

The output of the software component is simply a number indicating the similarity between the two given ontologies. The assumption is that the `content.rdf` files of both ontologies use the same set of instances (i.e. the same set of strings appears in the `r:resource` attributes of the `<link>` elements of both files). Concept identifiers (values of `r:id` of the `<Topic>` elements) may, however, be completely unrelated from one ontology to the next. The similarity value is printed to the standard output.

5. Future work

The approach presented here in section 3 assumes that we are comparing two ontologies based on the same set of instances (but with different sets of concepts, different assignment of instances to concepts and different arrangement of concepts into a hierarchy). One way to extend this approach would be to allow for comparison of ontologies based on different sets of instances. In this case it is no longer possible to take a pair of instances and observe where they are placed in one ontology and where in the other, because each ontology has its own separate set of instances. Assuming that each instance is represented by a textual document, some kind of matching would need to be introduced. Given two instances o_i and o_j from the ontology U, one might find a few nearest neighbours of o_i and o_j in the ontology V, and observe δ_V on the pairs of these nearest neighbours. However, this would introduce an additional level of time complexity.

Comparison of two ontologies could also be based on the principle of edit distance. In this case one is looking for a sequence of edit operations that can transform one ontology into the other, while minimizing the total cost (e.g. the number of edit operations). However, if the two ontologies have different sets of concepts (and possibly even different sets of instances), it might be difficult to efficiently find a minimum-cost sequence of edit operations. Some efficient algorithms for comparing ordered trees on the edit distance principle are known (see e.g. CHAWATHE *et al.*, 1996), but here we would be dealing with unordered trees.

Another direction that might be promising to explore would be ontology similarity measures based on information-theoretic principles. For example, the variation-of-information metric for comparing two flat partitions of a set of instances (MEILA, 2003) has been shown to have a number of desirable and theoretically appealing characteristics (MEILA, 2005). Essentially this metric treats cluster membership as a random variable; two different partitions of a set of instances are treated as two random variables and the mutual information between them is used as a measure of the similarity of the two partitions. This similarity measure could be extended to hierarchical partitions. It would need to roughly answer a question such as: How many bits of information do we need to convey in order to describe, for each instance, where it belongs in the second hierarchy, if we already know the position of all instances in the first hierarchy? A suitable coding scheme would need to be introduced; e.g. for each concept c of the first hierarchy, find the most similar concept c' in the second hierarchy; then, for each instance o from c, to describe its position in the second hierarchy, list a sequence of steps (up and down the is-a connections in the hierarchy) that leads from c' to the concept that actually contains the instance o.

From a purely algorithmic point of view, it would also be interesting to explore if the ontology similarity measure as currently defined in section 4.3 can be accurately computed in sub-quadratic time (in terms of the number of instances).

5.1 Evaluation without a golden standard

It would also be interesting to try evaluating an ontology "by itself" rather than comparing it to a golden standard. This type of evaluation would be useful in many contexts where a golden standard ontology is not available, which includes some of the SEKT case studies. One possibility is to have a partial golden standard, such as a list of important concepts but not a hierarchy; evaluation could then be based on precision and recall. Another scenario is if a golden standard is not available for our domain of interest but for some other domain, we can use that domain and its golden standard to evaluate/compare different ontology learning algorithms and/or tune their parameters, then use the resulting settings on the actual domain of our interest in the hope that the result will be a reasonable ontology, even though we don't have a golden standard to compare it to.

However, approaches that completely avoid the need for a golden standard could also be considered. In the case of "flat" partitions in traditional clustering, measures such as cluster compactness or inter-cluster distance are often used to evaluate a flat partition: instances from the same cluster should be close to each other, while instances from different clusters should be as far apart as possible. Measures of this sort could also be extended to hierarchical partitions. One could also envision using machine learning methods to evaluate a partition: the partition can be seen as dividing the set of instances into several disjoint classes, and we can try learning a classification model for each class. If the partition of instances into classes was reasonable, one would expect the resulting classifiers to perform better than if the partition was essentially random or unrelated to the attributes of the instances.

6. Relation to SEKT activities

The approaches presented in this report (in particular sections 4 and 5) are shaped in the way to enable a wide range of applications covering different ontology manipulation scenarios. In comparison with the more traditional approaches to ontology management (logic and linguistic based), our approach emphasizes some of the less frequently addressed issues in the semantic web literature such as scalability (size of the data), computational efficiency (e.g. through sampling), and relation to statistical approaches (e.g. from clustering evaluation). The main goal was to design approaches which will enable to solve real life tasks from the SEKT technical and case studies workpackages.

6.1 Relation to SEKT technical workpackages

The ontology definition (section 3) requires each instance to consist of a set of features, which connects to the results of the tasks T1.3 (Dealing with Different Data Types) and T1.4 (Language Issues). The main point of these tasks is to transform various more or less structured data types into a common feature-based representation from which we are able to build conceptual structures (such as ontologies). Data types include social networks, images, movies, sound, uncommon document representations (such as "language independent" document representation), as well as combinations of different data types (e.g. combining text and images). Tasks T1.3 and T1.4 in combination with the ontology evaluation framework developed in this deliverable enable us to evaluate ontologies involving a wide spectrum of object types including combinations of objects (by e.g. using techniques such as KCCA developed under task T1.4). Results of this type will be also used in forthcoming WP1 task T1.11 (Ontology Generation from Social Network Data) where the goal will be to deal with social networks and extract ontological structures based on social behavior from the data.

Task T1.5 (Modeling Human Expertise Based on Existing Ontologies) serves as a prerequisite for ontology evaluation (T1.6) and ontology learning (T1.7) tasks since it extracts invariant properties of the human built ontologies (in our case topic ontologies) which can be further used for tuning-up application specific evaluation functions for a target domain. We deal with some of these questions in the section 5.1 and combination of both approaches will be addressed in the forthcoming (year 3) tasks on ontology learning (task T1.12) where we plan to upgrade results from task T1.7 (Ontology Construction from Scratch).

Software for semiautomatic creation of ontologies built in the task T1.7 builds on the ideas from this report — the software package will in the next release implement several ontology evaluation measures based on the theoretical framework from section 3 to propose and guide the user when constructing the ontology.

This deliverable will serve as an important prerequisite for task T1.9 (Simultaneous Ontologies) where the goal will be to create different ontologies form the same set of instances. The ideas on how to structure the data in different ways to satisfy different application scenarios will stem directly out of sections 3, 4 and 5 of this report.

The fact that we emphasized scalability issues when evaluating ontologies will serve as input for the task T1.13. (Generation of Ontologies from Very Large Databases) where the goal will be to search for large ontological structures in an efficient way and where the evaluation functions will play a crucial role in the optimization procedure.

6.2 Relation to SEKT case studies tasks

SEKT has three case studies which all build on the results of technical workpackages. Since ontology evaluation provides fundamental insight in the quality and properties of the structured knowledge, we expect that all three case studies will benefit from the results of this task.

The BT Digital Library case study has several tasks where the results will be applied by the end of M24 of the project. In particular, we expect to use analytically guided ontology construction (based on the ideas from the sections 4 and 5) for extending topic ontologies for 5 million scientific and technical papers abstracts. The goal is to tune the evaluation function based on how the knowledge is structured directly from the existing topic ontology and to generate new parts of the ontology in the same fashion further on. Automatically tuned evaluation function will be a central part of the procedure for extending the structure.

The legal case study will benefit from the ontology evaluation in several ways. First, the manual ontology built in the case study will get evaluated with the ideas from sections 4 and 5 — the goal is to provide a recommender system for ontology revision based on the documents collected in the other part of this case study (from the Wolters Kluwer database) using software from task T1.7 (Ontology Learning from Scratch) and with adapted evaluation functions. Next, the database of approx. 100,000 legal documents collected within the legal case study (crawled from the Wolters Kluwer web site) will need to be arranged in an ontological structure — for this we will develop an appropriate evaluation function (based on the ideas from the sections 2, 3, 4 and 5) together with human experts which will be used in combination with the tools from the task T1.7 to develop appropriate structure.

Appendix - User Guide

The utility compares two ontologies and outputs a similarity measure in the range [0, 1]. Each ontology is defined in two input files (the structure file and the content file), as described in section 4.5. The following command-line parameters must be provided:

- "-ic1:*FileName*" specifies the (path and) file name of the `content.rdf` file for the first ontology
- "-is1:*FileName*" specifies the (path and) file name of the `structure.rdf` file for the first ontology
- "-ic2:*FileName*" and "-is2:*FileName*" analogously provide file names for the second ontology.

The following parameters control the similarity measure used:

- "-simOverlap" means that the overlap-based definition of δ will be used, as defined by eq. (2).
- "-simExp" means that the definition of δ from eq. (3) will be used. In this case, the "-alpha:*Value*" and "-beta:*Value*" parameters must also be provided to specify the values of the α and β parameters for eq. (3) (both should be nonnegative real numbers).
- "-sample:*Value*" means that, instead of computing the complete sum in eq. (1), i.e. for all pairs of instances, only a randomly selected subset of pairs will be used. The *Value* specifies the number of pairs to use. (See section 4.4.)

The parameter "-quiet" may be used to suppress all output except for the actual value of the similarity measure between the two input ontologies. This can be useful if the program output will be piped into another program.

Example:
```
OntSimilarity.exe -is1:MyStructure.rdf -ic1:MyContent.rdf
                  -is2:DmozStructure.rdf -is2:DmozContent.rdf
                  -simOverlap -sample:100000
```

This reads the two ontologies from the given input file and computes the similarity based on eq. (2) and a random sample of 100000 pairs of instances.

Bibliography and References

1. BLOEHDORN, S., HAASE, P., SURE, Y, VOELKER, J., BEVK, M., BONTCHEVA, K., ROBERTS, I., Report on the integration of ML, HLT and OM. SEKT Deliverable D.6.6.1, July 2005.
2. BREWSTER, C., ALANI, H., DASMAHAPATRA, S., WILKS, Y., Data driven ontology evaluation. Proceedings of Int. Conf. on Language Resources and Evaluation, Lisbon, Portugal, 26–28 May 2004.

3. BURTON-JONES, A., STOREY, V. C., SUGUMARAN, V., AHLUWALIA, P., A semiotic metrics suite for assessing the quality of ontologies. Accepted by *Data and Knowledge Engineering* (2004).
4. CHAWATHE, S. S., RAJARAMAN, A., GARCIA-MOLINA, H., WIDOM, J., Change Detection in Hierarchically Structured Information. *Proc. of the ACM SIGMOD Conference*, pp. 493–504, 1996.
5. DING, L., FININ, T., JOSHI, A., PAN, R., COST, R. S., PENG, Y., REDDIVARI, P., DOSHI, V., SACHS, J., Swoogle: A search and metadata engine for the semantic web. *Proc. 13th ACM Conference on Information and Knowledge Management*, pp. 652–659 (2004).
6. EHRIG, M., HAASE, P., HEFKE, M., STOJANOVIC, N., Similarity for ontologies — a comprehensive framework. *Proc. 13th European Conference on Information Systems*, May 2005.
7. FOX, M. S., BARBUCEANU, M., GRUNINGER, M., LIN, J., An organization ontology for enterprise modelling. In: M. Prietula et al. (eds.), *Simulating organizations: Computational models of institutions and groups*, AAAI/MIT Press, 1998, pp. 131–152.
8. GÓMEZ-PÉREZ, A. Some ideas and examples to evaluate ontologies. Knowledge Systems Laboratory, Stanford University, 1994.
9. GÓMEZ-PÉREZ, A. Towards a framework to verify knowledge sharing technology. *Expert Systems with Applications*, 11(4):519–529 (1996).
10. GUARINO, N., WELTY, C., Evaluating ontological decisions with OntoClean. *Communications of the ACM*, 45(2):61–65, February 2002.
11. HARTMANN, J., SPYNS, P., GIBOIN, A., MAYNARD, D., CUEL, R., SUÁREZ-FIGUEROA, M. C., SURE, Y., Methods for ontology evaluation. KnowledgeWeb (EU-IST Network of Excellence IST-2004-507482 KWEB), Deliverable D1.2.3, January 2005.
12. HAUSSLER, D., Convolution kernels on discrete structures. Technical report, Department of Computer Science, University of California at Santa Cruz, 1999.
13. LOZANO-TELLO, A., GÓMEZ-PÉREZ, A., Ontometric: A method to choose the appropriate ontology. *Journal of Database Management*, 15(2):1–18 (2004).
14. MAEDCHE, A., STAAB, S., Measuring similarity between ontologies. *Proc. 13th Conference on Information and Knowledge Management* (2002). LNAI vol. 2473.
15. MEILA, M., Comparing clusterings by the variation of information. *Proc. of the 16th Annual Conference on Computational Learning Theory*, 2003.
16. MEILA, M., Comparing clusterings — an axiomatic view. *Proc. of the International Conference on Machine Learning*, 2005.
17. MLADENIĆ, D., Machine Learning on non-homogeneous, distributed text data. Ph.D. thesis, University of Ljubljana, 1998.
18. PATEL, C., SUPEKAR, K., LEE, Y., PARK, E. K., OntoKhoj: a semantic web portal for ontology searching, ranking and classification. *Proc. 5th ACM Intl. Workshop on Web Information and Data Management*, New Orleans, LA, USA, pp. 58–61 (2004).
19. PORZEL, R., MALAKA, R., A task-based approach for ontology evaluation. *Proc. ECAI 2004 Workshop on Ontology Learning and Population*, pp. 9–16.
20. RADA, R., MILI, H., BICKNELL, E., BLETTNER, M., Development and application of a metric on semantic nets. IEEE Trans. on Systems, Man, and Cybernetics, 19(1):17–30 (1989).
21. RAND, W. M., Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66:846–850 (1971).

22. SPYNS, P., EvaLexon: Assessing triples mined from texts. Technical Report 09, STAR Lab, Brussels, Belgium, 2005.
23. SUPEKAR, K. A peer-review approach for ontology evaluation. *Proc. 8th International Protégé Conference*, Madrid, Spain, July 18–21, 2005.
24. VELARDI, P., NAVIGLI, R., CUCCHIARELLI, A., NERI, F., Evaluation of OntoLearn, a methodology for automatic learning of domain ontologies. In: P. Buitelaar, P. Cimiano, B. Magnini (eds.), *Ontology Learning from Text: Methods, Evaluation and Applications*, IOS Press, 2005.
25. VÖLKER, J., VRANDECIC, D., SURE, Y., Automatic evaluation of ontologies (AEON). *Proceedings of the 4th International Semantic Web Conference*, 2005.