Bridging the Gap from Text to Knowledge

Selected Contributions in Ontology Learning and Population from Text

Paul Buitelaar, Philipp Cimiano

On the "ontology" in ontology learning

In recent years, the field of ontology learning from text has attracted a lot of attention, resulting in a wide variety of approaches to the extraction of knowledge from textual data. Yet, results so far are still limited as the semantic gap between human language on the one hand and formalized knowledge on the other is significant. Knowledge formalized in the form of ontologies is declarative, explicit and in general monotonic and crisp. Knowledge expressed by human language is highly diluted, very implicit, vague and even defeasible.

As Brewster et al. [1] have correctly argued, when writing an article, authors assume a large body of background knowledge which they share with their community and potential readers, while focusing on a very specific aspect, i.e. on the specific message they want their text to convey. Thus, most of the knowledge in texts is actually very implicit and remains *"under the surface"*. Further, natural language lacks conceptual preciseness, allowing people to have very different conceptualizations and yet use the very same words to express them. In addition, for reasons of economy, people use language in a rather vague and underspecified way and are just precise enough to still allow reasonable communication. Thus, knowledge conveyed by means of language has to be considered as implicit, vague and defeasible in general and is consequently far away from current ontology models assuming knowledge defined declaratively, explicitly as well as in a crisp and monotonic manner.

By definition, an ontology is an explicit specification of a shared conceptualization (see [2] and [3]). In essence, it is thus a view on how the world or a specific domain is structured as agreed upon by the members of a community. Assuming that we have perfect natural language processing tools for extracting knowledge from text, it is still questionable whether we will be able to actually learn an ontology from this data as the conceptualization behind an ontology is typically assumed to be the result of an intentional process. Ontologies therefore cannot be "learned" by machines in the strict sense of the word as they lack intention and purpose. Instead, ontology learning may only support the human ontology engineers in defining their conceptualization of a particular part of the world, e.g. a technical domain, on the basis of empirical evidence derived from textual and other data.

If we adopt such a view of ontology learning, we have to conclude however that a number of important questions in this regard remain largely unanswered by the current literature:

- *textual evidence*: What kind of empirical textual evidence should an ontology engineer actually consider when modelling an ontology?

- *evidence-based agreement*: How can we foster the process of consensus building and agreement by presenting empirical evidence derived from data for different design choices?
- *data-driven ontology engineering*: On a more general note, what should the role of data-driven ontology learning be in the overall process of ontology engineering?
- *methodological integration*: How should ontology learning tools be integrated into a larger framework for ontology engineering from a methodological point of view?
- *user interface*: What is the best way to support an ontology engineer in presenting empirical evidence at the user interface level and what is the optimal way for a user to interact with such a system?

At least four different research communities may contribute to answering these questions: natural language processing, machine learning, knowledge representation/engineering and user interface design. In fact, it seems to us that the above questions can only be addressed through an interdisciplinary research program across these research communities. We will briefly elaborate why.

The natural language processing community has so far applied their best techniques to the problem of ontology learning, mainly for term extraction and for learning paradigmatic relations between terms such as synonymy (see [4] and [5]), hyperonymy (see [6,7]) and meronymy (see [8]). However, these are lexical relations and do not hold between concepts with explicitly defined intensions. Lexical relations do in fact not map straightforwardly to relations between concepts, e.g. A is a subconcept of B iff every A is also a B. In contrast, according to Lyons [9], hypernymy is defined as *"the relation which holds between a more specific, or subordinate, lexeme and a more general, or superordinate, lexeme."*, which is clearly not equivalent to the definition above in terms of subsumption of extension. Typically, hypernym relations are indicated through so called diagnostic frames. In the case of hypernymy, one useful diagnostic frame is *An X is a kind/type of Y"* (see [10]). However, such diagnostic frames clearly lack the necessary preciseness. First of all, they do not distinguish whether the terms are roles (in the sense of OntoClean [11]) or actually concepts. Thus, *student* and *person* can be actually found in such a diagnostic frame *A student is a person who studies*, while the first is clearly a (material) role and the second a type. Similar remarks hold for the meronymy relation. It is well-known in artificial intelligence that there are various types of part-of relations (compare [12]) that can clearly not be differentiated from each other within diagnostic frames. In summary, an important problem seems to be that there is neither a straightforward mapping between terms in language to concepts with a well-defined intension and extension nor can lexical relations be mapped to ontological relations in the general case (see also [13]). NLP research has in many cases ignored such intricate questions in knowledge acquisition and focused instead on learning paradigmatic relations between linguistic objects.

The machine learning community provides a large number of sound techniques for data-driven (inductive) learning but, with a few exceptions, is in fact quite opposed to the idea of learning ontologies. Ontologies are logical theories and declarative by nature. Machine learning is in principle concerned with de-

veloping analytical models that explain data. In its supervised fashion (compare [14]), such models serve prediction purposes, i.e. for classifying novel examples. In unsupervised learning, one aims to discover regularities or patterns in data such as homogeneous groups or clusters (see [15]) or general associations (see for instance [16]). Many techniques from unsupervised machine learning such as clustering and mining associations have been applied to ontology learning. Mädche and Staab have for example used association rules to discover relations between (lexicalizations) of concepts (compare [17]) and Cimiano et al. have used clustering techniques to group and hierarchically arrange words (see [18]). Most of the papers in this volume also apply machine learning techniques in some way, in particular clustering (Brunzel, Poesio et al.), classification (Poesio et al.), memory-based learning (Tanev et al.) as well as induction of patterns from examples (Pantel et al., Alfonseca et al.). However, analytical models as considered in machine learning are generally not declarative in the sense of a logical theory. Some branches of machine learning research have indeed aimed at learning declarative logical theories from data. This is the case for example for Inductive Logic Programming [19]. However, theories learned from data through ILP differ crucially from ontologies. The latter reflect a shared understanding of a domain of interest, produced as the byproduct of reflection and consensus within a certain community and thus represent a commitment to a specific conceptualization. For logical theories derived inductively from data, it seems unclear in how far they can be seen as shared or as really expressing a view. The most promising way of applying inductive techniques seems to be in ontology refinement. First blueprints in this direction can be found in the works of Lisi [20] and Rudolph et al. [21]. In general, it seems to us that an important avenue for future machine learning work in ontology learning is to systematically analyze the question how inductively derived models, classifications, relations etc. can support an ontology engineer to formulate or refine their conceptualization in the form of an ontology, seeing ontology learning always as an interactive and cooperative process between an ontology engineer and a system (see also the definition of ontology learning in [22]).

The knowledge representation community has focused traditionally on methods for efficient reasoning and inference, but to a large extent neglected the following issues: i) integrating insights from linguistics into ontology development (with the exception of some of the work on DOLCE [23]) ii) integrating ontology learning into methodologies for engineering ontologies, iii) integrating knowledge representation and inferencing paradigms which are closer to the way knowledge is expressed in human language (notable exceptions being the work on computing with words of Zadeh [24], the work on natural logic [25] or the conceptual graphs of Sowa [26]). The linguistics community has in fact developed category systems based on linguistic principles that could be integrated into ontologies, for example Vendler's verb categories [27] or the so called *'Aktionsarten'* [28]. Ontologists have largely neglected such distinctions which might be useful exactly in bridging the gap between text and knowledge. While there is some work on integrating machine learning into traditional knowledge acquisition and engineering methodologies such as CommonKADS [29], the integration of ontology learning with more recent ontology engineering methodologies such as On-To-Knowledge [30], DILIGENT [31] or METHONTOLOGY [32]) has not been approached to a satisfactory

extent. A first step in this direction is included in this volume (Paslaru-Bontas et al.) Methodological issues related to the interplay between linguistic analysis and ontology engineering are also addressed in this volume (Aussenac-Gilles et al.)

Finally, the contribution from the user interface community is urgently needed in ontology learning. We have argued above that ontology learning cannot be, by its very nature, fully automatic. On the contrary, ontology engineering is a highly interactive task in which a user interacts with a system that presents empirical textual evidence in support of the human task of modelling a particular domain. Novel user interface paradigms are needed here. First blueprints considering usability aspects can be found in the work of Wang et al. [33] and Missikoff et al. [34]. Unfortunately, we have no contribution on this included in this volume.

In summary, ontology learning research in which the "ontology" is taken serious requires a joint effort of various communities. Through this volume we therefore aim at forging stronger bonds between these by presenting promising research from the different communities in one collection. In this way we hope to have contributed to the development of a more integrated and cross-disciplinary approach to ontology learning. We hope that this book will stimulate further research in the field and encourage researchers to increasingly tackle also the harder challenges in ontology learning as outlined above.

Paul Buitelaar, Philipp Cimiano
Saarbrücken/Karlsruhe, November 2007

References

[1] C. Brewster, F. Ciravegna, and Y. Wilks. Background and foreground knowledge in dynamic ontology construction. In *Proceedings of the SIGIR Semantic Web Workshop*, 2003.

[2] T.R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, 1993.

[3] R. Studer, R. Benjamins, and D. Fensel. Knowledge engineering: Principles and methods. *Data Knowledge Engineering*, 25(1-2):161–197, 1998.

[4] P.D. Turney. Mining the web for synonyms: PMI-IR versus LSA on TOEFL. In *Proceedings of the 12th European Conference on Machine Learning (ECML)*, pages 491 – 502, 2001.

[5] D. Lin. Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (COLING-ACL)*, pages 768–774, 1998.

[6] M.A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING)*, pages 539–545, 1992.

[7] K. Ahmad and H. Fulford. Knowledge processing: Semantic relations and their use in elaborating terminology. Technical report, University of Surrey, 1992.

[8] M. Berland and E. Charniak. Finding parts in very large corpora. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 57–64, 1999.

[9] J. Lyons. *Semantics: Volume 1.* Cambridge University Press, 1977.

[10] D. Cruse. *Lexical Semantics.* Cambridge University Press, 1986.

[11] C.A. Welty and N. Guarino. Supporting ontological analysis of taxonomic relationships. *Data Knowledge Engineering (DKE)*, 39(1):51–74, 2001.

[12] A. Artale, E. Franconi, N. Guarino, and L. Pazzi. Part-whole relations in object-centered systems: An overview. *Data Knowledge Engineering*, 20(3):347–383, 1996.

[13] J. Völker, P. Hitzler, and P. Cimiano. Acquisition of OWL DL axioms from lexical resources. In *Proceedings of the European Semantic Web Conference (ESWC)*, pages 670–685, 2007.
[14] Tom Mitchell. *Machine Learning*. McGraw Hill, 1997.
[15] A.K. Jain and R.C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.
[16] T. Imielinski R. Agrawal and A.N. Swami. Mining association rules between sets of items in large databases. *SIGMOD*, 22(2):207–216, 1993.
[17] A. Mädche and S. Staab. Discovering conceptual relations from text. In *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI)*, pages 321–325, 2000.
[18] P. Cimiano, A. Hotho, and S. Staab. Learning concept hierarchies from text corpora using formal concept analysis. *Journal of Artificial Intelligence Research (JAIR)*, 24:305–339, 2005.
[19] N. Lavrac and S. Dzeroski. *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood, 1994.
[20] F. Lisi and F. Esposito. Two orthogonal biases for choosing the intensions of emerging concepts in ontology refinement. In *Proccedings of the European Conference on Artificial Intelligence (ECAI)*, pages 765–766, 2006.
[21] S. Rudolph, J. Völker, and P. Hitzler. Supporting lexical ontology learning by relational exploration. In *Proceedings of the International Conference on Conceptual Structures (ICCS)*, pages 488–491, 2007.
[22] A. Mädche and S. Staab. *Handbook of Ontologies*, chapter Ontology Learning, pages 173–190. Handbook of Information Systems. Springer, 2004.
[23] C. Masolo, S. Borgo, A. Gangemi, N. Guarino, and A. Oltramari. Ontology library (final). WonderWeb deliverable D18.
[24] L.A. Zadeh. From computing with numbers to computing with words – from manipulation of measurements to manipulation of perceptions. *IEEE Transactions on Circuits and Systems*, 45:105–119, 1999.
[25] L.S. Moss. Natural language, natural logic, natural deduction. Draft Available from the author.
[26] John F. Sowa. *Conceptual Structures: Information Processing in Mind and Machine*. Addison-Wesley, 1984.
[27] Z. Vendler. Verbs and times. *The Philosophical Review*, 66:143–160, 1957.
[28] B. Comrie. *Aspect: Introduction to the Study of Verbal Aspect and Related Problems*. Cambridge University Press, 1976.
[29] W. van de Velde. Machine learning issues in CommonKADS. KADS-II Project Deliverable D2.11, 1992.
[30] Y. Sure, S. Staab, and R. Studer. Methodology for development and employment of ontology-based knowledge management applications. *SIGMOD Record*, 31(4):18–23, 2002.
[31] D. Vrandecic, H.S. Pinto, Y. Sure, and C. Tempich. The DILIGENT knowledge processes. *Journal of Knowledge Management*, 9(5):85–96, 2005.
[32] M. Fernandez-Lopez, A. Gomez-Perez, and N. Juristo. METHONTOLOGY: From ontological art towards ontological engineering. In *Proceedings of the AAAI Spring Symposium on Ontological Engineering*, pages 33–40, 1997.
[33] Y. Wang, J. Völker, and P. Haase. Towards semi-automatic ontology building supported by large-scale knowledge acquisition. In *Proceedings of the AAAI Fall Symposium On Semantic Web for Collaborative Knowledge Acquisition*, pages 70–77, 2006.
[34] M. Missikoff, R. Navigli, and P. Velardi. The usable ontology: An environment for building and assessing a domain ontology. In *Proceedings of the International Semantic Web Conference (ISWC)*, pages 39–53, 2002.

Foreword

Gregory GREFENSTETTE [a]
[a] *Commissariat l'Energie Atomique, CEA LIST, Saclay, France*

Humans can use the internet easily thanks to World Wide Web browsers, which hide operating system details behind intuitive interfaces. In little over a decade, a world of ordinary people have become computer literate, using the internet to communicate, to find information, and for sundry daily tasks, such as buying books and travel. In the same time, billions of web pages have been created. And even with this glut of pages to wade through, people are good at finding and using information for their various needs.

Though we would like to have the computer understand things as we do, navigating the billions of web pages internet without our constant direction, and understanding and seeing matches as we do, this is beyond current computer technology. Computers, even though they house and transmit this information, are not very good at understanding and manipulating it.

The un-intelligence of computers comes from their basic functioning. Computers are good at matching zeros and ones. Their circuits can tell when things are exactly alike. But a computer is not good at looking through variety and detecting similarity. Take, for example, these two sentences which might describe the same event: *Thieves stole a valuable Picasso last night* and *Armed robbers made off with an invaluable Cubist-period painting yesterday.* We humans can recognize the similarity in them, since we know that Picasso produced Cubist paintings, that *thieves/robbers* and *made off/stole* are near synonyms, and that *last night* is part of *yesterday.* But for a computer, there is little in the text that matches. A computer has no model of these basic facts and relations. The computer needs lists of similar things, and the context to know when similarity holds and does not hold. It needs an explicit list of items in a domain, and an explicit list of relations between them. It needs a representation of text in which similar things are expressed in identical language.

The response of the computer scientist has been the idea of building a Semantic Web in which the information appearing on the Web would also contain additional annotation understandable by a machine. To make information understandable to a machine, it must be reduced to a space where similar things look exactly the same, in a *zero* and *one* sense. The variety of expression that humans use (and understand) in the description of something has to be reduced to a canonical set of words and expressions. The fact that a *robber* is a *thief* has to explicitly appear in a hierarchy of words that describes when one thing is a type of another thing. The relations between things in these lists have to be explicitly listed. The computer has to know that *Picasso* was a *painter* and that *painters paint paintings* by looking up these facts in a computer-manipulable data structure.

For the Semantic Web, an *ontology* is a data structure that describes the vocabulary in a certain domain, and that specifies the relation between the elements in that domain. The computer can exploit this structure because all the things that the computer needs to know are explicitly listed, or can be deduced from the links between items. For example, if the ontology contains the explicit relation that *thieves steal* and the explicit link that a *robber* is a type of *thief*, it can deduce that *robbers steal*. The promise of the Semantic Web is that future web pages will be annotated not only with bright colours and fancy fonts as they are now, but with annotation extracted from large domain ontologies that specify, to a computer in a way that it can exploit, what information is contained on the given web page. The presence of this information will allow software agents to examine pages and to make decisions about content as humans are able to do now.

This book describes the state-of-the-art in computer-based ontology construction and evaluation.

The classic method of building an ontology is to gather a committee of experts in the domain to be modelled by the ontology, and to have this committee agree on which concepts cover the domain, on which terms describe which concepts, on what relations exist between each concept and what the possible attributes of each concept are. The chapter written by Simperl et al. outlines the steps in building a functional ontology, with a description of tools available at each stage, and a detailed case study involving automating part of the ontology learning process.

All ontology learning systems begin with an ontology structure, which may just be an empty logical structure, and a collection of texts in the domain to be modelled. If the ontology is not empty, then existing terms and relations in the ontology can be used to create patterns to recognize new terms, and new relations between terms. An ontology learning system can be seen as an interplay between these three things: an existing ontology, a collection of texts, and lexical syntactic patterns, such as *A stole B*. Existing items in the ontology can be used to gather new texts, and to identify lexical patterns between ontology items in the text. Texts can be exploited by identifying new terms that should be added to the ontology, and new relations between existing terms. And new lexical patterns can be found by analyzing new texts, finding where and in what syntactic or lexical context known ontology items occur. With the ontology items, we find new texts. With the text, we find new patterns between existing ontology items. And with the new patterns, we find new ontology items.

One can also start from a simple list of domain terms, not yet structured in an ontology, as shown in the chapter by Navigli and Velardi. They describe how to take a domain glossary, using the Art and Architecture thesaurus as their example, and then to bootstrap an ontological structuring of these terms using part of an existing ontology, using the CIDOC core ontology for cultural heritage information exchange as their model.

If one does not have a term list, but only large document collections from the domain of interest, then term recognition techniques are used to find the entities to be stored in the ontology. A number of chapters in this book deal with recognizing the new elements that should go into a domain ontology (often called ontology population) and finding new relations between them (ontology

learning). Ruiz-Casado et al. demonstrate what can be done if the domain texts have a certain structure already. They treat Wikipedia, an evolving large scale user-modifiable dictionary. Since Wikipedia has a cross-linking structure, and a predictable format for subject identification and classification of articles, they exploit this structure to identify and class entities. They also attack the problem of disambiguating entities and extracting semantic relations between entities. Brunzel also uses document structure, using the HTML markup around words as an indication of what sequences of words should be candidates for ontology terms. Co-occurrence of terms in similar parts of the HTML structure are clues for discovering synonymy and sibling relations between these terms.

Pantel and Pennacchiotti, with their Espresso algorithm, show how new entities can be recognized and added to a knowledge base as defined in an ontology. They describe an iterative processes, starting with a few seed relations like *poodle* is a *dog*, then inducing new lexical syntactic patterns, selecting good patterns, and extracting new terms and relations. They also probe the web with their high-precision but low yield patterns to find more instances of terms and relations than could have been found in their local document collections. Each element in an ontology might also have a list of databases-like attributes, for example, *date-of-birth*, *location*, etc. Two chapters, by Tanev and Magnini and by Poesio and Almuhareb, describe other techniques for finding entity attributes and entity classes by probing the Web, which allow ontologies to be filled out using the largest document collection known. Aussenac-Gilles et al. detail their TERMINAE platform that implements four steps in the ontology building process using term recognition: adding new text to a document collection in a domain, extracting terms and relations, organizing these concepts in the network, and producing a normalized structure that can feed an ontology.

Sometimes one begins with an existing ontology, populated with entities, and one wants to enrich the ontology model automatically, by creating new relations between these entities. Ciaramita et al. demonstrate this approach by taking an existing molecular biology ontology GENIA, and showing how the known concepts, coupled with natural language processing technology and a corpus of text from that domain, can be used to extract new relations between these concepts. They then evaluate the validity of these newly learned relations for the domain.

Ontologies can describe more than just terms, attributes and the relations between terms. A popular language for describing ontologies is the Web Ontology Language OWL. OWL provides a rich logical description language for organizing concepts, such as being able to define disjoint sets, existence relations, and other logical conditions, that allow a computer to reason over the terms and relations found explicitly in an ontology, in order to deduce new information that might not be physically present on a web page, or in its Semantic Web annotations, but which are implicitly present because of the relations contained in the ontology. The chapter by Völker at al. explores the possibilities of learning these more complex logical relations, studying the cases of learning logical OWL representations of defining sentences (such as *Enzymes are proteins that catalyse chemical reactions*) and of learning disjoint classes of ontology objects.

All of the above tools and methods for learning ontologies can be evaluated. Dellschaft and Staab cover the range of options for evaluating the quality of the

techniques discussed in this book. Maynard et al. discuss evaluation, especially of entity discovery in ontology learning. They introduce a new measure, the Balanced Distance Metric, which measures the accuracy of placing a new term in an existing ontology hierarchy of terms, and show that it captures human intuition of quality better than existing measures.

Millions of human beings have contributed to making the World Wide Web a new and accessible repository of human knowledge. The promise of the Semantic Web is to make this information accessible and automatically exploitable by computers, too. The Semantic Web will only be a reality if we can create structured, unambiguous ontologies that model domain knowledge that computers can handle. The creation of vast arrays of such ontologies, to be used to mark-up web pages for the Semantic Web, can only be accomplished by computer tools that can extract and build large parts of these ontologies automatically. This book provides the state-of-art of many automatic extraction and modelling techniques for ontology building. The maturation of these techniques will lead to the creation of the Semantic Web.

Part I: Extracting Terms and Synonyms

The XTREEM Methods for Ontology Learning from Web Documents

Marko Brunzel
DFKI GmbH (German Research Centre for Artificial Intelligence GmbH)
Trippstadter Strasse 122, 67663 Kaiserslautern, Germany
marko.brunzel@dfki.de

Abstract. Ontology Learning is up to now dominated by techniques which use text as input. There are only few methods which use a different data source. The techniques which use highly structured data as input have the disadvantage that such data sources are rare. On the other side, there are enormous amounts of Web content present today.

We present the XTREEM (Xhtml TREE Mining) methods which enable Ontology Learning from Web Documents. Those methods rely on the semi-structure of Web Documents. The added value of Web document markup is exploited by the XTREEM methods. We show methods for the acquisition of terms, synonyms and semantic relations.

The XTREEM techniques are based on the structure of Web documents; they are domain and language independent. There is no need for NLP software nor for training. They do not rely on domain or document collection specific resources or background knowledge, such as patterns, rules or other heuristics; nor do they rely on manually assembling a document collection.

1. Introduction

In this chapter we present the XTREEM (Xhtml TREE Mining) methods for *Ontology Learning from Web Documents*. The XTREEM methods tackle different layers of the *Ontology Learning Layer Cake* [1,2]. In particular we show techniques for obtaining (1) terms, (2) synonyms and (3) semantic relations.

(1) Terms: XTREEM-T (XTREEM for Terms) finds domain specific vocabularies. XTREEM-T is able to find single word and multiword terms by means of a frequency statistic. The domain focus is as for all XTREEM approaches given by a simple query for which a Web document collection is obtained. We will show a manual evaluation on the results of XTREEM-T.

(2) Synonyms: XTREEM-S (XTREEM for Synonyms) aims to find promising candidates for synonymy. This is done by a standard procedure for synonym acquisition upon the innovative *Group-By-Path* data sets. The Group-By-Path methods which is described in section 3 provides a way to access Web documents which is different to traditional document processing. We will test weather the GBP data is helpful on synonym detection. We will perform an evaluation against a gold standard synonym reference.

(3) Semantic Relations: Almost all existing approaches for the acquisition of semantic relations focused on direct hierarchical relations. Those *sub-ordination relations*

exists between super-concepts and their sub-concepts. Between the sub-concepts of a common super-concepts also a *co-ordination relation* exists. Those *sibling relations* can be learned with statistics and data mining upon Group-By-Path data sets. XTREEM-SG, XTREEM-SP, XTREEM-SC and XTREEM-SA are methods which aim to structure terms on sibling relations.

Those methods are relying on the *markup* which is present in Web documents. Since the structure of Web documents is not bound to particular domains and languages the XTREEM methods are subsequently *domain and language independent*. In contrast to many other methods, they are working with a vocabulary of multiword terms.

In the following section we will present related work. Then we describe the Group-By-Path method. That method is the core operation within other XTREEM approaches described in the later sections. Then we will we provide explanations and experimental results of the XTREEM-T method in section 4, for XTREEM-S in section 5 as well as for the XTREEM approaches for obtaining sibling relations in section 6.

2. Related Work

First we present related work with respect to Ontology Learning from Web documents.Then we present related work regarding the usage of Web document structure. Then related work for the particular XTREEM approaches is presented.

Most of the existing ontology learning methods are used to perform ontology learning from text. For a overview on this topic see [1]. Even if those methods are processing semi-structured data, e.g., HTML Web documents, they are removing the markup and process plain text.

On the other side, there are those methods [3,4] where highly structured data such as dictionaries, database schema or UML schema are used as input. Those approaches have the disadvantage, that such highly structured data is rare.

A rather recent subfield of ontology learning uses semi-structured documents as input data [5]. There the added value, which is given by nowadays omnipresent semi-structured documents in HTML and XML standard, is incorporated. Kruschwitz [6,7] used the markup sections of Web documents to learn a domain model. The method of Kruschwitz uses the markup, but not the paths. In [8] list itemizations of HTML documents are used for acquiring hyponymy relations.

Besides the amount of structure which can be observed on the input data on, one can also identify subfields of ontology learning which are of relevance to our work as they use Web data[9,10].

2.1. Methods that use XML Structure

The idea of using structural similarities [11,12], including *path structures of XHTML/XML documents*, is used for several goals, such as clustering documents on structural similarities [13,14,15]. In contrast, we use the path information to infer siblings. The constitution of the paths is not used itself; no comparison with paths from other documents is performed with our approach.

2.2. Related Work on Terminology Acquisition

For an overview on terminology acquisition see [16,17,18]. With approaches relying on syntactic chunks (e.g., parser generated), XTREEM-T shares "finding boundaries on term expressions"; with n-gram based statistical approaches XTREEM-T shares the incorporation of large amounts of documents.

Compared to the 90's, when many of the terminology acquisition systems have been developed, now there is a big amount of (manually) marked-up Web content available. Nowadays there are mainly activities on term acquisition approaches in the biomedical domain [19,20], where special, high quality text corpora are used. But those approaches designed for rather pure text are not general applicable. The approaches, designed for high quality text, are likely to struggle with the noise present in Web documents, since conversion can not be expected to be perfect; Web documents navigational elements make the task even more different from pure text methods. The conversion from semi-structured text to pure plain text also eliminates information. This information we regard as valuable for term acquisition and not only for interpretation by the browser rendering the Web content. In [7] the markup of Web documents is used to learn a domain model, therefore also boundaries created by Web document structure are used, but not for the purpose of term acquisition.

In the field of ontology learning there are the approaches of [21,22,23,24,25] which tackle the terminology acquisition step seriously.

2.3. Related Work on Synonymy Detection

The detection of information about synonyms (synonym detection / synonym mining) deals with the task of finding words which are interchangeable. For this purpose the distributional hypothesis of Harris [26] is used, e.g, [27,28,29,30]. From Dorow [31] we have obtained the revised hypothesis that for terms to synonyms, they are only required to have similar contexts. LSA [32] was successfully applied for finding synonyms according to the TOEFL test [33].

2.4. Related Work on Detection of Sibling Relations

Upon plain text, *Hearst patterns* [34] are used to find relations among terms in text collections by means of matching several patterns. Also *co-hyponym relations* can be found with this approach. However the disadvantage is that such patterns are rare, the coverage is low, even on big document collections. Cimiano et al. also discover (co-)hyponym relations by finding and analyzing examples of Hearst patterns on the WWW [35,36]. Cimiano et al. did not exploit the markup of Web documents, which are treated like regular plain text.

The acquisition of co-hyponym semantics from text with association measures is ,e.g., performed by [37], but there the document structure is not used.

Now we present our own prior work. In [38] XTREEEM, an initial version of an approach using the Group-By-Path approach was introduced. There K-Means clustering was used to obtain sibling groups. This approach worked on an open vocabulary, the evaluation was only exemplary by human inspection. In [39] the Group-By-Path method was described formally and an evaluation against gold standards was conducted. There

clustering was used for processing. This method was called XTREEM-SG (XTREEM for Sibling Groups) since this methods aimed to obtain sets of sibling terms. In [40] additional additional term clustering is performed.

In [41], we presented XTREEM-SP (XTREEM for Sibling Pairs). XTREEM-SP is a method which focused on finding pairs of strongly related siblings by means of statistical association measures. A comprehensive *overview on associations measures* is given by Evert in [42]. In particular we use χ^2*-association*. Using χ^2-association for finding collocations is mentioned in [43], though its application in computational linguistics goes far back in time. Maedche [44,45] also incorporated association calculation, but he used the traditional Bag of Words vector space model, the structure of Web documents was not incorporated, he did not address siblings relations. In [46] we additionally applied frequent itemset mining to find n-ary sibling associations.

3. Web Content and the Group-By-Path Operation

The XTREEM methods are based on the markup that is present in almost all Web documents. Web documents are usually coded in the HTML standard. Authors use different nested tags to structure pieces of information in Web documents, as shown in table 1.

Now we describe the Group-By-Path operation [39] which is the core operation for all other XTREEM methods (with exception of XTREEM-T). Specifically we use the following definitions.

Table 1. Semantically related terms, located in different paragraphs or separated by other terms

Headings, located in different paragraphs	Highlighted keywords, separated by normal text
… WordNet Was developed … Germanet Analogous … …	… … there are different important standards for building the Semantic Web. … is RDF. … RDFS adds … whereas OWL is … …

Definition 1 (Web document) *A* Web document *or* Web page d *is a semi-structured document following the W3C XHTML standard*[1].

XHTML is an XML dialect, wherein the former HTML standard has been adopted to meet the XML requirements. Traditional legacy HTML documents are converted to XHTML documents, as is also done by all popular Web browsers. Hence, an XHTML document can be seen as a tree, text is represented by leaf nodes and the intermediate nodes are markup elements. We use the term text span to denote the textual contents, the character data sequences of XML elements. The XML elements formed by the tags we will denote as markup elements or tags.

[1] http://www.w3.org/TR/xhtml1/

```
<html>
<html><head>
<html><head>…
<html></head>
<html><body>
<html><body><h1>Lexical Resources …</h1>
<html><body><p>…</p>
<html><body><h2>WordNet</h2>
<html><body><p>Was developed …</p>
<html><body><h2>Germanet</h2>
<html><body><p>Analogous to WordNet for the English …</p>
<html><body>…
<html></body>
</html>
```

Figure 1. A XHTML Document viewed as a collection of Tag Path - Text-Span Pairs

Definition 2 (Tag Path) *Let M be the set of tags supported by XHTML and let d be a Web document according to Def. 1. A* tag path *p in d is a sequence of tags leading from the root tag element of d to a text span in d, i.e., p has the form $p =< m_1, m_2, \ldots, m_v >$, where $m_i \in M (i = 1, \ldots, v)$.*

We use the notation (p, e) to indicate that e is the text span to which p leads.

By this definition, p is a branch of an XHTML tree; for each m_i, m_{i+1} $(i = 1, \ldots, v-1)$ it holds that m_i is the tag surrounding m_{i+1}. A tag path is therefore a special kind of XPath[2] expression. Moreover, a document d is a collection of pairs of the form (p, e), where p is a tag path and e is the text span to which p leads. For example, consider the document in figure 1: In line 8, we see the tag path `<html><body><h2>` leading to the text span `Wordnet`.

Let $B = \{e_1, \ldots, e_r\}$ be a set of text spans. For one document several B can be found by the following Group-By-Path algorithm (Algorithm 1). This is different to traditional "text treatment", where for one text unit (e.g., document, paragraph or sentence) a corresponding "Bag of Words" is obtained. Here a B is obtained for each distinct path p of a document d. Let $A = \{B_1, \ldots, B_t\}$ be the collection which contains the sets of text spans. The following algorithm reflects the way A is obtained from d. This grouping operation is the core of the XTREEM procedures.

[2] http://www.w3.org/TR/xpath/

Algorithm 1 The Group-By-Path Algorithm [39] on a XHTML Web Document

Input: Web document d

Output: Collection A of sets of text spans $B_i = \{e_1, \ldots, e_t\}$

1: extract from d the set $Y = Y(D)$ of (p, e)-pairs, where p is a tag path according to *Definition 2:* and e is its target text span)
2: $A = \emptyset$
3: let Z be the set of tag paths in Y
4: **for all** $p \in Z$ **do**
5: set $B = \{e|(p, e)inY\}$
6: insert B to A
7: **end for**
return A

This algorithm creates all tag paths which lead to an text span (XML text element) in the Web document tree. Then those text spans are grouped together, which share a common tag path.

Now d is represented as a collection of text span sets. This operation is used in the methods described in section 5 and section 6.

Summary: The Group-By-Path approach performs a transition of a Web document from a tree, to a collection of "tag path" / "text span" pairs to a collection of text span sets.

4. Acquisition of Topic specific Vocabulary with XTREEM-T

The core of our approach is in the exploitation of Web document structure to find sibling relations. This procedure is described in section 6. To find those sibling relations many Web documents are processed. Then we investigate if those Web documents can also be used to find terms which are afterwards to be structured by sibling relations.

Up-to-date domain specific vocabularies are valuable resources; e.g., for the application fields ontology learning or text-mining. Usually such domain specific resources are not available; manually crafting the vocabulary without support of suited tools is infeasible. Though the acquisition of terms is the basic layer in ontology learning, it is important since the subsequent layers rely on the vocabulary and it is therefore desirable to obtain also valuable multiword terms from the beginning.

Automatically acquiring a vocabulary is the subject of term extraction, also referred to as term acquisition. Terms constituted by more than one word are also called multiword terms. The vocabulary to be acquired should include single word and multiword term expressions. The vocabulary we want to extract should not necessarily be limited to noun constructs also verbs and adjectives can be important for the lexical layer of an ontology.

Terminology acquisition, relying on statistics of large document collections, is enabled by the fact that large amounts of textual content are readily available. Within Web documents some "sequences of text" occur frequently "marked-up" by tags. We will show, that ordering the "sequences of text" according to their frequency in the collection separates promising term candidates from ordinary text spans. This makes this straight-

forward approach transparent. There are no hidden triggers which prevent the practical applicability for other domains and languages.

The terms obtained from Web documents are expected to be especially suited in the context of learning ontologies for the Semantic Web, since the concepts/terms from such an ontology should label Web content again. Shared markup can be used to facilitate shared conceptualizations.

There are many methods from the field of terminology acquisition tackling the objective of extracting term expressions. The adoption of existing methods to a new domain is laborious. This drawback exacerbated the broad incorporation in application areas like ontology learning and text mining. In contrast, our XTREEM-T approach does not rely on rules and background knowledge. XTREEM-T is domain and language neutral and operates on easily obtainable Web documents.

The availability of a corpus is normally a prerequisite. For some domains it is possible to get big document collections (e.g. Medline), for other topics assembling a feasible document collections is a problem on its own. Since the Web is a big source of content for nearly all topics one can think of, using the Web seems to be an alternative, especially when the document collection is not itself of interest, as it is sometimes the case for ontology learning.

4.1. Foundations of XTREEM-T

Term extraction is based on finding boundaries which separate promising candidates from not relevant sequences of tokens. Our approach uses the boundaries available in Web documents. Those boundaries are mostly manually created by millions of Web content authors. These boundaries are explicit through the markup in semi-structured Web documents. Though the markup is usually not created to make term boundaries explicit, large amounts of such markup boundaries can be helpful for terminology acquisition.

4.1.1. Term Boundaries from Web Content

Web content marked up with HTML tags contains textual data such has ``Here is <EM>marked up text</EM>''. The angle bracket limited tags, enclose sequences of text. Table 1 shows a exemplary fractions of (X)HTML Web content, figure 2 depicts a list of text spans obtained from the content of the second column of table 1 by chunking on the markup tags. When considering bigger amounts of Web content treated in this manner, we postulate that promising term expressions are more frequent and can therefore be obtained from a frequency statistic. E.g., only the text spans in line 2, 4, 5 and 7 are likely to become frequent within amounts of other text spans from a Web document collection.

4.2. The XTREEM-T Procedure

The data flow diagram depicted in figure 3 gives an overview of the XTREEM-T approach for obtaining a domain specific vocabulary (including multiword terms) from Web document collections.

In the following the individual steps of the XTREEM-T procedure are described.

1:	`... there are different important standards for building the`
2:	`Semantic Web`
3:	`. ... is`
4:	`RDF`
5:	`RDFS`
6:	`adds... whereas`
7:	`OWL`
8:	`is...`

Figure 2. List of Text-Spans derived from the XHTML code of the right Column of Table 1

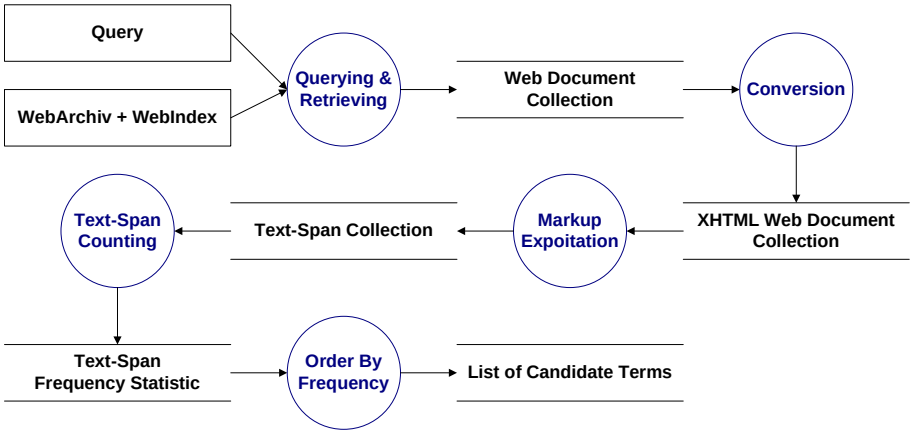

Figure 3. Flow Diagram for XTREEM-T - Vocabulary Detection from Web documents

4.2.1. Step 1 - Querying & Retrieving:

The XTREEM procedure operates on medium size (thousands of documents) and large size (millions of documents) Web document collections. Such a Web document collection is obtained by querying an Archive+Index facility on a query. The Archive+Index facility is a large collection of Web documents, obtained by Web crawling, whereupon a index is created. The query constitutes the domain of interest whereupon semantics should be discovered. It should therefore encircle the documents which are supposed to entail domain relevant content, e.g., ``tourism*'' or ``myocardial infarction''.

The Web document collection should be big enough to contain manifold occurrences of the desired concepts. This is not supposed to be a small manually handcrafted document collection; bigger amounts of Web content which have an appropriate coverage of the domain are more desirable. Here, recall is more important than precision. To obtain such a comprehensive Web document collection, alternatively a focused Web crawl can be performed. The Archive+Index Facility can be easily replaced by obtaining Web document references from the public search engine API's of the major Web search engines.

4.2.2. Step 2 - Conversion:

Most of the present Web documents do not yet fully comply with the proposed W3C standards. Non conformant documents are converted to XHTML documents as is done

by all popular Web browsers. This step is not primarily necessary for the objective of finding term expressions, but it ensures consistency, for every opening tag a corresponding closing tag is present, and eases the handling of Web documents in the next step.

4.2.3. Step 3 - Markup Exploitation:

The Web document can be interpreted as a collection of alternating sequences of textual data and markup sections (tags). We will refer to the textual sequences of text which are not markup as text spans. Only those text spans are of further interest for XTREEM-T. Multiple occurrences of whitespace in text spans are normalized to a single whitespace character; leading and trailing whitespace of text spans is removed. Additionally one may perform a cleaning of the text spans. For our experiments we only used alphabetic characters to eliminate punctuation and numbers, all characters have been converted to lower case. The text spans are constituted by different numbers of tokens and can also be seen as word n-grams of variable length.

4.2.4. Step 4 - Text-Span Counting:

For the text spans, obtained from the Web documents in Step 3, a frequency statistics is created. This frequency statistics contains text spans constituted by different numbers of whitespace separated tokens mixed together. For practical settings it is feasible to limit the length of the text spans to 5 to 10 tokens.

4.2.5. Step 5 - Order By Frequency:

From the text span frequency statistics a list of candidate term expressions is generated by ordering text spans according to their frequency within the Web document collection. According to our hypothesis, the top ranked text spans are term expressions. No stop word removal is required.

4.3. Experiments and Evaluation for XTREEM-T

The term acquisition research field lacks an agreed evaluation vocabulary. Even if such an evaluation vocabulary would be available, it would likely be bound to a specific document collection. For large Web document collection is also not feasible to manually assemble a gold standard which can then be used as a reference. As for other Web data driven approaches, a evaluation as it is usually expected for traditional corpus driven processing is not possible. E.g., though the retrieval quality of the major internet search engines is quite different, this difference was not measured quantitatively. This is due to several circumstances of Web data driven approaches. The amount of data is enormous, and it is not even clear what one would regard as a useful result. Despite that the quality can not be measured exactly, human users experience significant differences.

Because of the lacking gold standard vocabulary and the problems of evaluating Web data driven approaches, we will perform an exemplary manual evaluation on samples of term candidates. For our experiments we will vary:

- the domain of the vocabulary
- the size of the document collection
- the rank-range where evaluation is performed

4.3.1. Experimental Settings

Table 2 gives an overview of the experimental settings used for the evaluation. For all queries the XTREEM-T procedure was run. In the following we will refer to the 5 settings with Q1 to Q5. A "setting" is hereby created by a query which results in a Web document collection.

The rather small Web document collections resulting from Q1 and Q3 are obtained by querying the Web service API of Google. The other document collections, retrieved by the queries Q2, Q4 and Q5, are obtained by performing large domain focused Web crawls, ranging from several hundred thousands (Q4) to 10 million documents (Q2, Q5).

The processing was limited to term expressions of up to a length of 4 tokens (unigrams, bigrams, trigrams and quadgrams) since this is usually the upper bound used for term acquisition. For inspection, subsets of the most frequent text spans have been selected as term candidates. The human expert judged, whether the term candidate can be regarded as a valid expression - in the context of the corresponding domain.

For all settings Q1 to Q5 the top 1000 most frequent text spans have been evaluated. Additionally for Q5 also the text spans with rank 10001 to 11000 and 50001 to 51000 have been inspected to investigate the decrease of quality on low ranks.

Table 2. Domains reflected by query phrases and the resulting number of Web documents used for the experiments

Query Name	Domain	Query Phrase	Number of Documents
Q1	Ontology Ontologies Semantic Web	ontology OR ontologies OR "semantic Web"[3]	3,974
Q2	Ontology Ontologies	"ontolog*"	272,588
Q3	Myocardial Infarction	"acute myocardial syndrome" OR "myocardial syndrome" OR "acute myocardial" OR "myocardial infarction" OR "acute myocardial infarction"	1,037
Q4	Myocardial Infarction	"myocardial*"	42,768
Q5	Tourism	"accommodation"	1,612,108

4.3.2. Evaluation Criteria

The *precision* is the relative number of accepted candidates to the overall number of candidates evaluated.

$$precision = \frac{\#accepted}{\#accepted + \#rejected}$$

If the human expert was in doubt about a term candidate, it was rather regarded as rejected to be on the safe side.

4.3.3. Experimental Results

In figure 4 we see a exemplary list of obtained term candidates. Table 3 shows the precision obtained in the evaluation. The first 5 rows show the obtained results on the top 1000

most frequent text spans, evaluated whether they are domain relevant term expressions or not. The best results are obtained for Q5 with a precision of 77%. This is also the largest document collection. The high precision on the Web document collection from the tourism domain can be explained by the fact that many of the accepted terms are valid geographic expressions such as `new_zealand`, `venice` or `sunshine_coast`. Whether to regard such candidates as good or bad is an open issue, depending on the task. The worst results stem from Q2 with a precision of 40%. These lower results can be explained by the fact that the keyword, which constituted the document collection, is polysemous: there are a couple of terms belonging to "ontology in philosophy" such as `martin_heidegger` and `philosophy_of_mind` and not to "ontology in computer science". This shows the influence of assembling a document collection. Focusing search results by eliminating not wanted senses can be relatively easily done by adopting the query which constitutes the domain Web document collection.

Then we also evaluated lower rank regions of frequent text spans for Q5. There the precession values are lower than for the top 1000 most frequent text spans, but still reasonable good. The still high number of term expressions regarded as relevant is indicatory of the following not astonishing finding. Without further domain restrictions the vocabulary of the tourism domain (given by the query phrase "accommodation") is rather large. A vocabulary for the tourism domain, where also many proper names can be found is likely to consist of many thousands or even hundred thousands of terms. This is also the reason why an evaluation against the vocabulary of known tourism gold standard ontologies is not feasible since most of the acquired term candidates are not within the gold standard though they are valid domain relevant term expressions.

When looking at the results for multiword term expressions separately (numbers in parenthesis of table 3), it can bee seen that also multiword terms are captured with reasonable quality. For the lower evaluated rank regions, the results for multiword terms are even above those for unigrams.

Table 3. Evaluation results for term candidates, in parenthesis the results for multiword terms are shown additionally isolated

Query Name	Order Criterion	Evaluated Rank Region	Accepted	Rejected	Precision
Q1	frequency	1-1000	512 (148)	488 (165)	51% (47%)
Q2	frequency	1-1000	396 (60)	604 (223)	40% (21%)
Q3	frequency	1-1000	522 (214)	478 (277)	52% (44%)
Q4	frequency	1-1000	530 (197)	470 (256)	53% (43%)
Q5	frequency	1-1000	793 (240)	207 (93)	79% (72%)
Q5	frequency	10001-11000	619 (485)	381 (224)	62% (68%)
Q5	frequency	50001-51000	522 (497)	478 (300)	52% (62%)

4.4. Conclusion for XTREEM-T

In all performed experiments on term acquisition with XTREEM-T can be said: that approximately at least half of the candidates are regarded as relevant term expressions.

... , software, conferences, index, daml_oil, phone,
site_map, registration, tutorials, table_of_contents,
figure, about_us, help, conclusion, call_for_papers,
services, artificial_intelligence, program, at, university,
main, see, project, education, java, am,
ieee_intelligent_systems, pm, topic_maps, more, price,
pages, see_also, archives, background, privacy_policy,
download_now, feedback, tools, ontoWeb, iswc, applications,
availability, daml, uml, trackback, summary, technology,
information_retrieval, knowledge_representation,
dublin_core, books, platforms, ...

Figure 4. Exemplary list of obtained Term Expressions; Rank 80 to Rank 132 of Candidate Term Expressions for Document Collection 1

5. Acquisition of Synonyms with XTREEM-S

Synonyms are words with the same meaning or very similar meaning like *car* and *automobile*. Synonym words should be interchangeable in certain contexts. The acquisition of information about synonymy of terms is an important task in the field of lexical knowledge acquisition.

The hypothesis is that good synonym candidates to be synonyms are words which nearly never occur together but which have a very similar context. E.g., `car` occurs often together with `bike` and `bus`. `automobile` occurs often together with `bike` and `bus`. But `car` and `automobile` occur together rather seldom.

In [31], it was shown that this thesis does not hold true regularly. This is for certain circumstances. One reason is hat author's change between synonyms in a narrow distance of textual context window. It is also not uncommon that two synonymous words are used in close coordination, e.g., "Consequence in the form of `penalty` and `punishment` is the subject of the next chapter." (Example taken from [31]). Dorow revised the thesis that synonymous words do not occur together (*first order association*), to only require synonymous words to have a similar context (*second order association*). This was done on regular consecutive plain text. Since our GBP [39] approach enables to obtain sets of terms which have a different "constitution bias" than traditional text treatment (e.g., Bag of Words (BOW), the vector space model), we want to examine whether this data type yields suitable results by means of the updated hypothesis that synonymous words have similar contexts (second order association).

We will introduce XTREEM-S (XTREEM for Synonyms) an approach for obtaining information about synonymy between terms of a given vocabulary. XTREEM-S uses a standard procedure for calculation of second order association on a data set based on the GBP algorithm [39] described in section 3. In an experiment we will demonstrate that this approach is able to perform well in finding terms which are good candidates as synonyms compared to the traditional Bag of Words vector space model.

5.1. The XTREEM-S procedure

For finding synonyms by means of statistical processing the hypothesis of Dorow [31] is that candidates for synonyms are terms which occur together with a similar context

(of terms). The overall procedure for XTREEM-S is shown in the data flow diagram of figure 6.

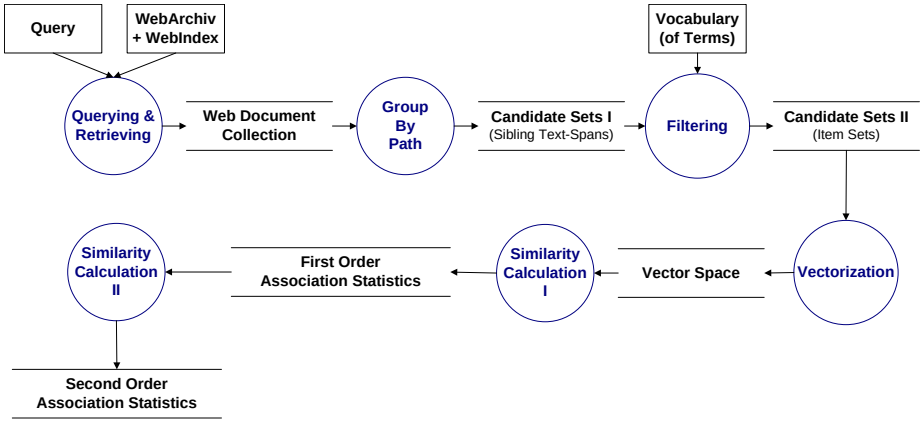

Figure 5. Data Flow Diagram for XTREEM-S: Obtaining a Association Statistics for Synonymy Detection

5.1.1. Step 1 - Querying & Retrieving:

The first step is the same as those of XTREEM-T, described in section 4.2.1. For a given query, a Web document collection is obtained. For our evaluation, the query "`tourism OR tourist''` was used to obtain a Web document collection of the tourism domain. This query resulted in a document collection of 1,468,324 documents.

5.1.2. Step 2 - Group-By-Path:

For each document the Group-By-Path algorithm [39], described in section 3 is applied. As result we obtain the collection of 13,177,526 text-span sets.

5.1.3. Step 3 - Filtering:

For the following steps we only consider all text-spans which are contained in a given vocabulary. Further we are only processing text span sets with an cardinality of at least two, other wise no co-occurrence can be observed at all.

As the vocabulary to be processed we took the terms of two tourism gold standard ontologies[4][5]. Both ontologies did not contain synonyms; therefore we additionally added the synonyms obtained for the basic vocabulary from Wordnet. This was done to ensure, that for the terms of the vocabulary a synonym relation exists for most of the terms. The enhanced vocabulary consists of 1786 terms. The synonyms which have been obtained from Wordnet are also used for evaluation as *gold standard synonym reference*.

From the 13,177,526 text-span sets found in the Group-By-Path step, 864,431 sibling term sets, which are constituted by at least two terms of the vocabulary, have been obtained.

[4] http://www.aifb.uni-karlsruhe.de/WBS/pci/TourismGoldStandard.isa
[5] http://www.aifb.uni-karlsruhe.de/WBS/pci/getess_tourism_annotation.daml

5.1.4. Step 4 - Vectorization:

The text-span sets obtained in the previous step are then spanned over the feature space. The feature space is given by the input vocabulary. For each term of the input vocabulary there is a corresponding vector spanned over all contexts (tag paths).

	Wordnet	Germanet	Euroword net	Semantic Web	…
DocumentA<html><body><h2>	1	1	0	0	…
DocumentB<html><body><table><h1>	1	0	1	0	…
DocumentC<html><body><p>…	0	0	0	1	…
…	…	…	…	…	…

Figure 6. Exemplary Fragment of a Vectorization

5.1.5. Step 5 - Association Calculation I:

For each pair wise combination of terms, the corresponding similarity is calculated by similarity function S_1. With S_1 the similarity of the corresponding context vectors is calculated. As a result, one obtains a symmetric matrix A_1 where the pair wise term similarities are stored. This matrix A_1 represents the *first order association* among terms. Similarity is given by joint occurrence within the same contexts.

As similarity function of our evaluation we used the cosine similarity.

5.1.6. Step 6 - Association Calculation II:

The matrix obtained by step 5 can also be treated as a list of vectors. For every term, there is a vector with the first order association to all other terms. Now, for each combination of first order association vectors, the similarity is calculated again by similarity function S_2. The result is then stored in a matrix A_2. This matrix A_2 represents the *second order association* among terms. Similarity is given by similar co-occurrence profiles (first order association vectors).

As similarity function of our evaluation for second order association calculation we used again cosine similarity.

5.2. Experiments and Evaluation for XTREEM-S

5.2.1. Evaluation Methodology

For the evaluation of synonymy detection in computational linguistics, there is the recent trend to use the TOEFL task. In this task, one has to select a synonymous word out of a given sets of 5 terms. For the evaluation of synonymy detection in the context of ontology learning, this is an unrealistic scenario, since for ontology learning usually a mid size vocabulary of several hundreds terms is to be processed. Choosing synonym candidates out of thousand candidate terms is much harder than to choose a synonym candidate out of 5. Instead we will perform an evaluation where no restriction is undertaken. This is a

much harder task, but this fits better to the context of ontology learning where no prior restriction can be expected in real world settings.

We will perform a gold standard evaluation. As reference, the synonym relations from Wordnet have been used. The synonym groups (synsets) obtained from Wordnet have been transformed into a collection of term pairs which stand in a synonym relation. The evaluation has a bias against the automatically generated results since Wordnet (the reference) also contains synonyms for other domains than the used tourism domain.

Object of the evaluation is a ranked list of automatically obtained concepts pairs, whereas the ranking is given according to the second order association strength of the term pairs. For each automatically obtained terms pair, it can be determined if this relation is also supported by the reference which gives a positive count. If a term pair is not supported by the reference a negative count is assumed. With this, for each position in the ranked list, recall and precision can be calculated.

5.2.2. Evaluation Criteria:

The recall is the number of already seen true synonym pairs ($\#positive$) to the number of synonym pairs given by the reference ($\#overall$). The precision is the number of true synonym pairs ($\#positive$) to the number of seen automatically generated pairs ($\#positive + \#negative$).

$$recall = \frac{\#positive}{\#overall}$$

$$precision = \frac{\#positive}{\#positive + \#negative}$$

For a ranked list of associated term pairs a chart line can be obtained by a series of measurements on precision values for several numbers of seen candidate term pairs. We evaluated the top-N candidate term pairs for several N. N increases from the left to the right (N=10, 20, 30, . . . , 100, 200, 300, . . . , 1000, 2000, 3000,. . . , 10000) .

5.2.3. Evaluation Results

In figure 7 the results of our experiment are shown. The GBP approach is contrasted to the traditional Bag of Words approach. GBP performs better than BOW. For the alternative BOW method, there are no synonyms at all within the top 100 evaluated candidate term pairs. For higher numbers of evaluated term pairs the lines approach to each other on low level. For GBP there is a region ranging up to the first top 400 synonym candidates, where a precision of over 10 percent can be achieved. This means: there is a rather small number of synonyms which can be found with an acceptable precision. Compared to the overall number of synonyms which are supposed to be present according to the reference, this is quit a small fraction. The corresponding recall values are low, e.g., while observing 400 term pairs, with a precision of 10.97 percent, the recall is only 3.09 percent. To obtain a recall of 9.43 percent, 10000 candidate pairs have to be inspected; the precision is then only 1.33 percent, which is practically unacceptable. For the traditional BOW approach the results are even worse.

The recall which was achieved with an acceptable precision is too low to present it in a diagram. This is a circumstance which is common to automatic approaches for findings synonyms; they can only find a small fraction of synonym relations.

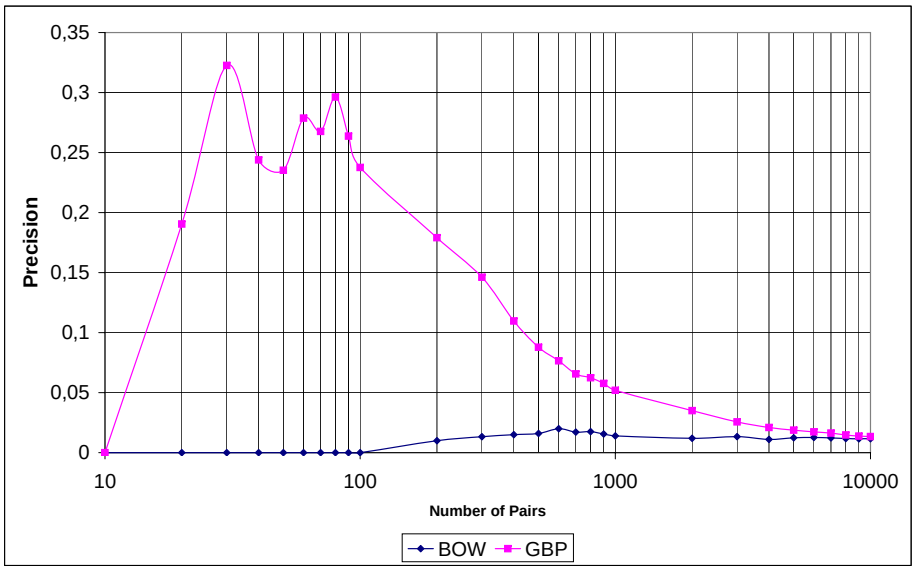

Figure 7. BOW vs. GBP on finding Synonyms: Precision over the Number of evaluated Candidates for Group-By-Path and Bag-of-Words, Second Order Association was used for Processing

5.3. Conclusion for XTREEM-S

On one side, with GBP a better quality was achieved than with a traditional Bag of Words approach. But even those improved results are bad for finding all synonyms which are to be found ideally. One can state that it is possible to find a rather small number of synonyms with acceptable precision.

6. Acquisition of Semantic Sibling Relations

The main contribution of XTREEM is in the field of obtaining semantic sibling relations.

6.1. Sibling Relations

Most approach for the acquisition of semantic relations focused on direct hierarchical relations. Those *sub-ordination relations* exists between super-concepts and their sub-concepts. E.g., in figure 8, between A and B and between A and C as well as between A and D a *Sub-ordination Relation* exists. Between B and C, B and D and between C and D a *Co-ordination Relation* can be observed. The co-ordination relation can be referred to as as *Sibling Relation*. To a set of siblings which are all siblings to each other we refer to as *sibling group*.

Prominent examples of semantic sibling relations are co-Hyponymy and co-meronymy. E.g., the hyponyms of a common hypernym can be referred to as co-hyponyms. Co-meronymy refers to the meronyms (parts) of a common holonym (whole).

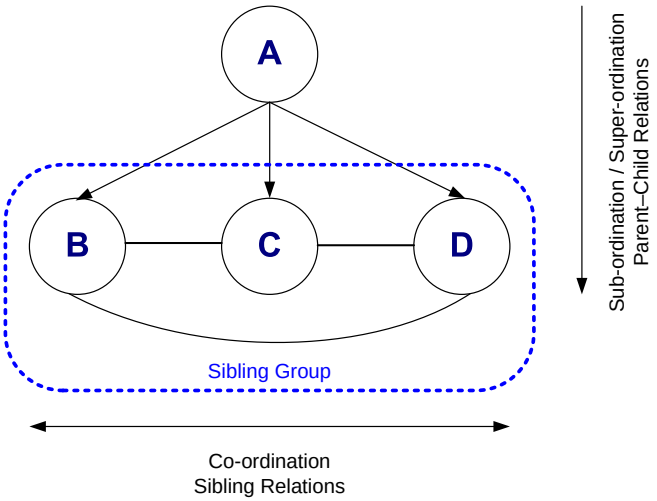

Figure 8. Subordination and Coordination Relations.

6.2. Finding Sibling Relations

For finding semantic sibling relations from GBP data, several processing types are possible. In figure 9 the general procedure is shown: Upon a given Web document collection, the Group-By-Path operation is applied. As an intermediate result, a collection of syntactically motivated sibling text-spans is obtained. By means of statistics and data mining semantic sibling terms can be obtained. In the following we will describe processing methods which we regard as most suitable for this task. Our description will not go into the details, they can be obtained from comphrehensive descriptions in recent articles [40,46].

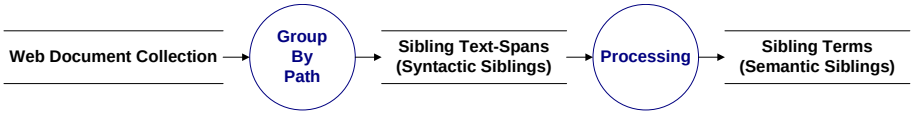

Figure 9. General Data Flow Diagram for XTREEM Procedures. First the Group-By-Path operation is applied upon a Web document Collection. Then the intermediate results are further processed ,e.g., by Clustering

One major distinction is if (1) binary sibling relations (sibling pairs) or if (2) n-ary sibling relations (sibling groups) are to be found.

(1) Sibling Pairs: The most simple form is just counting the co-occurrence within GBP generated terms sets. The co-occurrence frequency of terms can be used as a rough indicator for those terms being siblings to each other. More sophisticated indicators of association strength are the plethora of statistical association measures, which are used in computational linguistics. A comprehensive overview on those associations measures is given by Evert [42].

We performed and evaluated the acquisition of binary sibling relations with XTREEM-SP (XTREEM for Sibling Pairs) in [41] and its successor XTREEM-SA (XTREEM for Sibling Associations) in [46].

Sibling pairs are practically less interesting than sibling groups, but the calculation methods used to find binary relations are computational cheaper than those for finding n-ary relations.

(2) Sibling Groups: To find strong patterns of terms which co-occur as siblings, one can on one side perform (a) frequent item set mining or (b) clustering.

(a) Frequent Item Sets: The acquisition of frequent item sets according to the Apriori algorithm [47] is a method, where for a support threshold all frequent item sets satisfying this support criteria can be found. We applied this algorithm as described in [46]. The calculation of frequent item sets is computational cheaper than performing clustering, but the obtained results are significantly worse than those we obtained with clustering.

(b) Clustering: Clustering is an unsupervised machine learning method. There is a plethora of different clustering algorithms. Several differentiations on characteristic of clustering can be undertaken. One major distinction is the direction in which the data set is to be clustered. On one side, similar records (e.g., documents, sentences or tag paths (sibling term sets)) can be clustered together. For traditional text document clustering this clustering direction is referred to as *document clustering* or row-row clustering. The clustering algorithm works as a kind of lossy compression facility. The clusters have to be labeled by incorporating some kind of clustering labeling strategy. On the other side one can group similar terms (features) together. For traditional text clustering this clustering direction is referred to as *term clustering* or column-column clustering. Here clusters can be directly labeled with the terms constituting the cluster.

In [39] we have described XTREEM-SG where K-Means was used for performing row-row clustering (tag path clustering). In [40] we additionally performed column-column clustering. Column-column clustering (term clustering) yielded considerable better results than row-row clustering.

The automatic evaluation regarding a gold standard evaluation yielded an improvement of the state of the art on finding sibling relations from 14.4 percent [36] to 21.47 percent with row-row clustering [39] and even better by column-column clustering with 22,93 percent [40].

Figure 10 depicts the list of clusters which have been obtained for a vocabulary from the tourism domain. The size of the feature space is 693 terms. For K-Means the parameter K was chosen to be 200. Finally 128 clusters with at least 2 cluster members have been generated. There were two big clusters of terms which could not be well separated. The clusters shown in table 10 are constituted by 463 terms. As can be seen, there are plenty clusters where the semantic coherence can be recognized. On the other hand, there is still much room for optimization, since also many outliers can be found. The row-row clustering (term clustering) which yielded the best results has the disadvantage, that since clustering with hard cluster membership was used, terms can only occur in one sibling context at a clustering. This is problematic since first terms can be polysemous and second one term can have several valid sibling contexts. Incorporating a clustering algorithm with soft cluster membership can improve this circumstance.

Further, clustering algorithms can be divided into hierarchical and non-hierarchical algorithms. The K-Means algorithm which has been mentioned before produces a flat partitioning. In contrast, hierarchical clustering algorithms produce a hierarchy of clusters. Hierarchical clusterings can be further distinguished into divisive (top-down) and agglomerative (bottom-up) approaches.

Figure 10. Resulting Sibling Groups, produced by Term Clustering upon GBP Data

We have already applied Bi-Secting-K-Means, a divisive hierarchical clustering algorithm on row-row clustering. The results are worse than those obtained by K-Means [48]. But using Bi-Secting-K-Means which produces a hierarchy of clusters has the advantage, that the human user which consumes the clusters can browse the cluster hierarchy more easily than a long list of clusters. In the data flow diagram of figure 11 a procedure for obtaining a hierarchy of sibling groups by means of hierarchical clustering is shown.

Further we are investigating the application of agglomerative hierarchical clustering. The preliminary results show that agglomerative clustering yields better results (regard-

ing sibling relations based on XTREEM/GBP) than K-Means or Bi-Secting-K-Means. Agglomerative clustering has the major drawback, that the complexity of the algorithm is $n\hat{2}$ log n, which in contrast to K- Means with its linear complexity is impractical for such data sets as those we have used in our experiments.

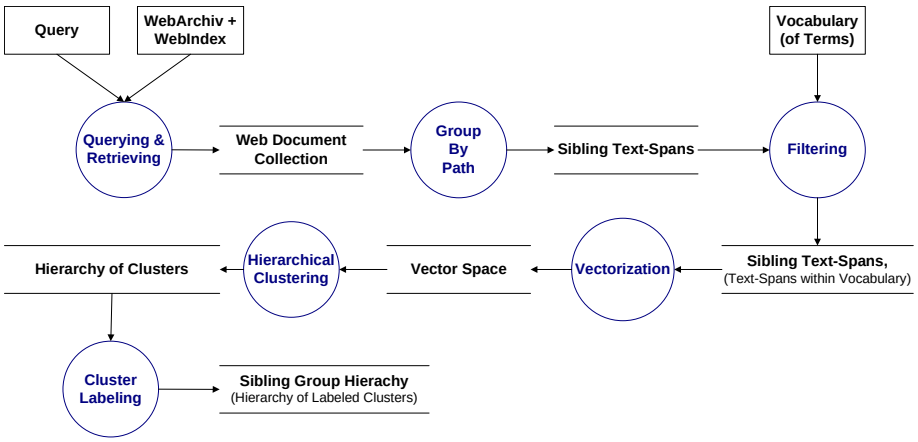

Figure 11. Data-Flow Diagram of the XTREEM-SGH procedure. A Hierarchy of Sibling Groups is acquired by applying Hierarchical Clustering upon a data set of Sibling text spans found by the Group-By-Path approach

For the vectorization which is used for clustering, usually a term weighting like TF-IDF [49] is performed. In [50] we have presented a *domain relevance enhanced term weighting* approach which is especially created for performing term weighting upon GBP data sets. This term weighting is a enhancement of the existing IDF weighting; additionally a domain relevance score is incorporated. The term frequencies which are used for IDF weighting are different for GBP data sets, on a certain fraction of tag paths also undesired terms are captured. To dim the influence of those terms also external information, which has been obtained by comparing the frequency in the domain document collection to the frequency in the general language, is incorporated.

6.3. Conclusion for finding Semantic Siblings

As we have shown in various publications, it is possible to find meaningful semantic sibling relations upon GBP generated term sets. Regarding a gold standard evaluation, the state of the art quality could be significantly improved. The best results for findings sibling groups have been obtained by performing term clustering [40]. This was done by using a clustering algorithm with an hard cluster membership. To overcome the limitation that a term can only belong to one sibling group in term clustering, it is desirable to incorporate a clustering algorithm with soft cluster membership.

The amount of clusters to be generated influences the abstraction forced on the constitution of the resulting sibling groups. For real world settings the expert may decide to handle the tradeoff between reachable quality and the amount of generated information according to his objectives of how detailed the semantics should be. This gold standard evaluation does not capture this aspect, but this can be seen by manually inspecting the results. Cimiano and Staab reported that the results of their approach get better quality

valuation by a human expert inspection; the same holds true for the results obtained with XTREEM. The found siblings are surprisingly meaningful. On the other side this is not astonishing, the results are based on many thousands, often hand crafted, manifestations on the WWW.

Since the XTREEM algorithms for finding sibling relations rely on Web document structure, the algorithms are language and domain independent. A further important advantage of those algorithms is that they can process multiword terms in the same way as words.

7. Conclusion and Future Work

In this chapter we have presented different methods for performing term, synonym and sibling extraction for ontology learning from Web documents. We have shown that several tasks in the ontology learning process can be successfully conducted using the structure of Web documents on a large scale.

First we showed that the markup of Web documents can be used to infer term expressions. The conclusion of a manual evaluation is that around half of the top ranked term candidates are indeed useful terms.

Then we showed that it is possible to extract a rather small fraction of synonyms with reasonable precision. The acquisition of synonyms at higher recall remains a difficult research task. Our finding that statistics based on Group-By-Path performed better than the traditional Bag of Words method should encourage further research, e.g., by incorporating more sophisticated processing methods.

Then we gave an overview on several methods which can be applied to obtain semantic sibling relations. The best results we could obtain are based by term clustering. Further improvements can be expected by hierarchical clustering algorithms. The acquisition of sibling semantics was up to now relative rare. For a complete acquisition of a term/concept hierarchy it is further desirable to combine our methods which perform well on finding co-ordination relations with methods that perform well on finding super-ordination sub-ordination relations.

We have shown the approach for the acquisition of terms, synonyms and siblings in isolation. For a practical setting, the terms found by XTREEM-T can be used as the input vocabulary for XTREEM-S and for the XTREEM approaches finding sibling relations.

All XTREEM approaches have the advantage that since they work on the structure of Web documents, they are language and domain independent. There is no need for existing background knowledge nor for training. Even the Web document collection whereupon the processing is performed can be obtained automatically from the Web.

The limit of the XTREEM approaches is that terms which are not or only rarely marked-up in the Web, because they may not be suited to be used solely, can not be identified. It is an issue of future work to investigate which fraction of a domain vocabulary and the corresponding semantic relations can not be captured this way.

References

[1] Paul Buitelaar, Philipp Cimiano, and Bernardo Magnini. *Ontology Learning from Text: Methods, Evaluation and Applications*, volume 123 of *Frontiers in Artificial Intelligence and Applications Series*. IOS Press, Amsterdam, 7 2005.

[2] Philipp Cimiano. *Ontology Learning and Population*. PhD thesis, Universität Karlsruhe, Institut für Angewandte Informatik und Formale Beschreibungsverfahren (AIFB), 2006.

[3] Vipul Kashyap. Design and creation of ontologies for environmental information retrieval. In Proceedings of the 12th Workshop on Knowledge Acquisition, Modeling and Management. Alberta, Canada., 1999.

[4] Ljiljana Stojanovic, Nenad Stojanovic, and Raphael Volz. Migrating data-intensive Web sites into the semantic Web. In *SAC '02: Proceedings of the 2002 ACM symposium on Applied computing*, pages 1100–1107, New York, NY, USA, 2002. ACM Press.

[5] Richi Nayak and Mohammed Javeed Zaki. Knowledge discovery from XML documents: Pakdd 2006 workshop proceedings. Lecture Notes in Computer Science, Springer Heidelberg, Vol. 3915, 2006.

[6] Udo Kruschwitz. Exploiting structure for intelligent Web search. In *HICSS '01: Proceedings of the 34th Annual Hawaii International Conference on System Sciences (HICSS-34)-Volume 4*, page 4010, Washington, DC, USA, 2001. IEEE Computer Society.

[7] Udo Kruschwitz. A rapidly acquired domain model derived from markup structure. In *Proceedings of the ESSLLI'01 Workshop on Semantic Knowledge Acquisition and Categorization*, Helsinki, 2001.

[8] Keiji Shinzato and Kentaro Torisawa. Acquiring hyponymy relations from Web documents. In *Proceedings of HLT-NAACL*, pages 73–80, 2004.

[9] Andreas Faatz and Ralf Steinmetz. Ontology enrichment with texts from the WWW. In *Proceedings of the First International Workshop on Semantic Web Mining at the ECML 2002*, Helsinki, 2002.

[10] Eneko Agirre, Olatz Ansa, Eduard Hovy, and David Martinez. Enriching very large ontologies using the WWW. In Proceedings of the Workshop on Ontology Construction of the European Conference of AI (ECAI). Berlin, Germany, 2000.

[11] Zhongping Zhang, Rong Li, Shunliang Cao, and Yangyong Zhu. Similarity metric for XML documents. In Proceedings of the Workshop on Knowledge and Experience Management, 2003.

[12] David Buttler. A short survey of document structure similarity algorithms. In Hamid R. Arabnia and Olaf Droegehorn, editors, *International Conference on Internet Computing*, pages 3–9. CSREA Press, 2004.

[13] Theodore Dalamagas, Tao Cheng, Klaas-Jan Winkel, and Timos Sellis. A methodology for clustering XML documents by structure. *Inf. Syst.*, 31(3):187–228, 2006.

[14] Andrea Tagarelli and Sergio Greco. Toward semantic XML clustering. In Joydeep Ghosh, Diane Lambert, David B. Skillicorn, and Jaideep Srivastava, editors, *Proceedings of the SIAM Data Mining Conference (SDM'06)*, pages 188–199. SIAM, 2006.

[15] Il-Hwan Choi, Bongki Moon, and Hyoung-Joo Kim. A clustering method based on path similarities of XML data. *Data & Knowledge Engineering*, Feb. 2006.

[16] Christian Jacquemin and Didier Bourigault. Term extraction and automatic indexing. In *The Oxford Handbook of Computational Linguistics*, chapter 33. 2003.

[17] Hans Friedrich Witschel. Terminology extraction and automatic indexing – comparison and qualitative evaluation of methods. In *Proc. of Terminology and Knowledge Engineering (TKE)*, 2005.

[18] Paul Deane. A nonparametric method for extraction of candidate phrasal terms. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 605–613, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

[19] Joachim Wermter and Udo Hahn. Finding new terminology in very large corpora. In *K-CAP '05: Proceedings of the 3rd international conference on Knowledge capture*, pages 137–144, New York, NY, USA, 2005. ACM Press.

[20] Audrey Baneyx, Jean Charlet, and Marie-Christine Jaulent. Building medical ontologies based on terminology extraction from texts: Methodological propositions. In Silvia Miksch, Jim Hunter, and Elpida T. Keravnou, editors, *AIME*, volume 3581 of *Lecture Notes in Computer Science*, pages 231–235. Springer, 2005.

[21] Lee Gillam and Mariam Tariq. Ontology via terminology? In Proceedings of Workshop on Terminology, Ontology and Knowledge Representation (Termino 2004), Lyon, France, 2004.

[22] Lee Gillam, Mariam Tariq, and Khurshid Ahmad. Terminology and the construction of ontology. Terminology 11 2005 , pp55-81. John Benjamins Publishing Company., 2003.

[23] Paola Velardi, Michele Missikoff, and Roberto Basili. Identification of relevant terms to support the construction of domain ontologies. In *Proceedings of the workshop on Human Language Technology and Knowledge Management*, pages 1–8, Morristown, NJ, USA, 2001. Association for Computational Linguistics.

[24] Paola Velardi, Paolo Fabriani, and Michele Missikoff. Using text processing techniques to automatically enrich a domain ontology. In *FOIS '01: Proceedings of the international conference on Formal Ontology in Information Systems*, pages 270–284, New York, NY, USA, 2001. ACM Press.

[25] Sophie Le Moigno, Jean Charlet, Didier Bourigault, Patrice Degoulet, and Marie-Christine Jaulent. Terminology extraction from text to build an ontology in surgical intensive care. In Proceedings of the ECAI 2002 workshop on NLP and ML for Ontology Engineering. Lyon., 2002.

[26] Harris. Mathematical language. Wiley, New York, e, 1968.

[27] Dekang Lin, Shaojun Zhao, Lijuan Qin, and Ming Zhou. Identifying synonyms among distributionally similar words. In Georg Gottlob and Toby Walsh, editors, *IJCAI*, pages 1492–1493. Morgan Kaufmann, 2003.

[28] Lonneke van der Plas and Jörg Tiedemann. Finding synonyms using automatic word alignment and measures of distributional similarity. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 866–873, Sydney, Australia, July 2006. Association for Computational Linguistics.

[29] Dayne Freitag, Matthias Blume, John Byrnes, Edmond Chow, Sadik Kapadia, Richard Rohwer, and Zhiqiang Wang. New experiments in distributional representations of synonymy. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 25–32, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

[30] Enrique Alfonseca Maria Ruiz-Casado and Pablo Castells. Using context-window overlapping in synonym discovery and ontology extension. In *RANLP-2005*, Sofia, Bulgaria, 2005.

[31] Beate Dorow. *A graph model for words and their meanings*. PhD thesis, University of Stuttgart, 2006.

[32] Thomas K. Landauer and Susan T. Dumais. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211–240, 1997.

[33] Peter D. Turney. Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL. In *Proceedings of the Twelfth European Conference on Machine Learning (ECML-2001)*, 2001.

[34] Marti A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics*, pages 539–545, Morristown, NJ, USA, 1992. Association for Computational Linguistics.

[35] Philipp Cimiano and Steffen Staab. Learning by googling. *SIGKDD Explorations*, 6(2):24–33, 2004.

[36] Philipp Cimiano and Steffen Staab. Learning concept hierarchies from text with a guided agglomerative clustering algorithm. In Chris Biemann and Gerhard Paas, editors, *Proceedings of the ICML 2005 Workshop on Learning and Extending Lexical Ontologies with Machine Learning Methods*, Bonn, Germany, August 2005.

[37] Gerhard Heyer, Martin Läuter, Uwe Quasthoff, Thomas Wittig, and Christian Wolff. Learning relations using collocations. In Alexander Maedche, Steffen Staab, Claire Nedellec, and Eduard H. Hovy, editors, *Workshop on Ontology Learning*, volume 38 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2001.

[38] Marko Brunzel and Myra Spiliopoulou. Discovering multi terms and co-hyponymy from xhtml documents with XTREEM. In Richi Nayak and Mohammed Javeed Zaki, editors, *KDXD*, volume 3915 of *Lecture Notes in Computer Science*, pages 22–32. Springer, 2006.

[39] Marko Brunzel and Myra Spiliopoulou. Discovering semantic sibling groups from Web documents with XTREEM-SG. In Steffen Staab and Vojtech Svátek, editors, *EKAW*, volume 4248 of *Lecture Notes in Computer Science*, pages 141–157. Springer, 2006.

[40] Marko Brunzel and Myra Spiliopoulou. Discovering semantic sibling clusters from Web documents with XTREEM-SG. submitted to Journal on Semantics of Data (JoDS VIII), special issue, 2007.

[41] Marko Brunzel and Myra Spiliopoulou. Discovering semantic sibling associations from Web documents with XTREEM-SP. In A. Min Tjoa and Juan Trujillo, editors, *DaWaK*, volume 4081 of *Lecture Notes in Computer Science*, pages 469–480. Springer, 2006.

[42] Stefan Evert. *The Statistics of Word Cooccurrences: Word Pairs and Collocations*. PhD thesis, University of Stuttgart, 2004.

[43] Christopher D. Manning and Hinrich Schtze. *Foundations of statistical natural language processing*. MIT Press, 1999.

[44] Alexander Maedche and Steffen Staab. Discovering conceptual relations from text. In *ECAI-2000 –Proceedings of the 13th European Conference on Artificial Intelligence*, pages 321–325. IOS Press, Amsterdam, 2000.

[45] Alexander Maedche and Steffen Staab. Ontology Learning for the Semantic Web. *IEEE Intelligent Systems*, 16(2):72–79, 2001.

[46] Marko Brunzel and Myra Spiliopoulou. Acquiring semantic sibling associations from Web documents. accepted for publication at the International Journal of Data Warehousing and Mining (IJDWM), special issue, 2007.

[47] Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Mining association rules between sets of items in large databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington D.C., May 1993.

[48] Marko Brunzel. Learning of semantic sibling group hierarchies - k-means vs. bi-secting-k-means. In *9th International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2007)September 3-7, 2007, Regensburg, Germany*, volume 4654 of *Lecture Notes in Computer Science*. Springer, 2007.

[49] Gerard Salton and Chris Buckley. Term weighting approaches in automatic text retrieval. Technical report, Cornell University, Ithaca, NY, USA, 1987.

[50] Marko Brunzel and Myra Spiliopoulou. Domain relevance on term weighting. In *12th International Conference on Applications of Natural Language to Information Systems (NLDB) June 27-29, 2007, CNAM, Paris, France*, volume 4592 of *Lecture Notes in Computer Science*. Springer, 2007.

Part 2: Taxonomy and Concept Learning

Extracting concept descriptions from the Web: the importance of attributes and values

[1]

Massimo POESIO [a] and Abdulrahman ALMUHAREB [b]
[a] *Center for Mind / Brain Sciences, Universitá di Trento, Italy*
and Language and Computation Group, University of Essex, UK
[b] *King Abdulaziz City for Science and Techology (KACST), Saudi Arabia*

Abstract. When extracting information about concepts from the Web, the problem is not recall, but precision: trying to identify which properties of a concept are genuinely distinctive. We discuss a series of experiments in empirical ontology using both unsupervised and supervised methods, showing that not all semantic relations we can extract from text are equally useful, and suggesting that attempting to identify concept **attributes** (parts, qualities, and the like) and their **values** results in better concept descriptions than those obtained by being less selective.

Keywords. Ontology learning from text, relation extraction, attributes and values

1. Introduction

The availability of huge amounts of textual data about concepts on the Web and in other web-size corpora has kickstarted a new area of research usually called ontology learning from text, which has the ambitious goal of developing methods for extracting from text full ontologies–or at least, taxonomies of conceptual knowledge. As this volume testifies, ontology learning from text is a very active area of research, and very many systems performing this task have been developed, including OntoLearn [1], Text2Onto [2] and other systems discussed in the volumes in this collection.

Our research is at the boundaries between ontology learning proper and work on (nominal) lexical acquisition in Computational Linguistics CL [3,4,5,6,7,8], in that the approach we are pursuing is perhaps best described as 'lexical acquisition meets relation extraction'. In some sense, our goals are more modest than those of systems like OntoLearn or Text2Onto: we are only interested in clustering what in psychology are called (nominal) **basic categories** such as **dog**, **cat**, **car**, and **truck** into what psychologists call their **superordinate categories**, such as **animal** or **vehicle**. From another point of view, however, our goal is quite ambitious: to carry out a form of what we call **empirical ontology**–namely, to use computational methods to gain insights into concepts that can supplement the use of evidence from psychology and the neural sciences in providing a test for theories of ontology.

Perhaps the most distinctive feature of the methods discussed in this paper is that we defined the relations to be extracted on the basis of views developed in Artificial Intelligence AI, linguistics, and philosophy about the 'intrinsic' properties of concepts – that we will call **attributes**. In other words, we believe that not all information is equally important to build good concept descriptions: from the clustering perspective at least– the perspective of identifying which concepts are similar, and which ones have distinct superordinate categories– 'less is more': not all information that can be gathered from a corpus is equally important, not even all semantic information.

The structure of this paper is as follows. We begin with a quick overview of work on concepts, on lexical acquisition, and on relation extraction. We then discuss first our unsupervised, then our supervised methods for building concept descriptions. A discussion follows.

2. Background

Concepts are viewed as complex mental objects characterized by a number of **attributes** or features in most theories of concepts developed in philosophy, psychology, linguistics and Artificial Intelligence (AI), even in those theories derived from the work of Wittgenstein and Rosch that do not subscribe to the view that concepts can be 'defined' in the Aristotelian sense.

The notion of 'attribute' assumed in philosophical work is typically semantic. For instance, according to Aristotle (in *Metaphysics*) the nature of an object can be described by four 'causes': the **material cause** (the material of which the object is composed), the **agentive cause** (what causes the object's movement or creation), the **formal cause** (what a thing is planned and intended to be –its essence and form), and the **final cause** ("that for the sake of which a thing exists, or is done"). This view of the nature of objects was adopted in linguistics by Pustejovsky [9], who developed **Generative Lexicon** theory according to which an integral part of a lexical entry is its **qualia structure** in the sense of Aristotle. Pustejovsky's qualia structure consists of four (types of) **roles** corresponding to Aristotle's four causes. The **formal role** is a complex of attributes specifying what type of object the concept denotes–its 'intrinsic qualities'. These include both its supertypes (its $\langle isa \rangle$) relations and attributes specifying its form. For instance, in the case of the concept **book**, the formal roles include the fact that a **book** is a **physical object** with certain qualities such as $\langle shape \rangle$ and $\langle color \rangle$. The second role is the **constitutive role**, specifying the stuff and parts that an object consists of. Again, in the case of **book**, the constitutive roles include the fact that a book is made of **paper**, that it has **chapters** and an **index**, etc.. The **telic role** specifies the purpose of the object–e.g., in the case of a book, **reading**. Finally, the **agentive role** specifies how the object was created: e.g., in the case of a book, by **writing**.

In Artificial Intelligence, theories of concepts based on a semantic notion of attribute have been developed in the area of formal ontology. For instance, Guarino [10] developed a theory of attributes according to which there are two types of attributes: **relational** and **non-relational**. Relational attributes include **qualities** such as $\langle color \rangle$ and $\langle position \rangle$ and **relational roles** such as $\langle son \rangle$ and $\langle spouse \rangle$. Non-relational attributes include **parts** such as $\langle wheel \rangle$ and $\langle engine \rangle$. Activities such as reading and writing for a book are not viewed as attributes in Guarino's classification. **Description logics** [11], the latest form

of the approach to semantic networks developed by Brachman, Levesque, and colleagues in papers such as [12], are logics developed to define concepts in terms of subsumption relations (**isa** links) and attributes.

Work on (nominal) concept acquisition from corpora in computational linguistics, on the other hand, is usually based on the distributional view of meaning derived from Wittgenstein via Firth [13], according to which the meaning of a word is specified by the other words with which this word co-occurs, not necessarily through the mediation of deeper semantic notions. This view of meaning led to the vector space model of semantic representation, proposed by Salton *et al.* [14] to represent the meaning of documents in Information Retrieval, but then adopted as a view of lexical meaning in the pioneering work of Schuetze [6], and then in work such as [4,5]. These models have proven rather successful at modelling synonymy, as shown e.g., by the results obtained on the TOEFL test [15]. Starting at least with Grefenstette [3], however, a modified view of vectorial spaces have been developing, in which the syntactic relation between a word and its neighbors (in the sense of dependency grammar) is taken into account in defining proximity [7,8].

In parallel with these efforts, however, there has been in CL a line of work dedicated to work on (supervised and unsupervised) extraction of semantic relations, beginning with the seminal work of Hearst [16] who developed unsupervised methods for the extraction of hyponymy relations using patterns. Hearst's proposals on hyponymy were followed up in [17,18], and additional work was carried out on part-of relations [19,20,21] and on other relations as part of the ACE program.[2]

More recently still, these methods for relation extraction have begun to be applied to ontology construction and population [22,23,24]; see also the papers in this volume, particularly the chapters by Aussenac *et al.*, Pantel and Pennacchiotti, and Voelker *et al.*. Our own approach [25,26,27,28,29] belongs to this line of work attempting to combine the two types of research: using relation extraction techniques to extract the 'features' to be used to describe vectors, but for ontology learning rather than ontology population. (We discuss related work in a later section.)

3. Attribute-based concept descriptions: an unsupervised approach

3.1. Attributes and values in concept descriptions

When less data were available–e.g., when extracting relational information from the Penn Treebank as done by Caraballo [17], the 64 million-word corpus used by Lin [7], or even from the 100 million-word British National Corpus used by [20]– one cannot afford to be choosy: every bit of information was necessary. But now, working with the Web or Web-sized corpora,[3] too much information is the problem.

Our approach to concept extraction is therefore based on the principle that sometimes 'less is more': extract only 'intrinsic' properties of concepts, as opposed to all information about them that can be found in a corpus. (This, of course, is predicated on the

[2] http://www.nist.gov/speech/tests/ace/index.htm

[3] We are not aware of any study attempting to establish what exactly is the threshold at which the phenomenon we are pointing out–elimination or reduction of the data sparsity problem, excessive data richness–begins to be displayed; we would guess not before 10 giga words.

assumption that there are more 'essential' properties of concepts–a point to which we will return in the discussion.) Specifically, what distinguishes our own approach to using relation extraction to acquire concept descriptions is the attempt to extract from corpora 'attributes' of concepts and their 'values', instead of all relations, being guided in this by works such as [10] and [9].

The methods for extracting attributes proposed in [25,26] are based on the observation that many such properties are dependent concepts realized using relational or functional nouns, and can therefore be used in constructions of the form "the X of the Y". Parts such as ⟨*wheels*⟩, for instance, are generally dependent concepts, hence the fact that **cars** have ⟨*wheels*⟩ can be expressed using the construction "the WHEEL of the CAR". This point can already be found in [30], where we can find the following test for attribute-hood. According to Woods, A is an attribute of concept C if we can say:

VALUE is a/the ATTRIBUTE of CONCEPT,

For example, the fact that we can say:

brown is a color of dogs

suggests that ⟨*color*⟩ may be an attribute of **dog**. Relational nouns potentially expressing attributes also occur in possessive constructions, as in "the CAR's WHEEL". On the basis of this observation, we extracted possible concept attributes searching in the Web for the two constructions above.[4]

Many concepts are characterized by special **values** of certain attributes: e.g., while all **fruit** have a ⟨*color*⟩ attribute, bananas are typically yellow, whereas strawberries are typically red. Thus we aimed to include in our concept descriptions, in addition to information about possible attributes of concepts, information about **values** of such attributes which are particularly distinctive of that concept.

It turns out that the contruction proposed by Woods above is not very common, even when the Web is used as a corpus. We found less than 500 instances of the constructions "VAL is the ATTR of CONCEPT" or "the ATTR of the CONCEPT is VAL" [29]. The construction suggested in [32],

"the VAL1 or VAL2 ATTR",

(as in "the RED or WHITE COLOR") has very high precision (almost 80%) but does not have very high recall either: we could not find in the Web any values for 3 out of 10 very common attributes [29]. So in the end we settled for considering as potential 'values' all prenominal modifiers occurring in constructions of the form:

"the VALUE CONCEPT is"

as in "the RED CAR is". This of course meant that we included among the 'values' a number of modifiers that could not really be considered values of any particular attribute (e.g., "trained" as a modifier of "horse") as well as a lot of information that is best described as collocational (e.g., "Trojan, Arabian, hobby" for "horse" again); we return to this point in the discussion.

[4] We looked for other constructions using the methods proposed by Hearst [31], but such constructions resulted in much lower precision for little recall gain.

3.2. Experimental results: text patterns

We tested the hypothesis that we would obtain better concept descriptions by concentrating on the constructions expressing information about attributes and values in a series of experiments discussed in [25,26,29,33]. We tested two ways of identifying these constructions: using simple textual patterns, as done in [31,19], and using a (dependency) parser–specifically the RASP parser developed by Briscoe and Carroll [34].

In all of the experiments described in this section we used the t-test as defined by [8] as a ranking function, defined as follows, where $t_{i,j}$ is the output weighted frequency for the $concept_i$ and the $feature_j$, N is the overall frequency, and C is a count function:

$$t_{i,j} \approx \frac{\frac{C(concept_i, feature_j)}{N} - \frac{C(concept_i) \times C(feature_j)}{N^2}}{\sqrt{\frac{C(concept_i, feature_j)}{N^2}}}$$

Our own tests confirmed Curran and Moens' finding that this measure outperformed other measures including simple frequency, mutual information, log likelihood ratio, and χ^2. As a similarity function we used the version of the Jaccard coefficient defined by [8] and shown below, where $t_{m,i}$ and $t_{n,i}$ are the weighted co-occurrence frequencies between $concept_m$ and $concept_n$ with $feature_i$, respectively.

$$Jaccard(concept_m, concept_n) = \frac{\sum_i (t_{m,i} \times t_{n,i})}{\sum_i (t_{m,i} + t_{n,i})}$$

Again, our results confirmed those by Grefenstette and by Curran and Moens that extended Jaccard outperforms other similarity measures including the cosine and Lin's similarity function [7]. We also always used as our clustering algorithm Repeated Bisections [35], a variant of K-means clustering, as implemented in the CLUTO clustering tool [36], as again we found it outperformed other clustering algorithms that we tested including EM, agglomerative clustering, and COBWEB.

In a preliminary experiment [25], we only used text patterns to extract information about the attributes and values of 214 relatively common nouns associated with synset belonging to 13 different WordNet classes. For finding attributes we used the following two Google patterns:

```
"the * of the C R"

"the C's * R"
```

where *C* is a concept, *R* is a restrictor such as *is* and *was*, and the wildcard denotes an unspecified attribute. For values we used the following Google pattern:

```
"[a|an|the] * C R"
```

In this experiment we found that attributes were as informative as values but that to achieve the best performance it was necessary to include in the concept descriptions a combination of the highest ranked attributes and values.

In a second esperiment [26] we compared attribute / value extraction using text patterns and using a parser. For this experiment we used a dataset of 402 concepts covering

all 21 WordNet unique beginners, and balanced for frequency (1/3 of the concepts in the set are high frequency as measured from the BNC, 1/3 are medium-frequency, 1/3 are low frequency) and ambiguity (1/3 of the concepts are highly ambiguous in the sense of having more than 4 senses in WordNet, 1/3 have between 2 and 3 senses, 1/3 have only 1 sense). (The dataset is shown in Appendix A.) The results were measured using **purity** and **entropy**, both of which measure the extent to which clusters are 'uniform', and are defined as follows.

Let S_r be a cluster, n_r the size of cluster S_r, q the number of classes in the dataset, n_r^i the number of concepts from the ith class that were assigned to the rth cluster, n the total number of concepts, and k is the number of clusters. Then the **purity** of cluster S_r, $P(S_r)$, is the ratio of the number $max_i(n_r^i)$ of elements in the 'dominant' category for S_r–the category with the greatest number of elements in that cluster–to the number n_r of elements in that cluster. A cluster containing only elements from one class will have purity 1. The **entropy** of cluster S_r, $E(S_r)$, is the standard entropy– a more comprehensive measure, that takes into account the entire distribution of categories in the cluster. Overall entropy and purity are the weighted sum of individual cluster entropies and purities respectively.

$$Entropy = \sum_{r=1}^{k} \frac{n_r}{n} E(S_r), \;\; where \;\; E(S_r) = -\frac{1}{log\, q} \sum_{i=1}^{q} \frac{n_r^i}{n_r} \, log \frac{n_r^i}{n_r}$$

$$Purity = \sum_{r=1}^{k} \frac{n_r}{n} P(S_r), \;\; where \;\; P(S_r) = \frac{1}{n_r} \, max_i(n_r^i)$$

The results using textual patterns (the same as in the first experiment) are shown in Table 1.

Measures	**Attributes**	**Values**	**Attributes and Values**
	21 Classes (402 Concepts)		
Purity	0.657	0.567	0.677
Entropy	0.335	0.384	0.296
Vector Size	24,178	94,989	119,167

Table 1. Clustering results using textual patterns

Here, we find that using vectors of attributes we can get better results than using vectors of 'values', even with vectors of a quarter of the size. And again, the best results are obtained by combining attributes and 'values'.

Looking at per-category purity, we find that we obtain perfect purity for *edible fruit*, *vehicle*, *illness*. Purity is above .8 for *animal*, *chemical element*, *creator*, *feeling* and *monetary unit*–almost all concrete categories. The worse purity is for abstract categories; among these, the worse two are *motivation* and *time*, with purity less than .4.

3.3. Experimental results: using a dependency parser

Text patterns are rigid: a pattern like "the * of the C is" cannot match, for instance, the construction "the SPEED of John's CAR is". Using a parser and finding instances of the

construction by searching in its output would allow us to find more constructions. On the other hand, this might introduce more errors, as well as making the method more language-dependent as we don't have dependency parsers for all languages.

For the second part of the experiment, we collected the documents found using the text patterns in the first part of the experiment, extracted the sentences that contained instances of the desired concept, and parsed them with RASP, obtaining results such as those shown in Figure 1.

```
(detmod, strawberry, the)            (dobj, like, strawberry)
(detmod, strawberry, a)              (ncmod, fruit, strawberry)
(ncsubj, be, strawberry)             (ncmod, strawberry, wood)
(ncmod, strawberry, fresh)           (ncsubj, grow, strawberry)
(ncmod, strawberry, wild)            (dobj, eat, strawberry)
(ncmod, plant, strawberry)           (ncsubj, have, strawberry)
(xcomp, be, strawberry)              (ncmod, strawberry, frozen)
(ncmod, strawberry, whole)           (ncmod, strawberry, cup)
(ncmod, strawberry, ripe)            (ncmod, strawberry, red)
(ncmod, strawberry, cultivated)      (dobj, have, strawberry)
(iobj, with, garnish, strawberry)    (ncmod, of, variety, strawberry)
(detmod, strawberry, an)             (ncmod, strawberry, modern)
(ncmod, variety, strawberry)         (ncmod, cultivar, strawberry)
(dobj, slice, strawberry)            (ncsubj, grow, strawberry, obj)
(ncmod, strawberry, large)           (ncmod, strawberry, big)
```

Figure 1. The most frequent grammatical relations for **strawberry**

RASP gives us the opportunity to study the usefulness for concept description of many other types of constructions in addition to those tested in the experiments with text patterns. The text patterns essentially extracted from text two (approximated) instances of what in RASP is called the `ncmod` grammatical relations: attribute patterns extracted cases which RASP would parse as

```
(ncmod, of, ATTR, CONCEPT),
```

(as in `(ncmod, of, color, strawberry)`), whereas value patterns extract cases that RASP would parse as

```
(ncmod, CONCEPT, VAL)
```

as in `(ncmod, strawberry, red)`. However, many other types of grammatical relations have been included in concept descriptions in the literature on using syntax-based relations for concept descriptions [3,7,8], including in particular direct objects and subjects. We could now compare the effect of using these additional grammatical relations. The results are shown in Table 2.

As the table shows, using all grammatical relations as part of the concept descriptions results in *less* purity than using only attributes and values–around a third of the features collected using RASP– or indeed just using attributes–less than a tenth of all features. In other words, these results indicate that "less is more": adding more information not necessarily results in better concept descriptions. The table also shows that using a parser results in slightly better performance than using text patterns, at the expense of less language-independence.

Measure	Grammatical relations subset					
	Attribute	**Value**	**Direct Obj**	**Subj**	**Attribute and Value**	**All**
Purity	0.656	0.600	0.613	0.555	0.701	0.614
Entropy	0.320	0.360	0.391	0.413	0.296	0.360
Vector Size	20,285	52,486	6,318	11,002	72,771	276,501

Table 2. Clustering results using different subsets of grammatical relations

4. Supervised attribute extraction

4.1. A first classification scheme for attributes

The patterns used for attributes in the experiments above match constructions expressing all sorts of semantic relations other than attribute-concept. The range of semantic relations expressed by the "the X of the Y" construction is illustrated in Figure 2 with examples of 'attributes' of **deer** collected by our unsupervised system discussed in Section 3. As the figure shows, this construction is used to express partitions of sets (hence the name **partitive construction**), ordinals, 'picture of' relations, naming, and other more complex relations. While some of this information may be considered highly distinctive of a concept like **deer** (e.g., it is hard to imagine that one would find discussions of the meaning of **bismuth**), none of it can be considered as expressing an 'attribute' of **deer**, in the intuitive sense of a 'defining property' of the concept.

```
the rest / majority of the deer
the first / last of the deer
the picture / image / photos of the deer
the cave / mountain / lake of the deer
the meaning of the deer [in Western philosophy / ... ]
```

Figure 2. Semantic relations expressed using the "the X of the Y" construction.

In order to make further progress towards the goal of extracting concept descriptions containing only 'proper' attributes, and excluding information such as those in Figure 2, it is necessary to be more clear about what we mean with 'attributes'–i.e., to have a theory of attributes, or at least a classification scheme specifying which relations count as proper attributes of concepts and which ones instead do not. Having this theory, or the classification, would allow us to develop supervised or unsupervised theories of concept extraction.

Unfortunately although the notion of 'attribute' has been part of philosophical theories of concepts and knowledge in philosophy since at least Aristotle, no fully worked out theory of attributes exists, nor a classification of attributes covering all concepts, in part also because philosophy and psychology tend to concentrate on a few categories of concepts. Partial theories can however be found in works such as [10] and [9] discussed earlier; these works can serve as a starting point for our study. (Inversely, one would hope of course that work on ontology extraction from text might contribute to formal work on ontology.)

Out of the two proposals of Guarino and Pustejovsky we developed a classification scheme for attributes that considers a relation in which a concept participates as an attribute if it belongs to one the following types:

qualities. These relations include Guarino's qualities and (some of) Pustejovsky's formal roles: e.g., the ⟨*weight*⟩ / ⟨*fusibility*⟩ / ⟨*solubility in Aqua Regia*⟩ of **gold**.

parts. These relations include Guarino's non-relational attributes, and Pustejovsky's constitutive roles, including that is parts such as the ⟨*wheel*⟩ of the **car** or the ⟨ *leg*⟩ of the **animal**, and 'stuff' such as the ⟨*gold*⟩ of the **ring**.

related objects. These relations relate objects to other independent objects with which however they are in strong association, such as the ⟨*nest*⟩ of the **bird**, and include Guarino's non-relational attributes other than parts and relational roles.

activities. These relations include Pustejovsky's telic and agentive roles such as the ⟨*reading*⟩ and ⟨*writing*⟩ of the book, but also other important activities such as the ⟨*publication*⟩ of the book.

related agents. Finally, these relations relate objects to agents that perform the activities above, such as the ⟨*writer*⟩, ⟨*reader*⟩ or ⟨*publisher*⟩ of a **book**.

Notice that ⟨*isa*⟩ relations were not included among attributes, although most work on ontology learning concentrates on this type of information [31,17]. This is in part precisely because there is no lack of insightful work on this topic, in part because this information plays a different epistemological function in the definition of concepts, in the sense of [37].

4.2. New experiments

In [28] we discussed the results obtained using a supervised classifier to classify potential attributes extracted from the Web in the five classes above, as well as the class 'not-an-attribute'.

We collected from the Web 20,000 candidate attributes for the 402 concepts in the dataset, kept the 4,728 that occurred more than 20 times, and collected for all of them four types of information:

morphological features, extracted through heuristics, such as the information that a particular noun might be derived from an adjective or a verb, which is useful to identify qualities and activities respectively [38];

question patterns, that is, the frequencies obtained by querying the Web with questions of the form "What is the ATTR of CONCEPT" or "When is the ATTR of CONCEPT";

features of features, i.e., the top attributes of these potential attributes, extracted using the same patterns that we used to extract attributes; and finally

feature use, that is, information about the respective frequency of the use of these nouns as attributes ("the ATTR of the *") or concepts ("the * of the CONCEPT").

We then hand-classified 1,155 of these feature vectors into six classes (the five classes above, and 'not an attribute'), and we trained two classifiers: a binary one just making the decision attribute / not attribute, and one classifying attributes into one of five classes. We evaluated these classifiers in three ways: (i) through cross-validation over the 1,155

hand-annotated features, (ii) by running them over around 400 additional feature vectors and hand-evaluating the results, and (iii) by using them to filter the potential attributes and clustering the original concepts using only the remaining attributes.

The binary classifier achieved an accuracy of 81.82% as evaluated through cross-validation, which corresponds to an F value of .892 at recognizing attributes and .417 at recognizing non-attributes. About the same results were obtained over the additional 400 attributes.

The 5-way classifier achieved an accuracy of around 80% at cross-validation, corresponding to an F value over .8 for quality, activity, and part / related object, of .95 for related agent, and .538 for not-an-attribute. Accuracy over the additional 400 attributes however was significantly lower at around 70%.

Table 3 compares the results obtained when clustering concepts using all attributes– i.e., the results with all attributes shown in Table 1– (second column) with the results obtained by filtering attributes using simple heuristics (third column) and, finally, the results obtained when clustering after having removed the attributes classified as 'not-attributes' by the binary classifier.

	All Candidate Attributes	Heuristic filtering	Filtering by classification
Purity	0.657	0.672	0.693
Entropy	0.335	0.319	0.302
Vector size	24,178	4,296	3,824

Table 3. Clustering with a supervised classifier

As the table shows, removing 'non-attributes' resulted in a significant improvement in the purity of the clustering. (The difference between the value of purity using all candidate attributes and using filtering by classification is significant.)

5. Related work

As mentioned above, relation extraction techniques have been used in work on ontology *population* such as [22,23,39]. The techniques used in these systems are primarily unsupervised, and the focus is on developing methods for acquiring new patterns for extracting the relevant information, as in KNOWITALL [23]. [39] propose an unsupervised method, but introduce a measure of the **reliability of pattern** used to identify which of the patterns found by the system are reliable.

The work most closely related to ours is probably Cimiano and Wenderoth's work on extracting qualia structures [40]. Cimiano and Wenderoth are also concerned with ontology learning rather than ontology population, and developed a completely unsupervised approach to extracting qualia structures by developing specific patterns for each of Pustejovsky's qualia. Apart from their approach being unsupervised, the main difference from the present work is that Cimiano and Wenderoth do not use the extracted information to build vectors for clustering purposes, hence the main evaluation is not in terms of clustering performance, but by comparing the information thus extracted with what is found in the literature, and by human evaluation.

6. Discussion

In a sense, the most exciting aspect of this type of research is that we can revisit all sorts of old philosophical chestnuts, but with the guidance of empirical evidence.

One example of 'philosophical' question is the question of what is an 'attribute'. We most certainly do not think that we found the definitive characterization of the notion of attribute. What we do think we have is evidence that attempting to identify the types of semantic relations that go under the name of 'attributes' or 'qualia' does seem useful from an ontology learning perspective (i.e., to draw a distinction between 'basic categories'). One might even wonder whether this work provides empirical evidence for Aristotle's notion of 'essence'–in the sense that concepts may have 'essential properties' that are best in distinguishing them from other concepts. (Of course, for some applications one may want to collect all available data about concepts.)

Another important question is how far to go in the direction of obtaining purely 'semantic' concept descriptions. The concept descriptions obtained with the methods discussed in this paper mix semantic and distributional information; would it make sense to try to 'clean up' these descriptions further? Again, perhaps a distinction needs to be drawn between information to be used for clustering concepts (which may be in part distributional) and information about the concepts to be used for question-answering purposes.

Up until now the terms 'concept' and 'noun' have been treated as synonymous, but of course this is not the case. A noun like *palm* will be associated with many concepts (e.g., 4 synsets in WordNet). This is to say that the concept descriptions obtained with the methods discussed here need to be *discriminated* in order to obtain actual concepts; we propose methods for doing this in [41,29].

Future work will include improving upon our theory of attributes, extracting additional types of information, trying out recent developments in semi-supervised relation extraction, and developing better evaluation methods.

References

[1] R. Navigli and P. Velardi. Learning domain ontologies from document warehouses and dedicated website. *Computational Linguistics*, 30(2), 2004.

[2] P. Cimiano and J. Voelker. Text2onto - a framework for ontology learning and data-driven change discovery. In *Proc. 10th International Conference on Applications of Natural Language to Information Systems (NLDB'2005)*, 2005.

[3] G. Grefenstette. *Explorations in Automatic Thesaurus Discovery*. Kluwer, 1994.

[4] K. Lund, C. Burgess, and R. A. Atchley. Semantic and associative priming in high-dimensional semantic space. In *Proc. of the 17th Annual Conference of the Cognitive Science Society*, pages 660–665, 1995.

[5] T. K. Landauer and S. T. Dumais. A solution to plato's problem: the latent semantic analysis theory of acquisition, induction and representation of knowledge. *Psychological Review*, 104(2):211–240, 1997.

[6] H. Schütze. *Ambiguity Resolution in Language Learning*. CSLI, Stanford, 1997.

[7] D. Lin. Automatic retrieval and clustering of similar words. In *Proc. of COLING-ACL*, 1998.

[8] J. R. Curran and M. Moens. Improvements in automatic thesaurus extraction. In *Proc. of the ACL Workshop on Unsupervised Lexical Acquisition*, pages 59–66, 2002.

[9] J. Pustejovsky. *The generative lexicon*. MIT Press, 1995.

[10] N. Guarino. Concepts, attributes and arbitrary relations. *Data and Knowledge Engineering*, 8:249–261, 1992.

[11] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge, 2003.

[12] Hector J. Levesque and Ronald J. Brachman. A fundamental tradeoff in knowledge representation and reasoning (revised version). In Ronald J. Brachman and Hector J. Levesque, editors, *Readings in Knowledge Representation*, pages 42–70. Morgan Kaufmann, San Mateo, California, 1985.

[13] J. Firth. A synopsis of linguistic theory 1930-55. In *Studies in Linguistic Analysis*, pages 1–32. The Philological Society, Oxford, 1957.

[14] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

[15] P. D. Turney. Mining the web for synonyms: PMI-IR versus LSA on the TOEFL. In *Proc. of 12th European Conference on Machine Learning (ECML)*, pages 491–502, 2001.

[16] M. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proc. of COLING*, pages 539–545, 1992.

[17] S. A. Caraballo. Automatic acquisition of a hypernym-labeled noun hierarchy from text. In *Proc. of the ACL*, pages 120–126, 1999.

[18] E. Riloff and J. Shepherd. A corpus-based approach for building semantic lexicons. In *Proc. of EMNLP-2*, 1997.

[19] M. Berland and E. Charniak. Finding parts in very large corpora. In *Proc. of the 37th ACL*, pages 57–64, University of Maryland, 1999.

[20] M. Poesio, T. Ishikawa, S. Schulte im Walde, and R. Vieira. Acquiring lexical knowledge for anaphora resolution. In *Proc. of the 3rd LREC*, Las Palmas, Canaria, 2002.

[21] R. Girju, A. Badulescu, and D. Moldovan. Learning semantic constraints for the automatic discovery of part-whole relations. In *Proc. of HLT*, 2003.

[22] P. Cimiano, S. Handschuh, and S. Staab. Towards the self-annotating web. In *Proc. 13th World Wide Web Conference*, pages 462–471, 2004.

[23] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A-M. Popescu, S. Shaked, S. Soderland, D. Weld, and A. Yates. Unsupervised named-entity extraction from the web: An experimental study. *Artificial Intelligence*, 165(1):91–134, 2005.

[24] G. Mann and D. Yarowsky. Unsupervised personal name disambiguation. In *Proc. CoNLL*, Edmonton, Alberta, 2003.

[25] A. Almuhareb and M. Poesio. Attribute- and value-based clustering of concepts. In *Proc. of EMNLP*, pages 158–165, Barcelona, July 2004.

[26] A. Almuhareb and M. Poesio. Concept learning and categorization from the web. In *Proc. of Annual Meeting of Cognitive Science Society*, Stresa (Italy), July 2005.

[27] M. Poesio and A. Almuhareb. Feature-based vs. property-based kr: An empirical evaluation. In A. C. Varzi and L. Vieu, editors, *Proc. of the Third International Conference on Formal Ontology in Information Systems (FOIS)*, Torino, November 2004.

[28] M. Poesio and A. Almuhareb. Identifying concept attributes using a classifier. In T. Baldwin and A. Villavicencio, editors, *Proc. of the ACL Workshop on Deep Lexical Semantics*, pages 18–27, Ann Arbor, Michigan, June 2005.

[29] A. Almuhareb. *Attributes in Lexical Acquisition*. PhD thesis, University of Essex, Department of Computer Science, 2006.

[30] W. A. Woods. What's in a link: Foundations for semantic networks. In Daniel G. Bobrow and Alan M. Collins, editors, *Representation and Understanding: Studies in Cognitive Science*, pages 35–82. Academic Press, New York, 1975.

[31] M. A. Hearst. Automated discovery of wordnet relations. In C. Fellbaum, editor, *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[32] V. Hatzivassiloglou and K. McKeown. Towards the automatic identification of adjectival scales: clustering adjectives according to meaning. In *Proc. of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 172–182, Ohio State University, 1993.

[33] A. Almuhareb and M. Poesio. Ontology learning through unsupervised attribute extraction from the web. In preparation, 2007.

[34] E. Briscoe and J. Carroll. Robust accurate statistical annotation of general text. In *Proc. of 3rd LREC*, pages 1499–1504, 2002.

[35] Y. Zhao and G. Karypis. Criterion functions for document clustering: Experiments and analysis. Technical Report 01-40, University of Minnesota, Department of Computer Science, Minneapolis, 2003.

[36] G. Karypis. CLUTO: A clustering toolkit. Technical Report 02-017, University of Minnesota, 2002. Available at http://www-users.cs.umn.edu/ karypis/cluto/.

[37] Ronald J. Brachman. On the epistemological status of semantic networks. In Nicholas V. Findler, editor, *Associative Networks: Representation and Use of Knowledge by Computers*, pages 3–50. Academic Press, New York, 1979.

[38] R. M. W. Dixon. *A new approach to English grammar, on semantic principles*. Oxford, 1991.

[39] P. Pantel and M. Pennacchiotti. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proc. of ACL / COLING*, pages 113–120, Sydney, 2006.

[40] P. Cimiano and J. Wenderoth. Automatically learning qualia structures from the web. In *Proc. ACL workshop on Deep Lexical Acquisition*, Ann Arbor, MI, 2005.

[41] A. Almuhareb and M. Poesio. MSDA: A word sense discrimination algorithm. In *Proc. of ECAI*, Riva del Garda, August 2006.

A. The 402 concepts dataset

WordNet Unique Beginner	Category	Concepts
animal	**animal**	bear, bull, camel, cat, cow, deer, dog, elephant, horse, kitten, lion, monkey, mouse, oyster, puppy, rat, sheep, tiger, turtle, zebra
possession	**assets**	allocation, allotment, capital, credit, dispensation, fund, gain, gold, hoard, income, interest, investment, margin, mortgage, payoff, profit, quota, taxation, trove, venture, wager
natural phenomenon	**atmospheric phenomenon**	airstream, aurora, blast, clemency, cloud, cloudburst, crosswind, cyclone, drizzle, fog, hurricane, lightning, rainstorm, sandstorm, shower, snowfall, thunderstorm, tornado, twister, typhoon, wind
substance	**chemical element**	aluminium, bismuth, cadmium, calcium, carbon, charcoal, copper, germanium, helium, hydrogen, iron, lithium, magnesium, neon, nitrogen, oxygen, platinum, potassium, silver, titanium, zinc
person	**creator**	architect, artist, builder, constructor, craftsman, designer, developer, farmer, inventor, maker, manufacturer, musician, originator, painter, photographer, producer, tailor
location	**district**	anchorage, borderland, borough, caliphate, canton, city, country, county, kingdom, land, metropolis, parish, prefecture, riverside, seafront, shire, state, suburb, sultanate, town, village
natural object	**edible fruit**	apple, banana, berry, cherry, grape, kiwi, lemon, mango, melon, olive, orange, peach, pear, pineapple, strawberry, watermelon

Table 4. The balanced dataset of 402 concepts used in the experiments (I)

WordNet Unique Beginner	Category	Concepts
plant	**tree**	acacia, casuarina, chestnut, cinchona, coco, conifer, fig, hornbeam, jacaranda, lime, mandarin, mangrove, oak, palm, pine, pistachio, rowan, samba, sapling, sycamore, walnut
artifact	**vehicle**	aircraft, airplane, automobile, bicycle, boat, car, cruiser, helicopter, motorcycle, pickup, rocket, ship, truck, van
feeling	**feeling**	anger, desire, fear, happiness, joy, love, pain, passion, pleasure, sadness, sensitivity, shame, wonder
act	**game**	baccarat, basketball, beano, bowling, chess, curling, faro, football, golf, handball, keno, lotto, nap, raffle, rugby, soccer, softball, tennis, volleyball, whist
state	**illness**	acne, anthrax, arthritis, asthma, cancer, cholera, cirrhosis, diabetes, eczema, flu, glaucoma, hepatitis, leukemia, malnutrition, meningitis, plague, rheumatism, smallpox
relation	**legal document**	acceptance, assignment, bill, bond, check, cheque, constitution, convention, decree, draft, floater, law, licence, obligation, opinion, rescript, sequestration, share, statute, straddle, treaty
quantity	**monetary unit**	cent, cordoba, dinar, dirham, dollar, drachma, escudo, fen, franc, guilder, lira, mark, penny, peso, pound, riel, rouble, rupee, shilling, yuan, zloty
motivation	**motivation**	compulsion, conscience, deterrence, disincentive, dynamic, ethics, impulse, incentive, incitement, inducement, life, mania, morality, motivator, obsession, occasion, possession, superego, urge, wanderlust

Table 5. The balanced dataset of 402 concepts used in the experiments (II)

WordNet Unique Beginner	Category	Concepts
cognition	**pain**	ache, backache, bellyache, burn, earache, headache, lumbago, migraine, neuralgia, sciatica, soreness, sting, stinging, stitch, suffering, tenderness, throb, toothache, torment
attribute	**physical property**	chill, coolness, deflection, diameter, extension, glow, heaviness, length, mass, momentum, plasticity, poundage, radius, reflexion, shortness, snap, stretch, temperature, visibility, weight
event	**social occasion**	ball, celebration, ceremony, commemoration, commencement, coronation, dance, enthronement, feast, fete, fiesta, fundraiser, funeral, graduation, inaugural, pageantry, party, prom, rededication, wedding
group	**social unit**	agency, branch, brigade, bureau, club, committee, company, confederacy, department, divan, family, house, household, league, legion, nation, office, platoon, team, tribe, troop
shape	**solid**	concavity, corner, crinkle, cube, cuboid, cylinder, dodecahedron, dome, droop, fluting, icosahedron, indentation, jag, knob, octahedron, ovoid, ring, salient, taper, tetrahedron
time	**time**	aeon, date, day, epoch, future, gestation, hereafter, menopause, moment, nonce, period, quaternary, today, tomorrow, tonight, yesterday, yesteryear

Table 6. The balanced dataset of 402 concepts used in the experiments (III)

Learning Expressive Ontologies

Johanna VÖLKER [1] and Peter HAASE and Pascal HITZLER
Institute AIFB, University of Karlsruhe, Germany

Abstract. The automatic extraction of ontologies from text and lexical resources has become more and more mature. Nowadays, the results of state-of-the-art ontology learning methods are already good enough for many practical applications. However, most of them aim at generating rather inexpressive ontologies, i.e. bare taxonomies and relationships, whereas many reasoning-based applications in domains such as bioinformatics or medicine rely on much more complex axiomatizations. Those are extremely expensive if built by purely manual efforts, and methods for the automatic or semi-automatic construction of expressive ontologies could help to overcome the knowledge acquisition bottleneck. At the same time, a tight integration with ontology evaluation and debugging approaches is required to reduce the amount of manual post-processing which becomes harder the more complex learned ontologies are. Particularly, the treatment of logical inconsistencies, mostly neglected by existing ontology learning frameworks, becomes a great challenge as soon as we start to learn huge and expressive axiomatizations. In this chapter we present several approaches for the automatic generation of expressive ontologies along with a detailed discussion of the key problems and challenges in learning complex OWL ontologies. We also suggest ways to handle different types of inconsistencies in learned ontologies, and conclude with a visionary outlook to future ontology learning and engineering environments.

Keywords. Ontology Learning, Reasoning, Ontology Evolution, Ontology Engineering

1. Introduction

During the last decade ontologies have become an important means for knowledge interchange and integration. They are used for corporate knowledge management, web portals and communities, semantic search, and web services. A couple of ontology languages have emerged as wide-spread means for ontology representation - among them the web ontology language, OWL, which provides a powerful formalism for knowledge representation and reasoning. OWL was proposed as a world-wide standard by the W3C committee, and several subsets of the OWL language with different expressivity have been defined in order to meet the demands of a great variety of semantic applications, and of course the great vision of the semantic web.

However, the realization of the semantic web as envisioned by Tim-Berners Lee is still hampered by the lack of ontological resources. Building ontologies is a difficult and time-consuming task. It usually requires to combine the knowledge of domain experts

[1]Corresponding Author: Johanna Völker, Institut AIFB, Universität Karlsruhe (TH), 76128 Karlsruhe, Germany; E-mail: voelker@aifb.uni-karlsruhe.de.

with the skills and experience of ontology engineers into a single effort with high demand on scarce expert resources. We believe that this bottleneck currently constitutes a severe obstacle for the transfer of semantic technologies into practice.

In order to address this bottleneck, it is reasonable to draw on available data, applying automated analyses to create ontological knowledge from given resources or to assist ontology engineers and domain experts by semi-automatic means. Accordingly, a significant number of ontology learning tools and frameworks has been developed aiming at the automatic or semi-automatic construction of ontologies from structured, unstructured or semi-structured documents. The current state-of-the-art in lexical ontology learning is able to generate ontologies that are largely *informal* or *lightweight* ontologies in the sense that they are limited in their expressiveness and often only consist of concepts organized in a hierarchy. While less expressive, informal ontologies have proven useful in certain application scenarios – an observation that also resonates with the so-called Hendler hypothesis [1]: "A little semantics goes a long way." – more and more people tend to see the future of semantic technologies in application scenarios such as e-business or bio-informatics which require large scale reasoning over complex domains. These knowledge-intensive applications even more than the semantic web depend on the availability of expressive, high-quality ontologies. However, both quality and expressivity of the ontologies which can be generated by the state-of-the-art ontology learning systems fail to meet the expectations of people who argue in favor of powerful, knowledge-intensive applications based on ontological reasoning. While it might seem infeasible to improve upon both at the same time, we argue that learning more expressive ontologies (e.g. by adding disjointness axioms) does not only yield sufficiently good results, but may also help in the task of automatic ontology evaluation, thus improving the overall quality of learned ontologies.

In this chapter, we present two complementary approaches to the automatic generation of expressive ontologies suitable for reasoning-based applications. The first approach is essentially based on a syntactic transformation of natural language definitions into description logic axioms. It hinges critically on the availability of sentences which have definitory character, like "*Enzymes are proteins that catalyse chemical reactions.*" Such sentences could be obtained e.g. from glossaries or software documentation related to the underlying ontology-based application. We exemplify this approach with definitions taken from a fishery glossary used in a case study at the *Food and Agriculture Organization of the United Nations* (FAO).

The second approach relies on a machine learning classifier for determining disjointness of any two classes. For its implementation, we developed a variety of different methods to automatically extract lexical and logical features which we believe to provide a solid basis for learning disjointness. These methods take into account the structure of the ontology, associated textual resources, and other types of data sources in order to compute the likeliness of two classes to be disjoint. The features obtained from these methods are used to build an overall classification model which we evaluated against a set of manually created disjointness axioms.

The support for reasoning is the major benefit of the use of expressive ontologies grounded in logics. Reasoning can be used in different phases of the lifecycle of an ontology. At runtime, reasoning allows to derive conclusions from the ontology, e.g. for the purpose of query answering over the ontology. At development time, reasoning can be used to validate the ontology and check whether it is non-contradictory, i.e. free of log-

ical inconsistencies. The more expressive the ontology language, the more precisely the intended meaning of a vocabulary can be specified, and consequently, the more precise conclusions can be drawn. Introducing disjointness axioms, for example, greatly facilitates consistency checking and the automatic evaluation of individuals in a knowledge base with regards to a given ontology.

Another particularly important topic for ontology learning is the challenge of dealing with inconsistencies. The reason lies in the fact that all ontology learning approaches generate knowledge that is afflicted with various forms of imperfection. The causes of imperfection may already be in the data sources from which the ontologies are generated, or they may be introduced by the ontology learning algorithms. To address this problem, we illustrate in this chapter how ontology learning can be combined with *consistent ontology evolution*, a reasoning-supported process to guarantee that the learned ontologies are kept consistent as the ontology learning procedures generate changes to the ontology over time.

The chapter is structured as follows. In the subsequent section we provide a brief introduction to the ontology language OWL and the main reasoning tasks in OWL as a Description Logic. In Section 3 we present ontology learning methods for learning expressive ontologies, particularly focusing on learning complex concept descriptions and disjointness axioms as two expressive language elements. In Section 4 we critically discuss the current state-of-the-art in learning expressive ontologies, analyzing problems and open issues. In Section 5 we show how inconsistencies can be dealt with in the context of ontology learning to guarantee a consistent evolution of the learned ontologies. We then propose a way of integrating ontology learning and evolution into the ontology lifecycle in Section 6. In Section 7 we present experiments in a concrete application scenario in the domain of fishery ontologies before concluding in Section 8.

2. OWL Ontologies and Reasoning Tasks

Traditionally, a number of different knowledge representation paradigms have competed to provide languages for representing ontologies, including most notably description logics and frame logics. With the advent of the OWL Web Ontology Language, developed by the Web Ontology Working Group and recommended by the World Wide Web Consortium (W3C), a standard for the representation of ontologies has been created. Adhering to this standard, we base our work on the OWL language (in particular OWL DL, as discussed below) and describe the developed formalisms in its terms.

2.1. OWL as a Description Logic

The OWL ontology language is based description logics, a family of class-based knowledge representation formalisms. In description logics, the important notions of a domain are described by means of concept descriptions that are built from *concepts* (also referred to as *classes*), *roles* (also referred to as *properties* or *relations*), denoting relationships between things, and *individuals* (also referred to as *instances*). It is now possible to state facts about the domain in the form of axioms. Terminological axioms make statements about how concepts or roles are related to each other, assertional axioms (sometimes also called *facts*) make statements about the properties of individuals of the domain.

We here informally introduce the language constructs of the description logic $\mathcal{SHOIN}$, the description logic underlying OWL DL. For the correspondence between our notation and various OWL DL syntaxes, see [2]. In the description logic $\mathcal{SHOIN}$, we can build complex classes from atomic ones using the following constructors:

- $C \sqcap D$ (intersection), denoting the concept of individuals that belong to both C and D,
- $C \sqcup D$ (union), denoting the concept of individuals that belong to either C or D,
- $\neg C$ (complement), denoting the concept of individuals that do not belong to C,
- $\forall R.C$ (universal restriction), denoting the concept of individuals that are related via the role R only with individuals belonging to the concept C,
- $\exists R.C$ (existential restriction), denoting the concept of individuals that are related via the role R with some individual belonging to the concept C,
- $\geq n\, R$, $\leq n\, R$ (qualified number restriction), denoting the concept of individuals that are related with at least (at most) n individuals via the role R.
- $\{c_1, \ldots, c_n\}$ (enumeration), denoting the concept of individuals explicitly enumerated.

Based on these class descriptions, axioms of the following types can be formed:

- concept inclusion axioms $C \sqsubseteq D$, stating that the concept C is a subconcept of the concept D,
- transitivity axioms $\mathsf{Trans}(R)$, stating that the role R is transitive,
- role inclusion axioms $R \sqsubseteq S$ stating that the role R is a subrole of the role S,
- concept assertions $C(a)$ stating that the individual a is in the extension of the concept C,
- role assertions $R(a, b)$ stating that the individuals a, b are in the extension of the role R,
- individual (in)equalities $a \approx b$, and $a \not\approx b$, respectively, stating that a and b denote the same (different) individuals.

Using the constructs above, we can make complex statements, e.g. expressing that two concepts are disjoint with the axiom $A \sqsubseteq \neg B$. This axioms literally states that A is a subconcept of the complement of B, which intuitively means that there must not be any overlap in the extensions of A and B.

In the design of description logics, emphasis is put on retaining decidability of key reasoning problems and the provision of sound and complete reasoning algorithms. As the name suggests, Description Logics are logics, i.e. they are formal logics with well-defined semantics. Typically, the semantics of a description logic is specified via *model theoretic semantics*, which explicates the relationship between the language syntax and the models of a domain.

An interpretation consists of a domain of interpretation (essentially, a set) and an interpretation function which maps from individuals, concepts and roles to elements, subsets and binary relations on the domain of interpretation, respectively. A description logic knowledge base consists of a set of axioms which act as constraints on the interpretations. The meaning of a knowledge base derives from features and relationships that are common in all possible interpretations. An interpretation is said *to satisfy a knowledge base*, if it satisfies each axiom in the knowledge base. Such an interpretation is called a *model* of the knowledge base. If there are no models, the knowledge base is said to be

inconsistent. If the relationship specified by some axiom (which may not be part of the knowledge base) holds in all models of a knowledge base, the axiom is said to be *entailed* by the knowledge base. Checking consistency and entailment are two standard reasoning tasks for description logics. Other reasoning tasks include computing the concept hierarchy and answering conjunctive queries.

2.2. Approaches to Dealing with Inconsistencies

Standard entailment as defined above is explosive, i.e. an inconsistent ontology has *all* axioms as consequences. Formally, if an ontology O is inconsistent, then for all axioms α we have $O \models \alpha$. In other words, query answers for inconsistent ontologies are completely meaningless, since for all queries the query answer will be *true*. To deal with the issue of potential inconsistencies in ontologies, we can choose from a number of alternative approaches [3]:

Consistent Ontology Evolution is the process of managing ontology changes by preserving the consistency of the ontology with respect to a given notion of consistency. The consistency of an ontology is defined in terms of consistency conditions, or invariants that must be satisfied by the ontology. The approach of consistent ontology evolution imposes certain requirements with respect to its applicability. For example, it requires that the ontology is consistent in the first place and that changes to the ontology can be controlled. In certain application scenarios, these requirements may not hold, and consequently, other means for dealing with inconsistencies in changing ontologies may be required.

Repairing Inconsistencies involves a process of diagnosis and repair: first the cause (or a set of potential causes) of the inconsistency needs to be determined, which can subsequently be repaired. Unlike the approach of consistent ontology evolution, repairing inconsistencies does not require to start with a consistent ontology and is thus adequate if the ontology is already inconsistent in the first place.

Reasoning with Inconsistent Ontologies does not try to avoid or repair the inconsistency (as in the previous two approaches), but simply tries to "live with it" by trying to return meaningful answers to queries, even though the ontology is inconsistent. In some cases consistency cannot be guaranteed at all and inconsistencies cannot be repaired, still one wants to derive meaningful answers when reasoning.

3. Learning Expressive OWL Ontologies

In the following we propose two complementary approaches to support the generation of expressive ontologies suitable for reasoning-based applications. After a brief overview of state-of-the-art methods for ontology learning we first present LExO, a prototypical implementation supporting the automatic generation of complex class descriptions from lexical resources (cf. Section 3.2). In Section 3.3, we focus on the task of creating disjointness axioms, and describe a classification-based approach using a combination of lexical and logical features for capturing disjointness.

3.1. Learning Ontology Elements and Basic Axioms

Ontology learning so far has focussed on the extraction of ontology elements such as concepts, instances or relations, as well as simple axioms. In this section, we briefly present some of the most frequently used methods for generating these types of primitives (for a more complete survey see, e.g. [4]).

3.1.1. Ontology Elements

Concepts and Instances. Different term weighting measures such as TFIDF, relative term frequency, entropy or C-value / NC-value [5] are used for identifying those terms which are most relevant for the domain of interest. Whereas the domain is modeled by a given document corpus each of the extracted noun phrases is assumed to represent either a concept or an instance. The distinction between concepts and instances is typically made depending on the part-of-speech information associated with its lexical representation, i.e. common and proper nouns.

General Relations. Approaches based on subcategorization frames rely on the assumption that ontological relationships are mostly represented by verbs and their arguments. Accordingly, selectional restrictions usually reflect domain and range restrictions of these relations [6,7]. In this line, Navigli and Velardi [8] extract taxonomic and non-taxonomic, i.e. general relations from glossaries and thesauri. Their approach is based on regular expressions further restricted by syntactic and semantic constraints, as well as a word sense disambiguation component which links extracted terms to WordNet. A more general approach which also considers attributive descriptions of concepts has been developed by Poesio and Almuhareb [9] who evaluated the use of a machine learning classifier for selecting the most distinctive attributes and relations for each concept. Unlabeled relations can be extracted by association rules which try to capture the semantic correlation of two elements based on their co-occurrences in the corpus. Unlike relations expressed by verbal or attributive phrases these anonymous relations have to be labeled by the ontology engineer in a post-processing step [10]. In addition to these two kinds of approaches, a number of methods for discovering particular types of relationships, e.g. part-of relations [11], have been developed so far.

3.1.2. Axioms

Subclass-of Relations. Many approaches for learning subclass-of relationships exploit hyponymy information in WordNet, or rely on Hearst patterns [12] as indicators for hyponymy relationships. Moreover, methods based on hierarchical clustering [13,14,15] and Formal Concept Analysis [16] have been developed for inducing taxonomies by grouping concepts with similar lexical context. Lexical context in its simplest form consists of the weighted co-occurrences of a term, but it may also include any kind of syntactic dependencies such as predicates, or prepositional complements associated with a given term.

Instance-of Relations. Distributional similarity, i.e. similarity based on lexical context, can be considered as an indicator for certain paradigmatic relationships, which makes it a suitable means for identifying, e.g. concept instantiation. Consequently, approaches such as [17] assign instances to the semantically most similar class by computing the contextual overlap. Other approaches to learning instance-of relationships rely on manually engineered or automatically acquired lexico-syntactic patterns [18,19].

```
( E1 () (fin) C
   ( 3 is (be) VBE i
      ( 2 farmer (~) N s
         ( 1 A (~) Det det )
      )
      ( 5 person (~) N pred
         ( E3 () (farmer) N subj 2 )
         ( 4 a (~) Det det )
         ( E0 () (fin) C rel
            ( 6 who (~) N whn 5 )
            ( 7 operates (operate) V i
               ( E4 () (who) N subj 5 )
               ( 9 farm (~) N obj
                  ( 8 a (~) Det det )
) ) ) ) ) )
```

Figure 1. Dependency Tree (Minipar)

```
rule: relative clause {
    arg_0:  //N
    arg_1:  arg_0 /C[@role='rel']
    arg_2:  arg_1 /V
    result: [equivalent 0 [and 0-1 2]]
}
rule: verb and object {
    arg_0:  //V
    arg_1:  arg_0 /N[@role='obj']
    result: [equivalent 0 [some 0-1 1]]
    result: [subObjectPropertyOf 0 0-1]
}
```

Figure 3. Transformation Rules

3.2. Learning Class Descriptions

LExO[2] (Learning EXpressive Ontologies) [20] is among the first approaches towards the automatic generation of ontologies featuring the full expressiveness of OWL DL. The core of LExO is a syntactic transformation of definitory natural language sentences into description logic axioms.

Given a natural language definition of a class, LExO starts by analyzing the syntactic structure of the input sentence. The resulting dependency tree is then transformed into a set of OWL axioms by means of manually engineered transformation rules. In the following, we provide a step-by-step example to illustrate the complete transformation process. For more (and more complicated) examples please refer to Section 7.

3.2.1. Example

Here, we assume that we would like to refine the description of the class *Farmer* which could be part of an agriculture ontology: *A farmer is a person who operates a farm.*

Initially, LExO applies the Minipar dependency parser [21] in order to produce a structured output as shown in Figure 1. Every node in the dependency tree contains information about the token such as its lemma (base form), its syntactic category (e.g. *N* (noun)) and grammatical role (e.g. *subj*), as well as its surface position. Indentation in this notation visualizes direct dependency, i.e. each child node is syntactically dominated by its parent.

This dependency structure is now being transformed into an XML-based format (see Figure 2) in order to facilitate the subsequent transformation process, and to make LExO more independent of the particular parsing component.

The set of rules which are then applied to the XML-based parse tree make use of XPath expressions for transforming the dependency structure into one or more OWL DL axioms. Figure 3 shows a few examples of such transformation rules in original syntax. Each of them consists of several arguments (e.g. *arg_1:...*), the values of which are defined by an optional prefix, i.e. a reference to a previously matched argument (*arg_0*), plus an XPath expression such as */C[@role='rel']* being evaluated relative to that prefix. The last lines of each transformation rule define one or more templates for OWL axioms, with variables to be replaced by the values of the arguments. Complex expressions such as *0-1* allow for "subtracting" individual subtrees from the overall tree structure. A more complete listing of the transformation rules we applied can be found further below.

[2] http://ontoware.org/projects/lexo/

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <C id="E1" pos="0">
    <VBE id="3" pos="3" role="i" phrase="is" base="be">
      <N id="2" pos="2" role="s" phrase="farmer">
        <Det id="1" pos="1" role="det" phrase="A"/>
      </N>
      <N id="5" pos="6" role="pred" phrase="person">
        <N id="E3" pos="4" role="subj" base="farmer" antecedent="2"/>
        <Det id="4" pos="5" role="det" phrase="a"/>
        <C id="E0" pos="7" role="rel">
          <N id="6" pos="8" role="whn" phrase="who" antecedent="5"/>
          <V id="7" pos="9" role="i" phrase="operates" base="operate">
            <N id="E4" pos="10" role="subj" base="who" antecedent="5"/>
            <N id="9" pos="12" role="obj" phrase="farm">
              <Det id="8" pos="11" role="det" phrase="a"/>
</N> </V> </C> </N> </VBE> </C> </root>
```

Figure 2. XML Representation of Dependency Tree

A minimal set of rules for building a complete axiomatization of the *Farmer* example could be, e.g. *Copula*, *Relative Clause* and *Transitive Verb Phrase* (see Table 1). The resulting list of axioms (see Figure 4) in KAON2[3] internal syntax is directly fed into the ontology management system which interprets the textual representation of these axioms, and finally builds an unfolded[4] class description as shown in Figure 5.

```
[equivalent lexo:a_farmer lexo:a_person_who_operates_a_farm]
[equivalent lexo:a_person_who_operates_a_farm [and lexo:a_person lexo:operates_a_farm]]
[equivalent lexo:operates_a_farm [some lexo:operates lexo:a_farm]]
```

Figure 4. Resulting Axioms

```
[equivalent lexo:a_farmer [and lexo:a_person [some lexo:operates lexo:a_farm]]]
```

Figure 5. Class Description (unfolded)

Obviously, all parts of this class description have to be normalized. After the normalization, the final, unfolded axiomatization in DL syntax reads:

$$\textit{Farmer} \equiv \textit{Person} \sqcap \exists \textit{operate.Farm}$$

Additionally, it may be necessary to map the ontology elements of the axiomatization to already existing content of the ontology before the results can be used to generate suggestions for ontology changes (cf. Section 6). As shown by the large body of research done in the domain of ontology mapping, this task is not trivial at all. Semantic ambiguities of labels (e.g. homonymy or polysemy), as well as the fact that a single entity or axiom in the ontology can have arbitrarily many lexicalizations – differing even in their syntactic category – make it necessary to consider a multitude of possible mappings. Moreover, idiomatic expressions, i.e. expressions the meaning of which cannot be directly derived from the meaning of their individual components, need to be treated properly. Therefore, in addition to integrating a state-of-the-art mapping framework, a significant degree of user involvement will be unavoidable in the end (see Section 6).

[3] `http://kaon2.semanticweb.org`

[4] By *unfolding*, a term borrowed from logic programming, we mean transformations like that of $\{A \equiv \exists R.B, C \equiv A \sqcap D\}$ to $\{C \equiv \exists R.B \sqcap D\}$. The specific for of output which we receive allows us to remove many of the newly generated class names by unfolding, in order to obtain a more concise output.

3.2.2. Transformation Rules

Table 1 gives an overview of the most frequently used transformation rules. Each row in the table contains the rule name (e.g. *Verb with Prepositional Complement*) and an expression describing the natural language syntax matched by that rule – like, for example, $V_0\ Prep_0\ NP(\textit{pcomp-n})_0$, where V_0 represents a verb, $Prep_0$ a preposition and $NP(\textit{pcomp-n})$ denotes a noun phrase acting as a prepositional complement. Please note that these expressions are very much simplified for the sake of presentation. The last column shows the OWL axioms generated in each case, where X denotes the atomic class name represented by the surface string of the complete expression matched by the regarding transformation rule.

It is important to emphasize that this set of rules is by no means exhaustive, nor does it define the only possible way to perform the transformation. In fact, there are many different modeling possibilities, and the choice and shape of the rules very much depends on the underlying application, the domain of interest or individual modeling preferences of the user (see example *Tetraploid* in Section 7).

Rule	Natural Language Syntax	OWL Axioms
Disjunction	NP_0 `or` NP_1	$X \equiv \text{NP}_0 \sqcup \text{NP}_1$
Conjunction	NP_0 `and` NP_1	$X \equiv \text{NP}_0 \sqcap \text{NP}_1$
Determiner	$\text{Det}_0\ \text{NP}_0$	$X \equiv \text{NP}_0$
Intersective Adjective	$\text{Adj}_0\ \text{NP}_0$	$X \equiv \text{Adj}_0 \sqcap \text{NP}_0$
Subsective Adjective	$\text{Adj}_0\ \text{NP}_0$	$X \sqsubseteq \text{NP}_0$
Privative Adjective	$\text{Adj}_0\ \text{NP}_0$	$X \sqsubseteq \neg\text{NP}_0$
Copula	NP_0 VBE NP_1	$\text{NP}_0 \equiv \text{NP}_1$
Relative Clause	NP_0 C(*rel*) VP_0	$X \equiv \text{NP}_0 \sqcap \text{VP}_0$
Number Restriction	V_0 Num NP(obj)_0	$X \equiv =\text{Num}\ \text{V}_0.\text{NP}_0$
Negation (not)	`not` $\text{V}_0\ \text{NP}_0$	$X \sqsubseteq \neg\exists \text{V}_0.\text{NP}_0$
Negation (without)	NP_0 `without` $\text{NP}(\textit{pcomp-n})_1$	$X \equiv \text{NP}_0 \sqcap \neg\text{with}.\text{NP}_1$
Participle	$\text{NP}_0\ \text{VP}(\textit{vrel})_0$	$X \equiv \text{NP}_0 \sqcap \text{VP}_0$
Transitive Verb Phrase	$\text{V}_0\ \text{NP}(\textit{obj})_0$	$X \equiv \exists \text{V}_0.\text{NP}_0$
Verb with Prep. Compl.	$\text{V}_0\ \text{Prep}_0\ \text{NP}(\textit{pcomp-n})_0$	$X \equiv \exists \text{V}_0\_\text{Prep}_0.\text{NP}_0$
Noun with Prep. Compl.	$\text{NP}_0\ \text{Prep}_0\ \text{NP}(\textit{pcomp-n})_1$	$X \equiv \text{NP}_0 \sqcap \exists \text{NP}_0\_\text{Prep}_0.\text{NP}_1$
...	...	...

Table 1. Transformation Rules

3.3. Learning Disjointness

The feasibility of learning disjointness based on simple lexical evidence in principle has already been shown in [22]. However, our experiments indicate that a single heuristic is not suitable for detecting disjointness with sufficiently high precision, i.e. better than an average human could do.

An extensive survey which we performed in order to collect experience with modeling disjoint classes revealed several problems frequently encountered by users who try to introduce disjointness axioms. Based on the results of this survey we developed a variety of different methods in order to automatically extract lexical *and* logical features which we believe to provide a solid basis for learning disjointness [27]. These methods take

into account the structure of the ontology, associated textual resources, and other types of data sources in order to compute the likeliness of two classes to be disjoint. The features obtained from these methods are used to train a classifier that decides whether any given pair of classes is disjoint or not. In the remainder of this Section, we will describe those features in more detail before concluding with a summary of our experiments and evaluation results.

3.3.1. Taxonomic Overlap

In description logics, two classes are disjoint *iff* their "taxonomic overlap"[5] *must* be empty. Because of the open world assumption in OWL, the individuals of a class do not necessarily have to *exist* in the ontology. Hence, the taxonomic overlap of two classes is considered not empty as long as there *could* be common individuals within the domain of interest which is modeled by the ontology, i.e. if the addition of such an individual does *not* generate an inconsistency.

We developed three methods which determine the likeliness for two classes to be disjoint by considering their overlap with respect to (i) *individuals* and *subclasses* in the ontology – or learned from a corpus of associated textual resources – and (ii) *Del.icio.us*[6] documents tagged with the corresponding class labels. An additional feature indicating disjointness is computed by determining whether (iii) any of the classes is *subsumed* by the other.

Ontology Individuals and subclasses which may serve as indicators for taxonomic overlap can be imported either from an ontology, or from a given corpus of text documents. In the latter case, `subclass-of` and `instance-of` relationships are extracted by different algorithms provided by the Text2Onto[7] ontology learning framework. A detailed description of these algorithms can be found in [23]. All taxonomic relationships – learned and imported ones – are associated with rating annotations $r_{\text{subclass-of}}$ (or $r_{\text{instance-of}}$ respectively) indicating the certainty of the underlying ontology learning framework in the correctness of its results. For imported relationships the confidence is 1.0.

The following feature $\mathrm{f}_{subclass-of}$ models the confidence for a pair (c_1, c_2) to be *not* disjoint based on the taxonomic overlap of c_1 and c_2 with respect to common subclasses (and in a similar way for instances):

$$\mathrm{f}_{subclass-of}(c_1, c_2) = \frac{\sum_{c \sqsubseteq c_1 \sqcap c_2} (r_{\text{subclass-of}}(c, c_1) \cdot r_{\text{subclass-of}}(c, c_2))}{\sum_{c \sqsubseteq c_1} r_{\text{subclass-of}}(c, c_1) + \sum_{c \sqsubseteq c_2} r_{\text{subclass-of}}(c, c_2)} \quad (1)$$

Del.icio.us Del.icio.us is a server-based system with a simple-to-use interface that allows users to organize and share bookmarks on the internet. It associates each URL with a description, a note, and a set of tags (i.e. arbitrary class labels). For our experiments, we collected $|U| = 75,242$ users, $|T| = 533,191$ tags and $|R| = 3,158,297$ resources, related by $|Y| = 17,362,212$ triples. The idea underlying the use of del.icio.us in this

[5] We use this notion to refer to the set of common individuals.

[6] `http://del.icio.us`

[7] `http://ontoware.org/projects/text2onto/`

case is that two labels which are frequently used to tag the same resource are likely to be disjoint, because users tend to avoid redundant labeling of documents.

In this case, we compute the confidence that c_1, c_2 are *not* disjoint as

$$\mathrm{f}_{del.icio.us}(c_1, c_2) = \frac{|\{d : c_1 \in t(d), c_2 \in t(d)\}|}{\sum_{c \in C} |\{d : c_1 \in t(d), c \in t(d)\}| + \sum_{c \in C} |\{d : c_2 \in t(d), c \in t(d)\}|} \tag{2}$$

where C is the set of all classes and $t(d)$ represents the set of del.icio.us *tags* associated with document d. The normalized number of co-occurrences of c_1 and c_2 (their respective labels to be precise) as del.icio.us tags aims at capturing the degree of association between the two classes.

Subsumption A particular case of taxonomic overlap is subsumption, which provides us with additional evidence with respect to the disjointness of two classes. If one class is a subclass of the other we assume these two classes to be *not* disjoint with a confidence equal to the likeliness $r_{\text{subclass-of}}$ associated with the `subclass-of` relationship.

3.3.2. Semantic Similarity

The assumption that a direct correspondence between the semantic similarity of two classes and their likeliness to be disjoint led to the development of three further methods: The first one implements the similarity measure described by [24] to compute the semantic similarity *sim* of two classes c_1 and c_2 with respect to *WordNet* [25]:

$$\mathrm{f}_{wordnet}(c_1, c_2) = \mathrm{sim}(s_1, s_2) = \frac{2 * \mathrm{depth}(\mathrm{lcs}(s_1, s_2))}{\mathrm{depth}(s_1) + \mathrm{depth}(s_2)} \tag{3}$$

where s_i denotes the first sense of c_i, $i \in \{1, 2\}$ with respect to WordNet, and $\mathrm{lcs}(s_1, s_2)$ is the least common subsumer of s_1 and s_2. The depth of a node n in WordNet is recursively defined as follows: $\mathrm{depth}(\mathrm{root}) = 1$, $\mathrm{depth}(\mathrm{child}(n)) = \mathrm{depth}(n) + 1$.

The second method measures the distance of c_1 and c_2 with respect to the given background *ontology* by computing the minimum length of a path of `subclass-of` relationships connecting c_1 and c_2.

$$\mathrm{f}_{ontology}(c_1, c_2) = \min_{p \in \mathrm{paths}(c_1, c_2)} \mathrm{length}(p) \tag{4}$$

And finally, the third method computes the similarity of c_1 and c_2 based on their *lexical context*. Along with the ideas described in [17] we exploit Harris' distributional hypothesis [26] which claims that two words are semantically similar to the extent to which they share syntactic contexts.

For each occurrence of a class label in a corpus of textual documents (see preliminaries of this section) we consider all the lemmatized tokens in the same sentence (except for stop words) as potential features in the context vector of the corresponding class. After the context vectors for both classes have been constructed, we assign weights to all

features by using a modified version of the TFIDF formula. It differs from the original version in that it aims at measuring the significance of terms with respect to the classes they co-occur with rather than the documents in which they are contained.

Let $v_c = (f_{c,1}, ...f_{c,n})$, $n \geq 1$ be the context vector of class c where each $f_{c,j}$ is the frequency of token t_j in the context of c. Then we define $TF_{c,j} = f_{c,j} \cdot (\sum_{1 \leq k \leq n} f_{c,k})^{-1}$ and $DF_j = |\sum_{c' \in C} f_{c',j} > 0|$, where C is the set of all classes. Finally, we get $f'_{c,j} = TF_{c,j} \cdot \log(|C| \cdot (DF_j)^{-1})$, hence $v'_c = (f'_{c,1}, ...f'_{c,n})$. Given the *weighted* context vectors v'_{c_1} and v'_{c_2} the confidence in c_1 and c_2 being *not* disjoint is defined as $f_{context}(c_1, c_2) = v'_{c_1} \cdot v'_{c_2} \cdot (\|v'_{c_1}\| \|v'_{c_2}\|)^{-1}$ which corresponds to the cosine similarity of v'_{c_1} and v'_{c_2}.

3.3.3. Patterns

Since we found that disjointness of two classes is often reflected by human language, we defined a number of lexico-syntactic patterns to obtain evidence for disjointness relationships from a given corpus of textual resources. The first type of pattern is based on enumerations as described in [22]. The underlying assumption is similar to the idea described in section 3.3.1, i.e. terms which are listed separately in an enumeration mostly denote disjoint classes. Therefore, from the sentence

The pigs, cows, horses, ducks, hens and dogs all assemble in the big barn, thinking that they are going to be told about a dream that Old Major had the previous night.[8]

we would conclude that *pig*, *cow*, *horse*, *duck*, *hen* and *dog* denote disjoint classes. This is because we believe that – except for some idiomatic expressions it would be rather unusual to enumerate overlapping classes such as *dogs* and *sheep dogs* separately which would result in semantic redundancy. More formally:

Given an enumeration of noun phrases $NP_1,\ NP_2,\ \ldots,\ (and|or)\ NP_n$ we conclude that the concepts $c_1,\ c_2,\ \ldots,\ c_k$ denoted by these noun phrases are pairwise disjoint, where the confidence $\mathrm{f}_{enumeration}(c_1, c_2)$ for the disjointness of two concepts c_1 and c_2 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.

The second type of pattern is designed to capture more explicit expressions of disjointness in natural language by phrases such as $either\ NP_1\ or\ NP_2$ or $neither\ NP_1\ nor\ NP_2$. For both types of patterns we compute the confidence for the disjointness of two classes c_1 and c_2 as follows:

$$\mathrm{f}_{pattern}(c_1, c_2) = \frac{\mathrm{freq}(c_1, c_2)}{\sum_{j \neq 1} \mathrm{freq}(c_1, c_j) + \sum_{i \neq 2} \mathrm{freq}(c_i, c_2)} \tag{5}$$

where $freq(c_i, c_j)$ is the number of patterns providing evidence for the disjointness of c_i and c_j with $0 \leq i, j \leq |C|$ and $i \neq j$.

3.3.4. Evaluation

We evaluated the approach by performing a comparison of learned disjointness axioms with a data set consisting of 2,000 pairs of classes, each of them manually tagged by

[8] George Orwell, *Animal Farm*, Secker & Warburg, London, 1945

Table 2. Evaluation against Majority Vote 100% (ADTree)

Dataset	P			R			F			Acc	$Acc_{majority}$
	+	−	avg.	+	−	avg.	+	−	avg.		
$Experts$	0.896	0.720	0.808	0.903	0.703	0.803	0.899	0.712	0.806	0.851	0.738
$Students$	0.866	0.790	0.828	0.942	0.599	0.771	0.903	0.681	0.792	0.851	0.734
$Avg.$	0.881	0.755	0.818	0.923	0.651	0.787	0.901	0.697	0.799	0.851	0.736
All	0.934	0.823	0.879	0.946	0.789	0.868	0.940	0.805	0.873	0.909	0.760

6 human annotators – 3 students and 3 ontology experts[9]. A 10-fold cross validation against those pairs of classes which were tagged identically by all the annotators showed an accuracy between 85.1% and 90.9%, which is significantly higher than the majority baseline (cf. Table 2).

In order to find out which classification features contributed most to the overall performance of the classifier we performed an analysis of our initial feature set with respect to the gain ratio measure. The ranking produced for data set C clearly indicates an exceptionally good performance of the features taxonomic overlap (Section 3.3.1), similarity based on WordNet and lexical context (Section 3.3.2), and del.icio.us (Section 3.3.1). The contribution of other features such as the one presented in Section 3.3.3 relying on lexico-syntactic patterns seems to be less substantial. However, as the classification accuracy tested on every single feature is always below the overall performance, the combination of all features is necessary to achieve a very good overall result.

4. Discussion

The syntactic transformation proposed in Section 3.2 creates a set of OWL axioms which can be used to extend the axiomatization of any given class in an ontology. Our naive implementation of this approach is as simple as efficient, but obviously requires a significant amount of manual or automatic post-processing. This is to a major extent due to a number of problems which relate to limitations of the linguistic analysis and the transformation process, as well as fundamental differences between lexical and ontological semantics. In the following we will discuss some of these problems in more detail, and present possible solutions.

Although the transformation takes into account some aspects of lexical semantics, it is certainly not capable of capturing much of the intension of the terms involved in the natural language expression that serves as an input for the transformation process. Much of the meaning of the resulting axioms is still brought in by the semantics of the underlying natural language terms. This does not necessarily constitute a significant problem as long as the semantics of the description logic expressions is sufficiently "in line" with the lexical semantics of the terms involved in their formation. Actually, the semantics of ontological elements – not of the constructs of the ontology language, but of the classes, properties and instances defined by means of these constructs – will always be grounded to some extent in natural language semantics.

[9]The complete data set is available from `http://ontoware.org/projects/leda/`.

As it is impossible to express all possible aspects of a concept's meaning by virtue of description logic axioms, natural language labels and comments undoubtedly play a key role in ontological knowledge representation. In fact, an ontology without natural language labels attached to classes or properties is almost useless, because without this kind of grounding it is very difficult, if not impossible, for humans to map an ontology to their own conceptualization, i.e. the ontology lacks human-interpretability.

However, a grounding of ontologies in natural language is highly problematic due to different semantics and the dynamic nature of natural language. It is important to mention that many problems linked to either of these aspects are not necessarily specific to ontology learning approaches such as the one we present in this chapter. Since the way people conceive and describe the world is very much influenced by the way they speak and vice-versa (also known as the *Sapir-Whorf hypothesis*), ontology engineering is often subject to our intuitive understanding of natural language semantics. Some problems that relate to differences between ontological and lexical semantics are discussed in the following.

Lexical Semantics. The semantics of lexical relations fundamentally differs from the model-theoretic semantics of ontologies. While lexical relations such as hyponymy, antonymy or synonymy are defined over lexemes, ontological relations are used for relating classes.[10] And it is not obvious in all cases how to map words – especially very abstract notions – to classes, as their extension often remains unclear.

For practical reasons it might be sensible to assume a correspondence between lexical relations and some types of axioms. Indeed, traditional ontology learning approaches often rely on information about hyponymy and synonymy for creating subsumption hierarchies [12], or meronymy for identifying part-of relationships [28]. However, a one-to-one mapping between lexical and ontological semantics is problematic for various reasons. Just to mention a few of them, true synonymy is very rare if existent at all in natural language. And since tiny differences in meaning may be significant depending on the modeling granularity of an ontology, synonymy cannot always be mapped to equivalence in a straightforward way. Second, lexical relations such as meronymy do not need to be transitive even if their ontological counterparts are. It is also important to mention that hyponymy in pattern-based ontology learning is often confused with *para-hyponymy* [29] as those patterns are not able to capture the necessity condition which holds for regular hyponymy[11]. Finally, one has to be aware of the fact that such a mapping between lexical and model-theoretic semantics may affect the formal correctness of an ontology – even more, if ontology learning or engineering exclusively relies on lexico-syntactic clues for inferring lexical relationships. Due to the informal character of natural language it is no trouble to say, for instance, "*A person is an amount of matter*". But from the perspective of formal semantics this might be problematic as pointed out by [30].

[10] For example, each of these classes could be associated with one or more natural language expressions describing the intended meaning (intension) of the class. And still, since hyponymy is not "transitive" over (near-)synonymy it is not necessarily the case that all mutually synonymous words associated with a subclass are hyponyms of all synonymous words associated with its superclass.

[11] Interestingly, this necessity condition parallels the *rigidity* constraint as defined by the OntoClean methodology [30]. A tool such as AEON [36] could therefore help to automatically detect both, cases of formally incorrect subsumption as well as para-hyponymy relationships (e.g. "A *dog* is a *pet*.").

Dynamics of Natural Language. Further problems with respect to the use of natural language in ontology engineering relate to the way in which semantics are defined. While ontologies have a clear model-theoretic semantics, the semantics of lexical relations is defined by so-called *diagnostic frames*, i.e. by typical sentences describing the context in which a pair of words may or may not occur given a certain lexical relation among them. This way of defining lexical relations does not guarantee for stable semantics, since natural languages, other than ontology representation languages, are dynamic. That means, each (*open-class*) word slightly changes its meaning every time it is used in a new linguistic context. These *semantic shifts*, if big enough, can affect the lexical relationships between any pair of words. And considering that natural language expressions are regularly used for the grounding of ontologies they can potentially lead to semantic "inconsistencies", i.e. conflicting intensional descriptions. This kind of inconsistencies can be avoided by more precise, formal axiomatizations of ontological elements. However, it is an open issue how many axioms are required to "pin down" the meaning of a given class or property.

The proposed approach for learning disjointness axioms (see Section 3.3) is affected by similar problems as it relies among others upon ontology learning methods for capturing the potential overlap of any two concepts. However, one of the main weaknesses of this approach might be that it crucially depends on the quality of the manually created training data sets. Our user study [27] revealed a number of difficulties and misunderstandings human ontology engineers had while creating disjointness axioms. For example, a simple taxonomy along with natural language labels was often not sufficient for disambiguating the sense of a given concept. And people were confused if the intensions of two concepts were disjoint while their extensions were not – or vice versa (e.g. *Woman* and *US President*).

5. Dealing with Inconsistencies in Ontology Learning

One of the major problems of learning ontologies is the potential introduction of inconsistencies. These inconsistencies are a consequence of the fact that it is inherent in the ontology learning process that the acquired ontologies represent *imperfect* information.

According to [31], we can distinguish three different causes of imperfection. 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 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 of some information. In particular, an 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.

Ignoring the fact that learned ontologies contain uncertain and thus potentially contradicting information would result in highly inconsistent ontologies, which do not allow meaningful reasoning. In the following we show how inconsistencies can be dealt with in the process of ontology learning. In particular, we show how the concept of consistent ontology evolution can be applied in the context of ontology learning. To begin with, we define the notion of consistency more precisely.

5.1. Logical Consistency

Logical consistency addresses the question whether the ontology is "semantically correct", i.e. does not contain contradicting information. We say *logical consistency* is satisfied for an ontology O if O is satisfiable, i.e. if O has a model. Please note that because of the monotonicity of the considered logic, an ontology can only become logically inconsistent by adding axioms: If a set of axioms is satisfiable, it will still be satisfiable when any axiom is deleted. Therefore, we only need to check the consistency for ontology change operations that add axioms to the ontology. Effectively, if $O \cup \{\alpha\}$ is inconsistent, in order to keep the resulting ontology consistent some of the axioms in the ontology O have to be removed or altered.

Example. Suppose, we have generated an ontology containing the following axioms: Pig $\sqsubseteq$ Mammal, Human $\sqsubseteq$ Mammal, Human $\sqsubseteq$ Biped (humans walk on two legs), Pig $\sqsubseteq$ Quadruped (pigs walk on two legs), Biped $\sqsubseteq \neg$Quadruped (Bipeds and Quadrupeds are disjoint), Pig(OldMajor). This ontology is logically consistent.

Suppose we now learn from some source that Old Major walks on two legs and want to add the axiom Biped(OldMajor). Obviously, this ontology change operation would result in an inconsistent ontology.

5.2. Consistent Ontology Evolution

As we have already discussed in Section 2.2, the most adequate approach to dealing with inconsistencies in ontology learning is by realizing a consistent evolution of the ontology. The goal of consistent ontology evolution is the resolution of a given ontology change in a systematic manner by ensuring the consistency of the whole ontology. It is realized in two steps:

1. *Inconsistency Localization*: This step is responsible for checking the consistency of an ontology with the respect to the ontology consistency definition. Its goal is to find "parts" in the ontology that do not meet consistency conditions;
2. *Change Generation*: This step is responsible for ensuring the consistency of the ontology by generating additional changes that resolve detected inconsistencies.

The first step essentially is a diagnosis process. There are different approaches how to perform the diagnosis step [32]. A typical way to diagnose an inconsistent ontology is to try to find a minimal inconsistent subontology, i.e. a minimal set of contradicting axioms. Formally, we call an ontology O' a *minimal inconsistent subontology* of O,

if $O' \subseteq O$ and O' is inconsistent and for all O'' with $O'' \subset O'$, O'' is consistent. Intuitively, this definition states that the removal of any axiom from O' will result in a consistent ontology. A simple way of finding a minimal inconsistent subontology is as follows: We start with one candidate ontology containing initially only the axiom that was added to the ontology as part of the change operation. As long as we have not found an inconsistent subontology, we create new candidate ontologies by adding axioms (one at a time) that are in some way connected with the axioms in the candidate ontology. One simple, but useful notion of connectedness is structural connectedness: We say that axioms are structurally connected if they refer to shared ontology entities. Once the minimal inconsistent ontology is found, it is by definition sufficient to remove any of the axioms to resolve the inconsistency.

In our previous example, a minimal inconsistent subontology would consist of the axioms Pig $\sqsubseteq$ Quadruped, Biped $\sqsubseteq$ $\neg$Quadruped, Pig(OldMajor), and Biped(OldMajor). The removal of any of the axioms would result in a consistent ontology.

While the removal of any of the axioms from a minimal inconsistent subontology will resolve the inconsistency, the important question of course is deciding *which* axiom to remove. This problem of only removing dispensable axioms requires some semantic selection functions capturing the relevance of particular axioms. These semantic selection functions can for example exploit information about the confidence in the axioms that allows us to remove "less correct" axioms. In the resolution of the changes we may decide to remove the axioms that have the lowest confidence, i.e. those axioms that are most likely incorrect. We are thus able to incrementally evolve an ontology that is (1) consistent and (2) captures the information with the highest confidence. For details of such a process and evaluation results, we refer the reader to [22].

Based on the discussions above, we can now outline an algorithm (c.f. Algorithm 1) to ensure the consistent evolution of a learned ontology.

Algorithm 1 Algorithm for consistent ontology learning

Require: A consistent ontology O
Require: A set of ontology changes OC
for all $\alpha \in OC, r_{\text{conf}}(\alpha) \geq t$ **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 $r_{\text{conf}}(\alpha') \leq r_{\text{conf}}(\alpha)$ **then**
        $\alpha^- := \alpha'$
      end if
    end for
    $O := O \setminus \{\alpha^-\}$
  end while
end for

Starting with some consistent ontology O, we incrementally add all axioms generated from the ontology learning process – contained in the set of ontology changes OC

– whose confidence is equal to or greater than a given threshold t. If adding the axioms leads to an inconsistent ontology, we localize the inconsistency by identifying a minimal inconsistent subontology. 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. It may be possible that one added axiom introduced multiple inconsistencies. For this case, the above inconsistency resolution has to be applied iteratively.

5.3. Context Information for the Resolution of Inconsistencies

Besides the general notion of confidence used above, we may rely on various other forms of contextual information to obtain a ranking of the axioms for the resolution of inconsistencies. In the following we discuss what kind of contextual information can be automatically generated by the ontology learning algorithms:

Axiomatic support. The axiomatic support can be defined as the relative number of times a particular axiom was generated by an ontology learning component. Since the methods which are applied by such a component can generate axioms of different complexity, it may be necessary to define the axiomatic support based on some normal form of the axioms.

Provenance information. Whenever ontologies are automatically generated from structured or unstructured resources, and in particular if these resources are part of the World Wide Web, the reliability of the results depends on the trustworthiness and quality of these resources. Therefore, associating provenance information with the learned ontology elements does not only increase the traceability of the results (as the user can track individual elements back to the resources they have been extracted from), but also helps to estimate the correctness of the results.

Mapping confidence. If the ontology learning task is to extend a given ontology (as opposed to generating an ontology from scratch) it can be necessary to map all newly introduced properties and class descriptions to already existing ontology elements. This helps to avoid unnecessary extensions to the ontology, but at the same time introduces additional uncertainty caused by incorrect mappings, as suggested in Section 3.2.

Rule reliability. Ontology learning approaches based on syntactic transformation rules (c.f. Section 3.2.2), or lexico-syntactic patterns [18] often make certain assumptions about the reliability of their rules or patterns. These (implicit or explicit) assumptions typically being supported by empirical data can be used to estimate the correctness of the ontology learning results.

Classifier confidence. Supervised ontology learning approaches such as the one presented in Section 3.3 rely on a classification model built from training examples. The classifier that can be constructed from this model will make predictions for previously unseen data (e.g. instances to be classified as belonging to a certain class, or pairs of classes being disjoint or not) with a confidence value that depends on the classifier type.

Relevance. In general, ontology learning from text is based on the assumption that the domain of interest, i.e. the domain to be modeled by the learned ontology, is given by means of the underlying document corpus. It therefore seems natural that approaches

such as [34] try to evaluate the quality, and in particular the domain coverage, of learned ontologies by comparing them to the corpus. Similarly, the relevance of an individual ontology element can be estimated based on the pieces of evidence (e.g. explicit mentions) in the corpus.

6. Integrating Learning and Evolution into the Ontology Lifecycle

In this section we sketch our vision of a semi-automatic ontology engineering process, which integrated our previously described methods for ontology learning and evolution along with an elaborate methodology. We describe the potential role of our approaches within this scenario and identify the missing components.

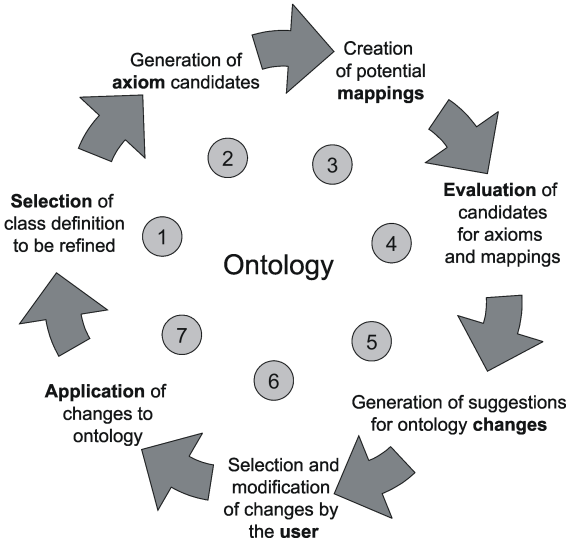

Figure 6. Ontology Evolution Process

The overall scenario we envision for the evolution of expressive OWL ontologies is a semi-automatic cyclic process of ontology learning, evaluation and refinement as depicted by Figure 6. The process starts with a relatively inexpressive ontology, possibly a bare taxonomy, which is supposed to be enriched and refined to meet the requirements, e.g. of a reasoning-based application. In each iteration of the process, the user selects the class to be refined, and optionally specifies appropriate resources for the ontology generation phase (Step 1) such as

- manual user input,
- comments contained in the ontology,
- definitions extracted from ontology engineering discussions by email or Wiki,
- software documentation of the underlying application,
- available glossaries and encyclopedias (e.g. Wikipedia), or
- textual descriptions of the domain which could be obtained by initiating a Google™search for definitions (e.g. "*define: DNS*").

A tool such as LExO (cf. Section 3.2) can analyze the given resources to identify and extract definitory sentences, i.e. natural language descriptions of the class previously selected by the user. These definitions are parsed and transformed into OWL DL axioms (Step 2) that can be presented to the user, if she wants to intervene at this point.

Otherwise, the system directly proceeds to the mapping phase which aims at relating the newly generated entities and axioms to elements in the initial ontology (Step 3). The outcome of this phase are a number of mapping axioms which can be added to the class axiomatization after being confirmed by the user. Then, methods for consistent ontology evolution check for logical inconsistencies or potential modeling errors (Step 4). Based on the learned axiomatization and additional mappings the system now suggests ontology changes or extensions to the user (Step 5). The user now revises the ontology by modifying or removing some of the axioms (Steps 6 and 7), before the whole process starts over again. Further entities, e.g. those introduced by previous iterations, can be refined until the user or application needs are satisfied.

When validating the ontology, it is certainly necessary to consider aspects beyond that of logical consistency. We point out two aspects which we judge to be of particular importance, namely how to aid the ontology engineer to ensure on the one hand a sufficiently high *quality* of the ontology and on the other hand the *completeness* of the modeling process in terms of the application domain.

Quality insurance will have to be based on previous work on the field of ontology evaluation. Since the automatic generation of expressive ontologies can potentially lead to a substantial increase in complexity, a simple manual revision of the ontology generated by a system such as the one described here might be infeasible. Therefore, we believe that automatic techniques for ontology evaluation will play a crucial role in the ontology learning and engineering cycle. These techniques could check, for instance, the ontology's coverage with respect to a domain-specific corpus [34] or its validity in terms of the OntoClean methodology [36].

In order to ensure completeness of the modeling process with regard to the application domain, a structured approach for an exhaustive exploration of complex relationships between classes is required. This can be realized, for example, by employing methods like *relational exploration* [37], which is an adaptation of attribute exploration from Formal Concept Analysis [38] to description logics. And finally, it might also be worthwhile to consider an integration of LExO with other learning approaches which could compensate for some of its limitations, e.g. with respect to the learnability of particular relations between roles [39], or disjointness axioms (see Section 3.3).

7. Experiments in an Application Scenario

In this section we discuss an application scenario of ontology learning in the context of a case study in the fishery's domain at the Food and Agriculture Organization (FAO) of the UN.

The FAO Fisheries department has several information and knowledge organization systems to facilitate and secure the long-term, sustainable development and utilization of the world's fisheries and aquaculture. In order to effectively manage the world's shared fish stocks and prevent overfishing, the FAO Fishery systems manage and disseminate statistical data on fishing, GIS data, information on aquaculture, geographic entities, de-

scription of fish stocks, etc. However, even though much of the data is "structured", it is not necessarily represented in a formal way, and some of the information resources are not available through databases but only as parts of websites, or as individual documents or images. Therefore, many or even all of these data sources could be better exploited by bringing together related and relevant information, along with the use of the fishery ontologies, to provide inference-based services for policy makers and national governments to make informed decisions.

A particular application developed within the NeOn project[12] is FSDAS (Fishery Stock Depletion Alert System), an ontology-driven decision support system for fisheries managers, assistants to policy makers and researchers. FSDAS will be a web-based intelligent agent that uses networked ontologies consisting of various fisheries, taxonomic, and geographical ontologies to aid users in discovering resources and relationships related to stock depletion and to detect probabilities of over-fishing. Fisheries ontologies, which bring together concepts from a number of existing knowledge organization systems, help to improve language-independent extraction and the discovery of information. Their development will allow for managing the complexity of fishery knowledge communities, their meaning negotiation and their deployment by worldwide authorities.

In order to achieve these goals, the ontological model needs to be shaped starting from highly structured FAO information systems, and to develop a learning capacity from this model to incorporate data and information from other less structured systems. Here, ontology learning becomes an integral part of the lifecycle of the fishery ontology. Further, in order for the FSDAS to be effective, it is important that the ontologies and resources it builds on are maintained and kept up-to-date, and that when applying changes to ontologies the consistency of the ontology is guaranteed.

7.1. Learning an Ontology for the Fishery Domain

We exemplify our approach by giving a number of axiomatizations automatically generated by means of LExO and the set of rules listed by Table 1. The example sentences are not artificial, but were selected from a fishery glossary provided by FAO.

1. ***Data:*** *Facts that result from measurements or observations.*
   $\text{Data} \equiv \text{Fact} \sqcap \exists \text{result\_from.}(\text{Measurement} \sqcup \text{Observation})$
2. ***InternalRateOfReturn:*** *A financial or economic indicator of the net benefits expected from a project or enterprise, expressed as a percentage.*
   $\text{InternalRateOfReturn} \equiv (\text{Financial} \sqcup \text{Economic}) \sqcap \text{indicator} \sqcap \exists \text{indicator\_of.}(\text{Net} \sqcap \text{Benefit} \sqcap \exists \text{expected\_from.}(\text{Project} \sqcup \text{Enterprise})) \sqcap \exists \text{expressed\_as.Percentage}$
3. ***Vector:*** *An organism which carries or transmits a pathogen.*
   $\text{Vector} \equiv \text{Organism} \sqcap (\text{carry} \sqcup \exists \text{transmit.Pathogen})$
4. ***Juvenile:*** *A young fish or animal that has not reached sexual maturity.*
   $\text{Juvenile} \equiv \text{Young} \sqcap (\text{Fish} \sqcup \text{Animal}) \sqcap \neg \exists \text{reached.}(\text{Sexual} \sqcap \text{Maturity})$
5. ***Tetraploid:*** *Cell or organism having four sets of chromosomes.*
   $\text{Tetraploid} \equiv (\text{Cell} \sqcup \text{Organism}) \sqcap =4 \text{ having.}(\text{Set} \sqcap \exists \text{set\_of.Chromosomes})$
6. ***Pair Trawling:*** *Bottom or mid-water trawling by two vessels towing the same net.*
   $\text{PairTrawling} \equiv (\text{Bottom} \sqcup \text{MidWater}) \sqcap \text{Trawling} \sqcap =2 \text{ trawling\_by.}(\text{Vessel} \sqcap \exists \text{tow.}(\text{Same} \sqcap \text{Net}))$

[12] http://www.neon-project.org

7. ***Sustained Use:*** *Continuing use without severe or permanent deterioration in the resources.*
   SustainedUse $\equiv$ Continuing $\sqcap$ Use $\sqcap$ $\neg\exists$use_with.((Severe $\sqcup$ Permanent) $\sqcap$ Deterioration $\sqcap$ $\exists$deterioration_in.Resources)
8. ***Biosphere****: The portion of Earth and its atmosphere that can support life.*
   Biosphere $\equiv$ Portion $\sqcap$ $\exists$portion_of.((Earth $\sqcap$ Its $\sqcap$ Atmosphere) $\sqcap$ $\exists$can_support.Life)

Some critical remarks and observations on the examples:

1. This is a simple example, which works out very well.
2. This example shows the complex axiomatizations which can be obtained using our approach. Here (and in other examples) we note that adjectives are so far interpreted as being intersective – we discuss this in Section 4. Another recurring problem is the generic nature of the role *of* which we tried to solve by designing the transformation rule in a way that it adds a disambiguating prefix to the preposition as a role name (*indicator_of*). Nevertheless, the output is a reasonable approximation of the intended meaning and would serve well as suggestion for an ontology engineer within an interactive process as we draft in Section 6.
3. This is a Minipar parse error. The desired solution would be
   Vector $\equiv$ Organism $\sqcap$ ($\exists$carry.Pathogen $\sqcup$ transmit.Pathogen).
4. Take particular attention to the handling of negation and of the present perfect tense.
5. The natural language sentence is actually ambiguous whether the number should be read as *exactly four* or *at least four*, and the role name *having* is certainly not satisfactory. Even more difficult is how *set of chromosomes* is resolved. A correct treatment is rather intricate, even if modeling is done manually. The class name *Chromosomes* should probably rather be a nominal containing the class name as individual – which cannot be modeled in OWL DL, but only in OWL Full. Note also that the cardinality restriction is used as a so-called *qualified* one, which is not allowed in OWL DL but is supported by most DL reasoners.
6. *Same* is difficult to resolve. In order to properly model this sentence, one would have to state that two different individuals of the class *Vessel* are connected to the same instantiation of *Net* by means of the *tow* role. This is not expressible in OWL DL as in the general case, such constructions would lead to undecidability.
7. Apart from the very generic role *in* and the problem with adjectives already mentioned, this is a complex example which works very well.
8. The possessive pronoun *its* would have to be resolved.

8. Conclusions

In this chapter we presented two conceptual approaches and implementations for learning expressive ontologies. LExO (cf. Section 3.2) constitutes a lexical approach to generating complex class descriptions from definitory sentences. It can be complemented by any general purpose ontology learning framework, or more specific solutions such as the approach presented in Section 3.3 aiming at the automatic generation of disjointness axioms.

Although we see a great potential in learning expressive ontologies, the discussion shows that there are still many open issues – technical, but also very fundamental ques-

tions. The most important ones according to our perception relate to the relationship of lexical and ontological semantics. Given a purely syntactical transformation such as the one presented in Section 3.2, it will be crucial to investigate at which stage of the process and in which manner particularities of both semantics have to be considered. Finally, we will have to answer the question where the principal limitations of our approaches with respect to the expressivity of the learned ontologies really are. It is reasonable to assume that at least some aspects of ontological semantics cannot (or not so easily) be captured by purely lexical ontology learning methods. However, we believe that a combination of lexical and logical approaches could help to overcome these limitations.

In any case, learning expressive ontologies for knowledge-intensive applications will demand a tighter integration of learning and reasoning at both development time and runtime. One of the most important questions is how potential inconsistencies in the ontology can be dealt with by consistent ontology evolution, for example (see Section 5). Finally, suitable methodologies for semi-automatic ontology engineering will be needed in order to combine ontology learning, evaluation and reasoning.

Acknowledgements

This research has been partially supported by the European Commission under contracts IST-2003-506826 SEKT, IST-2006-027595 NeOn, by the German Federal Ministry of Education and Research (BMBF) under the SmartWeb project 01IMD01B, and by the Deutsche Forschungsgemeinschaft (DFG) under the ReaSem project.

References

[1] J. Hendler, *On beyond ontology*, in: Keynote talk, Second International Semantic Web Conference, 2003.

[2] I. Horrocks and P. F. Patel-Schneider, *Reducing OWL Entailment to Description Logic Satisfiability*, in: Journal of Web Semantics, 1(4):345–357, 2004.

[3] P. Haase, F. van Harmelen, Z. Huang, H. Stuckenschmidt, and Y. Sure, *A framework for handling inconsistency in changing ontologies*, in: Proceedings of the 4th International Semantic Web Conference, ISWC 2005, Galway, Ireland, November 6-10, 2005, pages 353–367.

[4] P. Cimiano, J. Völker and R. Studer, *Ontologies on demand? : A description of the state-of-the-art, applications, challenges and trends for ontology learning from text*, in: Information Wissenschaft und Praxis, vol. 57, no. 6–7, pp. 315–320, 2006.

[5] K. Frantzi and S. Ananiadou, *The C-value / NC-value domain independent method for multi-word term extraction*, in: Journal of Natural Language Processing, 6(3), pp. 145–179, 1999.

[6] M. Ciaramita, A. Gangemi, E. Ratsch, J. Saric and I. Rojas, *Unsupervised Learning of Semantic Relations for Molecular Biology Ontologies*, in: P. Buitelaar, P. Cimiano, eds., Bridging the Gap between Text and Knowledge: Selected Contributions to Ontology Learning and Population from Text, Frontiers in Artificial Intelligence and Applications Series, IOS Press, *this volume*.

[7] A. Schutz, P. Buitelaar, *RelExt: A Tool for Relation Extraction from Text in Ontology Extension*, in: Proceedings of the 4th International Semantic Web Conference, (ISWC) pp. 593–606, 2005.

[8] R. Navigli and P. Velardi, *From Glossaries to Ontologies: Learning the Structure Hidden in the Text*, in: P. Buitelaar, P. Cimiano, eds., Bridging the Gap between Text and Knowledge: Selected Contributions to Ontology Learning and Population from Text, Frontiers in Artificial Intelligence and Applications Series, IOS Press, *this volume*.

[9] M. Poesio and A. Almuhareb, *Extracting concept descriptions from the Web: the importance of attributes and values*, in: P. Buitelaar, P. Cimiano, eds., Bridging the Gap between Text and Knowledge: Selected Contributions to Ontology Learning and Population from Text, Frontiers in Artificial Intelligence and Applications Series, IOS Press, *this volume*.

[10] A. Mädche and S. Staab, *Discovering Conceptual Relations from Text*, in: Proceedings of the European Conference on Artificial Intelligence (ECAI), pp. 321–325, 2000.
[11] M. Berland and E. Charniak, *Finding parts in very large corpora*, in: Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL), 1999.
[12] M. Hearst, *Automatic acquisition of hyponyms from large text corpora*, in: Proceedings of the 14th International Conference on Computational Linguistics, pp. 539-545, 1992.
[13] D. Faure and C. Nedellec, *Knowledge Acquisition of Predicate Argument Structures from Technical Texts Using Machine Learning: The System ASIUM*, in: Proceedings of the European Knowledge Acquisition Workshop (EKAW), pp. 329–334, 1999.
[14] S. A. Caraballo, *Automatic construction of a hypernym-labeled noun hierarchy from text*, in: Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 120–126, 1999.
[15] P. Cimiano and S. Staab, *Learning Concept Hierarchies from Text with a Guided Agglomerative Clustering Algorithm*, in: Proceedings of the ICML 2005 Workshop on Learning and Extending Lexical Ontologies with Machine Learning Methods, 2005.
[16] P. Cimiano, A. Hotho and S. Staab, *Learning Concept Hierarchies from Text Corpora using Formal Concept Analysis*, in: Journal of Artificial Intelligence Research (JAIR), 24, pp. 305–339, 2005.
[17] P. Cimiano and J. Völker, *Towards large-scale, open-domain and ontology-based named entity classification*, in: G. Angelova, K. Bontcheva, R. Mitkov, and N. Nicolov, editors, Proc. of the International Conference on Recent Advances in Natural Language Processing (RANLP), pp. 166–172, Borovets, Bulgaria, September 2005, INCOMA Ltd.
[18] P. Cimiano, S. Handschuh and S. Staab, *Towards the self-annotating web*, in: Proceedings of the 13th World Wide Web Conference, pp. 462–471, 2004.
[19] O. Etzioni, M. J. Cafarella, D. Downey, S. Kok, A.-M. Popesu, T. Shaked, S. Soderland, D.S. Weld and A. Yates, *Web-scale information extraction in KnowItAll (preliminary results)*, in: Proceedings of the 13th World Wide Web Conference, pp. 100–110, 2004.
[20] J. Völker, P. Hitzler and P. Cimiano, *Acquisition of OWL DL Axioms from Lexical Resources*, in: Proceedings of the 4th European Semantic Web Conference (ESWC'07), pp. 670–685, Springer, June 2007.
[21] D. Lin, *Dependency-based evaluation of minipar*, in: Proceedings of the Workshop on the Evaluation of Parsing Systems, 1998.
[22] P. Haase and J. Völker, *Ontology Learning and Reasoning – Dealing with Uncertainty and Inconsistency*, in: P.C.G. da Costa, K.B. Laskey, K.J. Laskey, M. Pool, Proceedings of the Workshop on Uncertainty Reasoning for the Semantic Web (URSW'05), pp. 45–55, November 2005.
[23] P. Cimiano and J. Völker, *Text2Onto – A Framework for Ontology Learning and Data-driven Change Discovery*, in: Andres Montoyo, Rafael Munoz, Elisabeth Metais, Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems (NLDB'05), volume 3513 of Lecture Notes in Computer Science, pp. 227–238, Springer, Alicante, Spain, June 2005.
[24] Z. Wu and M. Palmer, *Verb semantics and lexical selection*, in: 32nd Annual Meeting of the Association for Computational Linguistics, pp. 133–138, New Mexico, 1994.
[25] C. Fellbaum, *WordNet, an electronic lexical database*, MIT Press, 1998.
[26] Z. Harris, *Distributional structure*, in: J. Katz, editor, The Philosophy of Linguistics, pp. 26–47, New York, 1985, Oxford University Press.
[27] J. Völker, D. Vrandecic, Y. Sure and A. Hotho, *Learning Disjointness*, in: Proceedings of the 4th European Semantic Web Conference (ESWC'07), pp. 175–189, Springer, June 2007.
[28] M. Poesio, T. Ishikawa, S. Schulte im Walde and R. Vieira, *Acquiring lexical knowledge for anaphora resolution*, in: Proceedings of the 3rd Conference on Language Resources and Evaluation, 2002.
[29] D.A. Cruse, *Lexical Semantics*, Cambridge Textbooks in Linguistics, Cambridge University Press, New York, 1986.
[30] N. Guarino and C. A. Welty, *An Overview of OntoClean*, in: S. Staab and R. Studer, eds., The Handbook on Ontologies, pp. 151–172, Springer, 2004.
[31] A. Motro and P. Smets, *Uncertainty Management In Information Systems*, Springer, 1997.
[32] P. Haase and G. Qi, *An analysis of approaches to resolving inconsistencies in DL-based ontologies*, in: Proceedings of the International Workshop on Ontology Dynamics (IWOD'07), June, 2007.
[33] S. Schlobach, *Debugging and semantic clarification by pinpointing*, in: Proceedings of the 2nd European Semantic Web Conference (ESWC'05), volume 3532 of LNCS, pp. 226–240, Springer, 2005.
[34] C. Brewster, H. Alani, S. Dasmahapatra and Y. Wilks, *Data Driven Ontology Evaluation*, in: Proceedings of the Lexical Resources and Evaluation Conference (LREC'04), pp. 641–644, Lisbon, Portugal, 2004.

[35] M. Ehrig, P. Haase, N. Stojanovic, and M. Hefke, *Similarity for ontologies - a comprehensive framework*, in: Proceedings of the 13th European Conference on Information Systems, May 2005.
[36] J. Völker, D. Vrandecic and Y. Sure, *Automatic Evaluation of Ontologies (AEON)*, in: Y. Gil, E. Motta, V. R. Benjamins, M. A. Musen, Proceedings of the 4th International Semantic Web Conference (ISWC'05), volume 3729 of LNCS, pp. 716–731, Springer Verlag Berlin-Heidelberg, November 2005.
[37] S. Rudolph, *Exploring relational structures via FLE*, in: Wolff, K.E., Pfeiffer, H.D., Delugach, H.S., eds.: Conceptual Structures at Work, 12th International Conference on Conceptual Structures, volume 3127 of LNCS, Springer, pp. 196–212, 2004.
[38] B. Ganter and R. Wille, *Formal Concept Analysis – Mathematical Foundations*, Springer, Berlin, 1999.
[39] D. Lin and P. Pantel: *DIRT – discovery of inference rules from text*, in: Knowledge Discovery and Data Mining, pp. 323–328, 2001.

From Glossaries to Ontologies: Extracting Semantic Structure from Textual Definitions

Roberto NAVIGLI [a,1] and Paola VELARDI [a]
[a] *Università di Roma "La Sapienza", Roma, Italy.*

Abstract. Learning ontologies requires the acquisition of relevant domain concepts and taxonomic, as well as non-taxonomic, relations. In this chapter, we present a methodology for automatic ontology enrichment and document annotation with concepts and relations of an existing domain core ontology. Natural language definitions from available glossaries in a given domain are processed and regular expressions are applied to identify general-purpose and domain-specific relations. We evaluate the methodology performance in extracting hypernymy and non-taxonomic relations. To this end, we annotated and formalized a relevant fragment of the glossary of Art and Architecture (AAT) with a set of 10 relations (plus the hypernymy relation) defined in the CRM CIDOC cultural heritage core ontology, a recent W3C standard. Finally, we assessed the generality of the approach on a set of web pages from the domains of history and biography.

Keywords. Ontology learning, Semantic relation learning, Glossary formalization

Introduction

The Semantic Web [1], i.e. the vision of a next-generation web where content is conceptually indexed, requires applications to process and exploit the semantics implicitly encoded in on-line and off-line resources. The large-scale, automatic semantic annotation of web documents based on well-established domain ontologies would allow Semantic Web applications to emerge and gain acceptance. Wide coverage ontologies are indeed available for general applications (e.g. WordNet[2], CYC[3], SUMO[4]), however semantic annotation in unconstrained areas seems still out of reach for state-of-the-art systems. Domain-specific ontologies are preferable since they would limit the semantic coverage needed and make the applications feasible.

Recently, certain web communities began to exploit the benefits deriving from the application of Semantic Web techniques. Accordingly, they spent a remarkable efforts to conceptualize their competence domain through the definition of a core ontology, i.e. a

[1] Corresponding Author: Roberto Navigli, Dipartimento di Informatica, Via Salaria, 113 - 00198 Roma Italy; E-mail: navigli@di.uniroma1.it.

[2] http://wordnet.princeton.edu

[3] http://www.opencyc.org

[4] http://www.ontologyportal.org

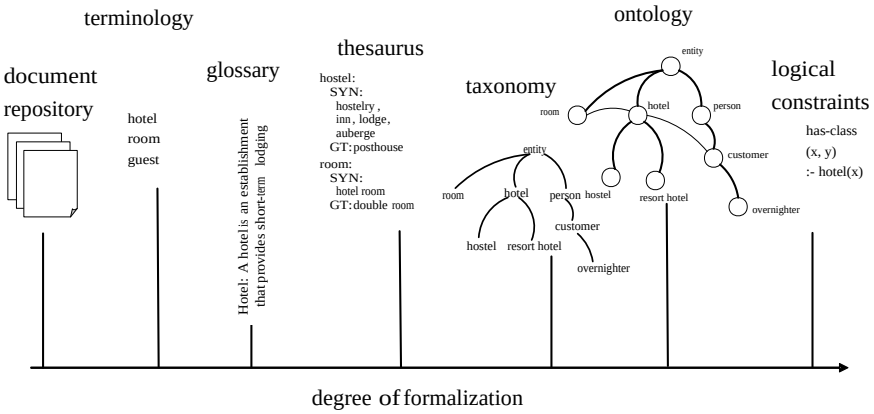

Figure 1. The different degrees of formalization: from unstructured textual content to ontology and logical rules.

basic ontology consisting of the minimal concepts and relations required to understand the other domain concepts. Relevant examples are in the area of enterprise modeling [2,3] and cultural heritage [4]. Core ontologies are indeed a necessary starting point to model in a principled way the concepts, relations and axioms of a given domain. But in order for an ontology to be really usable in applications, it is necessary to enrich the core structure with the thousands of concepts and instances that "make" the domain.

While several ontology learning approaches extract concepts and relation instances directly from (web) documents, i.e. from unstructured texts (see the Chapter by Poesio and Almuhareb), in this chapter we present a methodology which relies on the existence of a domain glossary. In Figure 1 we show the different degrees of knowledge formalization: from unstructured texts to terminologies, glossaries, thesauri, taxonomies, ontologies, and logic rules. Our assumption allows us to shift the focus from fully unstructured texts to glossaries, which encode textual definitions for domain terms. This assumption drastically reduces the chance of extracting inadequate information (especially, information from inappropriate sources), and to focus on the formalization of textual definitions. Specifically, the methodology presented hereafter automatically annotates a glossary G with the semantic relations of an existing core ontology O. The annotation of documents and glossary definitions is performed using regular expressions, a widely adopted text mining approach. However, while in the literature regular expressions seek mostly for patterns at the lexical and part-of-speech level, we defined expressions enriched with syntactic and semantic constraints. A word sense disambiguation algorithm, SSI [5], is used to automatically replace the high-level semantic constraints specified in the core ontology with fine-grained sense restrictions, using the sense inventory of WordNet, a general purpose lexicalized ontology. From each gloss g of a term t in the glossary G, we extract one or more semantic relation instances $R(C_t, C_w)$, where R is a relation in O, C_t and C_w are respectively the domain and range of R. The concept C_t corresponds to its lexical realization t, while C_w is the concept associated with a word w in G, captured by a regular expression.

The annotation process allows to automatically enrich O with an existing glossary in the same domain of O, since each pair of term and gloss (t, g) in the glossary G

is transformed into a formal definition, compliant with O. Furthermore, the very same method can be used to automatically annotate free text with the concepts and relations of the enriched ontology O'. We experimented with our methodology in the cultural heritage domain, since for this domain several well-established resources are available, like the CIDOC-CRM core ontology, the Art and Architecture Thesaurus (AAT), and others.

The chapter is organized as follows: in Section 1 we present the CIDOC and the other resources used in this work. In Section 2 we describe in detail the ontology enrichment algorithm. In Section 3 we provide a performance evaluation on a subset of CIDOC properties and a sub-tree of the AAT thesaurus. Related literature is examined in Section 4.

1. Semantic and Lexical Resources in the Cultural Heritage Domain

In this section we describe the semantic and lexical resources in the cultural heritage domain that have been used in this work.

1.1. The CIDOC CRM

We adopted as a core ontology O the CIDOC Conceptual Reference Model (CIDOC CRM) [4], a formal core ontology whose purpose is to facilitate the integration and exchange of cultural heritage information between heterogeneous sources. It is currently being elaborated to become an ISO standard. In its current version (4.0) the CIDOC includes 84 taxonomically structured concepts (called *entities*) and a flat set of 141 semantic relations, called *properties*. Entities, i.e. concepts, are defined in terms of their subclass and super-class relations in the CIDOC hierarchy, and an informal description of the entity is provided. Properties are defined in terms of domain (the class for which a property is formally defined) and range (the class that comprises all potential values of a property), e.g.: property 46, labelled *is composed of (forms part of)*, has E19 Physical Object as domain and E42 Object Identifier as range.

To make the CIDOC CRM usable by a computer program, we replaced specifications written in natural language with formal ones. For each property R, we created a tuple $R(C_d, C_r)$ where C_d and C_r are the domain and range entities specified in the CIDOC reference manual.

1.2. The AAT thesaurus

We adopted as a domain glossary G the Art and Architecture Thesaurus (AAT), a controlled vocabulary for use by indexers, catalogers, and other professionals concerned with information management in the fields of art and architecture. In its current version, it includes more than 133,000 terms, descriptions, bibliographic citations, and other information relating to fine art, architecture, decorative arts, archival materials, and material culture. An example is reported in Table 1. We manually mapped the top AAT concepts to CIDOC entities, as shown in Table 2. As a result, CIDOC properties can be applied to connect pairs of concepts in AAT which satisfy the CIDOC domain and range constraints, i.e. the CIDOC CRM can be used as a core ontology for AAT.

Table 1. An entry from the Art and Architecture Thesaurus (AAT) glossary.

Concept name: Maestà
Definition: Refers to a work of a specific iconographic type, depicting the Virgin Mary and Christ Child enthroned in the center with saints and angels in adoration to each side. The type developed in Italy in the 13^{th} century and was based on earlier Greek types. Works of this type are typically two-dimensional, including painted panels (often altarpieces), manuscript illuminations, and low-relief carvings.
Hierarchical Position:
Objects Facet
Visual and Verbal Communication
Visual Works
<visual works>
<visual works by subject type>
maestà

Table 2. Mapping between AAT and CIDOC.

AAT topmost	**CIDOC entities**
Top concept of AAT	CRM Entity (E1), Persistent Item (E77)
Styles and Periods	Period (E4)
Events	Event (E5)
Activities Facet	Activity (E7)
Processes/Techniques	Beginning of Existence (E63)
Objects Facet	Physical Stuff (E18), Physical Object (E19)
Artifacts	Physical Man-Made Stuff (E24)
Materials Facet	Material (E57)
Agents Facet	Actor (E39)
Time	Time-Span (E52)
Place	Place (E53)

1.3. Additional Resources

To apply semantic constraints on the words of a definition (as clarified in the next Section), we need additional resources. WordNet [6] is used to verify that certain words in a gloss fragment f satisfy the domain and range constraints of $R(C_d, C_r)$ in the CIDOC. In order to do so, we manually linked the WordNet topmost concepts to the CIDOC entities. For example, the concept E19 (Physical Object) is mapped to the WordNet synset "object, physical object". Furthermore, we created a gazetteer of named entities by extracting names from DMOZ[5], a large human-edited directory of the web, the Union List of Artist Names[6] (ULAN) and the Getty Thesaurus of Geographic Names[7] (GTG) provided by the Getty institute, along with the AAT.

[5] http://dmoz.org/about.html

[6] http://www.getty.edu/research/conducting_research/vocabularies/ulan/

[7] http://www.getty.edu/research/tools/vocabulary/tgn

2. Automated Ontology Enrichment: from Glossaries to Ontologies

In this Section we describe in detail the method for automatic semantic annotation and ontology enrichment in the cultural heritage domain. Let G be a glossary, t a term in G and g the corresponding natural language definition (gloss) in G. The main steps of the algorithm are the following:

1. A pre-processing step (part-of-speech tagging and Named Entity Recognition).
2. Annotation of sentence segments with CIDOC properties.
3. Formalization of glosses.

2.1. Pre-processing

2.1.1. Part-of-speech tagging

Each input gloss is processed with a part-of-speech tagger, TreeTagger[8]. As a result, for each gloss $G = w_1\ w_2 \ldots w_n$, a string of part-of-speech tags $p_1, p_2, \ldots, p_n$ is produced, where $p_i \in P$ is the part-of-speech tag chosen by TreeTagger for word w_i, and $P = \{ N, A, V, J, R, C, P, S, W \}$ is a simplified set of syntactic categories (respectively, nouns, articles, verbs, adjectives, adverbs, conjunctions, prepositions, symbols, and wh-words).

2.1.2. Named Entity Recognition

We augmented TreeTagger with the ability to capture named entities of locations, organizations, persons, numbers and time expressions. In order to do so, we use regular expressions [7] in a rather standard way, therefore we omit details. When a named entity string $w_i\ w_{i+1} \ldots w_{i+j}$ is recognized, it is transformed into a single term and a specific part of speech denoting the kind of entity is assigned to it (L for cities (e.g. Venice), countries and continents, T for time and historical periods (e.g. Middle Ages), O for organizations and persons (e.g. Leonardo Da Vinci), B for numbers, etc.).

2.2. Annotation of Sentence Segments with CIDOC Properties

We now present an algorithm for the annotation of gloss segments with properties grounded on the CIDOC-CRM relation model. Given a gloss G and a property R[9], we define a *relation checker* c_R taking as input G and producing as output a set F_R of fragments of g annotated with the property R: <**R**>f</**R**>. The selection of a fragment f to be included in the set F_R is based on different kinds of constraints:

- a part-of-speech constraint $p(f$, pos-string) matches the part-of-speech (pos) string associated with the fragment f against a regular expression (pos-string), specifying the required syntactic structure.
- a lexical constraint $l(f, k$, lexical-constraint) matches the lemma of the word in k-th position of f against a regular expression (lexical-constraint), constraining the lexical conformation of words occurring within the fragment f.

[8] TreeTagger is available from: http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger

[9] In what follows, we adopt the CIDOC terminology for relations and concepts, i.e. properties and entities, respectively.

- semantic constraints on domain and range $s_D(f$, semantic-domain) and $s(f, k$, semantic-range) are valid, respectively, if the term t and the word in the k-th position of f match the semantic constraints on domain and range imposed by the CIDOC, i.e. if there exists at least one sense of t, C_t, and one sense of w, C_w, such that: $R_{kind-of^*}(C_d, C_t)$ and $R_{kind-of^*}(C_r, C_w)$[10].

More formally, the annotation process is defined as follows: a relation checker c_R for a property R is a logical expression composed with constraint predicates and logical connectives, using the following production rules:

$$
\begin{aligned}
c_R &\rightarrow s_D(f, \textit{semantic-domain}) \wedge c'_R \\
c'_R &\rightarrow \neg c'_R | (c'_R \vee c'_R) | (c'_R \wedge c'_R) \\
c'_R &\rightarrow p(f, \textit{pos-string}) | l(f, k, \textit{lexical-constraint}) | s(f, k, \textit{semantic-range})
\end{aligned}
$$

where f is a variable representing a sentence fragment. Notice that a relation checker must always specify a semantic constraint s_D on the domain of the relation R being checked on fragment f. Optionally, it must also satisfy a semantic constraint s on the k-th element of f, the range of R. For example, the following excerpt of the checker for the *is-composed-of* relation P46 in CIDOC:

$$
\begin{aligned}
c_{\textit{is-composed-of}}(f) = s_D(f, \textit{physical object\#1}) \\
\wedge p(f, \text{"}(V)_1(P)_2 R?A?CRJVN^*(N)_3\text{"}) \\
\wedge l(f, 1, \text{"}\hat{}(\textit{consisting}|\textit{composed}|\textit{comprised}|\textit{constructed})\$\text{"}) \\
\wedge l(f, 2, \text{"}\textit{of}\text{"}) \wedge s(f, 3, \textit{physical object\#1})
\end{aligned}
$$

reads as follows: "the fragment f is valid if it consists of a verb in the set { consisting, composed, comprised, constructed }, followed by a preposition 'of', a possibly empty number of adverbs, adjectives, verbs and nouns, and terminated by a noun interpretable as a physical object in the WordNet concept inventory". The first predicate, s_D, requires that also the term t whose gloss contains f (i.e., its domain) be interpretable as a physical object.

Notice that some letter in the regular expression specified for the part-of-speech constraint is enclosed in parentheses. This allows it to identify the relative positions of words to be matched against lexical and semantic constraints, as shown graphically in Figure 2.

The checker recognizes, among others, the following fragments (the words whose part-of-speech tags are enclosed in parentheses are indicated in bold):

- **(consisting)**$_1$ **(of)**$_2$ semi-precious **(stones)**$_3$ (matching part-of-speech string: **(V)**$_1$**(P)**$_2$ J**(N)**$_3$);

[10] $R_{kind-of^*}$ denotes zero, one, or more applications of $R_{kind-of}$.

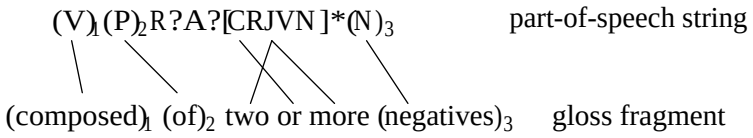

Figure 2. Correspondence between parenthesized part-of-speech tags and words in a gloss fragment.

- **(composed)**$_1$ **(of)**$_2$ **(knots)**$_3$ (matching part-of-speech string: **(V)**$_1$**(P)**$_2$**(N)**$_3$).

As a second example, an excerpt of the checker for the *consists-of* (P45) relation is the following:

$$\begin{aligned} c_{consists-of}(f) = {} & s_D(f, \textit{physical object\#1}) \wedge p(f, \text{"}(V)_1(P)_2A?JN, VC^*(N)_3\text{"}) \\ & \wedge l(f, 1, \text{"}\hat{}(\textit{make}|\textit{do}|\textit{produce}|\textit{decorated})\$\text{"}) \\ & \wedge l(f, 2, \text{"}\hat{}(\textit{of}|\textit{by}|\textit{with})\$\text{"}) \wedge \neg s(f, 3, \textit{color\#1}) \\ & \wedge \neg s(f, 3, \textit{activity\#1}) \\ & \wedge (s(f, 3, \textit{material\#1}) \vee s(f, 3, \textit{solid\#1}) \vee s(f, 3, \textit{liquid\#1})) \end{aligned}$$

recognizing, among others, the following phrases:

- **(made)**$_1$ **(with)**$_2$ the red earth pigment **(sinopia)**$_3$ (matching part-of-speech string: **(V)**$_1$**(P)**$_2$AJNN**(N)**$_3$);
- **(decorated)**$_1$ **(with)**$_2$ red, black, and white **(paint)**$_3$ (matching part-of-speech string: **(V)**$_1$**(P)**$_2$JJCJ**(N)**$_3$).

Notice that in both checkers $c_{is-composed-of}$ and $c_{consists-of}$ semantic constraints are specified in terms of WordNet sense numbers (*material#1*, *solid#1* and *liquid#1*), and can also be negative (¬*color#1* and ¬*activity#1*). The motivation is that CIDOC constraints are coarse-grained due to the small number of available core concepts: for example, the property *P45 consists-of* simply requires that the range belongs to the class *Material (E57)*. Using WordNet for semantic constraints has two advantages: first, it is possible to write more fine-grained (and hence more reliable) constraints, second, regular expressions can be re-used, at least in part, for other domains and ontologies. In fact, several CIDOC properties are rather general-purpose.

2.3. Formalization of Glosses

The annotations generated in the previous step are the basis for extracting property instances to enrich the CIDOC CRM with a conceptualization of the AAT terms. In general, for each gloss g defining a concept C_t, and for each fragment $f \in F_R$ of g annotated with the property R: <**R**>f</**R**>, it is possible to extract one or more *property instances* in the form of a triple $R(C_t, C_w)$, where C_w is the *concept* associated with a term or multi-word expression w occurring in f (i.e. its language realization) and C_t is the *concept* associated with the defined term t in AAT. For example, from the definition of *tatting* (a kind of lace) the algorithm automatically annotates the phrase *composed of*

knots, suggesting that this phrase specifies the *range* of the is-composed-of property for the term *tatting*:

$$R_{is-composed-of}(C_{tatting}, C_{knot})$$

In this property instance, $C_{tatting}$ is the *domain* of the property (a term in the AAT glossary) and C_{knot} is the *range* (a specific term in the definition g of *tatting*). Selecting the concept associated with the domain is rather straightforward: glossary terms are in general not ambiguous, and, if they are, we simply use a numbering policy to identify the appropriate concept. In the example at hand, $C_{tatting} = tatting\#1$ (the first and only sense in AAT). Therefore, if C_t matches the domain restrictions in the regular expression for R, then the domain of the relation is considered to be C_t. Selecting the range of a relation is instead more complicated. The first problem is to select the correct words in a fragment f. Only certain words of an annotated gloss fragment can be exploited to extract the range of a property instance. For example, in the phrase "depiction of fruit, flowers, and other objects" (from the definition of *still life*), only *fruit*, *flowers*, *objects* represent the range of the property instances of kind *depicts (P62)*.

When writing relation checkers, as previously described, we can add markers of ontological relevance by specifying a predicate $r(f, k)$ for each relevant position k in a fragment f. The purpose of these markers is precisely to identify words in f whose corresponding concepts are in the range of a property. For instance, the checker (1) $c_{is-composed-of}$ from the previous paragraph is augmented with the conjunction: $\wedge\, r(f, 3)$. We added the predicate $r(f, 3)$ because the third parenthesis in the part-of-speech string refers to an ontologically relevant element (i.e. the candidate *range* of the *is-composed-of* property).

The second problem is that words that are candidate ranges can be ambiguous, and they often are, especially if they do not belong to the domain glossary G. Considering the previous example of the property *depicts*, the word *fruit* is not a term of the AAT glossary, and it has 3 senses in WordNet (the fruit of a plant, the consequence of some action, an amount of product). The property *depicts*, as defined in the CIDOC, simply requires that the range be of type *Entity (E1)*. Therefore, all the three senses of *fruit* in WordNet satisfy this constraint. Whenever the range constraints in a relation checker do not allow a full disambiguation, we apply the SSI algorithm [5], a word sense disambiguation algorithm based on structural pattern recognition, available on-line[11]. The algorithm is applied to the words belonging to the segment fragment f and is based on the detection of relevant semantic interconnection patterns between the appropriate senses. These patterns are extracted from a lexical knowledge base that merges WordNet with other resources, like word collocations, on-line dictionaries, etc. For example, in the fragment "depictions of fruit, flowers, and other objects" the following properties are created for the concept *still_life#1*:

$$R_{depicts}(still\_life\#1, fruit\#1)$$
$$R_{depicts}(still\_life\#1, flower\#2)$$
$$R_{depicts}(still\_life\#1, object\#1)$$

[11] SSI is an on-line knowledge-based WSD algorithm accessible from http://lcl.uniroma1.it/ssi. The on-line version also outputs the detected semantic connections (as those in Figure 3).

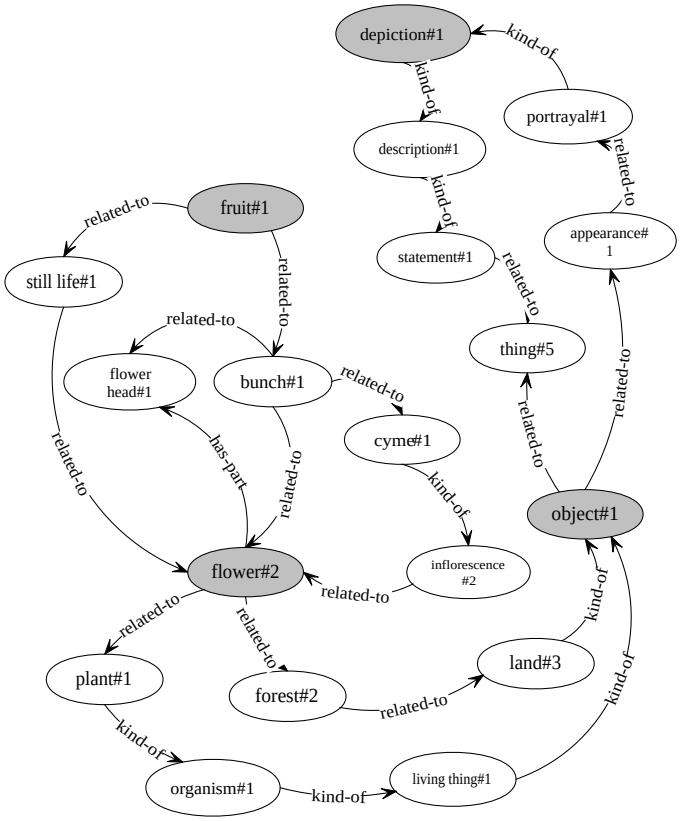

Figure 3. Semantic Interconnections selected by the SSI algorithm when given the word list: "depiction, fruit, flower, object".

Some of the semantic patterns supporting this sense selection are shown in Figure 3.

A further possibility is that the range of a relation *R* is a concept *instance*. We create concept instances if the word w extracted from the fragment f is a named entity. For example, the definition of *Venetian lace* is annotated as "Refers to needle lace created <**current-or-former-location**> in Venice</**current-or-former-location**> [...]". As a result, the following triple is produced:

$$R_{has-current-or-former-location}(Venetian\_lace\#1, Venice{:}city\#1)$$

where $Venetian\_lace$#1 is the concept label generated for the term *Venetian lace* in the AAT and Venice is an instance of the concept *city#1* (*city*, *metropolis*, *urban center*) in WordNet.

2.4. Taxonomy Validation

The ontology resulting from the methodology presented above can be viewed and validated with the aid of a web application developed in our laboratory, namely the Taxon-

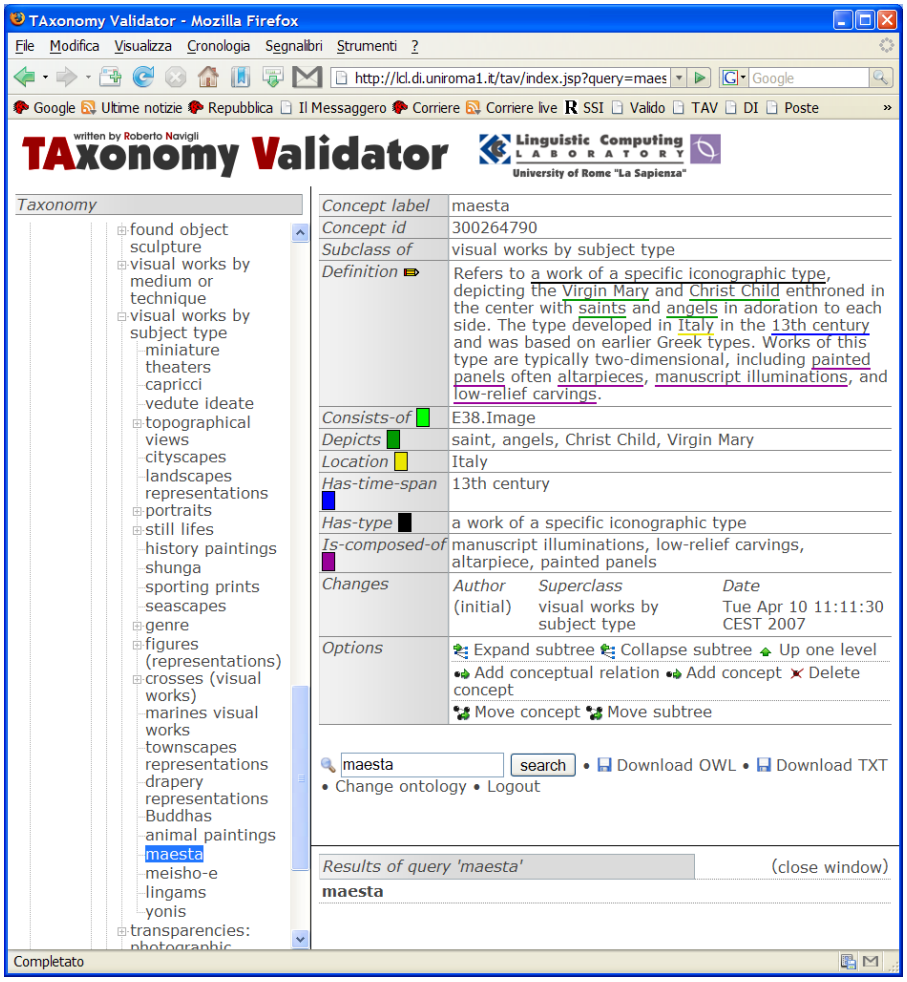

Figure 4. A screenshot of TAV, a tool for the collaborative validation of taxonomies.

omy VAlidator (TAV)[12]. In Figure 4 we show a screenshot of the tool: the taxonomy is shown in the left pane, whereas the right pane shows information about the selected concept. Gloss fragments are highlighted in different colors, one for each distinct semantic relation. The user can delete and move concepts up and down the taxonomy, and, more in general, search and edit the ontology. The tool has been extensively used in our experiments to assess the quality of the automatically-acquired taxonomic relations. In [8], TAV has also been used in a collaborative way in the context of the INTEROP Network of Excellence. However, the tool does not yet support the validation of non-taxonomic relations.

[12]TAV is available from: http://lcl.uniroma1.it/tav

3. Evaluation

Evaluating the quality of ontologies is particularly difficult, due to the fact that there is no prescribed way to account for a domain of interest (see the Chapter by Dellschaft and Staab in this volume). In this section we provide an evaluation of the methodology for taxonomic (Section 3.1) and non-taxonomic relations (Section 3.2). We further describe an experiment to test the generality of the approach (Section 3.3).

3.1. Evaluation of Taxonomic Relation Learning

As a first experiment, we developed a relation checker for the *is-a* taxonomic relation (i.e. *hypernymy*). We randomly selected 500 glosses from the Visual Works subtree of the AAT thesaurus, and applied the relation checker to the pre-processed gloss to determine its performance in the identification of the appropriate relation instances. Using TAV (see Section 2.4), we calculated that the checker correctly identified 474 out of 500 hypernyms, achieving 94.8% accuracy.

In AAT, the hypernym relation is already available, since AAT is a thesaurus, not a glossary. This allowed us to compare the extracted hypernyms with those already available in the thesaurus. When applying these patterns to the AAT we found that in 34% of the cases the automatically extracted hypernym is the same as in AAT, and in 26% of the cases, either the extracted hypernym is more general than the one defined in AAT, or the contrary, with respect to the AAT hierarchy. This result quite favorably compares with available results in the literature (see Section 4). Several kinds of gaps between textual glosses and manually-defined hypernyms are thoroughly discussed by Ide and Véronis [9].

3.2. Evaluation of Non-Taxonomic Relation Learning

It is commonly agreed that learning hypernyms is easier than learning non-taxonomic relations. In this Section, we describe our experiments on several kinds of non-taxonomic semantic relations from CIDOC-CRM. Since the CIDOC-CRM model formalizes a large number of fine-grained properties (precisely, 141), we selected a subset of properties for our experiments (reported in Table 3). We wrote a relation checker for each property in the Table. By applying the checkers in cascade to a gloss g, a set of annotations is produced. The following is an example of an annotated gloss for the term *vedute*:

Refers to detailed, largely factual topographical views, especially <**has-time-span**>18th-century <**/has-time-span**> Italian paintings, drawings, or prints of cities. The first vedute probably were <**carried-out-by**>painted by northern European artists<**/carried-out-by**> who worked <**has former-or-current-location**>in Italy<**/has former-or-current-location**><**has-time-span**>in the 16th century<**/has-time-span**>. The term refers more generally to any painting, drawing or print <**depicts**>representing a landscape or town view <**/depicts**> that is largely topographical in conception.

Figure 5 shows a more comprehensive graph representation of the outcome for the concepts *vedute#1* and *maestà#1* (see the gloss in Table 1).

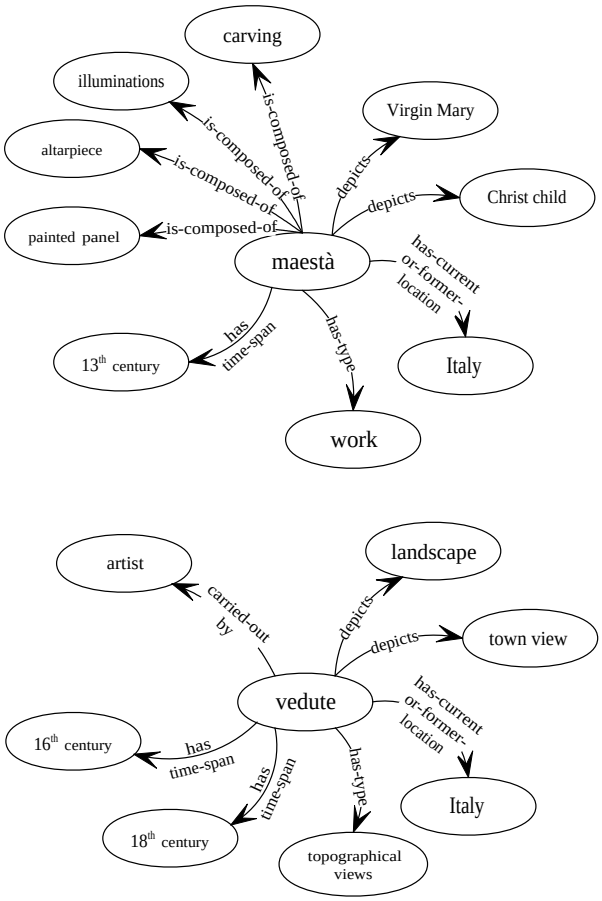

Figure 5. Extracted conceptualisation (in graphical form) of the terms *maestá#1* and *vedute#1* (sense numbers are omitted for clarity).

To evaluate the methodology described in Section 2 we considered 814 glosses from the *Visual Works* sub-tree of the AAT thesaurus[13], containing a total of 27,925 words. The authors wrote the relation checkers by tuning them on a subset of 122 glosses, and tested their generality on the remaining 692. The test set was manually tagged with the subset of the CIDOC-CRM properties shown in Table 3 by two annotators with adjudication (requiring a careful comparison of the two sets of annotations). We performed two experiments: in the first, we evaluated the *gloss annotation task*, in the second the *property instance extraction task*, i.e. the ability to identify the appropriate domain and range of a property instance. In the case of the gloss annotation task, for evaluating each piece of information we adopted the measures of "*labeled*" *precision* and *recall*. These measures are commonly used to evaluate parse trees obtained by a parser [10] and allow the rewarding of good partial results. Given a property R, labeled precision is the number of words annotated correctly with R over the number of words annotated automatically

[13] The resulting OWL ontology is available at http://lcl.uniroma1.it/tav

Table 3. A subset of the relations from the CIDOC-CRM model.

Property	Domain	Range	**Example**
P26 - moved to	Move	Place	P26(installation of public sculpture, public place)
P27 - moved from	Move	Place	P27(removal of cornice pictures, wall)
P53 - has former or current location	Physical Stuff	Place	P53(fancy pictures, London)
P55 - has current location	Physical Object	Place	P55(macrame, Genoa)
P46 - is composed of (is part of)	Physical Stuff	Physical Stuff	P46(lace, knot)
P62 - depicts	Physical Man-Made Stuff	Entity	P62(still life, fruit)
P4 - has time span	Temporal Entity	Time Span	P4(pattern drawings, Renaissance)
P14 - carried out by (performed)	Activity	Actor	P14(blotted line drawings, Andy Warhol)
P92 - brought into existence by	Persistent Item	Beginning of Existence	P92(aquatints, aquatint process)
P45 - consists of (incorporated in)	Physical Stuff	Material	P45(sculpture, stone)

with R, while labeled recall is the number of words annotated correctly with R over the total number of words manually annotated with R.

Table 4 shows the results obtained by applying the checkers to tag the test set (containing a total number of 1,328 distinct annotations and 5,965 annotated words). Note that here we are evaluating the ability of the system to assign the correct tag to every word in a gloss fragment f, according to the appropriate relation checker. We choose to evaluate the tag assigned to single words rather than to a whole phrase, because each misalignment would count as a mistake even if the most part of a phrase was tagged correctly by the automatic annotator. The second experiment consisted in the evaluation of the property instances extracted. Starting from 1,328 manually annotated fragments of 692 glosses, the checkers extracted an overall number of 1,101 property instances. We randomly selected a subset of 160 glosses for evaluation, from which we manually extracted 344 property instances. Two aspects of the property instance extraction task had to be assessed:

1. the extraction of the appropriate *range words* in a gloss, for a given property instance;
2. the precision and recall in the extraction of the appropriate *concepts* for both *domain* and *range* of the property instance.

An overall number of 233 property instances were automatically collected by the checkers, out of which 203 were correct with respect to the first assessment (87.12% precision (203/233), 59.01% recall (203/344)). In the second evaluation, for each property instance $R(C_t, C_w)$ we assessed the semantic correctness of both the concepts C_t and C_w. The appropriateness of the concept C_t chosen for the domain must be evaluated, since, even if a term t satisfies the semantic constraints of the domain for a property R, it still can be the case that a fragment f in g does not refer to t, like in the following

Table 4. Precision and Recall of the gloss annotation task.

Property	Precision		Recall	
P26 - moved to	84.95%	(79/93)	64.23%	(79/123)
P27 - moved from	81.25%	(39/48)	78.00%	(39/50)
P53 - has former or current location	78.09%	(916/1173)	67.80%	(916/1351)
P55 - has current location	100.00%	(8/8)	100.00%	(8/8)
P46 - composed of	87.49%	(944/1079)	70.76%	(944/1334)
P62 - depicts	94.15%	(370/393)	65.26%	(370/567)
P4 - has time span	91.93%	(547/595)	76.40%	(547/716)
P14 - carried out by	91.71%	(343/374)	71.91%	(343/477)
P92 - brought into existence	89.54%	(471/526)	62.72%	(471/751)
P45 - consists of	74.67%	(398/533)	57.60%	(398/691)
Average performance	85.34%	(4115/4822)	67.81%	(4115/6068)

example:

pastels (visual works) – *Works of art*, typically on a paper or vellum support, to which designs are applied using crayons made of ground pigment held together with a binder, typically oil or water and gum.

In this example, *ground pigment* refers to *crayons* (not to *pastels*). The evaluation of the semantic correctness of the domain and range of the property instances extracted led to the final figures of 81.11% (189/233) precision and 54.94% (189/344) recall, due to 9 errors in the choice of C_t as a domain for an instance $R(C_t, C_w)$ and 5 errors in the semantic disambiguation of range words w not appearing in AAT, but encoded in WordNet (as described in the last part of Section 3).

3.3. Evaluating the Generality of the Approach

A final experiment was performed to evaluate the generality of the approach presented in this chapter.

As already remarked, the same procedure used for annotating the glosses of a thesaurus can be used to annotate web documents. Our objective in this last experiment was to:

- Evaluate the ability of the system to annotate fragments of web documents with CIDOC relations;
- Evaluate the domain dependency of the relation checkers, by letting the system annotate documents not in the cultural heritage domain.

We selected 5 documents at random from an historical archive and an artist's biographies archive[14] including about 6,000 words in total, about 5,000 of which in the historical domain. We then ran the automatic annotation procedure on these documents and we evaluated the result, using the same criteria as in Table 4. Table 5 presents the results of the experiment. Only 5 out of 10 properties had at least one instance in the analysed documents. It is remarkable that, especially for the less domain-dependent properties,

[14] http://historicaltextarchive.com and http://www.artnet.com/library

Table 5. Precision and Recall of a web document annotation task.

Property	Precision		Recall	
P53 - has former or current location	79.84%	(198/248)	77.95%	(198/254)
P46 - composed of	83.58%	(112/134)	96.55%	(112/116)
P4 - has time span	78.32%	(112/143)	50.68%	(112/221)
P14 - carried out by	60.61%	(40/66)	-	-
P45 - consists of	85.71%	(6/7)	37.50%	(6/16)
Average performance	78.26%	(468/598)	77.10%	(468/607)

the precision and recall of the algorithm is still high, thus showing the generality of the method. Notice that the historical documents influenced the result much more than the artist biographies, because of their reduced size.

In Table 5 the recall of *P14 (carried out by)* is omitted. This is motivated by the fact that this property, in a generic domain, corresponds to the agent relation ("an active animate entity that voluntarily initiates an action"[15]), while in the cultural heritage domain it has a more narrow interpretation (an example of this relation in the CIDOC handbook is: "the painting of the Sistine Chapel (E7) *was carried out by* Michelangelo Buonarroti (E21) *in the role of* master craftsman (E55)"). However, the domain and range restrictions for P14 correspond to an agent relation, therefore, in a generic domain, one should annotate as "carried out by" almost any verb phrase with the subject (including pronouns and anaphoric references) in the class Human.

4. Related Work and Conclusions

In this chapter we presented a method, based on the use of regular expressions, to automatically annotate the glosses of a thesaurus, the AAT, with the properties (conceptual relations) of a core ontology, the CIDOC-CRM. The annotated glosses are converted into OWL concept descriptions and used to enrich the CIDOC.

Several methods for ontology population and semantic annotation described in literature (e.g. [11,12,13,14]) use regular expressions to identify named entities, i.e. concept *instances*. Other methods extract hypernym relations using syntactic and lexical patterns [15,16] or supervised clustering techniques [17]. Evaluation of hypernymy learning methods is usually performed by a restricted team of experts, on a limited set of terms, with hardly comparable results, usually well over 40% error rate [18,19]. When the evaluation is an attempt to replicate the structure of an already existing taxonomy, the error rate is over 50-60% [20].

Semantic annotation with relations other than hypernymy are surveyed in [21], and again, regular expressions are a commonly used technique. Reeve and Han's survey presents a table to compare systems performance, but in absence of well-established data sets of annotated documents, a fair comparison among the various techniques is not possible. Similarly, comparing the performance of our system with those surveyed in [21] is not particularly meaningful.

As far as the adopted ontology learning technique is concerned, in our work we automatically formalize *concepts* (not simply instances or taxonomies, as in most literature)

[15] http://www.jfsowa.com/ontology/thematic.htm

compliant with the semantics of a well-established core ontology, the CIDOC (the interested reader can refer to the Chapters by Maynard et al. and by Tanev and Magnini in this volume for ontology population techniques). In contrast, the entire area of Information Extraction deals with the extensional acquisition of concepts (i.e. concept instances), rather than with an intensional formalization.

The method presented in this chapter is unsupervised, in the sense that it does not need manual annotation of a significant fragment of text. However, it relies on a set of manually written regular expressions, based on lexical, part-of-speech, and semantic constraints. The structure of regular expressions is rather more complex than in similar works using regular expressions, especially for the use of automatically verified semantic constraints. The issue is however how much these expressions generalize to other domains:

1. A first problem is the availability of lexical and semantic resources used by the algorithm. The most critical requirement of the method is the availability of sound *core ontologies*, which hopefully will be produced by other web communities stimulated by the recent success of CIDOC CRM. On the other side, *in absence of an agreed conceptual reference model, no large scale annotation is possible at all*. As for the other resources used by our algorithm, glossaries, thesaura and gazetteers are widely available in "mature" domains. If not, we developed a methodology, described in [22], to automatically create a glossary in novel domains (e.g. enterprise interoperability), extracting definition sentences from domain-relevant documents and authoritative web sites.
2. The second problem is about the generality of regular expressions. Clearly, the relation checkers that we defined are tuned on the CIDOC properties, however many of these properties are rather general (especially locative and temporal relations) and could easily apply to other domains, as demonstrated by the experiment on automatic annotation of historical archives in Table 4. Furthermore, the method used to verify semantic constraints is fully general, since it is based on WordNet and a general-purpose, untrained semantic disambiguation algorithm, SSI.

Finally, the authors are convinced that automatic pattern-learning methods often require non-trivial human effort just like manual methods (because of the need of annotated data, careful parameter setting, etc.), and furthermore they are unable to combine in a non-trivial way different types of features (e.g. lexical, syntactic, semantic). A practical example is the full list of automatically learned hypernymy-seeker patterns provided in [16]. The complexity of these patterns is certainly lower than the regular expression structures used in this work, and many of them are rather intuitive, they could have easily written by hand. However, we believe that our method can be automated to some limited degree (for example, semantic constraints can be learned automatically), a research line we are currently exploring.

Acknowledgements

This work was partially funded by the Interop Network of Excellence (508011), 6^{th} European Union FP.

References

[1] T. Berners-Lee, J. Hendler, and O. Lassila, The Semantic Web, *Scientific American*, 284(5) (2001).

[2] M. S. Fox, M. Barbuceanu, M. Gruninger, and J. Lin, An Organisation Ontology for Enterprise Modeling, In *Simulating Organizations: Computational Models of Institutions and Groups*, M. Prietula, K. Carley and L. Gasser (Eds), Menlo Park CA: AAAI/MIT Press (1997), 131–152.

[3] M. Uschold, M. King, S. Moralee and Y. Zorgios, The Enterprise Ontology, In *The Knowledge Engineering Review*, **13** (1998).

[4] M. Doerr, The CIDOC Conceptual Reference Module: An Ontological Approach to Semantic Interoperability of Metadata, *AI Magazine*, **24**(3) (2003).

[5] R. Navigli and P. Velardi, Structural Semantic Interconnections: a knowledge-based approach to word sense disambiguation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **27**(7) (2005), 1063–1074.

[6] G. A. Miller, WordNet: a lexical database for English, *Communications of the ACM*, **38**(11) (1995), 39–41.

[7] J. E. F. Friedl, *Mastering Regular Expressions*, O'Reilly (1997).

[8] P. Velardi, R. Navigli and M. Pétit, Semantic Indexing of a Competence Map to support Scientific Collaboration in a Research Community, In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Hyderabada, India (2007).

[9] N. Ide, J. Véronis, Refining Taxonomies Extracted from Machine-Readable Dictionaries. In Hockey, S., Ide, N. *Research in Humanities Computing II*, Oxford University Press (2003), 145–59.

[10] E. Charniak, Statistical Techniques for Natural Language Parsing, *AI Magazine*, **18**(4) (1997), 33–44.

[11] M. Thelen and E. Riloff, A Bootstrapping Method for Learning Semantic Lexicons using Extraction Pattern Contexts, In *Proceedings of the Conference on Empirical Methods in Natural Language Processing* (2002).

[12] M. E. Califf and R.J. Mooney, Bottom-up relational learning of pattern matching rules for information extraction, *Machine Learning research*, **4**(2) (2004), 177–210.

[13] P. Cimiano, G. Ladwig and S. Staab, Gimme the context: context-driven automatic semantic annotation with C-PANKOW, In *Proceedings of the 14th International WWW Conference*, Chiba, Japan (2005).

[14] A. G. Valarakos, G. Paliouras, V. Karkaletsis and G. Vouros, Enhancing Ontological Knowledge through Ontology Population and Enrichment, In *Proceedings of the* 14^{th} *EKAW conference*, Springer-Verlag (2004), 144–156.

[15] R. Snow, D. Jurafsky, and A. Y. Ng, Learning syntactic patterns for automatic hypernym discovery, In *NIPS* (2005).

[16] E. Morin and C. Jacquemin, Automatic acquisition and expansion of hypernym links, *Computer and the Humanities*, **38** (2004), 363–396.

[17] V. Kashyap, C. Ramakrishnan, T. Rindflesch, Toward (Semi)-Automatic Generation of Bio-medical Ontologies, In *Proceedings of American Medical Informatics Association* (2003).

[18] S. A. Caraballo, Automatic construction of a hypernym-labeled noun hierarchy from text, In *Proceedings of* 37^{th} *Annual Meeting of the Association for Computational Linguistics* (1999), 120–126.

[19] A. Maedche V. Pekar and S. Staab, Ontology learning part One: On Discovering Taxonomic Relations from the Web, In *Web Intelligence*, Chapter 1, Springer, 2002.

[20] D. Widdows, Unsupervised methods for developing taxonomies by combining syntactic and statistical information, In *Proceedings of HLT-NAACL*, Edmonton, Canada (2003), 197–204.

[21] L. Reeve and H. Han, Survey of Semantic Annotation Platforms, In *Proceedings of the* 20^{th} *Annual ACM Symposium on Applied Computing* (2005).

15

[22] R. Navigli and P. Velardi, Automatic Acquisition of a Thesaurus of Interoperability Terms, In *Proceedings of* 16^{th} *IFAC World Congress*, Praha, Czech Republic (2005).

Part III: Learning Relations

Unsupervised Learning of Semantic Relations for Molecular Biology Ontologies

Massimiliano CIARAMITA [a,1], Aldo GANGEMI [b], Esther RATSCH [c], Jasmin ŠARIĆ [d] and Isabel ROJAS [e]

[a] *Yahoo! Research Barcelona, Spain*
[b] *ISTC-CNR, Roma, Italy*
[c] *University of Würzburg, Germany*
[d] *Boehringer Ingelheim Pharma GmbH & Co. KG, Germany*
[e] *EML-Research gGmbH, Heidelberg, Germany*

Abstract. Manual ontology building in the biomedical domain is a work-intensive task requiring the participation of both domain and knowledge representation experts. The representation of biomedical knowledge has been found of great use for biomedical text mining and integration of biomedical data. In this chapter we present an unsupervised method for learning arbitrary semantic relations between ontological concepts in the molecular biology domain. The method uses the GENIA corpus and ontology to learn relations between annotated named-entities by means of several standard natural language processing techniques. An in-depth analysis of the output evaluates the accuracy of the model and its potentials for text mining and ontology building applications. The proposed learning method does not require domain-specific optimization or tuning and can be straightforwardly applied to arbitrary domains, provided the basic processing components exist.

Keywords. Unsupervised ontology learning, semantic relations, natural language processing, biology, bioinformatics

1. Introduction

Bioinformatics is one of the most active fields for text mining applications due to the fast rate of growth of digital document collections such as Medline [2] where more than 500,000 publications are added every year. The ultimate goal of text mining in bioinformatics is the automatic discovery of new knowledge about complex biomedical scientific problems. As an example, Swanson & Smalheiser [1] discovered a previously unnoticed correlation between migraine and magnesium by comparing complementary biomedical literatures.

[1] Corresponding Author: Yahoo! Research Barcelona, Ocata 1, 08003, Barcelona, Catalunya, Spain; E-mail: massi@yahoo-inc.com.

[2] Pubmed/Medline: http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?DB=pubmed

Current approaches to text mining are mainly based on the application of natural language processing (NLP) and machine learning. Text mining concerns the acquisition of relevant information contained within documents by means of *information extraction* methods. The starting point is a conceptualization of the domain; e.g., a *domain ontology*, which specifies relevant *concepts* as well as *semantic relations*, e.g., is-a, part-of, and more complex relations encoding important interactions between concepts. Given the domain ontology, information extraction techniques can be applied to recognize where, in the documents, concepts are instantiated by specific entities, and where important interactions are expressed by linguistic structures.

Several ontologies which define concepts and structural semantic relations (e.g., is-a) are available. However, there is a need for ontologies that specify relevant arbitrary semantic relations between concepts. In other words, information about fundamental attributes of concepts and patterns of interactions between concepts, which constitutes the basic "world-knowledge" relative to the specific domain. For example, that "Cell express-the-receptor-for Protein" or that "Virus replicate-in Cell". In this chapter we discuss a method for enriching an existing ontology with arbitrary semantic relations which are strongly associated with ordered pairs of concepts. The method was originally introduced in [2], here we present the original formulation of the learning method and experimental evaluation, and present additional discussion also concerning further developments of our method.

The method concerns the implementation of an unsupervised system that combines an array of off-the-shelf NLP techniques such as syntactic parsing, collocation extraction and selectional restriction learning. The system was applied to a corpus of molecular biology literature, the GENIA corpus [3], and generated a list of labeled binary relations between pairs of GENIA ontology concepts. An in-depth analysis of the learned templates shows that the model, characterized by a very simple architecture, has good accuracy and can be easily applied in text mining and ontology building applications.

In the next section we describe the problem of learning relations from text and related work. In Section 3 we describe our system and the data used in our study in detail. In Section 4 we discuss the evaluation of the system's output.

2. Problem statement and related work

The GENIA ontology contains concepts related to gene expression and its regulation, including cell signaling reactions, proteins, DNA, and RNA. Much work in bioinformatics has focused on *named-entity recognition* (NER), or *information extraction* (IE)[3], where the task is the identification of sequences of words that are instances of a set of concepts. As an example, one would like to recognize that "NS-Meg cells", "mRNA" and "EPO receptor" are, respectively, instances of the GENIA classes "Cell_line", "RNA_family_or_group" and "Protein_molecule" in Example 1 below:

(1) "Untreated [$_{\text{Cell\_line}}$ NS-Meg cells] expressed [$_{\text{RNA\_family\_or\_group}}$ mRNA] for the [$_{\text{Protein\_molecule}}$ EPO receptor]"

[3]The task of information extraction should in principle focus more on the identification of relations involving entities. However, as Rosario & Hearst [4] point out much of the work in this area has in fact addressed primarily the entity detection problem.

A natural extension of NER is the extraction of relations between entities. NER and relation extraction can provide a better support for mining systems; e.g., patterns of entities and relations could be compared across document collections to discover new informative pieces of evidence concerning previously overlooked phenomena. Currently most work on relation extraction in bioinformatics applies hand-built rule-based extraction patterns; e.g., Friedman et al. [5] on identifying molecular pathways and Šarić et al. [6] on finding information about protein interactions by using a manually-built ontology similar to that described in [7]. One limitation of rule-based information extraction is that systems tend to have good precision but low recall. Machine learning-oriented work has focused on extracting manually-compiled lists of target relations; e.g., Rosario and Hearst [4] address the relation extraction problem as an extension of NER and use sequence learning methods to recognize instances of a set of 6 manually predefined relations about "Diseases" and "Treatments". These systems yield good precision and recall but still requires sets of relations between classes be defined first.

Yet another problem which deals with semantic relations is that addressed by Craven and Kumlien [8] who present a model for finding extraction patterns for 5 binary relations involving proteins. A similar work is that of Pustejovsky et al. [9] on automatically extracting "inhibit" relations. Semantic relations have been used as templates, or guiding principles, for the generation of database schemata [10]. Another application of ontological relations is that of consistency checking of data in molecular biology databases to individuate errors in the knowledge base (e.g., by checking the consistency of the arguments) or to align different databases.

Text mining systems involving relations require predefined sets of relations that have to be manually encoded, a job which is complex, expensive and tedious, and that as such can only guarantee narrow coverage – typically a handful of relations and one pair of classes, and thus neglect informative and useful relations. The goal of our system is to automatically generate all relevant relations found in a corpus between all ontological concepts defined in the ontology. Such systems would also be valuable to ontologists since ontology building and evaluation are becoming more and more automatized activities and most of the corpus-based work has focused only on structural relations such as is-a and part-of [11,12]. Another related problem is that of "ontologizing" automatically harvested semantic relations, i.e., to link them to existing semantic repositories. Pantel and Pennacchiotti [13], in this same volume, present an overview of the area and propose an accurate method for this task.

Our approach is related to those of Reinberger et al. [14] and Rinaldi et al. [15]. Both works differ with ours in the following aspects. First, they both rely on heuristic means to identify relations, Reinberger et al. focus on subject-verb-object patterns, while the method of Rinaldi et al. identifies relations by means of manually-created patterns. Our method is not limited to a pre-defined set of patterns, and propose a simple and clear way for representing, ranking and selecting arbitrary relations. Secondly, we present a simple and principled solution to assigning a score to candidate relations and filtering out unreliable ones by means of hypothesis testing. Finally, we investigate a method for generalizing the relations arguments to their superordinate classes based on corpus evidence using techniques for learning selectional preferences. A closer comparison of our method and Reinberger et al.'s, which describe a similar evaluation, can be found in Section 4.

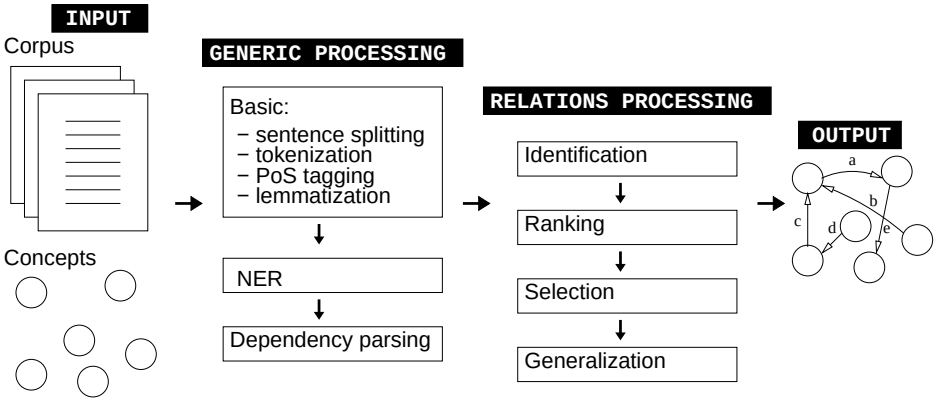

Figure 1. Overview of the proposed relation learning system.

3. Learning relations from text

3.1. Overview

Figure 1 illustrates schematically our method. The system for learning relations takes as input a corpus and a set of classes. A basic pre-processing step involves sentence splitting, tokenization, PoS tagging and lemmatization. The next step consists in identifying entity mentions in the documents. Since the focus of the experiments presented in this paper was the relations selection process we used the GENIA corpus in which named-entities corresponding to ontology concepts have been manually identified[4]. However, suitable corpus data can also be generated automatically using an appropriate NER system and basic NLP tools. The corpus data is then parsed so that relations can be defined and extracted based on the syntactic structure of the sentences. In this paper we used a constituent syntactic parser [16], however for efficiency and simplicity, a viable alternative would be to use directly a dependency parser since dependency treebanks are nowadays available in several languages [17]. Using the dependency structure of the sentence the methods generates a set of candidate relations which are assigned a score. The relations for which there is strong supporting evidence in the corpus are selected and can be added to the original ontology. Thus the model outputs a set of templates that involve pairs of GENIA ontology classes and a semantic relation. For example, a template might be "Virus infect Cell". In the remaining of this section we illustrate the resources used as input to our system, the GENIA corpus and ontology, and describe the system components in detail.

3.2. Corpus and ontology concepts

The GENIA ontology was built to model cell-signaling reactions in humans with the goal of supporting information extraction systems. It consists of a taxonomy of 46 nominal concepts with underspecified taxonomic relations, see Figure 2. We refer to concepts also with the terms "label" or "tag". The ontology was used to semantically annotate

[4]The previous pre-processing steps, sentence segmentation, PoS tagging, etc., have been carried out as well.

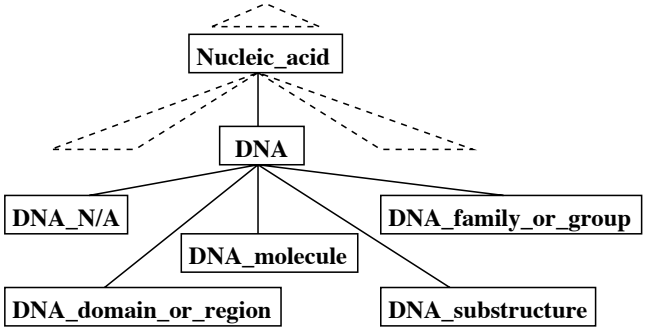

Figure 2. A small fraction of the GENIA ontology. Continuous lines represent unspecified taxonomic relations, dashed lines represent other regions.

biological entities in the GENIA corpus. We used version G3.02 consisting of 2,000 articles, 18,546 sentences, roughly half a million word tokens, and 36 types of labels. This corpus has complex annotations for disjunctive/conjunctive entities, for cases such as "erythroid, myeloid and lymphoid cell types". We excluded sentences that contained only instances of complex embedded conjunctions/disjunctions and also excessively long sentences (more than 100 words). The final number of sentences was 18,333 (484,005 word tokens, 91,387 tags). Many tags have nested structures; e.g. "[$_{\text{Other\_name}}$ [$_{\text{DNA}}$ IL-2 gene] expression]". For these cases we only considered the innermost entities, although the external labels contain useful information and should eventually be considered.

One potential drawback of the GENIA ontology is the relatively small number of biological concepts and their coarse granularity which causes groups of similar but distinct entities to be assigned to the same class. Some relations fit very well to subsets of the entities of the related concepts, whereas they don't fit well for other entities of the same concept. For example, the concept "DNA_domain_or_region" contains sequences with given start and end positions, as well as promoters, genes, enhancers, and the like. Even if promoters, genes, and enhancers are pieces of sequences too (with start and end positions), they also are functional descriptions of sequences. Therefore, different statements can be made about such kinds of DNA domains or regions and (pure) sequences. The relation "DNA_domain_or_region encodes Protein_molecule" makes sense for genes, but not for enhancers, and may make sense or not for (pure) sequences, depending on their (unknown) function. However, suitable annotated resources are scarce, and in this respect the GENIA corpus is unique in that it provides extensive named-entity annotations which can be used to train appropriate NER systems (cf. [18]). Recent work on extensions and refinements of the GENIA ontology, such as xGENIA [19] may lead to augmenting the resolution of the output of system like ours, given the extended annotations and classifications of concepts and relations.

3.3. Relations as dependency paths

The 18,333 sentences were parsed with a statistical constituent parser [16].[5] Since we are interested in relations that connect entities as chunks we want to avoid that the parser

[5] It took roughly three hours on a Pentium 4 machine to parse the target sentences.

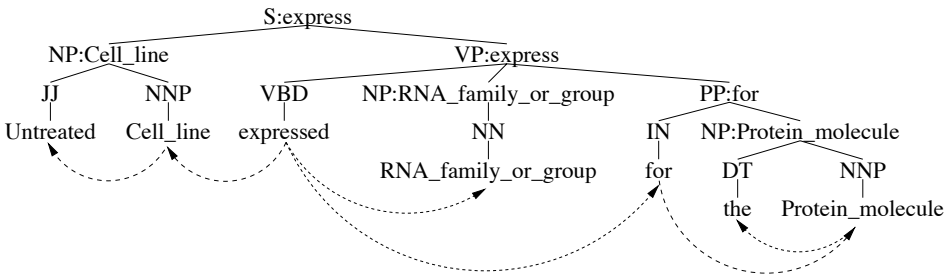

Figure 3. Parse tree for the sentence of Example 1. Entities are substituted with their tags. Phrases are labeled with their syntactic heads. The dependency graph is depicted with dashed directed edges pointing to the governed element.

analyzes an entity that is split among different phrases. This can happen because entity names can be fairly long, complex and contain words that are unknown to the parser. To avoid this problem we substituted the entity tags for the actual named-entities; the result can be seen in Figure 3 which shows the substitution and the relative parse tree for the sentence of Example 1. Trees obtained in this way are simpler and don't split entities across phrases. Alternatively, if it is important to retrieve the internal structure of the entities as well, the detected entities might be used as soft features rather than atomic tokens, which can help the parser in dealing with unknown words (e.g., as in [20]).

For each tree we generated a *dependency graph*: each word[6] is associated with one *governor*, defined as the *syntactic head*[7] of the phrase closest to the word that differs from the word itself. For example, in Figure 3 "Cell_line" is governed by "express", while "Protein_molecule" is governed by the preposition "for".

Similarly to what has been proposed for the task of recognizing paraphrase sentences [22], the dependency structure can be used to formalize the notion of semantic relation between two entities. A relation r between two entities c_i and c_j in a tree is the path between c_i and c_j following the dependency relations. As an example, in Figure 3 the path between "Cell_line" and "Protein_molecule" is "←express→for→". There is a path for every pair of entities in the tree. Paths can be considered from both directions, since the reverse of a path from A to B is a path from B to A. A large number of different patterns can be extracted, overall we found 172,446 paths in the dataset. For the sake of interpretability of the system's outcome we focused on a subset of these patterns. We selected paths from c_i to c_j where $j > i$ and the pivotal element, the word with no incoming arrows, is a verb v. In addition we imposed the following constraints: c_i is governed by v under an S phrase (i.e., is v's surface subject, SUBJ), e.g., "Cell_line" in Figure 3; and one of the following six constraints holds:

1. c_j is governed by v under a VP (i.e., is v's direct object, DIR_OBJ), e.g., "RNA_family_or_group" in Figure 3;
2. c_j is governed by v under a PP (i.e., is v's indirect object, IND_OBJ), e.g. "Protein_molecule" in Figure 3;

[6] Morphologically simplified with the "morph" function from the WordNet library [21], plus morphological simplifications from UMLS.

[7] The word whose syntactic category determines the syntactic category of the phrase; e.g., a verb for a verb phrase (VP), a noun for a noun phrase (NP), etc.

3. c_j is governed by v's direct object noun (i.e., is a modifier of the direct object, DIR_OBJ_MOD), e.g. "Virus" in "... influenced *Virus* replication";
4. c_j is governed by v's indirect object noun (i.e., is the indirect object's modifier, IND_OBJ_MOD), e.g., "Protein_molecule" in "..was induced by *Protein_molecule* stimulation";
5. c_j is governed by a PP which modifies the direct object (DIR_OBJ_MOD_PP); e.g., "Protein_molecule" in ".. induce overproduction of *Protein_molecule*";
6. c_j is governed by a PP which modifies the indirect object (IND_OBJ_MOD_PP); e.g., "Lipid" in "..transcribed upon activation with *Lipid*".

In the sentence of the previous example, see Figure 3, we identify two good patterns: "SUBJ←express→DIR_OBJ" between "Cell_line" and "RNA_family_or_group", and "SUBJ←express→for→IND_OBJ", between "Cell_line" and "Protein_molecule". It is important to notice that this selection is only necessary for evaluation purposes. However, all relations are retrieved and scored and could be used for mining purposes, although they might not be easy to interpret by inspection. Overall we found 7,189 instances of such relations distributed as follows:

Type	Counts	RelFreq
SUBJ-DIR_OBJ	1,746	0.243
SUBJ-IND_OBJ	1,572	0.219
SUBJ-DIR_OBJ_MOD_PP	1,156	0.161
SUBJ-DIR_OBJ_MOD	943	0.131
SUBJ-IND_OBJ_MOD_PP	911	0.127
SUBJ-IND_OBJ_MOD	861	0.120

The data contained 485 types of entity pairs, 3,573 types of patterns and 5,606 entity pair-pattern types.

3.4. Ranking and selection of relations

Let us take A to be an ordered pair of GENIA classes; e.g. A = (Protein_domain, DNA_domain_or_region), and B to be a pattern; e.g., B = SUBJ←bind→DIR_OBJ. Our goal is to find relations strongly associated with ordered pairs of classes, i.e., bi-grams AB. This problem is similar to finding *collocations*; e.g., multi-word expressions such as "real estate", which form idiomatic phrases. Accordingly the simplest method would be to select the most frequent bi-grams. However many bi-grams are frequent because either A or B, or both, are frequent; e.g., SUBJ←induce→DIR_OBJ is among the most frequent pattern for 37 different pairs. Since high frequency can be accidental and, additionally, the method doesn't provide a natural way for distinguishing relevant from irrelevant bi-grams, we use instead a simple statistical method.

As with collocations a better approach is to estimate if A and B occur together more often than at chance. One formulates a *null hypothesis* H_0 that A and B do not occur together more frequently than expected at chance. Using corpus statistics the probability of $P(AB)$, under H_0, is computed and H_0 is rejected if $P(AB)$ is beneath the significance level. For this purpose we used a chi-square test. For each observed bi-gram we created a contingency table of the frequencies AB, ¬AB, A¬B, and ¬A¬B; e.g., for A = Protein_molecule-DNA_domain_or_region, and B = SUBJ←bind→DIR_OBJ, the table computed from the corpus contains the values 6, 161, 24 and 6,998 (for AB, ¬AB,

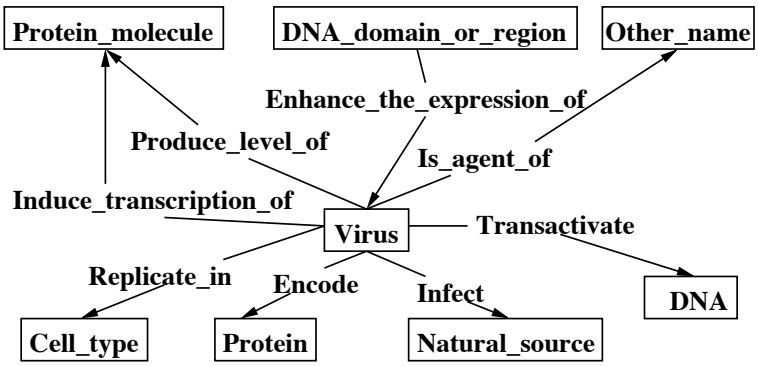

Figure 4. The "Virus" concept with the selected and generalized relations, and related concepts, in the enriched ontology.

A¬B, and ¬A¬B, respectively). The chi-square test compares the observed frequencies vs. the frequencies expected under H_0. Together with the test we use the log-likelihood chi-squared statistic: [8]

$$G^2 = 2 \sum_{i,j} o_{ij} \log \frac{o_{ij}}{e_{ij}} \tag{2}$$

where i and j range over the rows and columns of the contingency table, and the expected frequencies are computed off the marginal frequencies in the table. Hence the value generated by the statistic can be used as a score to rank the candidate relations, and a principled way of selecting the most reliable ones. In the previous example G^2 is equal to 16.43, which is above the critical value 7.88 for $\alpha = 0.005$, hence B is accepted as a relevant pattern for A. The following table shows the three highest ranked class pairs for pattern B. There is strong evidence that entities of the protein type tend to bind DNA locations, which is a reasonable conclusion:

(3)

B = SUBJ←bind→DIR_OBJ		
A	G^2	Select
Protein_domain-DNA_domain_or_region	16.43	YES
Protein_family_or_group-DNA_d._or_r.	13.67	YES
Virus-Protein_molecule	7.84	NO

In our study we used $\alpha = 0.005$. In general, α is an adjustable parameter which might be set on held-out data in order to maximize an objective function. We also ignored bigrams occurring less than 2 times and pairs A, patterns B, occurring less than 4 times. Overall there are 487 suitable AB pairs, 287 (58.6%) have a value for G^2 higher than α.

3.5. Generalization of relations

Relations can share similar arguments as in the case of "bind" in Example (3) above, where, in both significant cases, the direct object is "DNA domain or region" while the

[8] Dunning [23] argues that G^2 is more appropriate than Pearson's X^2 with sparse data; here they produce similar rankings.

subject is some kind of protein. This can be evidence that, in fact, there is a more general relation holding between super-ordinates of the arguments found after relation ranking and selection. Thus, it is desirable, when possible, to learn more general relations such as "Protein SUBJ←bind→DIR_OBJ DNA", because the learned ontology is more compact and has greater generalization power, i.e., relations apply to more entities. Finding such generalizations is similar to learning *selectional restrictions* of predicates, that is, the preferences that predicates place on the semantic category of their arguments; e.g., that "eat" prefers objects that are "foods". Several methods have been proposed for learning such restrictions; e.g., see [24] for an overview. We used the method proposed in [25] which is both accurate and simple, and is also based on hypothesis testing and frequency estimates related to those used in the relation selection step . We used the taxonomy defined in the GENIA ontology, see Figure 2, to generalize arguments of the learned patterns.[9]

Clark and Weir define an algorithm, $top(c, r, s)$, which (adjusting the terminology to our case) takes as input a relation r, a class c and a syntactic slot s, and returns a class c' which is c itself or one of its ancestors, whichever provides the best generalization for $p(r|c, s)$. The method uses the chi-squared test to check if the probability $p(r|c, s)$ is significantly different from $p(r|c', s)$, where c' is the parent of c. If this is false then $p(r|c', s)$ is supposed to provide a good approximation for $p(r|c, s)$, which is interpreted as evidence that (r, s) holds for c' as well. The procedure is iteratively applied until a significant difference is found. The last class considered is the output of the procedure, the concept that best summarizes the class that r "selects" in syntactic slot s. We computed the frequencies of patterns involving superordinate classes summing over the frequencies, from the GENIA corpus, of all descendants of that class for that pattern.

For each relation r, slot s and class c, learned in the selection stage, we used Clark and Weir's method to map c to $top(c, r, s)$. We again used the G^2 statistic and the same α value of 0.005. Using these maps we generalized, when possible, the original 287 patterns learned. The outcome of this process was a set of 240 templates, 153 of which had generalized arguments. As an example, the templates above "Protein_domain binds DNA_domain_or_region" and "Protein_family_or_group binds DNA_domain_or_region" are mapped to the generalized template "Protein binds DNA". Figure 4 depicts the set of labeled relations the concept "Virus" is involved in, and the respective paired concepts, after relation selection and generalization. As the figure shows relations can involve generalized concepts; e.g., the right argument of the template"Virus transactivate DNA" involves DNA, an internal node in the GENIA ontology (see Figure 2), which has been generalized automatically.

In [26] Cimiano et al. investigate further the issue of determining the appropriate level of abstraction for binary relations extracted from a corpus and present a detailed review of the existing techniques, suggesting also a new analysis of different evaluation measures.

[9] Four of the 36 GENIA corpus class labels, namely, "DNA_substructure", "DNA_N/A", "RNA_substructure" and "RNA_N/A", have no entries in the GENIA ontology, we used them as subordinates of " DNA" and "RNA", consistently with "Protein_N/A" and "Protein_substructure" which in the ontology are subordinates of "Protein".

4. Evaluation

We discuss now an evaluation of the model carried out by a biologist and an ontologist, both familiar with GENIA. The biological evaluation focuses mainly on the *precision* of the system; namely, the percentage of all relations selected by the model that, according to the biologist, express correct biological interactions between the arguments of the relation. From the ontological perspective we analyze semantic aspects of the relations, mainly the consistency with the GENIA classes.

4.1. Biological evaluation

The output of the relation selection process (see Section 3.4) is a set of 287 patterns, composed of an ordered pair of classes and a semantic relation. 91 of these patterns, involving in one or both arguments the class "Other_name", were impossible to evaluate and excluded altogether. This GENIA class is a placeholder for very different sorts of subconcepts, which have not yet been partitioned and structured. Relations involving "Other_name" (e.g., "treat") might prove correct for a subset of the entities tagged with this label (e.g., "inflammation") but false for a different subset (e.g., "gene expression"). Of the remaining 196 patterns 76.5% (150) are correct, i.e., express valid biological facts such as "Protein_molecule induce-phosphorylation-of Protein_molecule", while 23.5% (43) are incorrect, e.g. "Protein inhibit-expression-of Lipid". Evaluation involved the exhaustive inspection of the original sentences to verify the intended meaning of the pattern and spot recurring types of errors. Half of the mistakes (22) depend on how we handle coordination, which causes part of the coordinated structure to be included in the relation. For example, the first two DNA entities in the noun phrase "DNA, DNA, and DNA" are governed by the head DNA rather than by, say, the main verb. Thus wrong relations such as "Protein bind-DNA DNA" are generated in addition to good ones such as "Protein bind DNA". Fixing this problem would involve either a more sophisticated handling of coordinated structures or, more simply, filtering out redundant relations in a post processing step. Finally, 5 errors involved the class "Other_name" embedded somewhere within the relation[10], suggesting again generalizations that cannot be judged with enough confidence. The remaining errors are probably due to sparse data problems. In this respect it would probably be beneficial to apply a NER system to a larger unannotated corpus to produce more data and consequently more reliable distributional information. Arguably the use of automatically generated entity labels would introduce errors and noise in the process, however it is reasonable to expect that significantly larger amounts of data would generate larger numbers of good relations at the top of the relation rankings.

Finally, we notice that, although the GENIA ontology was intended to be a model of cell signaling reactions, it lacks important concepts such as *signaling pathway*. This leads to some errors as in the following case: "An intact TCR signaling pathway is required for p95vav to function.". In this case we derive the relation: "Protein_molecule is-required-for Protein_molecule" since only "TCR" is annotated as "Protein_molecule" neglecting *signaling pathway*.

To the best of our knowledge we can compare these results with one other study. Reinberger et al. [14] evaluate – also by means of experts – 165 subject-verb-object

[10] In other words, the label "Other_name" was found as part of the relation itself as in "Protein bind-Other_name DNA".

relations, extracted from data similar to ours[11] but with a different approach. They report an accuracy of 42% correct relations. Their method differs from ours in three respects: relations are extracted between nouns rather than entities (i.e., NER is not considered), a shallow parser is used instead of a full parser, and relations are selected by frequency rather than by hypothesis testing. A direct comparison of the methods is not feasible. However, if the difference in accuracy reflects the better quality of our method this is likely to depend on any, or on a combination, of those three factors.

As far as the generalization of relations is concerned we first removed all relations involving "Other_name" (40 out of 153), which do not have super-ordinates nor subordinates, and evaluated if the remaining 113 generalized patterns were correct. Of these, 60 (53.1%) provided valid generalizations; e.g., "Protein_molecule induce-phosphorylation-of Amino_acid_monomer" is mapped to "Protein induce-phosphorylation-of Amino_acid_monomer". Excluding mistakes caused by the fact that the original relation is incorrect, over-generalization seems mainly due to the fact that the taxonomy of the GENIA ontology is not simply a is-a hierarchy; e.g., "DNA_substructure" is not a kind of "DNA", and "Protein" is not a kind of "Amino_acid". Generalizations such as selectional restrictions instead seem to hold mainly between classes that share a relation of inclusion. In order to support this kind of inference the structural relations between GENIA classes would need to be clarified.

4.2. Ontological assessment

The 150 patterns validated by the expert are potential new components of the ontology. We compiled GENIA, including the newly learned relations, in OWL (Ontology Web Language [27]) to assess its properties with ontology engineering tools. Ignoring "Other_name", the GENIA taxonomy branches from two root classes: "Source" and "Substance". GENIA classes, by design, tend to be *mutually exclusive*, meaning that they should be logically disjoint. Our main objective is to verify the degree to which the new relations adhere to this principle.

To analyze the relations we map, "Source" and "Substance" to equivalent classes of another more general ontology. Ideally, the alignment should involve an ontology of the same domain such as TAMBIS [28]. Unfortunately TAMBIS scatters the subordinates of "Source" (organisms, cells, etc.) across different branches, while "Substance" in TAMBIS does not cover protein and nucleic acid-related subordinates of "Substance" in GENIA.[12] In GENIA substances are classified according to their chemical features rather than biological role, while sources are biological locations where substances are found and their reactions take place. This distinction assumes a stacking of ontology layers within the physical domain where the biological is superimposed to the chemical level. This feature of GENIA makes it suitable for alignment with DOLCE-Lite-Plus (DLP, http://dolce.semanticweb.org), a simplified translation of the DOLCE foundational ontology [29]. DLP specifies a suitable distinction between "chemical" and "biological" objects. It features about 200 classes, 150 relations and 500 axioms and has been used in various domains including bio-medicine [30].

We aligned "Source" and "Substance" to the biological and chemical classes in DLP. There are 78 types of relations out of 150, 58% of them (45) occur only with one pair of

[11] The SwissProt corpus, 13 million words of Medline abstracts related to genes and proteins.

[12] Notice that we are not questioning the quality of TAMBIS, but only its fitness for aligning GENIA.

classes, i.e., are monosemous, while 33 have multiple domains or ranges, i.e., are polysemous. Since the root classes of GENIA are disjoint we checked if there are polysemous relations whose domain or range mix up subclasses of "Source" with subclasses of "Substance". Such relations might not imply logical inconsistency but raise doubts because they suggest the possibility that a class of entities emerged from the data, which is the union of two classes that by definition should be disjoint. Interestingly, there are only 4 such relations out of 78 (5.1%); e.g., "encode", whose subject can be either "Virus" or "DNA". In biology, DNA encodes a protein, but biologists sometimes use the verb "metonymically". By saying that a virus encodes a protein, they actually mean that a virus' genome contains DNA that encodes a protein. The small number of such cases suggests that relations emerging from corpus data are consistent with the most general classes defined in GENIA.

At a finer semantic level relations are composed as follows: 54 (68%) are *eventive*, they encode a conceptualization of chemical reactions as events taking place in biological sources; 81% of the relations between biological and chemical classes are eventive, supporting the claim made in GENIA that biologically relevant chemical reactions involve both a biological and chemical object. Non-eventive relations have either a *structural* (e.g. "Consist-of"), *locative* ("Located-in"), or *epistemological* meaning ("identified-as").

5. Discussion and conclusion

In this chapter we presented a study on learning semantic relations from text in the domain of molecular biology. We proposed a system, see Figure 1, which takes as input a corpus of documents and a set of concepts, applies several language processing steps and generates a set of candidate relations which are then ranked, selected and possibly generalized by means of corpus statistics and hypothesis testing.

The method is based on the idea that relations can be represented as syntactic dependency paths between an ordered pair of named-entities. The most complex steps of our method thus involve parsing and entity detection. In this work we used a full constituent parser, however relations can be straightforwardly extracted directly from dependency trees for which accurate linear time parsers exist (e.g., see [20]). The other relatively complex step involves entity detection, for which also accurate linear time algorithms exist; e.g., based on discriminative Hidden Markov Models. Therefore the pre-processing computational cost of this dependency/entity-based approach is quite reasonable, while for the other necessary NLP steps there exist good publicly available resources for English and also several other languages.

We empirically investigated our method using the GENIA corpus and ontology The results of a biological and ontological analysis of the acquired relations are positive and promising. Arguably this type of method works well if the goal is precision rather than recall. That is, by imposing sufficiently conservative thresholds it is likely that the top ranked results will be accurate. However, other aspects need to be addressed beyond precision, in particular it would be important to evaluate the recall, i.e., the coverage, of the system. This task is problematic because it requires, in principle, considering a very large number of discarded relations. Other aspects that would be interesting to evaluate are the precision of alternative selection criteria, and the usefulness of automati-

cally learned relations in text mining. Another aspect which needs to be addressed is the identification of synonymic relations; e.g., in the context of Protein-Protein interaction "positively-regulate" is equivalent to "activate", "up-regulate", "derepress", "stimulate" etc. As a start, by representing relations as dependency paths one could frame this problem straightforwardly as that of finding paraphrases (e.g. as in [22]).

As a final remark, we highlight the fact that the method we propose can be applied fully unsupervised and domain-independent. Our method involves only one adjustable parameter, the confidence level α which can be set by default to standard conservative level in hypothesis testing, here we used $\alpha = .005$. Thus, by design, our method is, in principle, language and domain independent, provided the necessary NLP tools exist, although the quality of the output might differ in different domains.

Acknowledgments

We would like to thank the members of the Laboratory for Applied Ontology (LOA-CNR) for useful discussions, and the Klaus Tschira Foundation for their financial support.

References

[1] D.R. Swanson and N.R. Smalheiser. An interactive system for finding complementary literatures: A stimulus to scientific discovery. *Journal of Artificial Intelligence Research*, 12:271–315, 2000.

[2] M. Ciaramita, A. Gangemi, E. Ratsch, J. Šarić, and I. Rojas. Unsupervised learning of semantic relations between concepts of a molecular biology ontology. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, 2005.

[3] Y. Ohta, Y. Tateisi, J. Kim, H. Mima, and J. Tsujii. The GENIA corpus: An annotated research abstract corpus in the molecular biology domain. In *Proceedings of Human Language Technology Conference*, 2002.

[4] B. Rosario and M. Hearst. Classifying semantic relations in bioscience text. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, 2004.

[5] C. Friedman, P. Kra, H. Yu, M. Krauthammer, and A. Rzhetsky. GENIES: A natural-language processing system for the extraction of molecular pathways from journal articles. *Bioinformatics*, 17(1), 2001.

[6] J. Šarić, L.J. Jensen, R. Ouzounova, I. Rojas, and P. Bork. Extraction of regulatory gene expression networks from PubMed. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, 2004.

[7] E. Ratsch, J. Schultz, J. Šarić, P. Cimiano, U. Wittig, U. Reyle, and I. Rojas. Developing a protein interactions ontology. *Comparative and Functional Genomics*, 4(1):85–89, 2003.

[8] M. Craven and J. Kumlien. Constructing biological knowledge bases by extracting information from text sources. In *Proceedings of the 7th International Conference on Intelligent Systems for Molecular Biology*, 1999.

[9] J. Pustejovsky, J. Castaño, J. Zhang, B. Cochran, and M. Kotechi. Robust relational parsing over biomedical literature: Extracting inhibit relations. In *Proceedings of the Pacific Symposium on Biocomputing*, 2002.

[10] I. Rojas, L. Bernardi, E. Ratsch, R. Kania, U. Wittig, and J. Šarić. A database system for the analysis of biochemical pathways. *Silico Biology*, 2, 2007.

[11] M. Berland and E. Charniak. Finding parts in very large corpora. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, 1999.

[12] P. Pantel and D. Ravichandran. Automatically labeling semantic classes. In *Proceedings of the Human Language Technology and North American Chapter of the Association for Computational Linguistic Conference*, 2004.

[13] P. Pantel and M. Pennacchiotti. Automatically harvesting and ontologizing semantic relations. In P. Buitelaar and P. Cimiano, editors, *Bridging the Gap between Text and Knowledge - Selected Contributions to Ontology Learning and Population from Text*. IOS Press, 2007. THIS VOLUME.

[14] M-L Reinberger, P. Spyns, and A.J. Pretorius. Automatic initiation of an ontology. In *Proceedings of ODBase 2004*, 2004.

[15] F. Rinaldi, G. Schneider, K. Kaljurand, M. Hess, and M. Romacker. An environment for relation mining over richly annotated corpora: The case of GENIA. *BMC Bioinformatics*, 7, 2006.

[16] E. Charniak. A maximum-entropy-inspired parser. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, 2000.

[17] S. Buchholz and E. Marsi. Introduction to CoNNL-X shared task on multilingual dependency parsing. In *Proceedings of 10th Conference on Computational Natural Language Learning*, 2006.

[18] J. Kazama, T. Makino, Y. Ohta, and J. Tsujii. Tuning support vector machines for biomedical named entity recognition. In *Proceedings of the Workshop on Natural Language Processing in the Biomedical Domain*, 2002.

[19] R. Rak, L. Kurgan, and M. Reformat. xGENIA: A comprehensive OWL ontology based on the GENIA corpus. *Bioinformation*, 1(9):360–362, 2007.

[20] M. Ciaramita and G. Attardi. Dependency parsing with second-order feature maps and annotated semantic information. In *Proceedings of the 10th International Conference on Parsing Technology*, 2007.

[21] C. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA, 1998.

[22] D. Lin and P. Pantel. DIRT - Discovery of inference rules from text. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, 2001.

[23] T. Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1), 1993.

[24] M. Light and W. Greiff. Statistical models for the induction and use of selectional preferences. *Cognitive Science*, 87, 2002.

[25] S. Clark and D. Weir. Class-based probability estimation using a semantic hierarchy. *Computational Linguistics*, 28, 2002.

[26] P. Cimiano, M. Hartung, and E. Ratsch. Finding the appropriate generalization level for binary relations extracted from the Genia corpus. In *Proceedings of the International Conference on Language Resources and Evaluation*, pages 161–169, 2006.

[27] D. McGuinness and F. van Harmelen. Owl web ontology language overview. In *W3C Recommendations: http://www.w3c.org/TR/owl-features/*, 2004.

[28] R. Stevens, P. Baker, S. Bechhofer, A. Jacoby G. Ng, N.W. Paton, C.A. Goble, and A. Brass. TAMBIS: Transparent access to multiple bioinformatics information sources. *Bioinformatics*, 16(2), 2000.

[29] A. Gangemi, Guarino N., C. Masolo, and A. Oltramari. Sweetening WordNet with DOLCE. *AI Magazine*, 24(3), 2003.

[30] J. Šarić, E. Ratsch, I. Rojas, R. Kania, U. Wittig, and A.Gangemi. Modelling gene expression. In *Proceedings of the Workshop on Models and Metaphors from Biology to Bioinformatics Tools*, 2004.

Part IV: Ontology Population

NLP Techniques for Term Extraction and Ontology Population

Diana MAYNARD [1], Yaoyong LI and Wim PETERS
Dept. of Computer Science, University of Sheffield, UK

Abstract.
This chapter investigates NLP techniques for ontology population, using a combination of rule-based approaches and machine learning. We describe a method for term recognition using linguistic and statistical techniques, making use of contextual information to bootstrap learning. We then investigate how term recognition techniques can be useful for the wider task of information extraction, making use of similarity metrics and contextual information. We describe two tools we have developed which make use of contextual information to help the development of rules for named entity recognition. Finally, we evaluate our ontology-based information extraction results using a novel technique we have developed which makes use of similarity-based metrics first developed for term recognition.

Keywords. information extraction, ontology population, term recognition,

1. Introduction

In semantic web applications, ontology development and population are tasks of paramount importance. The manual performance of these tasks is labour- and therefore cost-intensive, and would profit from a maximum level of automation. For this purpose, the identification and extraction of terms that play an important role in the domain under consideration, is a vital first step.

Automatic term recognition (also known as term extraction) is a crucial component of many knowledge-based applications such as automatic indexing, knowledge discovery, terminology mining and monitoring, knowledge management and so on. It is particularly important in the healthcare and biomedical domains, where new terms are emerging constantly.

Term recognition has been performed on the basis of various criteria. The main distinction we can make is between algorithms that only take the distributional properties of terms into account, such as frequency and tf/idf [1], and extraction techniques that use the contextual information associated with terms. The work described here concentrates on the latter task, and describes algorithms that compare and measure context vectors, exploiting semantic similarity between terms and candidate terms. We then proceed to investigate a more general method for information extraction, which is used, along with term extraction, for the task of ontology population.

[1]Corresponding Author: Diana Maynard: Dept. of Computer Science, University of Sheffield, 211 Portobello St, Sheffield, UK; E-mail: diana@dcs.shef.ac.uk

Ontology population is a crucial part of knowledge base construction and maintenance that enables us to relate text to ontologies, providing on the one hand a customised ontology related to the data and domain with which we are concerned, and on the other hand a richer ontology which can be used for a variety of semantic web-related tasks such as knowledge management, information retrieval, question answering, semantic desktop applications, and so on.

Ontology population is generally performed by means of some kind of ontology-based information extraction (OBIE). This consists of identifying the key terms in the text (such as named entities and technical terms) and then relating them to concepts in the ontology. Typically, the core information extraction is carried out by linguistic pre-processing (tokenisation, POS tagging etc.), followed by a named entity recognition component, such as a gazetteer and rule-based grammar or machine learning techniques. Named entity recognition (using such approaches) and automatic term recognition are thus generally performed in a mutually exclusive way: i.e. one or other technique is used depending on the ultimate goal. However, it makes sense to use a combination of the two techniques in order to maximise the benefits of both. For example, term extraction generally makes use of frequency-based information whereas typically named entity recognition uses a more linguistic basis. Note also that a "term" refers to a specific concept characteristic of a domain, so while a named entity such as Person or Location is generic across all domains, a technical term such as "myocardial infarction" is only considered a relevant term when it occurs in a medical domain: if we were interested in sporting terms then it would probably not be considered a relevant term, even if it occurred in a sports article. As with named entities, however, terms are generally formed from noun phrases (in some contexts, verbs may also be considered terms, but we shall ignore this here).

The overall structure of the chapter covers a step by step description of the natural task extension from term extraction into more general purpose information extraction, and therefore brings together the whole methodological path from extraction, through annotation to ontology population.

2. A Similarity-based Approach to Term Recognition

The TRUCKS system [2] introduced a novel method of term recognition which identified salient parts of the context surrounding a term from a variety of sources, and measured their strength of association with relevant candidate terms. This was used in order to improve on existing methods of term recognition such as the C/NC-Value approach [3] which used largely statistical methods, plus linguistic (part-of-speech) information about the candidate term itself. The NC-Value method extended on the C-Value method by adding information about frequency of co-occurrence with context words. The SNC-Value used in TRUCKS includes contextual and terminological information and achieves improved precision (see [4] for more details).

In very small and/or specialised domains, as are typically used as a testbed for term recognition, statistical information may be skewed due to data sparsity. On the other hand, it is also difficult to extract suitable semantic information from such specialised corpora, particularly as appropriate linguistic resources may be lacking. Although contextual information has previously been used, e.g. in general language [5], and in the NC-Value method, only shallow semantic information is used in these cases. The TRUCKS

approach, however, identifies different elements of the context which are combined to form the Information Weight [2], a measure of how strongly related the context is to the candidate term. This Information Weight is then combined with statistical information about a candidate term and its context, acquired using the NC-Value method. Note that both approaches, unlike most other term recognition approaches, result in a ranked list of terms rather than making a binary decision about termhood. This introduces more flexibility into the application, as the user can decide at what level to draw the cut-off point. Typically, we found that the top 1/3 of the list produces the best results.

The idea behind using the contextual information stems from the fact that, just as a person's social life can provide valuable insight about their personality, so we can gather much information about a term by analysing the company it keeps. In general, the more similar context words are to a candidate term, the stronger the likelihood of the term being relevant. We can also use this same kind of criteria to perform term disambiguation, by choosing the meaning of the term closest to that of its context [6].

2.1. Acquiring Contextual Information

The TRUCKS system builds on the NC-Value method for term recognition, by incorporating contextual information in the form of additional weights. We acquire three different types of knowledge about the context of a candidate term: syntactic, terminological, and semantic. The NC Value method is first applied to the corpus to acquire an initial set of candidate terms.

Syntactic knowledge is based on *boundary words*, i.e. the words immediately before and after a candidate term. A similar method (the *barrier word* approach [7,8]) has been used previously to simply accept or decline the presence of a term, depending on the syntactic category of the barrier or boundary word. Our system takes this a stage further by - rather than making a binary decision - allocating a weight to each syntactic category based on a co-occurrence frequency analysis, to determine how likely the candidate term is to be valid. For example, a verb occurring immediately before a candidate term is statistically a much better indicator of a true term than an adjective is. By a "better indicator", we mean that a candidate term occurring with this context is more likely to be valid. Each candidate term is then assigned a syntactic weight, calculated by summing the category weights for all the context boundary words occurring with it.

Terminological knowledge concerns the terminological status of context words. A context word which is also a term (which we call a *context term*) is likely to be a better indicator of a term than one which is not also a term itself. This is based on the premise that terms tend to occur together. Context terms are determined by applying the NC-Value method to the whole corpus and selecting the top 30% of the resulting ranked list of terms. A context term (CT) weight is produced for each candidate term, based on its total frequency of occurrence with other context terms.

The CT weight is formally described as follows:

$$CT(a) = \sum_{d \epsilon T_a} f_a(d) \tag{1}$$

where
a is the candidate term,
T_a is the set of context terms of a,

d is a word from T_a,
$f_a(d)$ is the frequency of d as a context term of a.

Semantic knowledge is based on the idea of incorporating semantic information about terms in the context. We predict that context words which are not only terms, but also have a high degree of similarity to the candidate term in question, are more likely to be relevant. This is linked to the way in which sentences are constructed. Semantics indicates that words in the surrounding context tend to be related, so the more similar a word in the context is to a term, the more informative it should be.

Our claim is essentially that if a context word has some contribution towards the identification of a term, then there should be some significant correspondence between the meaning of that context word and the meaning of the term. This should be realised as some identifiable semantic relation between the two. Such a relation can be exploited to contribute towards the correct identification and comprehension of a candidate term. A similarity weight is added to the weights for the candidate term, which is calculated for each term / context term pair. This similarity weight is calculated using a new metric to define how similar a term and context term are, by means of their distance in a hierarchy. For the experiments carried out in [4], the UMLS semantic network was used [9].

While there exist many metrics and approaches for calculating similarity, the choice of measure may depend considerably on the type of information available and the intended use of the algorithm. A full discussion of such metrics and their suitability can be found in [4], so we shall not go into detail here. Suffice it to say that:

- Thesaurus-based methods seem a natural choice here, because to some extent they already define relations between words.
- Simple thesaurus-based methods fail to take into account the non-uniformity of hierarchical structures, as noted by [10].
- Methods such as information content [10] have the drawback that the assessment of similarity in hierarchies only involves taxonomic (is-a) links. This means that they may exclude some potentially useful information.
- General language thesauri such as WordNet and Roget's Thesaurus are only really suitable for general-language domains, and even then have been found to contain serious omissions. If an algorithm is dependent on resources such as this, it can only be as good as is dictated by the resource.

2.2. Similarity Measurement in the TRUCKS System

Our approach to similarity measurement in a hierarchy is modelled mainly on the EBMT (Example-Based Machine Translation)-based techniques of Zhao [11] and Sumita and Iida [12]. This is based on the premise that the position of the MSCA (Most Specific Common Abstraction)[2] within the hierarchy is important for similarity. The lower down in the hierarchy the MSCA, the more specific it is, and therefore the more information is shared by the two concepts, thus making them more similar. We combine this idea with that of semantic distance [13,14,15]. In its simplest form, similarity is measured by edge-counting – the shorter the distance between the words, the greater their similarity. The MSCA is commonly used to measure this. It is determined by tracing the respective paths of the two words back up the hierarchy until a common ancestor is found. The

[2] also known as Least Common Subsumer or LCS

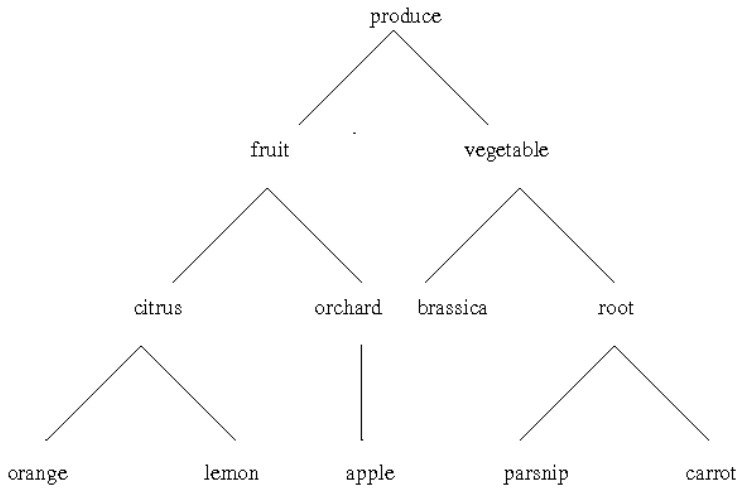

Figure 1. Fragment of a food network

average distance from node to MSCA is then measured: the shorter the distance to the MSCA, the more similar the two words. We combine these two ideas in our measure by calculating two weights: one which measures the distance from node to MSCA, and one which measures the vertical position of the MSCA. Note that this metric does of course have the potential drawback mentioned above, that only involving taxonomic links does mean the potential loss of information. However, we claim that this is quite minimal, due to the nature of the quite restricted domain-specific text that we deal with, because other kinds of links are not so relevant here. Futhermore, distance-based measures such as these are dependent on a balanced distribution of concepts in the hierarchy, so it is important to use a suitable ontology or hierarchy.

To explain the relationship between network position and similarity, we use the example of a partial network of fruit and vegetables, illustrated in Figure 1. Note that this diagram depicts only a simplistic is-a relationship between terms, and does not take into account other kinds of relationships or multidimensionality (resulting in terms occurring in more than one part of the hierarchy due to the way in which they are classified). We claim that the height of the MSCA is significant. The lower in the hierarchy the two items are, the greater their similarity. In the example, there would be higher similarity between *lemon* and *orange* than between *fruit* and *vegetable*. Although the average distance from *lemon* and *orange* to its MSCA (*citrus*) is the same as that from *fruit* and *vegetable* to its MSCA (*produce*), the former group is lower in the hierarchy than the latter group. This is also intuitive, because not only do *lemon* and *orange* have the *produce* feature in common, as *fruit* and *vegetable* do, but they also share the features *fruit* and *citrus*.

Our second claim is that the greater the horizontal distance between words in the network, the lower the similarity. By horizontal distance, we mean the distance between two nodes via the MSCA. This is related to the average distance from the MSCA, since the greater the horizontal distance, the further away the MSCA must be in order to be common to both. In the food example, *carrot* and *orange* have a greater horizontal distance than *lemon* and *orange*, because their MSCA (*produce*) is further away from them

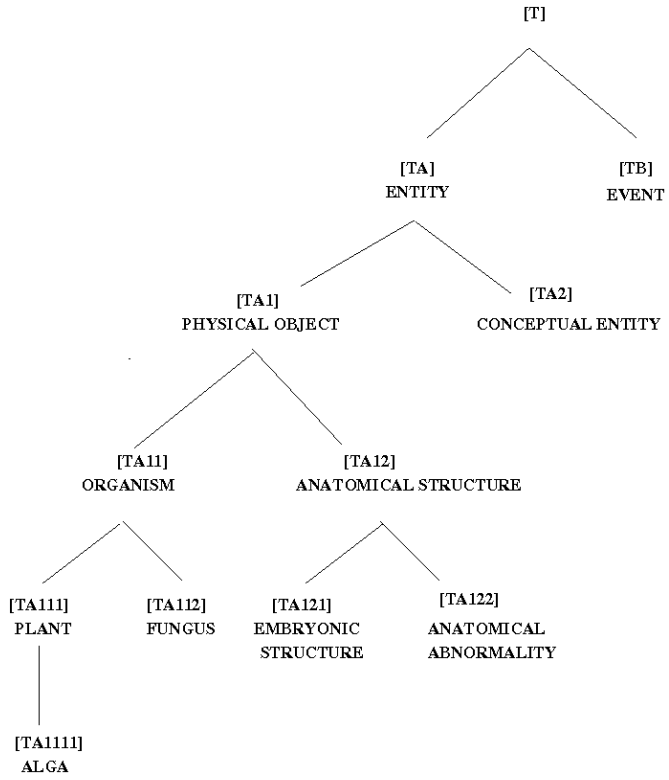

Figure 2. Fragment of the Semantic Network

than the MSCA of *lemon* and *orange* (*citrus*). Again, it is intuitive that the former are less similar than the latter, because they have less in common.

Taking these criteria into account, we define the following two weights to measure the vertical position of the MSCA and the horizontal distance between the nodes:

- *positional*: measured by the combined distance from root to each node
- *commonality*: measured by the number of shared common ancestors multiplied by the number of words (usually two).

The nodes in the Semantic Network are coded such that the number of digits in the code represents the number of leaves descended from the root to that node, as shown in Figure 2, which depicts a small section of the UMLS Semantic Network. Similarity between two nodes is calculated by dividing the commonality weight by the positional weight to produce a figure between 0 and 1, 1 being the case where the two nodes are identical, and 0 being the case where there is no common ancestor (which would only occur if there were no unique root node in the hierarchy). This can formally be defined as follows:

$$\mathrm{sim}(w_1 \ldots w_n) = \frac{\mathrm{com}(w_1 \ldots w_n)}{\mathrm{pos}(w_1 \ldots w_n)} \tag{2}$$

where

com($w_1 ... w_n$) is the commonality weight of words 1...n

pos($w_1 ... w_n$) is the positional weight of words 1...n.

It should be noted that the definition permits any number of nodes to be compared, although usually only two nodes would be compared at once. Also, it should be made clear that similarity is not being measured between terms themselves, but between the semantic types (concepts) to which the terms belong. So a similarity of 1 indicates not that two terms are synonymous, but that they both belong to the same semantic type.

3. Moving from Term to Information Extraction

There is a fairly obvious relationship between term recognition and information extraction, the main difference being that information extraction may also look for other kinds of information than just terms, and it may not necessarily be focused on a specific domain. Traditionally, methods for term recognition have been strongly statistical, while methods for information extraction have focused largely on either linguistic methods or machine learning, or a combination of the two. Linguistic methods for information extraction (IE), such as those used in GATE [16], are generally rule-based, and in fact use methods quite similar to those for term extraction used in the TRUCKS system, in that they use a combination of gazetteer lists and hand-coded pattern-matching rules which use contextual information to help determine whether such "candidate terms" are valid, or to extend the set of candidate terms. We can draw a parallel between the use of gazetteer lists containing sets of "seed words" and the use of candidate terms in TRUCKS: the gazetteer lists act as a starting point from which to establish, reject, or refine the final entity to be extracted.

3.1. Information Extraction with ANNIE

GATE, the General Architecture for Text Engineering, is a framework providing support for a variety of language engineering tasks. It includes a vanilla information extraction system, ANNIE, and a large number of plugins for various tasks and applications, such as ontology support, information retrieval, support for different languages, WordNet, machine learning algorithms, and so on. There are many publications about GATE and ANNIE – see for example [17]. This is not the focus of this paper, however, so we simply summarise here the components and method used for rule-based information extraction in GATE.

ANNIE consists of the following set of processing resources: tokeniser, sentence splitter, POS tagger, gazetteer, finite state transduction grammar and orthomatcher. The resources communicate via GATE's annotation API, which is a directed graph of arcs bearing arbitrary feature/value data, and nodes rooting this data into document content (in this case text).

The **tokeniser** splits text into simple tokens, such as numbers, punctuation, symbols, and words of different types (e.g. with an initial capital, all upper case, etc.), adding a "Token" annotation to each. It does not need to be modified for different applications or text types.

The **sentence splitter** is a cascade of finite-state transducers which segments the text into sentences. This module is required for the tagger. Both the splitter and tagger are generally domain and application-independent.

The **tagger** is a modified version of the Brill tagger, which adds a part-of-speech tag as a feature to each Token annotation. Neither the splitter nor the tagger is a mandatory part of the NE system, but the annotations they produce can be used by the semantic tagger (described below), in order to increase its power and coverage.

The **gazetteer** consists of lists such as cities, organisations, days of the week, etc. It contains some entities, but also names of useful key words, such as company designators (e.g. "Ltd."), titles (e.g. "Dr."), etc. The lists are compiled into finite state machines, which can match text tokens.

The **semantic tagger** (or JAPE transducer) consists of hand-crafted rules written in the JAPE pattern language [18], which describe patterns to be matched and annotations to be created. Patterns can be specified by describing a specific text string or annotation (e.g. those created by the tokeniser, gazetteer, document format analysis, etc.).

The **orthomatcher** performs coreference, or entity tracking, by recognising relations between entities. It also has a secondary role in improving NE recognition by assigning annotations to previously unclassified names, based on relations with existing entities.

ANNIE has been adapted to many different uses and applications: see [19,20,21] for some examples. In terms of adapting to new tasks, the processing resources in ANNIE fall into two main categories: those that are domain-independent, and those that are not. For example, in most cases, the tokeniser, sentence splitter, POS tagger and orthographic coreference modules fall into the former category, while resources such as gazetteers and JAPE grammars will need to be modified according to the application. Similarly, some resources, such as the tokeniser and sentence splitter, are largely language-independent (exceptions may include some Asian languages, for example), and some resources are more language-dependent, such as gazetteers.

3.2. Using contextual information to bootstrap rule creation

One of the main problems with using a rule-based approach to information extraction is that rules can be slow and time-consuming to develop, and an experienced language engineer is generally needed to create them. This language engineer typically needs also to have a detailed knowledge of the language and domain in question. Secondly, it is easy with a good gazetteer list and a simple set of rules to achieve reasonably accurate results in most cases in a very short time, especially where recall is concerned. For example, our work on surprise languages [20] achieved a reasonable level of accuracy on the Cebuano language with a week's effort and with no native speaker and no resources provided. Similarly, [22] achieved high scores for recognition of locations using only gazetteer lists. However, achieving very high precision requires a great deal more effort, especially for languages which are more ambiguous than English.

It is here that making use of contextual information is key to success. Gazetteer lists can go a long way towards initial recognition of common terms; a set of rules can boost this process by e.g. combining elements of gazetteer lists together, using POS information combined with elements of gazetteer lists (e.g. to match first names from a list with probable surnames indicated by a proper noun), and so on. In order to resolve

ambiguities and to find more complex entity types, context is necessary. Here we build on the work described in Section 2, which made use of information about contextual terms to help decide whether a candidate term (extracted initially through syntactic tagging) should be validated.

There are two tools provided in GATE which enable us to make use of contextual information: the gazetteer lists collector and ANNIC. These are described in the following two sections.

3.3. Gazetteer lists collector

The GATE gazetteer lists collector [23] helps the developer to build new gazetteer lists from an initial set of annotated texts with minimal effort. If the list collector is combined with a semantic tagger, it can be used to generate context words automatically. Suppose we generate a list of Persons occurring in our training corpus. Some of these Persons will be ambiguous, either with other entity types or even with non-entities, especially in languages such as Chinese. One way to improve Precision without sacrificing Recall is to use the lists collector to identify from the training corpus a list of e.g. verbs which typically precede or follow Persons. The list can also be generated in such a way that only verbs with a frequency above a certain threshold will be collected, e.g. verbs which occur less than 3 times with a Person could be discarded.

The lists collector can also be used to improve recognition of entities by enabling us to add constraints about contextual information that precedes or follows candidate entities. This enables us to recognise new entities in the texts, and forms part of a development cycle, in that we can then add such entries to the gazetteer lists, and so on. In this way, noisy training data can be rapidly created from a small seed corpus, without requiring a large amount of annotated data initially.

Furthermore, using simple grammar rules, we can collect not only examples of entities from the training corpus, but also information such as the syntactic categories of the preceding and following context words. Analysis of such categories can help us to write better patterns for recognising entities. For example, using the lists collector we might find that definite and indefinite articles are very unlikely to precede Person entities, so we can use this information to write a rule stipulating that if an article is found preceding a candidate Person, that candidate is unlikely to be a valid Person. We can also use lexical information, by collecting examples of verbs which typically follow a Person entity. If such a verb is found following a candidate Person, this increases the likelihood that such a candidate is valid, and we can assign a higher priority to such a candidate than one which does not have such context.

3.4. ANNIC

The second tool, ANNIC (ANNotations In Context) [24], enables advanced search and visualisation of linguistic information. This provides an alternative method of searching the textual data in the corpus, by identifying patterns in the corpus that are defined both in terms of the textual information (i.e. the actual content) and of metadata (i.e. linguistic annotation and XML/TEI markup). Essentially, ANNIC is similar to a KWIC (KeyWords In Context) index, but where a KWIC index provides simply text in context in response to a search for specific words, ANNIC additionally provides linguistic information (or other annotations) in context, in response to a search for particular linguistic patterns.

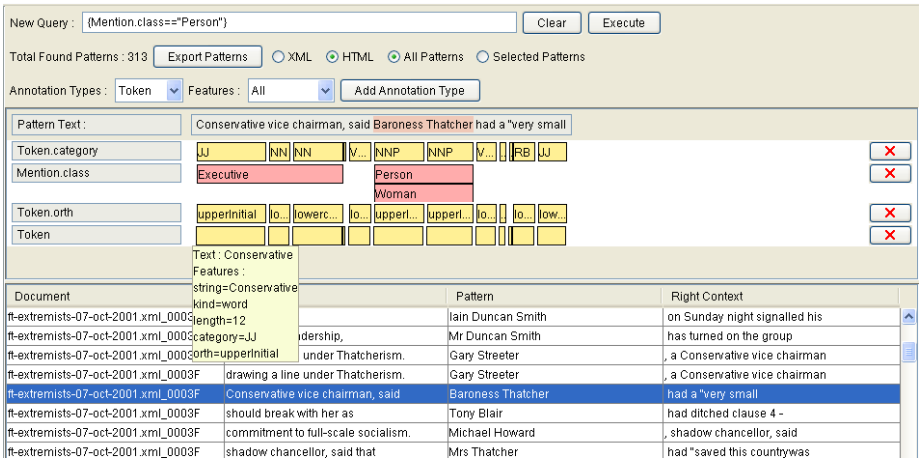

Figure 3. ANNIC Viewer

ANNIC can be used as a tool to help users with the development of JAPE rules by enabling them to search the text for examples using an annotation or combination of annotations as the keyword. Language engineers have to use their intuition when writing JAPE rules, trying to strike the ideal balance between specificity and coverage. This requires them to make a series of informed guesses which are then validated by testing the resulting ruleset over a corpus. ANNIC can replace the guesswork in this process with a live analysis of the corpus. Each pattern intended as part of a JAPE rule can easily be tested directly on the corpus and have its specificity and coverage assessed almost instantaneously.

Figure 3 shows a screenshot of ANNIC in use. The bottom section in the window contains the patterns along with their left and right context concordances, while the top section shows a graphical visualisation of the annotations. ANNIC shows each pattern in a separate row and provides a tool tip that shows the query that the selected pattern refers to. Along with its left and right context, it also lists the name of documents that the patterns come from. The tool is interactive, and different aspects of the search results can be viewed by clicking on appropriate parts of the GUI.

ANNIC can also be used as a more general tool for corpus analysis, because it enables querying the information contained in a corpus in more flexible ways than simple full-text search. Consider a corpus containing news stories that have been processed with a standard NE system such as ANNIE. A query like

{Organization} ({Token})*3 ({Token.string==’up’}|{Token.string==’down’}) ({Money} | {Percent})

would return mentions of share movements like “BT shares ended up 36p” or “Marconi was down 15%”. Locating this type of useful text snippets would be very difficult and time consuming if the only tool available were text search. Clearly it is not just information extraction and rule writing that benefits from the visualisation of contextual information in this way. When combined with the TRUCKS term extraction technique, we can use it to visualise the combinations of term and context term, and also to investigate other possible sources of interesting context which might provide insight into further refinement of the weights. We can also very usefully combine ANNIC with the

gazetteer list collector described in Section 3.3 in order to again visualise other sources of contextual information worth collecting.

4. From Traditional to Ontology-Based Information Extraction

Ontology-Based IE (OBIE) is one of the technologies used for semantic annotation, which is essentially about assigning to entities in the text links to their semantic descriptions. This sort of metadata provides both class and instance information about the entities. One of the important differences between traditional IE and OBIE is the use of a formal ontology rather than a flat lexicon or gazetteer structure. This may also involve reasoning.

4.1. OBIE Systems

There are a number of what we describe as ontology-oriented IE systems, which, unlike ontology-based ones, do not incorporate ontologies into the system, but either use them as a bridge between the IE output and the final annotation (as with AeroDAML) or rely on the user to provide the relevant information through manual annotation (as with the Amilcare-based tools).

AeroDAML [25] applies IE techniques to automatically generate DAML annotations from web pages. It links proper nouns and common types of relations with classes and properties in a DAML ontology. It makes use of an ontology in order to translate the extraction results into a corresponding RDF model.

Amilcare [26] is an IE system which has been integrated in several different semantic annotation tools, such as OntoMat [27], which combines a manual annotation tool with an IE system running in the background. It uses supervised rule learning to adapt to new domains and applications given human annotated texts (training data). It treats the semantic annotations as a flat set of labels, thus ignoring the further knowledge in the ontology. Amilcare uses GATE's NLP components in order to obtain linguistic information as features for the learning process.

One of the problems with these annotation tools is that they do not provide the user with a way to customise the integrated language technology directly. While many users would not need or want such customisation facilities, users who already have ontologies with rich instance data will benefit if they can make this data available to the IE components. However, this is not possible when traditional IE methods like Amilcare are used, because they are not aware of the existence of the user's ontology.

The more serious problem however, as discussed in the S-CREAM system [27], is that there is often a gap between the IE output annotations and the classes and properties in the user's ontology. The solution proposed by the developers was to write logical rules to resolve this. For example, an IE system would typically annotate `London` and `UK` as locations, but extra rules are needed to specify that there is a containment relationship between the two. However, rule writing of this kind is too difficult for most users and therefore ontology-based IE is needed, as it annotates directly with the classes and instances from the user's ontology.

In response to these problems, a number of OBIE systems have been developed. Magpie [28] is a suite of tools which supports semantic annotation of web pages. It is

fully automatic and works by matching the text against instances in the ontology. The SemTag system [29] is similar in approach to Magpie as it annotates texts by performing lookup against the TAP ontology. It also has a second, disambiguation phase, where SemTag uses a vector-space model to assign the correct ontological class or determine that this mention does not correspond to a class in TAP. The problem with both systems is that they are not able to discover new instances and are thus restricted in terms of recall.

The PANKOW system [30] exploits surface patterns and the redundancy on the Web to categorise automatically named entities found in text with respect to a given ontology. Its aim is thus primarily ontology population rather than annotation. PANKOW has recently been integrated with MAGPIE [31].

OntoSyphon [32] is similar to PANKOW and uses the ontology as the starting point in order to carry out web mining to populate the ontology with instances. It uses the ontology structure to determine the relevance of the candidate instances. However, it does not carry out semantic annotation of documents as such.

The KIM system [33] produces annotations linked both to the ontological class and to the exact individual in the instance base. For new (previously unknown) entities, new identifiers are allocated and assigned; then minimal descriptions are added to the semantic repository. KIM has a rule-based, human-engineered IE system based on GATE's ANNIE, which uses the ontology structure during pattern matching and instance disambiguation. The only shortcoming of this approach is that it requires human intervention in order to adapt it to new ontologies.

To summarise, all these systems use the ontology as their target output, and the ontology-based ones also use class and instance information during the IE process. While KIM and OntoSyphon do make use of the ontology structure, the former is a rule-based, not a learning approach, whereas the latter does not perform semantic annotation, only ontology population.

5. Evaluation of Ontology-Based Information Extraction

Traditionally, information extraction is evaluated using Precision, Recall and F-Measure. However, when dealing with ontologies, such methods are not really sufficient because they give us a binary decision of correctness, i.e. they classify the result as either right or wrong. This is fine for traditional IE, because an element identified as a Person is either correct or incorrect (measured by Precision), and elements which should be identified as Person are either identified or not (measured by Recall). When making an ontological classification, however, the distinction is a bit more fuzzy. For example if we misclassify an instance of a Researcher as a Lecturer, we are clearly less wrong than missing the identification (and classification) altogether, and we are also somehow less wrong than if we had misclassified the instance as a Location. Credit should therefore be given for partial correctness. Traditionally, this is sometimes achieved by allocating a half weight to something deemed partially correct, but this is still insufficient to give a proper distinction between degrees of correctness. We therefore adopt an approach based on similarity between Key (the gold standard) and Response (the output of the system).

5.1. A Distance-based Metric for Evaluation

We developed the Balanced Distance Metric (BDM) [34] in order to address this problem. This metric has been designed to replace the traditional "exact match or fail" metrics with a method which yields a graded correctness score by taking into account the semantic distance in the ontological hierarchy between the compared nodes (Key and Response).

The semantic distance is adapted from the semantic weight used in the TRUCKS system, but takes into account also some normalisation – something which was not considered in the original TRUCKS weight. In the BDM, each of the paths has been normalised with two additional measurements, of which the first is the average length of the chains in which key and response concepts occur. The longer a particular ontological chain is, the more difficult it is to consistently pick out a particular class for annotation [35]. The second normalization is the introduction of the branching factor (i.e. number of descendants) of the relevant nodes in the ontology. This is also an indication of the level of difficulty associated with the selection of a particular ontlogical class relative to the size of the set of candidates. These normalizations will make the penalty that is computed in terms of node traversal within our metric relative to the semantic density of the chains.

Another similar metric which has been proposed for this task is Learning Accuracy (LA) [36], which was originally developed to measure how well an item had been classified in an ontology. Learning Accuracy has a major flaw for our purposes, however, in that it does not take into account the depth of the key concept in the hierarchy, considering essentially only the height of the MSCA (Most Specific Common Abstraction) and the distance from the response to the MSCA. This means that however far away the key is from the MSCA, the metric will give the same outcome. The BDM is more balanced in this respect, because it takes the relative specificity of the taxonomic positions of the key and response into account in the score, but it does not distinguish between the specificity of the key concept on the one hand, and the specificity of the response concept on the other. For instance, the key can be a specific concept (e.g. 'car'), whereas the response can be a more general concept (e.g. 'relation'), or vice versa, and the result will be the same. This is not the case with the Learning Accuracy metric.

The BDM is computed on the basis of the following measurements:

- CP = the shortest length from root to the most specific common parent, i.e. the most specific ontological node subsuming both Key and Response)
- DPK = shortest length from the most specific common parent to the Key concept
- DPR = shortest length from the most specific common parent to the Response concept
- n1: average chain length of all ontological chains containing Key and Response.
- n2: average chain length of all ontological chains containing Key.
- n3: average chain length of all ontological chains containing Response.
- BR: the branching factor of each relevant concept, divided by the average branching factor of all the nodes from the ontology, excluding leaf nodes.

The complete BDM formula is as follows:

$$BDM = \frac{BR(CP/n1)}{BR(CP/n1) + (DPK/n2) + (DPR/n3)} \tag{3}$$

As with the similarity weight described in Section 2.2, the measure provides a score somewhere between 0 and 1 for the comparison of key and response concepts with respect to a given ontology. If a concept is missing or spurious, BDM is not calculated since there is no MSCA. If the key and response concepts are identical, the score is 1 (as with Precision and Recall). Overall, in case of an ontological mismatch, this method provides an indication of how serious the error is, and weights it accordingly.

The BDM itself is not sufficient to evaluate our populated ontology, because we need to preserve the useful properties of the standard Precision and Recall scoring metric. Our APR metric (Augmented Precision and Recall) combines the traditional Precision and Recall with a cost-based component (namely the BDM). We thus combine the BDM scores for each instance in the corpus, to produce Augmented Precision, Recall and F-measure scores for the annotated corpus, calculated as follows:

$$AP = \frac{BDM}{n + Spurious} \qquad and \qquad AR = \frac{BDM}{n + Missing} \tag{4}$$

while F-measure is calculated from Augmented Precision and Recall as:

$$F - measure = \frac{AP * AR}{0.5 * (AP + AR)} \tag{5}$$

5.2. Experiments with OBIE evaluation

The BDM metric has been evaluated in various ways in order to compare it with other metrics for evaluation and to test scalability issues. For the evaluation, a semantically annotated corpus was created for use as a gold standard. This is known as the OntoNews corpus [37]. This semantically annotated corpus consists of 292 news articles from three news agencies (The Guardian, The Independent and The Financial Times), and covers the period of August to October, 2001. The articles belong to three general topics or domains of news gathering: International politics, UK politics and Business.

The ontology used in the generation of the ontological annotation process is the PROTON ontology[3], which has been created and used in the scope of the KIM platform[4] for semantic annotation, indexing, and retrieval [33]. The ontology consists of around 250 classes and 100 properties (such as partOf, locatedIn, hasMember and so on). PROTON has a number of important properties: it is domain-independent, and therefore suitable for the news domain, and it is modular (comprising both a top ontology and a more specific ontology).

The aim of the experiments carried out on the OntoNews corpus was, on the one hand, to evaluate a new learning algorithm for OBIE, and, on the other hand, to compare the different evaluation metrics (LA, flat traditional measure, and the BDM).

The OBIE algorithm learns a Perceptron classifier for each concept in the ontology. Perceptron [38] is a simple yet effective machine learning algorithm, which forms the basis of most on-line learning algorithms. Meanwhile, the algorithm tries to keep the difference between two classifiers proportional to the cost of their corresponding concepts in the ontology. In other words, the learning algorithm tries to classify an instance as correctly as it can. If it cannot classify the instance correctly, it then tries to classify it with

[3] http://proton.semanticweb.org

[4] http://www.ontotext.com/kim

another concept with the least cost associated with it relative to the correct concept. The algorithm is based on the Hieron, a large margin algorithm for hierarchical classification proposed in [39]. See [40] for details about the learning algorithm and experiments.

We experimentally compared the Hieron algorithm with the SVM learning algorithm (see e.g. [41]) for OBIE. The SVM is a state of the art algorithm for classification. [42] applied SVM with uneven margins, a variant of SVM, to the traditional information extraction problem and achieved state of the art results on several benchmarking corpora. In the application of SVM to OBIE, we learned one SVM classifier for each concept in the ontology separately and did not take into account the structure of the ontology. In other words, the SVM-based IE learning algorithm was a flat classification in which the structure of concepts in the ontology was ignored. In contrast, the Hieron algorithm for IE is based on hierarchical classification that exploits the structure of concepts.

As the OntoNews corpus consists of three parts (International politics, UK politics and Business), for each learning algorithm two parts were used as training data and another part as test data. Note that although the tripartition of the corpus indicates three distinct and topically homogeneous parts of the corpus, these parts are used as training and testing data for the comparison of different algorithms, and not their performance. For this purpose, semantic homogeneity does not play a role.

For each experiment we computed three F_1 values to measure the overall performance of the learning algorithm. One was the conventional micro-averaged F_1 in which a binary reward was assigned to each prediction of instance — the reward was 1 if the prediction was correct, and 0 otherwise. We call this flat_F_1 since it does not consider the structure of concepts in the ontology. The other two measures were based on the BDM and LA values, respectively, which both take into account the structure of the ontology.

	flat_F_1	BDM_F_1	LA_F_1
SVM	73.5	74.5	74.5
Hieron	74.7	79.2	80.0

Table 1. Comparison of Hieron and SVM for OBIE

Table 1 presents the experimental results for comparing the two learning algorithms SVM and Hieron. We used three measures: conventional micro-averaged flat_F_1 (%), and the two ontology-sensitive augmented F_1 (%) based respectively on the BDM and LA, BDM_F_1 and LA_F_1. In this experiment, the International-Politics part of the OntoNews corpus was used as the test set, and the other two parts as the training set.

Both the BDM_F_1 and LA_F_1 are higher than the flat_F_1 for the two algorithms, reflecting the fact that the latter only counts the correct classifications, while the former two not only count the correct classifications but also the incorrect ones. However, the difference for Hieron is more significant than that for SVM, demonstrating an important difference between the two methods — the SVM based method just tried to learn a classifier for one concept as well as possible, while the Hieron based method not only learned a good classifier for each individual concept but also took into account the relations between the concepts in the ontology during the learning.

In terms of the conventional flat_F_1, the Hieron algorithm performed slightly better than the SVM. However, if the results are measured by using the ontology-sensitive measure BDM_F_1 or LA_F_1, we can see that Hieron performed significantly better than

SVM. Clearly, the ontology-sensitive measures such as the BDM_F_1 and LA_F_1 are more suitable than the conventional flat_F_1 to measure the performance of an ontology-dependent learning algorithm such as Hieron.

In order to analyse the difference between the three measures, Table 2 presents some examples of entities predicted incorrectly by the Hieron based learning system, their key labels, and the similarity between the key label and predicted label measured respectively by the BDM and the LA. Note that in all cases, the flat measure produces a score of 0, since it is not an exact match.

No.	Entity	Predicted label	Key label	BDM	LA
1	Sochi	Location	City	0.724	1.000
2	Federal Bureau of Investigation	Organization	GovernmentOrganization	0.959	1.000
3	al-Jazeera	Organization	TVCompany	0.783	1.000
4	Islamic Jihad	Company	ReligiousOrganization	0.816	0.556
5	Brazil	Object	Country	0.587	1.000
6	Senate	Company	PoliticalEntity	0.826	0.556
7	Kelly Ripa	Man	Person	0.690	0.667

Table 2. Examples of entities misclassified by the Hieron based system

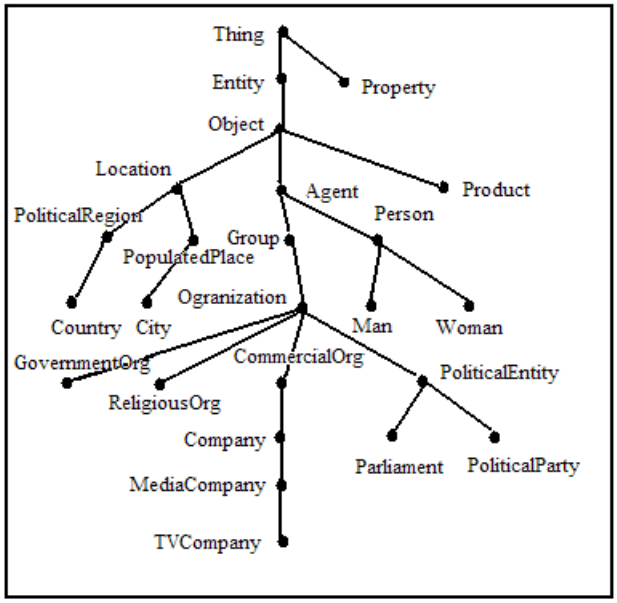

Figure 4. Subset of the PROTON ontology

All the concepts and their relations involved in Table 2 are illustrated in Figure 4, which presents a part of the PROTON ontology. This ontology section starts with the root node *Thing*, and has 10 levels of concepts with *TVCompany* as the lowest level concept. Note that the graph does not show all the child concepts for most of the nodes presented.

The conventional flat measure assigned each case a zero similarity because the examples were misclassified and the measure does not consider the structure of labels. On the other hand, both the LA and BDM take into account the structure of labels and measure the degree of a misclassification based on its position in the ontology. Hence they assign a non-zero value to a misclassification in most cases. Note that zero would be assigned in the case where the MSCA is the root node. In our experiments, all the concepts used were below the node "Entity" and so we used its immediate upper node "Thing" as root[5]. This meant that CP (the depth of the MSCA) was always at least 1, and hence there is no zero value for BDM or LA in our experiments. This is because we consider that if an entity's instance is recognised but with the wrong type, the system should have a non-zero reward because it at least recognised the instance in the first place. However, this could be changed according to the user's preference.

However, BDM and LA adopt different mechanisms in consideration of the ontology structure. In particular, the LA assigns the maximal value 1 if the predicted label is an ancestor concept of the key label, regardless of how far apart the two labels are within the ontological chain. In contrast, the BDM takes into account the similarity of two concepts in the ontology and assigns a distance-dependent value. The difference is demonstrated by the examples in the table. For example, in the Proton ontology, the predicted label *Organization* is the parent concept of the key label *GovernmentOrganization* in the second example, and in the third example the same predicted label *Organization* is 4 concepts away from the key label *TVCompany*. Hence, the BDM value of the second example is higher than the BDM value of the third example. In the first example, the predicted label *Location* is 3 concepts away from the key label *City* but its BDM value is lower than the corresponding value in the third example, mainly because the concept *Location* occupies a higher position in the Proton ontology than the concept *Organization*. Similarity is thus lower because higher concepts are semantically more general, and therefore less informative.

Another difference between the BDM and LA is that the BDM considers the concept densities around the key concept and the response concept, but the LA does not. The difference can be shown by comparing the fourth and the sixth examples. They have the same predicted label *Company*, and their key labels *ReligiousOrganization* and *PoliticalEntity* are two sub-concepts of *Organization*. Therefore, the positions of the predicted and key labels in the two examples are very similar and hence their LA values are the same. However, their BDM values are different — the BDM value of the fourth example is a bit lower than the BDM value of the sixth example. This is because the concept *PoliticalEntity* in the sixth example has two child nodes but the concept *ReligiousOrganization* in the fourth example has no child node, resulting in different averaged lengths of chains coming through the two concepts.

The BDM value in the fifth example is the lowest among the examples, mainly because the concept *Object* is in the highest position in the ontology among the examples. These differences in BDM scores show the effects of the adoption of chain density and branching factor as penalty weights in the computation of the score. These reflect the level of difficulty associated with the selection of a particular ontlogical class relative to the size of the set of candidates.

[5] "Thing" subsumes both "Entity" and "Property"

5.3. Discussion and Future work

The initial observation from our experiments is that binary decisions are not good enough for ontology evaluation, when hierarchies are involved. We propose an Augmented Precision and Recall measure that takes into account the ontological distance of the response to the position of the key concepts in the hierarchy. For this purpose we have developed an extended variant of Hahn's Learning Accuracy measure, called Balanced Distance Metric, and integrated this with a standard Precision and Recall metric. We have performed evaluations of these three metrics based on a gold standard corpus of news texts annotated according to the PROTON ontology, and conclude that both the BDM and LA metrics are more useful when evaluating information extraction based on a hierarchical rather than a flat structure. Furthermore, the BDM appears to perform better than the LA in that it reflects a better error analysis in certain situations.

Although the BDM gives an intuitively plausible score for semantic similarity on many occasions, it can be argued that in some cases it does not correlate well with human judgement. Examples 4 and 6 in Table 2 show counter-intuitively high similarity values for combinations of key and wrongly predicted labels, particularly in comparison with example 7. Note that as mentioned earlier, they are still better than the LA in that they distinguish different values for the two examples. From a human perspective, they also seem more wrong than the erroneous classification in Example 7, and slightly more wrong than those in examples 1 and 3. This indicates a need for further tuning the BDM score with additional cost-based metrics, in order to meet human judgement criteria. In such cases, this could entail the integration of a rule which boosts similarity scores for concepts within the same ontological chain (in a more subtle way than LA), and which lowers the score for concept pairs that occur in different chains. Work will continue on further experiments with the integration of such rules, including assessment of the correlation between BDM scores and human intuition.

6. Conclusion

In this chapter we have investigated NLP techniques for term extraction and ontology population, using a combination of rule-based approaches and machine learning. Starting from an existing method we developed for term recognition using contextual information to bootstrap learning, we have shown how such techniques can be adapted to the wider task of information extraction. Term recognition and information extraction, while quite similar tasks in many ways, are generally performed using very different techniques. While term recognition generally uses primarily statistical techniques, usually combined with basic linguistic information in the form of part-of-speech tags, information extraction is usually performed with either a rule-based approach or machine learning, or a combination of the two. However, the contextual information used in the TRUCKS system for term recognition can play an important role in the development of a rule-based system for ontology-based information extraction, as shown by the development of the GATE tools described in this chapter. Furthermore, the similarity metric used in TRUCKS to determine a semantic weight for terms forms the basis for a new evaluation metric for information extraction (BDM), which uses similarity between the key and response instances in an ontology to determine the correctness of the extraction.

Experiments with this metric have shown very promising results and clearly demonstrate a better evaluation technique than the Precision and Recall metrics used for traditional (non-ontology-based) information extraction applications.

Acknowledgements

This work has been partially supported by the EU projects KnowledgeWeb (IST-2004-507482), SEKT (IST-2004-506826) and NeOn (IST-2004-27595).

References

[1] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.

[2] D.G. Maynard and S. Ananiadou. Identifying terms by their family and friends. In *Proc. of 18th International Conference on Computational Linguistics (COLING)*, Saarbrücken, Germany, 2000.

[3] K.T. Frantzi and S. Ananiadou. The C-Value/NC-Value domain independent method for multi-word term extraction. *Journal of Natural Language Processing*, 6(3):145–179, 1999.

[4] D. G. Maynard. *Term Recognition Using Combined Knowledge Sources*. PhD thesis, Mamchester Metropolitan University, UK, 2000.

[5] G. Grefenstette. *Explorations in Automatic Thesaurus Discovery*. Kluwer Academic Publishers, 1994.

[6] D.G. Maynard and S. Ananiadou. Term sense disambiguation using a domain-specific thesaurus. In *Proc. of 1st International Conference on Language Resources and Evaluation (LREC)*, pages 681–687, Granada, Spain, 1998.

[7] D. Bourigault. Surface grammatical analysis for the extraction of terminological noun phrases. In *Proc. of 14th International Conference on Computational Linguistics (COLING)*, pages 977–981, Nantes, France, 1992.

[8] S.J. Nelson, N.E. Olson, L. Fuller, M.S. Tuttle, W.G. Cole, and D.D. Sherertz. Identifying concepts in medical knowledge. In *Proc. of 8th World Congress on Medical Informatics (MEDINFO)*, pages 33–36, 1995.

[9] NLM. Unified Medical Language System (UMLS). Technical report, National Library of Medicine, http://www.nlm.nih.gov/research/umls/umlsmain.html.

[10] P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proc. of 14th International Joint Conference on Artificial Intelligence*, pages 448–453, Montreal, Canada, 1995.

[11] Gang Zhao. *Analogical Translator: Experience-Guided Transfer in Machine Translation*. PhD thesis, Dept. of Language Engineering, UMIST, Manchester, England, 1996.

[12] E. Sumita and H. Iida. Experiments and prospects of example-based machine translation. In *Proc. of 29th Annual Meeting of the Association for Computational Linguistics*, pages 185–192, Berkeley, California, 1991.

[13] G. Rigau, J. Atserias, and E. Agirre. Combining unsupervised lexical knowledge methods for word sense disambiguation. In *Proc. of ACL/EACL*, pages 48–55, Madrid, Spain, 1997.

[14] A. Smeaton and I. Quigley. Experiments on using semantic distances between words in image caption retrieval. In *Proc. of 19th International Conference on Research and Development in Information Retrieval*, Zurich, Switzerland, 1996.

[15] J-Y. Magadur and G. Tabuteau. Semantic disambiguation in an information retrieval system. In *NLP+IA 96*, pages 148–154, Moncton, Canada, 1996.

[16] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*, 2002.

[17] D. Maynard, V. Tablan, H. Cunningham, C. Ursu, H. Saggion, K. Bontcheva, and Y. Wilks. Architectural Elements of Language Engineering Robustness. *Journal of Natural Language Engineering – Special Issue on Robust Methods in Analysis of Natural Language Data*, 8(2/3):257–274, 2002.

[18] H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, and C. Ursu. *The GATE User Guide*. `http://gate.ac.uk/`, 2002.

[19] D. Maynard and H. Cunningham. Multilingual Adaptations of a Reusable Information Extraction Tool. In *Proceedings of the Demo Sessions of EACL'03*, Budapest, Hungary, 2003.

[20] D. Maynard, V. Tablan, K. Bontcheva, and H. Cunningham. Rapid customisation of an Information Extraction system for surprise languages. *Special issue of ACM Transactions on Asian Language Information Processing: Rapid Development of Language Capabilities: The Surprise Languages*, 2003.

[21] D. Maynard. Multi-source and multilingual information extraction. *Expert Update*, 2003.

[22] A. Mikheev, M. Moens, and C. Grover. Named Entity Recognition without Gazetteers. In *Proceedings of EACL*. Bergen, Norway, 1999.

[23] D. Maynard, K. Bontcheva, and H. Cunningham. Automatic Language-Independent Induction of Gazetteer Lists. In *Proceedings of 4th Language Resources and Evaluation Conference (LREC'04)*, 2004.

[24] N. Aswani, V. Tablan, K. Bontcheva, and H. Cunningham. Indexing and Querying Linguistic Metadata and Document Content. In *Proceedings of Fifth International Conference on Recent Advances in Natural Language Processing (RANLP2005)*, Borovets, Bulgaria, 2005.

[25] P. Kogut and W. Holmes. AeroDAML: Applying Information Extraction to Generate DAML Annotations from Web Pages. In *First International Conference on Knowledge Capture (K-CAP 2001), Workshop on Knowledge Markup and Semantic Annotation*, Victoria, B.C., 2001.

[26] F. Ciravegna and Y. Wilks. Designing Adaptive Information Extraction for the Semantic Web in Amilcare. In S. Handschuh and S. Staab, editors, *Annotation for the Semantic Web*. IOS Press, Amsterdam, 2003.

[27] S. Handschuh, S. Staab, and F. Ciravegna. S-CREAM — Semi-automatic CREAtion of Metadata. In *13th International Conference on Knowledge Engineering and Knowledge Management (EKAW02)*, pages 358–372, Siguenza, Spain, 2002.

[28] J. Domingue, M. Dzbor, and E. Motta. Magpie: Supporting Browsing and Navigation on the Semantic Web. In N. Nunes and C. Rich, editors, *Proceedings ACM Conference on Intelligent User Interfaces (IUI)*, pages 191–197, 2004.

[29] S. Dill, J. A. Tomlin, J. Y. Zien, N. Eiron, D. Gibson, D. Gruhl, R. Guha, A. Jhingran, T. Kanungo, S. Rajagopalan, and A. Tomkins. SemTag and Seeker: Bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the* 12^{th} *International Conference on World Wide Web (WWW2003)*, pages 178–186, Budapest, Hungary, May 2003.

[30] P. Cimiano, S. Handschuh, and S. Staab. Towards the Self-Annotating Web. In *Proceedings of WWW'04*, 2004.

[31] M. Dzbor, E. Motta, and J. Domingue. Opening up magpie via semantic services. In *Proceedings of ISWC 2004*, Hiroshima, Japan, 2004.

[32] L. K. McDowell and M. Cafarella. Ontology-Driven Information Extraction with OntoSyphon. In *5th Internal Semantic Web Conference (ISWC'06)*. Springer, 2006.

[33] A. Kiryakov, B. Popov, D. Ognyanoff, D. Manov, A. Kirilov, and M. Goranov. Semantic annotation, indexing and retrieval. *Journal of Web Semantics, ISWC 2003 Special Issue*, 1(2):671–680, 2004.

[34] D. Maynard. Benchmarking ontology-based annotation tools for the semantic web. In *UK e-Science Programme All Hands Meeting (AHM2005) Workshop "Text Mining, e-Research and Grid-enabled Language Technology"*, Nottingham, UK, 2005.

[35] E. Agirre and G. Rigau. Word sense disambiguation using conceptual density. In *Proc. of 16th International Conference on Computational Linguistics*, volume 1, pages 16–23, Copenhagen, Denmark, 1996.

[36] U. Hahn and K. Schnattinger. Towards text knowledge engineering. In *Proc. of 15th National Conference on Artificial Intelligence (AAAI-98)*, pages 524–531, Menlo Park, CA, 1998. MIT Press.

[37] W. Peters, N. Aswani, K. Bontcheva, and H. Cunningham. Quantitative Evaluation Tools and Corpora v1. Technical report, SEKT project deliverable D2.5.1, 2005.

[38] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

[39] O. Dekel, J. Keshet, and Y. Singer. Large Margin Hierarchical Classification. In *Proceedings of the 21st International Conference on Machine Learning (ICML-2004)*, Canada, 2004.

[40] Y. Li, K. Bontcheva, and H. Cunningham. Perceptron-like learning for ontology based information extraction. Technical report, University of Sheffield, Sheffield, UK, 2006.

[41] N. Cristianini and J. Shawe-Taylor. *An introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.

[42] Y. Li, K. Bontcheva, and H. Cunningham. SVM Based Learning System For Information Extraction. In M. Niranjan J. Winkler and N. Lawerence, editors, *Deterministic and Statistical Methods in Machine Learning*, LNAI 3635, pages 319–339. Springer Verlag, 2005.

Weakly Supervised Approaches for Ontology Population

Hristo TANEV [a,1] and Bernardo MAGNINI [b]
[a] *IPSC-JRC, Ispra, Italy*
[b] *ITC-irst, Trento, Italy*

Abstract. We present a weakly supervised approach to automatic ontology population from text and compare it with two other unsupervised approaches. In our experiments we populate a part of our ontology of Named Entities. We considered two high level categories - *geographical locations* and *person names* and ten sub-classes for each category. For each sub-class we automatically learn a *syntactic model* from a list of training examples and a parsed corpus. A novel syntactic indexing method allowed us to use large quantities of syntactically annotated data. The syntactic model for each named entity sub-class is a set of weighted *syntactic features*, i.e. words which typically co-occur with the members of the class in the corpus. The method is weakly supervised, since no manually annotated corpus is used in the learning process. The syntactic models are used to classify the unknown Named Entities in the test set. The method achieved promising results, i.e. 65% accuracy, and outperforms significantly the other two approaches.

Keywords. Ontology Population, Syntactic Network, Named Entity Classification, Ontology, Machine Learning, Semantic Web, Knowledge Acquisition

1. Introduction

Automatic ontology population (OP) from texts has recently emerged as a new field of application for knowledge acquisition techniques (see, among others, [1]). The final goal of OP is the construction of an *ontological knowledge base* - a repository of facts about real-life objects and concept instances which follows the structure of an ontology. A rather different task is ontology learning (OL), where new concepts and relations are supposed to be acquired, with the consequence of changing the definition of the ontology itself (see, for instance, [2]). For example, via OL a system may automatically acquire the knowledge that "actor" is a concept and an "is-a" relation holds between it and the concept "person"; on the other hand, OP may learn automatically that "Robert De Niro" is an instance of the concept "actor".

Although there is no commonly accepted definition for the OP task, a useful approximation has been suggested in [3] as ontology Driven Information Extraction, where, in place of a template to be filled, the goal of the task is the extraction and classification of instances of concepts and relations defined in an ontology. A good example for an

[1]Corresponding Author: Hristo Tanev, IPSC - JRC, via Fermi 1, 21020 Ispra (VA), Italy; E-mail: htanev@gmail.com.

ontology driven information extraction infrastructure is the semantic annotation platform KIM [4]. Recently, Magnini et.al. [5] presented another more formal definition of the OP task based on *text mentions*.

The first aspect of OP, extraction and classification of concept instances, is typically attacked via Named Entity recognition, term extraction, term clustering (e.g. [6] and [7]), and term categorization (e.g. [8]). The second aspect - extraction and classification of relation instances is subject of Relation and Event Extraction ([9], [10], and [11]). In this paper we address the problem of concept instances classification relevant to the first aspect of OP.

In this paper OP is defined in the following scenario. Given a set of terms $T = t_1, t_2, ..., t_n$, a document collection D, where terms in T are supposed to appear, and a set of predefined classes $C = c_1, c_2, ..., c_m$ denoting concepts in an ontology, each term t_i has to be assigned to the proper class in C. For the purposes of the experiments presented in this paper we assume that (i) classes in C are mutually disjoint and (ii) each term is assigned to just one class.

As we have defined it, OP shows a strong similarity with Named Entity Recognition and Classification (NERC). However, a major difference is that in NERC each occurrence of a recognized term has to be classified separately, while in OP it is the term, independently of the context in which it appears, that has to be classified.

While Information Extraction, and NERC in particular, have been addressed prevalently by means of supervised approaches, ontology population is typically attacked in an unsupervised way. As many authors have pointed out (e.g. [12]), the main motivation is the fact that in OP the set of classes is usually larger and more fine grained than in NERC (where the typical set includes Person, Location, Organization, GPE, and a Miscellanea class for all other kind of entities). In addition, by definition, the set of classes in C changes as a new ontology is considered, making the creation of annotated data almost impossible practically.

According with the demand for weakly supervised approaches to OP, we propose a method, called *Class-Example*, which learns a classification model from a set of classified terms, exploiting lexico-syntactic features. Unlike most of the approaches which consider pair wise similarity between terms ([12], [6]), the Class-Example method considers the similarity between a term t_i and a set of training examples which represent a certain class. This results in a great number of class features and opens the possibility to exploit more statistical data, such as the frequency of appearance of a class feature in different training terms.

In order to show the effectiveness of the Class-Example approach, it has been compared against two different approaches: (i) a pattern-based unsupervised approach, in the style of [13]; (ii) an unsupervised approach described in [12] that considers the name of the class as a pivot word for acquiring relevant contexts for the class (we refer to this method as Class-Word). Results of the comparison show that the Class-Example method outperforms significantly the other two methods, making it appealing even considering the need of supervision.

Although the Class-Example method we propose is applicable in general, in this paper we show its usefulness when applied to terms denoting Named Entities. The motivation behind this choice is the practical value of Named Entity classifications, as for instance, in Questions Answering and Information Extraction. Moreover, some Named Entity classes, including names of writers, athletes, and politicians, dynamically change

over the time, which makes it impossible to represent their instances in a static ontological knowledge base.

The rest of the paper is structured as follows. Section 2 describes state-of-the-art methods in ontology population. Section 3 presents the three approaches to the task we have compared. Section 4 introduces the Syntactic Network, a formalism used for the representation of syntactic information and exploited in both the Class-Word and the Class-Example approaches. Section 5 reports on the experimental settings, results obtained, and discusses the three approaches. Section 6 concludes the paper and suggests directions for future work.

2. Related Work

There are two main paradigms in the term classification approaches relevant to the OP task. The first one is ontology population using patterns [13], rules, or the term structure [2]. The second paradigm is represented by the context feature approaches [12].

Pattern based approaches search for phrases which explicitly show that there is an "is-a" or other relation between two words, e.g. "the ant is an insect" or "ants and other insects". However, such phrases do not appear frequently in a text corpus. For this reason, some approaches use the Web [14]. In [2] the head-matching heuristic is introduced, according to which, if a $term_1$ is in the head of $term_2$, then there is an "is-a" relation between them: For example "christmas tree" is a kind of "tree". A weakly supervised pattern-based system is described in [15]. This system takes on its input several seed pairs which represent certain relation and learns generalized linear patterns which express this relation. In [16] the *Espresso* system is presented which uses patterns to extract relations in a weakly supervised manner and next maps them to WordNet. Some approaches like [4] make use of more complex grammar rules instead of simple patterns to perform mapping to an existing ontology.

A further step in this direction is implemented in Learning Cyc [17] where semantic formulae are translated into natural language patterns using Cyc's Natural Language Generation machinery. Another semantic-based OP system is Artequakt, described in [18]. This system uses the knowledge from an ontology to learn new facts and relations.

Context feature approaches use a corpus to extract features from all the contexts in which the instances of a semantic class are mentioned. Context features may be superficial [19] or syntactic [6], [7]. Comparative evaluation in [12] shows that the syntactic features lead to better performance. Feature weights are calculated by Machine Learning algorithms [19] or using statistical measures like Point Wise Mutual Information or the Jaccard coefficient [6].

A hybrid approach which uses pattern based, term structure, and context feature methods is presented in [20].

There are two types of OP methods according to the manual intervention they require: Unsupervised approaches like the one described in [12] or supervised approaches which in general use manually tagged training corpus, e.g. [19]. Low performance is a common feature of the state-of-the-art unsupervised OP approaches. On the other hand, supervised approaches provide higher accuracy, but require the manual construction of a training set - this significantly limits their scalability and flexibility. The OP systems which use rules and knowledge bases like [18] may also be assigned to the class of the

supervised methods, since they require manual construction of rules for classification and reasoning. Such semantic-based approaches suffer even more from scalability and flexibility problems, since each adaptation to a new domain requires adding new facts to the knowledge base or writing new rules.

3. Weakly supervised approaches for Ontology Population

In this section we present three ontology population approaches: two of them are virtually unsupervised - a pattern-based approach described in [13] and a context feature method reported in [12] to which we will refer as *Class-Word*; finally, we describe a new weakly supervised approach for ontology population which accepts as a training data lists of instances for each class under consideration. This method we call *Class-Example*

3.1. Pattern-based approach

This approach was described first by M. Hearst in [13]. The main idea is that if a noun t is an instance or a hyponym of the concept c, then in a text corpus we may expect the occurrence of phrases like *such c as t,...*. In our experiments for ontology population we used the patterns described in the Hearst's paper plus the pattern *t is (a | the) c*:

1. t is (a | the) c
2. such c as t
3. such c as (NP,)*, (and | or) t
4. t (,NP)* (and | or) other c
5. c, (especially | including) (NP,)* t

For each instance from the test set t and for each concept c we instantiated the patterns and searched with them in the corpus. If a pattern which is instantiated with a concept c and a noun t appears in the corpus, then we assume that t is an instance or a hyponym of c. For example, if the term to be classified is "Etna" and the concept is "mountain", one of the instantiated patterns will be "mountains such as Etna"; if this pattern is found in the text, then "Etna" is considered to be a "mountain". If this method assigns a term to several categories, we choose the one which co-occurs most often with the term.

3.2. Class-Word approach

Cimiano and Völker describe in [12] an unsupervised approach for ontology population based on context similarity between each concept c and a term to be classified t. For example, in order to conclude how much "Etna" is an appropriate instance of the class "mountain", this method finds the feature-vector similarity between the contexts of the word "Etna" and the contexts of the word "mountain". Each instance from the test set T is assigned to one of the classes in the set C. Features are collected from $Corpus$ and the classification algorithm on Figure 1 is applied. In this algorithm the context vectors v_t and v_c are feature vectors whose elements represent weighted context features of the term t (e.g. "Etna") and the concept word c (e.g. "mountain").

The problem with this approach is that the context distribution of a name (e.g. "Etna") is different than the context distribution of the class word (e.g. "mountain").

PROCEDURE CLASSIFY(T, C, $Corpus$)
for all t in T **do**
  $v_t = getContextVector(t, Corpus)$
end for
for all c in C **do**
  $v_c = getContextVector(c, Corpus)$
end for
for all t in T **do**
  $classes[t] = argmax_{c \in C} sim(v_t, v_c)$
end for
return $classes[]$

Figure 1. Unsupervised algorithm for Ontology Population.

For example, the more generic word "mountain" may be used to form a meaningful phrases like "mountain sports" and "mountain shoes"; on the oher hand, phrases like "Etna sports" and "Etna shoes" have much lower probability to appear in a text. Another shortcoming of the Class-Word approach is that a single word provides only a limited quantity of contextual data.

3.3. Syntactic features

Cimiano and Völker evaluate different context features and prove that syntactic features work best. Therefore, in our experimental settings we considered only such features extracted from a corpus parsed with a dependency parser. Unlike the original approach of Cimiano and Völker, which relies on pseudo-syntactic features, we used features extracted from dependency parse trees. Moreover, we used virtually all the words connected syntactically to a term, not only the modifiers.

A syntactic feature is a pair: *(word, syntactic relation)* (see [6]), for example ("invent", *subject-of*). We consider two feature types: *First order features*, which are directly connected with the test examples in the dependency parse trees of $Corpus$; *second order features*, which are connected to the training or test instances indirectly by skipping one word (the verb) in the dependency tree. As an example, let's consider two sentences: "Edison invented the phonograph" and "Edison created the phonograph". If "Edison" is a name to be classified, then two first order features of this name exist - ("invent", *subject-of*) and ("create", *subject-of*). One second order feature can be extracted - ("phonograph", *object-of+subject*); it co-occurs two times with the word "Edison".

3.4. Weakly Supervised Class-Example Approach

The approach we present here uses the same processing stages as the one presented in Figure 1 and relies on context features extracted from a corpus. However, the Class-Example algorithm receives as an additional input parameter the sets of training examples $Train(c)$ for each class $c \in C$. These training sets are simple lists of instances (i.e. terms denoting Named Entities) without any context. They can be acquired automatically or semi-automatically from an existing ontology or gazetteer. To facilitate their acquisition, the Class-Example approach imposes no restrictions on the training examples - they can be ambiguous and have different frequencies. However, they have to ap-

pear in $Corpus$ (in our experimental settings - at least twice). For example, for the class "mountain" training examples are: "Everest", "Mauna Loa", etc.

The algorithm learns from each training set $Train(c)$ a single feature vector v_c. We used syntactic features in our expreriments, therefore we call v_c a *syntactic model of the class*. In our algorithm, the statement

$$v_c = getContextVector(c, Corpus)$$

on Figure 1 is substituted with

$$v_c = getSyntacticModel(Train(c), Corpus)$$

.

For each class c, a set of context features $F(c)$ are collected by finding the union of the features extracted from each occurrence in the corpus of each training instance in $Train(c)$. Next, the feature vector v_c is constructed: If a feature is not present in $F(c)$, then its corresponding coordinate in v_c has value 0; otherwise, it has a value equal to the feature weight.

The weight of a feature $f_c \in F(c)$ is calculated in three steps:

1. First, the co-occurrence of f_c with the training set is calculated:

$$weight_1(f_c) = \sum_{t \in Train(c)} \alpha \cdot log(\frac{P(f_c, t)}{P(f_c).P(t)})$$

   where $P(f_c, t)$ is the probability that feature f_c co-occurs with t, $P(f_c)$ and $P(t)$ are the probabilities that f_c and t appear in the corpus, $\alpha = 14$ for syntactic features with lexical element noun and $\alpha = 1$ for all the other syntactic features. The α parameter reflects the linguistic intuition that nouns are more informative than verbs and adjectives which in most cases represent generic predicates. The optimal values of α were automatically learned from the training data.
2. We normalize the feature weights, since we observed that they vary significantly between different classes:
   First, for each class c we find the feature with maximal weight and denote its weight with $W_{max}(c)$,

$$W_{max}(c) = max_{f_c \in F(c)} weight_1(f_c)$$

   Next, the weight of each feature $f_c \in F(c)$ is normalized by dividing it with $W_{max}(c)$:

$$weight_N(f_c) = \frac{weight_1(f_c)}{W_{max}(c)}$$

3. To obtain the final weight of f_c, we divide $weight_N(f_c)$ by the number of classes in which this feature appears. This is motivated by the intuition that a feature which appears in the syntactic models of many classes is not a good class predictor.

$$weight(f_c) = \frac{weight_N(f_c)}{|Classes(f_c)|}$$

where $|Classes(f_c)|$ is the number of classes for which f_c is present in the syntactic model.

As shown in Figure 1, the classification uses a similarity function $sim(v_t, v_c)$ whose arguments are the feature vector of a term v_t and the feature vector v_c for a class c. We defined the similarity function as the dot product of the two feature vectors: $sim(v_t, v_c) = v_c \cdot v_t$. To simplify the calculation, the vectors v_t are binary (i.e. the feature value is 1 if the feature is present and, 0-otherwise), while the features in the syntactic model vectors v_c receive weights according to the approach described in this section.

4. Representing Syntactic Information

Due to efficiency issues, state-of-the-art syntactic based approaches for OP use superficial syntactic features extracted from a relatively small data set ([12]) or the Web ([7]). However, Almuhareb and Poesio ([7]) say that it would be better to use a full parsing instead of that, but it is "computationally much more expensive". We used a full parser to parse a relatively large off-line news corpus of about half a gigabyte. Our approach extracts from this corpus the syntactic features for the Class-Word and the Class-Example methods. Using a large off-line data set mitigates the problem of data sparseness and unlike the Web-based approaches, it allows for fast feature extraction thus making possible to perform large-scale learning.

Syntactic Pre-processing. We used MiniPar - a statistically based full dependency parser [21] to process the corpus. The obtained dependency structures are directed labelled graphs whose vertices represent words and the edges between them represent syntactic relations like *subject*,*object*, *modifier*, etc. Examples for two dependency structures - g_1 and g_2, are shown in Figure 2: They represent the sentences "John loves Mary" and "John loves Jane"; labels s and o on their edges stand for $subject$ and $object$ respectively. The syntactic structures generated by MiniPar are dendroid (tree-like), but still cycles appear in some cases. We performed some normalizations on the output of MiniPar; the most important of them was to put the prepositions on the arcs as labels, instead of treating them as separate nodes.

Syntactic Network. In order to find information efficiently in a large corpus of millions of syntactic graphs, we had to index them using an appropriate model. Building a classic index at word level was not an option, since we had to search for syntactic structures, not words. On the other hand, indexing syntactic relations (i.e. word pair and the relation between the words) would be useful, but still does not resolve the problem, since in many cases we search for more complex structures than a relation between two words: for example, when we have to find which words are syntactically related to a Named Entity composed of two words, we have to search for syntactic structures which consists of three vertices (two for the Named Entity and one for the related word) and two edges (one connects the two words representing the Named Entity and one connects the entity to the related word). To solve this problem, we present a more elaborated model for representation of a set of labelled graphs, called *Syntactic Network* (*SyntNet* for short).

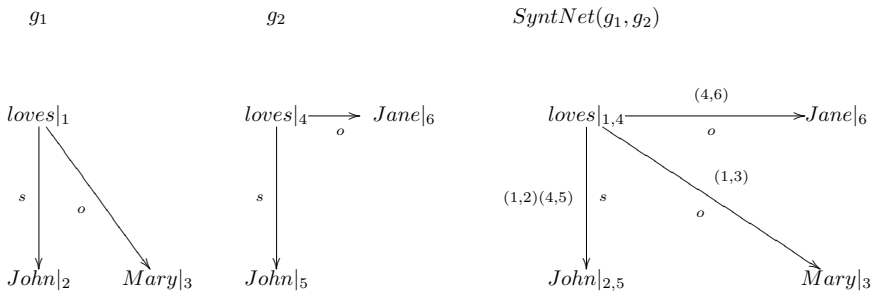

Figure 2. Two syntactic graphs and their Syntactic Network.

The model is inspired by a representation schema which we presented earlier in [22], however SyntNet allows for more efficient processing. The scope of our model is to represent a set of labelled graphs through one aggregate structure in which the isomorphic sub-structures overlap. When SyntNet represents a syntactically parsed text corpus, its vertices are labelled with words from the text while edges represent syntactic relations from the corpus and are labelled accordingly.

An example is shown on Figure 2, where two syntactic graphs, g_1 and g_2, are merged into one aggregate representation $SyntNet(g_1, g_2)$. Vertices labelled with equal words in g_1 and g_2 are merged into one generalizing vertex in $SyntNet(g_1, g_2)$. For example, the vertices with label $John$ in g_1 and g_2 are merged into one vertex $John$ in $SyntNet(g_1, g_2)$.

Edges are merged in a similar way: $(loves, John) \in g_1$ and $(loves, John) \in g_2$ are represented through one edge $(loves, John)$ in $SyntNet(g_1, g_2)$.

Each vertex in g_1 and g_2 is labelled additionally with a numerical index which is unique for the graph set. Numbers on vertices in $SyntNet(g_1, g_2)$ show which vertices from g_1 and g_2 are merged in the corresponding SyntNet vertices. For example, vertex $loves \in SyntNet(g_1, g_2)$ has a corresponding index set $\{1, 4\}$ which means that vertices 1 and 4 are merged in it. In a similar way the edge $(loves, John) \in SyntNet(g_1, g_2)$ is labelled with two pairs of indices $(4, 5)$ and $(1, 2)$, which shows that it represents two edges: the edge between vertices 4 and 5 and the edge between 1 and 2.

Two properties of SyntNet are important: First, isomorphic sub-structures from all the graphs represented via a SyntNet are mapped into one structure. This allows us to easily find all the occurrences of multiword terms and named entities. Second, using the numerical indices on vertices and edges, we can efficiently calculate which structures are connected syntactically to the training and test terms. As an example for structure tracing, let's try to calculate in which constructions the word "Mary" appears considering $SyntNet$ on Figure 2. First, in SyntNet we can directly observe that there is the relation *loves* $\rightarrow$ *Mary* labelled with the pair $1 \rightarrow 3$ - therefore this relation appears once in the corpus. Next, tracing the numerical indices on the vertices and edges we can find a path from "Mary" to "John" through "loves". The path passes through the following numerical indices: $3 \leftarrow 1 \rightarrow 2$, this means that there is one appearance of the structure "John loves Mary" in the corpus, spanning through vertices $1, 2$, and 3. Such a path through the numerical indices cannot be found between "Mary" and "Jane" which means that they do not appear in the same syntactic construction in the corpus.

It can be shown that each syntactic structure can be found in the SyntNet in $O(MaxCO \cdot log_2 MaxCO + log_2|arcs|)$ time, where $MaxCO$ is the maximal number

of occurrences of a syntactic pair (syntactic pair is a pair of words connected by a syntactic arc, for example $loves \xrightarrow{o} Mary$) and $|arcs|$ is the number of the arcs in SyntNet, that is the number of the different syntactic pairs in the corpus. We present more detailed complexity analysis in the Appendix to this paper.

SyntNet is built incrementally in a straightforward manner: Each new vertex or edge added to the network is merged with the identical vertex or edge, if such already exists in SyntNet. Otherwise, a new vertex or edge is added to the network. The time necessary for building a SyntNet is proportional to the number of the vertices and the edges in the represented graphs (and does not otherwise depend on their complexity).

We used the properties of SyntNet in order to trace efficiently the occurrences of Named Entities in the parsed corpus, to calculate their frequencies, to find the syntactic features which co-occur with these Named Entities, as well as the frequencies of these co-occurrences. Moreover, the SyntNet model allowed us to extract more complex, second order syntactic features which are connected indirectly to the terms in the training and the test set.

5. Experimental settings and results

	macro P (%)	*macro R (%)*	*macro F (%)*	*micro F(%)*
patterns	18	6	9	10
Class-Word	32	41	33	42
Class-Example	67	63	62	65

Table 1. Comparison of different approaches.

We have evaluated all the three approaches described in Section 3. The same evaluation settings were used for the three experiments. The source of features was a news corpus of about half a gigabyte. The corpus was parsed with MiniPar and a Syntactic Network representation was built from the dependency parse trees produced by the parser. Syntactic features were extracted from this SyntNet.

We considered two high-level Named Entity categories: Locations and Persons. For each of them five fine-grained sub-classes were taken into consideration. For locations: *mountain*, *lake*, *river*, *city*, and *country*; for persons: *statesman*, *writer*, *athlete*, *actor*, and *inventor*.

For each class under consideration we created a test set of Named Entities using WordNet 2.0 and Internet sites like Wikipedia. For the Class-Example approach we also provided training data using the same resources. WordNet was the primary data source for training and test data. The examples from it were extracted automatically. To do this, for each concept under consideration, we took the instances of this concept by considering its hyponyms which begin with uppercase letters and which are leaves of the taxonomy tree, rooted in this concept. We used the Internet to get additional examples for some classes.

For example, Wikipedia has articles such as “List of inventors” which enumerate certain concept instances. Other Web pages also contain similar lists. We created automatic text extraction scripts for these Web sites. We manually processed their output, when it was necessary.

In total, the test data comprised 280 Named Entities which were not ambiguous and appeared at least twice in the corpus.

For the Class-Example approach we provided a training set of 1194 names. The only requirement to the names in the training set was that they appear at least twice in the parsed corpus. They were allowed to be ambiguous and no manual post-processing or filtering was carried out on this data.

For both context feature approaches (i.e. Class-Word and Class-Example), we used the same type of syntactic features and the same classification schema, namely the one described in Section 3.3. This was done in order to compare better the approaches.

Results from the comparative evaluation are shown in Table 1. For each approach we measured macro and micro average precision, macro and micro average recall, and macro and micro average F-measure.

The micro measures consider all the terms across all the classes together: The micro average precision is calculated by dividing the number of the correctly classified terms to the number of all the terms which were classified. The micro average recall is the number of the correctly classified terms divided by the total number of terms. The micro F is calculated from the micro precision and the micro recall. In the experiments with Class-Word and Class-Example approaches we assigned a class to each term from the test set. That is why micro average precision is equal to the micro average recall and both of them are equal to the micro F - all these three measures refer to the percent of the instances classified correctly. It was not the case with the pattern-based approach, where some entities were not classified.

The macro average measures were calculated as average of the corresponding measures for each class. For example, if we have x instances of the class *actor* in the test set, the system recognizes y instances of the class *actor* and z of them are recognized correctly, then the precision of populating the *actor* class is z/y, the recall is z/x, and the F measure is calculated from the precision and the recall. We calculated precision, recall, and F measure for each class, then the macro precision was calculated as the average of the precisions of all the classes, the macro recall was set to the average of the recalls and macro F was set to the average of the F measures.

The first row of Table 1 shows the results obtained with superficial patterns. The second row presents the results from the Class-Word approach. The third row shows the results of our Class-Example method.

The pattern-based approach showed low performance, similar to the random classification, for which macro and micro F=10%. Patterns succeeded to classify correctly only instances of the classes "river" and "city". For the class "city" the patterns reached precision of 100% and recall 65%; for the class "river" precision was high (i.e. 75%), but recall was 15%.

The Class-Word approach showed significantly better performance (macro F=33%, micro F=42%) than the pattern-based approach.

The performance of the Class-Example (62% macro F and 65% micro F) is much higher than the performance of Class-Word (29% increase in macro F and 23% in micro F).

A more detailed evaluation of the Class-Example approach is shown in Table 2. Results vary between different classes: The highest F is measured for the class "country" - 89% and the lowest is for the class "inventor" - 18%. However, the class "inventor" is an exception - for all the other classes the F measure is over 50%. Another difference

	P (%)	*R (%)*	*F (%)*
mountain	58	78	67
lake	75	50	60
river	69	55	61
city	56	76	65
country	86	93	89
locations macro	69	70	68
statesman	42	72	53
writer	93	55	69
athlete	90	47	62
actor	90	73	80
inventor	12	33	18
persons macro	65	56	57
total macro	67	63	62
total micro	65	65	65
category location	83	91	87
category person	95	89	92

Table 2. Performance of the Class-Example approach

may be observed between the Location and Person classes: Our approach has a better performance for the locations (68% vs. 57% macro F). Although different classes had different number of training examples, we observed that the performance for a class does not depend on the size of its training set. We think, that the variation in performance between categories is due to the different specificity of their textual contexts. As a consequence, some classes tend to co-occur with more specific syntactic features, while for other classes this is not true.

Additionally, we measured the performance of our approach considering only the macro-categories "Location" and "Person". For this purpose we did not run another experiment, we rather used the results from the fine-grained classification and grouped the already obtained classes. Results are shown in the last two rows of table 2: It turns out that the Class-Example method makes very well the difference between "location" and "person" - 90% of the test instances were classified correctly between these categories.

Syntactic Features. We looked at the syntactic model of every concept produced by the Class-Example approach and found that in most of the cases our approach weights the syntactic features in a relevant manner. For example, the top ranked features for the class "river" are:

- *bank of* **[NAME]**
- **[NAME]** *river*
- **[NAME]** *region*
- **[NAME]** *town*
- **[NAME]** *cruise*
- *canoe down* **[NAME]**
- *cross* **[NAME]**
- **[NAME]** *basin*
- *road along* **[NAME]**

(Here the syntactic features are represented in their linearized text form.) We also reached the conclusion that some of the undesired features come from ambiguity of the name. For example, one of the top ranked feature for the class "mountain" was found to be **[NAME]** *brand*; this is because some of the mountain names can be name of brands too. One way to resolve this problem is to perform text filtering, for example for the class "mountain" we can use only geographical texts.

We did not show in the presented evaluation table the results obtained with the second-order syntactic features, since it turned out that it is impossible to learn appropriate weights for these features using the training set. However, we experimented with different weights and measured the performance on the test set. The optimal weight configuration for the test set resulted in a macro F of 62% and micro F of 68%, adding 3% to the micro F.

6. Conclusions and future work

In this paper we presented a new weakly supervised approach for ontology population, called Class-Example, and confronted it with two other methods. Experimental results show that the Class-Example approach has best performance. In particular, it reached 65% of accuracy, outperforming in our experimental framework the state-of-the-art Class-Word method by 42%. The presented weakly supervised Class-Example approach takes on the input simple lists of training instances which can be automatically acquired from existing knowledge bases, dictionaries, and gazetteers. This makes our weakly supervised methodology applicable on larger scale than supervised approaches, still having significantly better performance than the unsupervised ones. The 42% advantage of the Class-Example versus the Class-Word approaches may be viewed as a clue to the correctness of our assumption that the context distrubution of the class instances is different than the context distribution of the word representing the class name.

In our experimental framework we used syntactic features extracted from dependency parse trees generated by a full parser - MiniPar. We put forward Syntactic Network - a novel model for representation of a syntactically parsed corpus. This model allows for performing efficient and comprehensive extraction of syntactic features from a parsed corpus. Empirical observations not described in this paper lead us to the conclusion that the performance of an ontology population system improves with the increase of the types of syntactic features under consideration. In this clue, the Syntactic Network model played an important role in the ontology population process, since it allowed us to use different types of relations. This model made it feasible to exploit without limitations the expressive power of the full parser and to use as a corpus large amounts of text.

In our future work we consider applying our ontology population methodology to more semantic categories and to experiment with other types of syntactic features, as well as other types of feature-weighting formulae and learning algorithms. A bootstrapping OP appoach can be developed on the basis of Class-Example method in which the new classified instances are used as training ones in a consecutive learning iteration.

In the framework of Information Extraction and Question Answering the Class-Example method can be used to populate domain specific ontologies which back up fine-grained Named Entity recognition and answer extraction.

References

[1] P. Buitelaar, P. Cimiano, and B. Magnini: Ontology Learning from Text: Methods, Evaluation and Applications, IOS Press, Amsterdam, The Netherlands, 2005.

[2] P. Velardi, R. Navigli, A. Cuchiarelli, and F.Neri: Evaluation of Ontolearn, a Methodology for Automatic Learning of Domain Ontologies, *Ontology Learning from Text: Methods, Evaluation and Applications*, pages 92–106, IOS Press, 2005.

[3] K. Bontcheva and H. Cunningham: The Semantic Web: A New Opportunity and Challenge for HLT, *Proceedings of the Workshop Human Language Technologies for the Semantic Web and Web Services at International Semantic Web Conference (ISWC)*, 2003.

[4] B. Popov, A. Kiryakov, A. Kirilov, D. Manov, D. Ognyanoff, and M. Goranov: KIM - Semantic Annotation Platform, *The SemanticWeb - International Semantic Web Conference (ISWC)*, Springer-Berlin, Volume 2870, pages 834–849, 2003.

[5] B. Magnini, E. Pianta, O. Popescu, and M. Speranza: Ontology Population from Textual Mentions: Task Definition and Benchmark, *Proceedings of the 2nd Workshop on Ontology Learning and Population (OLP2): Bridging the Gap between Text and Knowledge*, Sidney, Australia, 2006.

[6] D. Lin: Automatic Retrieval and Clustering of Similar Words, *In Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics (COLING-ACL'98)*, pages 768–773, Montreal, Canada, 1998.

[7] A. Almuhareb and M. Poesio: Attribute-based and Value-based Clustering: An Evaluation, *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*, pages 158–165, Barcelona, Spain, 2004.

[8] H. Avancini, A. Lavelli, B. Magnini, F. Sebastiani, and R. Zanoli: Expanding Domain-Specific Lexicons by Term Categorization, *Proceedings of the 2003 ACM Symposium on Applied Computing (SAC)*, March, 2003.

[9] D. Zelenko, C. Aone, and A. Richardella: Kernel Methods for Relation Extraction, *Journal of Machine Learning Research* **3**, pages 1083–1106, 2003.

[10] L. Romano, M. Kouylekov, I. Szpektor, I. Dagan, and A. Lavelli: Investigating a Generic Paraphrase-based Approach for Relation Extraction, *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Trento, Italy, 2006.

[11] A. Yakushiji, Y. Tateisi, Y. Miyao, and J. Tsujii: Event Extraction from Biomedical Papers Using a Full Parser, *Pacific Symposium Biocomputing*, pages 408–419, 2001

[12] P. Cimiano and J. Völker: Towards Large-scale, Open-domain and Ontology-based Named Entity Classification, *Proceedings of Recent Advances in Natural Language Processing (RANLP)*, pages 166–172, Borovets, Bulgaria, 2005.

[13] M. Hearst: Automated Discovery of WordNet Relations, *WordNet: An Electronic Lexical Database*, pages 131–152, MIT Press, 1998.

[14] S. Schlobach, M. Olsthoorn, and M. de Rijke: Type Checking in Open-Domain Question Answering, *Proceedings of 16th European conference on Artificial Intelligence(ECAI)*, 2004.

[15] M. Ruiz-Casado, E. Alfonseca, M. Okubura, and P. Castells: Information Extraction and Semantic Annotation of Wikipedia, *Bridging the Gap between Text and Knowledge - Selected Contributions to Ontology Learning and Population from Text*, editors: P. Buitelaar and P. Cimiano, IOS-Press, 2007, THIS VOLUME

[16] P. Pantel and M. Pennacchiotti: Automatically Harvesting and Ontologizing Semantic Relations, *Bridging the Gap between Text and Knowledge - Selected Contributions to Ontology Learning and Population from Text*, editors: P. Buitelaar and P. Cimiano, IOS-Press, 2007, THIS VOLUME

[17] C. Matuszek, M. Witbrock, R.C. Kahlert, J. Cabral, D. Schneider, P. Shah, and D. Lenat: Searching for Common Sense: Populating Cyc^{TM} from the Web, *Proceedings of the Twentieth National Conference on Artificial Intelligence*, pages 1430–1435, Pittsburgh, Pennsylvania, 2005.

[18] H. Alani, S. Kim, D.E. Millard, M.J. Weal, W. Hall, P.H. Lewis, and N.R. Shadbolt: Automatic Ontology-Based Knowledge Extraction from Web Documents, *IEEE Intelligent Systems*, 18(1), pages 14–21, 2003

[19] M. Fleischman and E. Hovy: Fine Grained Classification of Named Entities, *Proceedings of the 19th International Conference on Computational Linguistics (COLING)*, pages 267–273, Taipei, Taiwan, 2002.

[20] P. Cimiano, A. Pivk, L.S. Thieme, and S. Staab: Learning Taxonomic Relations from Heterogeneous Sources of Evidence, *Ontology Learning from Text: Methods, Evaluation and Applications*, pages 59–

76, IOS Press, 2005

[21] D. Lin: Dependency-based Evaluation of MiniPar, *Proceedings of Workshop on the Evaluation of Parsing Systems*, Granada, Spain, 1998.

[22] I. Szpektor, H. Tanev, I. Dagan, and B. Coppola: Scaling Web-based Acquisition of Entailment Relations, *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*, pages 41–48, Barcelona, Spain, 2004.

Appendix: Complexity analysis of the structure tracing in the SyntNet model

In our experiments the SyntNet was stored in a database. Let's estimate the complexity of calculating how many times a structure appears in the corpus, using the database. As an example we will follow how the occurrences of the constructioin $John \overset{s}{\leftarrow} loves \overset{o}{\rightarrow} Mary$ are traced in SyntNet on Figure 2.

First, we have to find between which pairs of vertices the arcs $John \overset{s}{\leftarrow} loves$ and $loves \overset{o}{\rightarrow} Mary$ appear. From $SyntNet(g_1, g_2)$ on Figure 2 we can see that the first arc has two index pairs assigned $(4, 5)$ and $(1, 2)$; the second arc has one index pair $(1, 3)$. In the SyntNet database there is an inverted index which maps each arc to these index pairs, therefore the time for searching and collecting the occurrences of an arc a is $log_2(|arcs|) + CountOccurrences(a)$, where $|arcs|$ is the number of arcs in the SyntNet, and $CountOccurrences(a)$ is the number of the occurrences of a in the corpus. For example, $CountOccurrences(John \overset{s}{\leftarrow} loves) = 2$.

Since the index pairs denote arc occurrences, the first index of each pair is an initial vertex of an arc occurrence. The set of the first indices of the arc's index pair we call *initial index set*. The initial index set of $John \overset{s}{\leftarrow} loves$ is $\{4, 1\}$ and the initial index set of $loves \overset{o}{\rightarrow} Mary$ is $\{1\}$. For each arc a the cardinality of the initial index set is at most $CountOccurrences(a)$, therefore the number of operations for extraction of the initial index set of a and ignoring its duplicates is $O(CountOccurrences(a) \cdot log_2 CountOccurrences(a))$.

We calculate the intersection of the initial index sets of the two arcs: $\{1\} \cap \{4, 1\} = \{1\}$. We refer to this intersection as *intersection index set*. The indices in this intersection refer to these vertices which are initial for the occurrences of both arcs. In this example, the intersection contains only one vertex with index 1. This means that the construction $John \overset{s}{\leftarrow} loves \overset{o}{\rightarrow} Mary$ appears once in the corpus and it is rooted in vertex number 1. As we stated, the initial index set of each arc a has no more than $CountOccurrences(a)$ elements. Intersection between two sets each having no more than n elements requires at most $O(n \cdot log_2 n)$ operations. If $MaxCO$ is the maximal value of $CountOccurrences(a)$ for the arcs from a given SyntNet, then intersection will require no more than $O(MaxCO \cdot log_2 MaxCO)$ operations for each pairs of arcs from SyntNet.

When we obtain the intersection index set, we may go back to the pairs of indices of the two arcs and consider all index pairs which begin with some of the elements from the intersection index set. We refer to this last operation as *index set propagation*. In this example the index set propagation will return the pairs $(1, 2)$ and $(1, 3)$. In this example the propagation is necessary to compute the occurrences of the three-vertices construction as a whole. Propagation requires at most $O(MaxCO)$ operations, since each arc under consideration has at most $MaxCO$ index pairs.

Taking into account all the estimates, the upper bound of the the number of operations necessary to calculate the occurrences of a construction of two arcs is

$$O(MaxCO \cdot log_2 MaxCO + log_2 |arcs|)$$

In a similar manner, using sequences of intersections and propagations we may trace the occurrence of syntactic constructions of any type. If the size of these syntactic constructions is limited by a constant, it may be shown that the complexity estimate $O(MaxCO \cdot log_2 MaxCO + log_2|arcs|)$ is valid also in this general case when the syntactic graphs from the corpus do not contain cycles. When cyclic constructions appear, it is necessary to add control in order to avoid infinite loops in the intersection and propagation cycles; this may lead to increased complexity.

Information Extraction and Semantic Annotation of Wikipedia

Maria RUIZ-CASADO [a,1], Enrique ALFONSECA [a] Manabu OKUMURA [b] and Pablo CASTELLS [a]

[a] *Computer Science Department, Universidad Autonoma de Madrid, Spain*
[b] *Precision and Intelligence Laboratory, Tokyo Institute of Technology, Japan*

Abstract. An architecture is proposed that, focusing on the Wikipedia as a textual repository, aims at enriching it with semantic information in an automatic way. This approach combines linguistic processing, Word Sense Disambiguation and Relation Extraction techniques for adding the semantic annotations to the existing texts.

Keywords. Wikipedia, Semantic Annotation, Relation Extraction

Introduction

The vast amount of information stored in on-line sources is pushing forward the efforts in researching new techniques to process it effectively. The last decade has witnessed many successful systems for Information Extraction and Information Retrieval able to cope with large data repositories, but the extraction of semantic information from web data is still far from being fully solved. Therefore, we can recently observe increasing interest in Information Extraction tools able to learn efficiently how to annotate on-demand from unlabelled data [1,2]. On the other hand, the appearance of new types of web content such as wikis, web forums and blogs, provides a new source of textual information with an underlying structure that can be exploited to increase the performance of automatic annotation systems. Thus, for example, existing wikis may be used as external knowledge sources for improving the precision of existing systems [3,4] or the wikis themselves may be extended automatically with semantic information [5]. Web forums can also be analysed to obtain information on social interaction [6].

The availability of semantic annotations would have applications in many fields. Two examples are ontology-based information retrieval [7] or semantic work environments [8]. The explicit annotation with relations (syntactically and semantically motivated) also improves the performance of Question-Answering systems [9,10,11]. Some authors have proposed that the semantic annotations can be added manually to documents following the wiki philosophy of collaboration in large communities, combined with lowering the technical barriers. The annotations may be as simple as tags attached to hyperlinks [12]. But, even if we take into consideration that many wiki systems have a

[1] Corresponding Author: E-mail: Maria.Ruiz@uam.es.

very large community of users, the semantic tagging can still be considered a bottleneck: placing labels in a large amount of existing content, sometimes also rapidly evolving, can be too costly if it has to be done manually. In February 2007, the Wikipedia already exceed 200,000 articles in at least 10 languages. If all these entries were to be extended with semantic annotations manually in a reasonable amount of time, the cost would be enormous, if not unfeasible. The problem is more acute when annotating personal or company-owned textual databases, where the number of users and information managers may be small and the amount of data comparatively very large.

In answer to this need, the use of semi-automatic annotation procedures has been the object of extensive research. In general, semantic tags provide the same information that a human reader can elicit (sometimes unconsciously) from an untagged natural-language text: names of locations, people, organisations and other entities mentioned in the text, events and relations involving them, spatiotemporal attributes for those events, etc. For instance, a text may convey that a particular person works for a specific company at some date. The automation of labelling natural language texts can benefit from partly structured information that these may have, such as tables, lists, or LaTeX, HTML or XML tags. Wrappers [13,14] take advantage of this structure in tagging portions of the documents. However, they are not applicable to any document that does not follow a fixed structure. In order to be applicable in more general cases, the use of Natural Language Processing techniques such as Word-Sense Disambiguation (WSD), Named Entity Recognition and Classification (NERC), co-reference resolution, Term Identification (TI) or relation extraction have proved their usefulness.

This chapter describes a proposed architecture that, focusing on the Wikipedia as a textual repository, aims at enriching it with semantic information in a fully automatic way. This approach is based on many of the above-mentioned NLP techniques, and a special focus is placed on the identification of relations between entities in the texts. The structure of the chapter is as follows: Section 1 provides a brief overview of related systems; Section 2 describes the proposed architecture and details internally some of its components; and, finally, Section 3 concludes the chapter.

1. Related work

Wikipedia, from its very beginning, attracted the attention of the research community as a resource to study social and collaborative interactions and trust [15,16]. More recently, there is an increasing interest in using the Wikipedia as a linguistic resource, with applications as varied as Named Entity Recognition [17], Named Entity Disambiguation [3], generation of parallel corpora [18], Question Answering [19], and co-reference resolution and semantic similarity calculation [20].

Concerning the automatic annotation of the Wikipedia, to our knowledge, there is no other work reported addressing the task of annotating it automatically with Semantic Web formalisms to this extent, including the disambiguation and classification of the entries in an ontology, and the extraction of taxonomic and non-taxonomic relations. However, there are several works that attempt to partially annotate it with either taxonomic relations [21], general relations [22] or generic "common-sense" RDF statements [23], amongst other works.

In the particular case of the automatic identification of relations in unrestricted text, there is already much work, although not necessary confined to the particular case of the

Wikipedia. Some use machine-readable dictionaries [24,25,26], WordNet glosses [27,28, 29] or just free text, by analysing distributional properties of words [30,31,32,33]. The use of lexical or lexicosyntactic patterns has been proposed more than a decade ago to discover ontological and non-taxonomic relations between concepts. Hand-made regular expressions have been used to extract hyponymy or part-of relations [34,35,36]. Kietz et al. [37] quantify the error rate of a similar approach for extracting taxonomic relations at 32%. Other approaches learn patterns that express non-taxonomic relations, such as company merge relations [38].

Systems that learn these lexical patterns using web corpora have the advantage that the training corpora can be collected easily and automatically. Several approaches have been proposed recently [39,40,10,41,42], having various applications in mind: Question-Answering [10], multi-document Named Entity Coreference [43], and the generation of biographical information [44]. Some procedures that are specially relevant with respect to our work are those by Pantel and Pennacchiotti [42,45] and Blohm et al. [46].

Other architectures for automatic semantic annotation of unrestricted text [47] are TextToOnto [48], OntoLT [49], KIM [50], C-PANKOW [51] and Text2Onto [52].

2. General architecture

The proposed architecture has been designed to be general enough to be applicable to any kind of texts. However, for the experiments, we have chosen to use the Wikipedia, because of the following reasons:

- Wikipedia is an encyclopedia, and as such it is organised in entries, each one having a more or less fixed structure: each entry contains a title of the term described, the first paragraph provides a brief definition of that term, and the remaining text further elaborates it.
- It already contains some annotations explicitly provided by the users, such as a (rather noisy) hierarchy of categories, or information on polysemous terms with so-called *disambiguation pages*, which might be useful for extending the proposed architecture.
- Wikipedia has already proven useful as a linguistic resource [3].

On the other hand, Wikipedia is a work-in-progress, and as such it contains errors, for instance duplicated entries, or dangling links, i.e. hyperlinks to Wikipedia entries that have not been created yet. Throughout this paper we shall make the distinction between existing Wikipedia pages (those that have been created and contain text) and non-existing Wikipedia pages (those that are pointed to by at least one hyperlink but still have not been created).

The general architecture of the proposed approach for the acquisition and annotation of semantics from Wikipedia is depicted in Figure 1. The system proceeds in the following way:

1. A central Knowledge Base stores the Wikipedia entries and all annotations obtained about them automatically produced by our system. Initially, it contains a lexical semantic network to which the Wikipedia entries will be associated, in order to have some world knowledge to depart from. The current implementation uses WordNet [53], although other lexical semantic networks would also be valid.

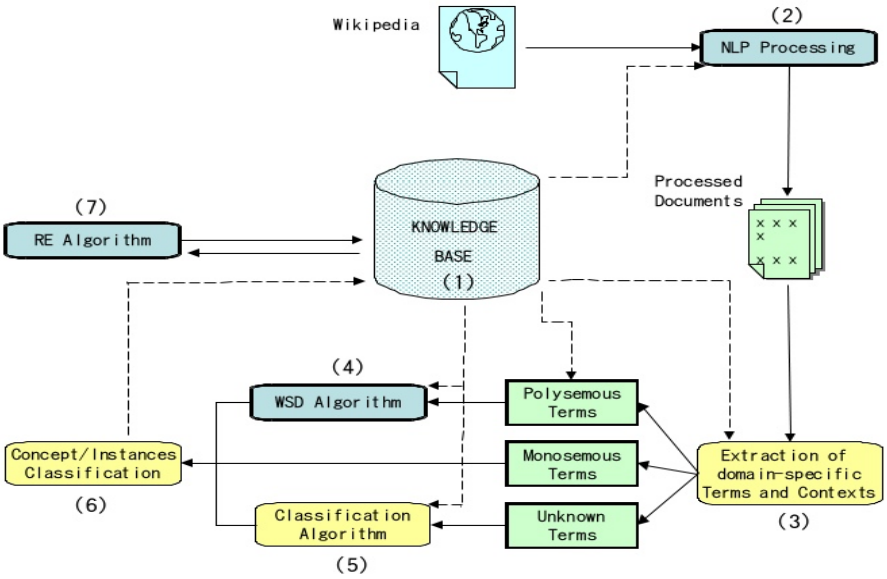

Figure 1. Architecture of the system for the automatic annotation of the Wikipedia with semantic relations.

2. A static dump of the Wikipedia is processed with NLP tools. All the information obtained in this step is directly encoded in XML inside the downloaded Wikipedia entries, so it can be easily accessed by the following modules. Because that is just syntactic information (and not semantic), it is not placed yet inside the knowledge base.
3. A set of relevant terms is extracted from the Wikipedia pages, including (a) the titles of existing pages; (b) the target of hyperlinks to pages that do not exist yet; and (c) named entities identified by the NLP tools that do not have an associated page, representing names of people, locations or organisations. These terms are now contrasted with the lexical semantic network stored in the central repository, and classified in three types:
   - those terms that do not appear in the semantic network, which are labelled as *unknown* (for the moment),
   - those terms that appear both in the Wikipedia and in the semantic network with just one meaning. These are assumed to be monosemous, and to have the same meaning as in the network. For those terms for which there is an entry in the Wikipedia, the entry is immediately associated to the corresponding node in the semantic network,
   - those terms that appear with several senses in the semantic network or in Wikipedia, which are labelled as *polysemous* terms.
4. The terms from step 3 that are polysemous are subject to a disambiguation process in order to discover which Wikipedia entries correspond to each of the meanings of those words in WordNet.
5. The terms from step 3 that are labelled as unknown are fed into a classification module that tries to classify them inside WordNet by learning taxonomic relations

(hyperonymy, *is-a* relations) between them. Furthermore, a procedure to learn non-taxonomic relations is also run on all the entries (7).

6. The new *is-a* relations learnt in step 5 are classified as *subclass* or *instance-of* relations.

The system annotates the Wikipedia entries in the format of the Semantic Wikipedia, by annotating the Wikipedia hyperlinks with labels.

The following subsections further elaborate some of these steps.

2.1. Linguistic processing

The computational linguistics tools chosen for the experiments is the wraetlic NLP toolkit [54] that provides information on tokenisation, sentence splitting, morphological analysis, part-of-speech tagging, Named Entity Recognition and Classification, chunking and partial parsing. As indicated, the annotations are encoded in XML inside the Wikipedia entries, which makes it convenient when they are needed for analysis later on. For illustration, Figure 2 shows the annotations added to a sample sentence after having been analysed by the tools. As can be seen, the syntactic analysis performed is only partial, as prepositional phrase attachment or coordination are not resolved by the parser.

Using the above-mentioned modules, and choosing the Maximum-Entropy Named Entity Classifier (the fastest one), the toolkit can process a dump of the English Wikipedia (approx. 1,300,000 articles) in about two weeks on a 800 MHz PC laptop with 1 GB of RAM memory.

2.2. Terminology extraction

Traditional Term Identification systems [55,56,57] study different properties of domain-dependent texts in order to identify terms that are relevant for a particular domain. Typical approaches combine lexicosyntactic patterns and various statistical analysis of term distributions (whether a term is used uniformly throughout the text or appears in bursts), relative frequency in domain-specific texts compared to general-purpose corpora, and word association metrics to identify multiword expressions.

In our particular case, dealing with the Wikipedia, we are interested in annotating relationships between concepts that are relevant enough to possibly have their own entry in the encyclopedia. Some of these entries may be general-purpose concepts (e.g. cat or house), others are domain-specific concepts (e.g. pterodactyl or thiocyanic acid), and many others are proper names referring to particular instances of those concepts (e.g. Pterodactyl (film) or William Shakespeare). In order to extract these terms, statistical techniques based on domain-specific corpus-analysis are not very appropriate because in the Wikipedia we can find entries from any possible domain of knowledge. If we compare the relative frequency of these terms with the frequency in a general corpus, such as the British National Corpus, many of the general-purpose terms defined in the Wikipedia will not be extracted as terms, and we want to be able to annotate them with semantic information as well.

Furthermore, word-association metrics like Mutual Information may be helpful to find multiword concepts and instances, but they are not so useful to find terms that appear with a low frequency throughout the encyclopedia, such as the names of not well-known people.

```
<p id="1">
  <s id="2">
    <np appositive="yes" id="1822">
      <np det="definite" head="yes" id="3" number="singular" person="3">
        <w c="w" id="4" pos="DT">The</w>
        <w c="w" id="5" pos="JJ">Asian</w>
        <w c="w" id="6" pos="NN" stem="giant">giant</w>
        <w c="w" head="yes" id="7" pos="NN" stem="hornet">hornet</w>
      </np>
      <np head="yes" id="1820">
        <w c="brackets" id="8" pos="(">(</w>
        <np head="yes" id="9">
          <w c="w" id="10" pos="NNP" stem="Vespa">Vespa</w>
          <w c="w" id="11" pos="NN" stem="mandarinium">mandarinia</w>
        </np>
        <w c="brackets" id="12" pos=")">)</w>
      </np>
    </np>
    <w c="," id="13" pos=",">,</w>
    <np det="definite" id="14" number="singular" person="3">
      <np det="none" id="17" number="singular" person="3" case="genitive">
        <w c="w" id="15" pos="DT">the</w>
        <w c="w" head="yes" id="16" pos="NN" stem="world">world</w>
        <w c="ctr" id="19" pos="POS">'s</w>
      </np>
      <w c="w" id="20" pos="JJS">largest</w>
      <w c="w" head="yes" id="21" pos="NN" stem="hornet">hornet</w>
    </np>
    <w c="," id="22" pos=",">,</w>
    <vbar id="23" tense="finite" time="present">
      <w c="w" head="yes" id="24" lexhead="yes" pos="VBZ" stem="be">is</w>
    </vbar>
    <np det="indefinite" id="25" number="singular" person="3">
      <w c="w" id="26" pos="DT">a</w>
      <w c="w" head="yes" id="27" pos="NN" stem="native">native</w>
    </np>
    <w c="w" id="28" pos="IN">of</w>
    <np det="none" id="29" number="singular" person="3">
      <w c="w" id="30" pos="JJ">temperate</w>
      <w c="w" id="31" pos="CC">and</w>
      <w c="w" head="yes" id="32" pos="JJ">tropical</w>
      <np entity="location" id="33">
        <w c="w" id="34" pos="NNP" stem="Eastern">Eastern</w>
        <w c="w" id="35" pos="NNP" stem="Asium">Asia</w>
      </np>
    </np>
  </s>
</p>
```

Figure 2. Sample sentence with shallow syntactic information: *The Asian giant hornet (Vespa mandarinia), the world's largest hornet, is a native of temperate and tropical Eastern Asia.*

On the other hand, we can benefit from the structure of the Wikipedia to find easily most of the terms that are described in the Wikipedia and for which we can extract relations:

- The titles of the entries, after some cleanup, are good candidates to be relevant terms.

- All the links to (as yet) non-existing Wikipedia pages, because some user considered that those terms are relevant enough to have their own entries. Also, it might be possible to extract relations between the terms defined in the entries containing those links and the non-existing pages.
- All the Named Entities identified during the linguistic processing step, which are already classified as people, locations and organisations. We assume that any person, place or organisation is potentially important enough as to have its own entry in the Wikipedia.

All the terms considered are obtained from either manual annotations or the output of a Named Entity tagger, which attains an F-score of 96% for people and locations, and 92% for organisations [54]. Therefore, we can assume that the list of terms is highly precise. Even though coverage could be somewhat improved with statistical techniques for identifying domain-dependent terms, the Wikipedia is currently so large that we assume that this list is broad enough.

Next, all these terms are compared to WordNet, the lexical semantic network with which the Knowledge Base is initialised, and they are grouped in the three categories mentioned before:

- Those that appear with only one meaning both in Wikipedia (i.e. there is just one entry for them) and in WordNet (i.e. there is only one synonym set). For all of these, we assume that they are monosemous terms and they are used with the same meaning in both places, so the WordNet synset is associated in the Knowledge Base to the Wikipedia entry. The motivation for this is to assume that, if only one sense of the term has been chosen to create a Wikipedia entry and to be included in WordNet, it is probably the most salient meaning of that term, given that both Wikipedia and WordNet are general-purpose resources and are, in principle, not biased towards any particular domain.
  Previous experiments [58] considered a sample of 180 terms with one meaning both in the Simple English Wikipedia and in WordNet, and a manual evaluation indicated that they really referred to the same concept in 98.33% of the cases. For instance, an erroneous case is that of *Cheddar*. In the English Wikipedia there is just one entry called *Cheddar*, referring to the village in Sommerset, and in WordNet there is one synset, referring to the cheese. In this case it should be possible to find the error with further processing, because that WordNet synset contains a synonym, *Cheddar cheese*, which would be attached to a different Wikipedia entry.
- Those that appear with more than one meaning either in the Wikipedia or in WordNet. These are labelled as polysemous, and are later processed by the disambiguation module (Section 2.3).
- Those terms that do not appear in WordNet, which are categorised as unknown: the procedure to classify them in WordNet is described in Section 2.4.

2.3. Word Sense Disambiguation

If there are several entries in the Wikipedia with the same title, or if several WordNet synsets contain the name of an entry, it is necessary to disambiguate the meaning of the title in order to associate the entries and the synsets. Polysemous titles in the Wikipedia can be discovered because either two titles only differ by a brief explanation between

Table 1. A (non-thorough) listing of different entries in the Wikipedia about persons called John Smith.

John Smith (Ontario MP)	John Smith (Ontario MPP)	John Smith (UK politician)
John Smith (Welsh politician)	John Smith (US politician)	John Smith (Conservative politician)
John Smith (actor)	John Smith (actor 2)	John Smith (BBC)
John Smith (Clockmaker)	John Smith (comics)	John Smith (filmmaker)
John Smith (guitarist)	John Smith (Scientologist)	John Smith (mathematician)
John Smith (dentist)	John Smith (baseball player)	John Smith (footballer)
John Smith (wrestler)	John Smith (1832-1911)	John Smith (1781-1854)
John Smith (Platonist)	John Smith (missionary)	John Smith (brewer)
John Smith (Medal of Honor recipient, 1880)	John Smith (VC)	

parentheses, or because there is a disambiguation page. Table 1 shows an example of the first case: there are more than twenty entries about a person called *John Smith*, where the entry titles can be distinguished by the small explanation between parenthesis. In this case, there is also a disambiguation page listing all these entries and a few others, including *John Smith of Jamestown*, the English soldier and colony leader. In this case, there is only one synset in WordNet that contains *John Smith*, so it is necessary to contrast all the mentioned entries with the definition and semantic relations of WordNet's John Smith:

> Smith, John Smith – (English explorer who helped found the colony at Jamestown, Virginia; was said to have been saved by Pocahontas (1580-1631))

The problem of matching Wikipedia entries and WordNet synsets is a particular case of a classical problem in Natural Language Processing called *Word Sense Disambiguation* (WSD) [59]. In WSD, similarity metrics between the word to disambiguate and each candidate sense are usually used for carrying out the task. Different approaches use co-occurrence information [60], all WordNet relations [61], or just the taxonomic *is-a* relation [62], with various success rates. WordNet glosses [63] and Machine Learning algorithms [64,65,66] are also useful in calculating a semantic similarity.

Our disambiguation task is generally easier than a general WSD in unrestricted text. Because both Wikipedia entries and WordNet glosses contain term definitions, those that refer to the same entity will probably highlight the same features. It is much more difficult to discover whether *John Smith*, used in the middle of a sentence in general text, refers to any of the Wikipedia entries, than to match a particular definition of John Smith with one of the entries' definitions. That is why the disambiguation accuracy is expected to be much higher than the accuracy typically obtained by general-purpose WSD systems.

In our case, we want to find a similarity metric between encyclopedia entries and WordNet synsets. If they refer to the same concept, we can expect that there will be much in common between the two definitions. This is the reason why the approach followed is mainly a comparison between the two glosses, inspired in [67]. It consists of the following steps:

1. Represent the Wikipedia entry as a vector e using the Vector Space Model, where each dimension corresponds to a word, and the coordinate for that dimension is the frequency of the word in the entry.
2. Let $S = \{s_1, s_2, ..., s_n\}$ be the set of WordNet synsets containing the term defined in the Wikipedia entry.
3. Represent each synset s_i as the set of words in its gloss: $G_i = \{t_1, t_2, ..., t_{k_i}\}$, including their frequencies.

4. Let $N = 1$
5. Extend the sets G_i with the synonym words in each synset s_i and its hyperonyms to a depth of N levels.
6. Weight each term t in every set G_i by comparing the frequency of t in G_i with its frequency in the glosses for the other senses. In this way, a numerical vector v_i, containing the term weights, is calculated for each G_i. In the experiments, two weight functions have been tried: tf·idf and χ^2.
7. Represent each Wikipedia entry as the set E of words in its first paragraph (which is usually a short definition of the term being defined). If the length of the first paragraph is below a threshold θ, continue adding paragraphs until it exceeds it. Next, weight the terms in E using a weight function to obtain a numerical vector w_j.
8. Apply a greedy algorithm to disambiguate the Wikipedia entries and the WordNet senses: while there are entries that are not disambiguated, choose the pair (w_j, v_i) such that the similarity between w_j and v_i is the largest. Two similarity metrics between the two vectors have been tested: the dot product and the cosine, to check whether the normalisation performed by the cosine could affect the results.
   If there is a tie between two or more senses, N is incremented and the procedure goes back to step 5.

Other weight functions (e.g. the t-score) and similarity functions (e.g. the Jacquard coefficient and the Dice coefficient) are planned to be tested as future work.

In the final settings, this disambiguation was also extended with a simple procedure to identify the genus word in the definitions, both in WordNet and in the Wikipedia entry [26], using simple patterns. So, for example, if the Wikipedia entry for *John Smith of Jamestown* would have started with the sentence

> John Smith was an English explorer.

then *explorer* would have been identified as the genus word in the entry. Because the WordNet synset containing *John Smith* has as a hyperonym *explorer*, that would have been further positive evidence to associate the entry to the synset. We currently double the similarity score between an encyclopedia entry and a WordNet synset if they have the same genus word or the entry's genus word is a hyperonym of the synset.

In an experiment with 180 polysemous Wikipedia entries from the Simple English Wikipedia, the best accuracy obtained was 84% without the genus word heuristic [58], and 87% when it was included, using the dot product as similarity metric.

2.4. Classification of unknown words and Relation Extraction

The modules for classifying unknown words and for extracting relations share a common technique for learning lexicosyntactic patterns, so they are explained together in this section. Actually, classifying a word inside a taxonomy can be studied as learning taxonomic *is-a* relations between that word and the concepts in the taxonomy, so in that sense it would be a special kind of relation extraction.

The idea for classifying a new term X in an ontology is to learn patterns such as *"An X is a Y"* or *"X, a kind of Y,"* and try to disambiguate Y as some of the concepts in the ontology. If Y has a hyperlink to another entry, then it has been already disambiguated in the previous step.

Similarly, for non-taxonomic relations, if we want to learn the birth-date relation, possible patterns are *"X was born in Y at location"* or *"X (location, Y)"*.

Once these patterns have been learnt, they can be applied to the entries categorised as unknown to try to find their hyperonyms. Non-taxonomic relations between the terms can also be obtained with these patterns. The learning process is divided in three steps: *pattern extraction*, *pattern generalisation* and *pattern scoring*, described below.

2.4.1. Pattern extraction

The aim of this step is the extraction of patterns relating two concepts. The process is slightly different for hyperonymy and for non-taxonomic relations. In the case of hyperonymy relations, the strategy is to find pairs of concepts (t, f) that co-occur in the same sentence, that have already been disambiguated and that have a hyperonymy relation in WordNet. The process is the following:

1. For each term t in the Wikipedia, with an entry definition d, we select every term f such that
   - t and f co-occur in the same sentence.
   - In d there is a hyperlink pointing to the definition of f.
   - f is a hyperonym of t in WordNet.
2. Extract a context from the sentence, around f and t, including at most five words at their left-hand side, all the words in between them, and at most five words at their right. The context never jumps over sentence boundaries, which are marked with the symbols BOS (*Beginning of sentence*) and EOS (*End of sentence*).
3. The two related terms are marked as `<hook>` and `<target>`.

The condition about the hyperlink guarantees that f has already been disambiguated with respect to WordNet with a high precision.

In the case of non-taxonomic relations, the procedure is more similar to other web-based rote extractors [44,10]: for each relation, the user provides the system with a seed list of related pairs. For instance, for the relation birth year, one such pair might be *(Darwin, 1812)*. For each of these pairs, the system:

1. Submits a query to a search engine containing both elements, e.g. *Dickens AND 1812*, and downloads a number of documents to build the training corpus.
2. For any sentence in that corpus containing both elements, the system extracts a context around them in the same way as it was done for the hyperonymy relation: at most five words at each side, not crossing sentence boundaries.

The output of this step is, for each relation, a list of free-text patterns that are expected to represent it. For illustration, in the case of hyperonymy, if the entry for *Dalmatian* contains the sentence *A Dalmatian is a dog that has a white coat with black spots*, the pattern produced would be the following: `A/DT <hook> is/VBZ a/DT <target> that/IN has/VBZ a/DT white/JJ coat/NN`. Note that the words are annotated with part-of-speech tags, using the labels defined for the Penn Treebank [68], and information of Named Entity types is also encoded in the pattern. Figure 3 shows some of the patterns found for some relations.

Birth year:

BOS/BOS <hook> (/(<target> -/- number/entity)/) EOS/EOS
BOS/BOS <hook> (/(<target> -/- number/entity)/) British/JJ writer/NN
BOS/BOS <hook> was/VBD born/VBN on/IN the/DT first/JJ of/IN time_expr/entity ,/, <target> ,/, at/IN location/entity ,/, of/IN
BOS/BOS <hook> (/(<target> -/-)/) a/DT web/NN guide/NN

Birth place:

BOS/BOS <hook> was/VBD born/VBN in/IN <target> ,/, in/IN central/JJ location/entity ,/,
BOS/BOS <hook> was/VBD born/VBN in/IN <target> date/entity and/CC moved/VBD to/TO location/entity
BOS/BOS Artist/NN :/, <hook> -/- <target> ,/, location/entity (/(number/entity -/-
BOS/BOS <hook> ,/, born/VBN in/IN <target> on/IN date/entity ,/, worked/VBN as/IN

Author-book:

BOS/BOS <hook> author/NN of/IN <target> EOS/EOS
BOS/BOS Odysseus/NNP :/, Based/VBN on/IN <target> ,/, <hook> 's/POS epic/NN from/IN Greek/JJ mythology/NN
BOS/BOS Background/NN on/IN <target> by/IN <hook> EOS/EOS
did/VBD the/DT circumstances/NNS in/IN which/WDT <hook> wrote/VBD "/" <target> "/" in/IN number/entity ,/, and/CC

Capital-country:

BOS/BOS <hook> is/VBZ the/DT capital/NN of/IN <target> location/entity ,/, location/entity correct/JJ time/NN
BOS/BOS The/DT harbor/NN in/IN <hook> ,/, the/DT capital/NN of/IN <target> ,/, is/VBZ number/entity of/IN location/entity
BOS/BOS <hook> ,/, <target> EOS/EOS
BOS/BOS <hook> ,/, <target> -/- organization/entity EOS/EOS

Figure 3. Example patterns extracted from the training corpus for each several kinds of relations.

2.4.2. Pattern generalisation

In order to identify the portions in common between the patterns, and to generalise them, the following pseudo-code is applied:

1. Store all the patterns in a set $\mathcal{P}$.
2. Initialise a set $\mathcal{R}$ as an empty set.
3. While $\mathcal{P}$ is not empty,
   (a) For each possible pair of patterns, calculate the edit distance between them (allowing for three edit operations: insertion, deletion or replacement).
   (b) Take the two patterns with the smallest distance, p_i and p_j.
   (c) Remove them from $\mathcal{P}$, and add them to $\mathcal{R}$.
   (d) Obtain the generalisation of both, p_g, as described hereafter.
   (e) If p_g does not have a wildcard adjacent to the hook or the target, add it to $\mathcal{P}$.
4. Return $\mathcal{R}$

At the end, $\mathcal{R}$ contains all the initial patterns and those obtained while generalising the previous ones. The motivation for step (*e*) is that, if a pattern contains a wildcard adjacent to either the hook or the target, it will be impossible to know where the hook or the target starts or ends. For instance, when applying the pattern `<hook> is a * <target>` to a text, the wildcard prevents the system from guessing where the hyperonym starts. This procedure is similar to the one described by Pantel et al. [69].

So as to calculate the similarity between two patterns, a slightly modified version of the dynamic programming algorithm for *edit-distance* calculation [70] is used. The distance between two patterns A and B is defined as the minimum number of changes that have to be done to the first one in order to obtain the second one. The calculation is carried on by filling a matrix $\mathcal{M}$, as shown in Figure 4 (left). At the same time that we calculate the edit distance matrix, it is possible to fill in another matrix $\mathcal{D}$, in which we record which of the choices was selected at each step: insertion, deletion, replacement or no edition. This will be used later to obtain the generalised pattern. We have used the following four characters:

- `I` means an insertion in the first pattern produces the second one.
- `R` means that it is necessary to remove a token.
- `E` means that the corresponding tokens are equal, so no edition is required.
- `U` means that the corresponding tokens are unequal, so a replacement is needed.

A: `is the well known location`
B: `is the classic location`

$\mathcal{M}$	**0**	**1**	**2**	**3**	**4**
0	0	1	2	3	4
1	1	0	1	2	3
2	2	1	0	1	2
3	3	2	1	1	2
4	4	3	2	2	2
5	5	4	3	3	2

$\mathcal{D}$	**0**	**1**	**2**	**3**	**4**
0		I	I	I	I
1	R	**E**	I	I	I
2	R	R	**E**	I	I
3	R	R	R	**U**	I
4	R	R	R	**R**	U
5	R	R	R	R	**E**

Figure 4. Example of the edit distance algorithm. *A* and *B* are two word patterns; $\mathcal{M}$ is the matrix in which the edit distance is calculated, and $\mathcal{D}$ is the matrix indicating the choice that produced the minimal distance for each cell in $\mathcal{M}$.

Figure 4 shows an example for two patterns, *A* and *B*, containing respectively 5 and 4 tokens. $\mathcal{M}(5,4)$ has the value 2, indicating the distance between the two complete patterns. The two editions may be: replacing `known` by `classic`, and removing `well`.

After calculating the edit distance between two patterns *A* and *B*, we can use matrix $\mathcal{D}$ to obtain a generalised pattern, which should maintain the common tokens shared by them. The procedure used is the following:

- Every time there is an insertion or a deletion, the generalised pattern will contain a wildcard, indicating that there may be anything in between.
- Every time there is replacement, the generalised pattern will contain a disjunction of both tokens. Replacement is only allowed if the two tokens involved share the same coarse-grain part-of-speech (e.g. both are nouns, both are adjectives, etc.)
- Finally, in the positions where there is no edit operation, the token that is shared between the two patterns is left unchanged.

The patterns in the example will produce the generalised pattern

```
  is/VBD the/DT well/RB known/JJ location/entity
is/VBD the/DT     classic/JJ       location/entity
-------------------------------------------------
  is/VBD the/DT * known|classic/JJ location/entity
```

The generalisation of these two patterns produces one that can match a wider variety of sentences, so we should always take care in order not to over-generalise. A few examples of generalisations of hyperonymy patterns are the following:

```
<hook> is/VBZ a/DT <target>

<hook> is/VBZ a/DT type/NN of/IN <target>

<hook> is/VBZ a|the/DT <target> for|in|of|that/IN

A|An|The/DT <hook>/NNP is/VBZ a|the/DT <target> that/WDT
disjunction-of-verbs/VBZ

<hook> is/VBZ the/DT disjunction-of-superlative-adjectives/JJS <target>
on/IN Earth|earth/NNP
```

2.4.3. Pattern scoring: background

As can be seen in the previous examples, the patterns obtained in the previous step cover all the space between very specific and constrained patterns and very general patterns. Very general patterns may extract many results with a low precision, while very restricted patterns may have a high precision but a small recall. In this step, an estimate of the precision of each pattern is calculated. The starting point is related work [44,10] in which patterns obtained in this way are scored using the following procedure: for each pair *(hook,target)* in the seed list:

1. Download a separate corpus from the web, called *hook corpus*, using a query that contains just the hook of the relation.
2. Apply the previous patterns to the hook corpus, calculate the precision of each pattern as the number of times it identifies a target related to the hook divided by the total number of times the pattern appears.

To illustrate this process, let us suppose that we want to learn patterns to identify birth years. We may provide the system the seed pair (*Dickens*, *1812*). From the corpus downloaded for training, the system extracts sentences such as

Dickens was born in 1812
Dickens (1812 - 1870) was an English writer
Dickens (1812 - 1870) wrote Oliver Twist

The system then generates the patterns and identifies that the contexts of the last two sentences are very similar, so it also produces a generalisation of those and appends it to the list:

```
<hook> was born in <target>
<hook> ( <target> - 1870 ) was an English writer
<hook> ( <target> - 1870 ) wrote Oliver Twist
<hook> ( <target> - 1870 )
```

The system needs to estimate automatically the precision of the extracted patterns, in order to keep the best ones. So as to measure these precision values, a hook corpus would be downloaded using the hook *Dickens* as the only query word, and the system would look for appearances of the patterns in this corpus. For every occurrence in which the hook of the relation is Dickens, if the target is 1812 it will be deemed correct, and otherwise it will be deemed incorrect (e.g. in *Dickens was born in Portsmouth*).

In our initial experiments we observed initially that this procedure for calculating the precision of the patterns is unreliable in some cases. For example, the following patterns are reported by Ravichandran and Hovy [10] for identifying the relations Inventor, Discoverer and Location:

Relation	Prec.	Pattern
Inventor	1.0	<target> 's <hook> and
Inventor	1.0	that <target> 's <hook>
Discoverer	0.91	of <target> 's <hook>
Location	1.0	<target> 's <hook>

In the particular application in which they are used (relation extraction for Question Answering), they are useful because there is initially a question to be answered that indicates whether we are looking for an invention, a discovery or a location. However, if we want to apply them to unrestricted relation extraction, we have the problem that the same pattern, the genitive construction, represents all these relations, apart from the most common use indicating possession.

Relation name	Seed-list	Cardinality	Hook-type	Target-type	Web queries
birth year	birth-date.txt	n:1	entity	entity	$1 was born in $2
death year	death-date.txt	n:1	entity	entity	$1 died in $2
birth place	birth-place.txt	n:1	entity	entity	$1 was born in $2
country-capital	country-capital.txt	1:1	entity	entity	$2 is the capital of $1
author-book	author-book.txt	n:n	entity	unrestricted	$1 is the author of $2
director-film	director-film.txt	1:n	entity	unrestricted	$1 directed $2, $2 directed by $1

Table 2. Example rows in the input table for the system.

If patterns like these are so ambiguous, then why do they receive so high a precision estimate? One reason is that the patterns are only evaluated for the same hook for which they were extracted. To illustrate this with an example, let us suppose that we obtain a pattern for the relation *located-at* using the pairs *(New York, Chrysler Building)*. The genitive construction can be extracted from the context *New York's Chrysler Building*. Afterwards, when estimating the precision of this pattern, only sentences containing *<target> 's Chrysler Building* are taken into account. Because of this, most of the pairs extracted by this pattern may extract the target *New York*, apart from a few that extract the name of the architect that built it, *van Allen*. Thus we can expect that the genitive pattern will receive a high precision estimate as a *located-at* pattern.

For our purposes, however, we want to collect patterns for several relations such as *writer-book*, *painter-picture*, *director-film*, *actor-film*, and we want to make sure that the obtained patterns are only applicable to the desired relation. Patterns like *<target> 's <hook>* are very likely to be applicable to all of these relations at the same time, so we would like to be able to discard them automatically by assigning them a low precision.

Therefore, we propose the following three improvements to this procedure:

1. Collecting not only a *hook corpus* but also a *target corpus* should help in calculating the precision. In the example of the *Chrysler building*, we have seen that in most cases that we look for the pattern *'s Chrysler building* the previous words are *New York*, and so the pattern is considered accurate. However, if we look for the pattern *New York's*, we shall surely find it followed by many different terms representing different relations, and the precision estimate will decrease.
2. Testing the patterns obtained for one relation using the hook and target corpora collected for other relations. For instance, if the genitive construction has been extracted as a possible pattern for the *writer-book* relation, and we apply it to a corpus about painters, the rote extractor can detect that it also extracts pairs with painters and paintings, so that particular pattern will not be very precise for that relation.
3. Many of the pairs extracted by the patterns in the hook corpora were not evaluated at all when the hook in the extracted pair was not present in the seed lists. To overcome this, we propose to use the web to check whether the extracted pair might be correct, as shown below.

2.4.4. Pattern scoring: implementation

In our implementation, the rote extractor starts with a table containing some information about the relations for which we want to learn patterns. This procedure needs a little more information than just the seed list, which is provided as a table in the format displayed in Table 2. The data provided for each relation is the following: (a) The **name of the relation**, used for naming the output files containing the patterns; (b) the name of the

file containing the **seed list**; (c) the cardinality of the relation. For instance, given that many people can be born on the same year, but for every person there is just one birth year, the cardinality of the relation *birth year* is n:1; (d) the **restrictions** on the hook and the target. These can be of the following three categories: *unrestricted*, if the pattern can extract any sequence of words as hook or target of the relation, *Entity*, if the pattern can extract as hook or target only things of the same entity type as the words in the seed list (as annotated by the NERC module), or *PoS*, if the pattern can extract as hook or target any sequence of words whose sequence of part-of-speech labels was seen in the training corpus; and (e) a sequence of **queries** that could be used to check, using the web, whether an extracted pair is correct or not.

We assume that the system has used the seed list to extract and generalise a set of patterns for each of the relations using training corpora [10,71]. Our procedure for calculating the patterns' precision is as follows:

1. For every relation,
   (a) For every *hook*, collect a *hook corpus* from the web.
   (b) For every *target*, collect a *target corpus* from the web.
2. For every relation r,
   (a) For every pattern P, collected during training, apply it to every hook and target corpora to extract a set of pairs.
   For every pair $p = (p_h, p_t)$,
   - If it appears in the seed list of r, consider it correct.
   - If it appears in the seed list of other relations, consider it incorrect.
   - If the hook p_h appears in the seed list of r with a different target, and the cardinality is 1:1 or n:1, consider it incorrect.
   - If the target p_t appears in r's seed list with a different hook, and the cardinality is 1:1 or 1:n, consider it incorrect.
   - Otherwise, the seed list does not provide enough information to evaluate p, so we perform a test on the web. For every query provided for r, the system replaces \$1 with p_h and \$2 with p_t, and sends the query to Google. The pair is deemed correct if and only if there is at least one answer.

   The precision of P is estimated as the number of extracted pairs that are supposedly correct divided by the total number of pairs extracted.

In this step, every pattern that did not apply at least twice in the hook and target corpora is also discarded.

Example After collecting and generalising patterns for the relation *director-film*, we apply each pattern to the hook and target corpora collected for every relation. Let us suppose that we want to estimate the precision of the pattern

<target> 's <hook>

and we apply it to the hook and the target corpora for this relation and for *author-book*. Possible pairs extracted are *(Woody Allen, Bananas), (Woody Allen, Without Fears), (Charles Dickens, A Christmas Carol)*. Only the first one is correct. The rote extractor proceeds as follows:

- The first pair appears in the seed list, so it is considered correct.
- Although *Woody Allen* appears as hook in the seed list and *Without Fears* does not appear as target, the second pair is still not considered incorrect because the *directed-by* relation has n:n cardinality.

- The third pair appears in the seed list for *writer-book*, so it is directly marked as incorrect.
- Finally, because the system has not yet made a decision about the second pair, it queries Google with the sequences

  Woody Allen directed Without Fears
  Without Fears directed by Woody Allen

  Because neither of those queries provide any answer, it is considered incorrect.

In this way, it can be expected that the patterns that are equally applicable to several relations, such as *writer-book*, *director-film* or *painter-picture* will attain a low precision because they will extract many incorrect relations from the corpora corresponding to the other relations.

Finally, in the particular case of hyperonymy, patterns are scored using a similar algorithm, but considering as gold standard not an initial list of seed pairs but the relations existing in WordNet:

1. Two corpora are collected automatically from the Internet, one containing t (the *hook corpus*) and another one containing f (the *target corpus*).
2. The patterns are tested on all the hook and target corpora, and all relations involving each hook and each target are extracted.
3. To estimate the precision of each pattern, for each pair (A, B) extracted by the pattern,
   (a) If B is a hyperonym of A in WordNet, it is considered correct.
   (b) If B and A both appear in WordNet, but B is not a hyperonym of A, it is considered incorrect.
   (c) If either B or A do not appear in WordNet, a query is sent to the Google search engine for "$A\ is\ a\ B$", "$A\ is\ the\ B$" or "$A\ is\ B$". Hopefully, given the vastness of the web, if A is a hyponym of B there will be some web page containing any of those simple patterns and there will be some results. Therefore, the extracted pair is judged correct if and only if there is at least one result for any of those three queries. Previous work shows that this procedure provides a good estimate of the real accuracy of the patterns in many cases [72], and it only requires us to assume that, for every relationship, the user knows beforehand just one or two simple lexical patterns that express it.

The patterns with estimated precision levels below a user-defined threshold are discarded.

2.4.5. Evaluation of the procedure to automatically score patterns

The disambiguation of Wikipedia entries and relation extraction is still an ongoing work and has not finished yet at the time of writing this chapter. Therefore, the results provided here refer to the evaluation of non-taxonomic patterns obtained from the web using the rote extractor approach, and taxonomic patterns obtained from a subset of the Wikipedia.

Table 3 shows the number of patterns obtained for each relation. Note that the generalisation procedure applied produces new (generalised) patterns that are added to the set of original patterns, but no pattern is removed, so they all are evaluated. This is why the set of patterns increases after the generalisation. The filtering criterion was to keep the patterns that applied at least twice on the test corpus.

Relation	Seeds	Extr.	Gener.	Filt.
Birth year	244	2374	4748	30
Death year	216	2178	4356	14
Birth place	169	764	1528	28
Death place	76	295	590	6
Author-book	198	8297	16594	283
Actor-film	49	739	1478	3
Director-film	85	6933	13866	200
Painter-painting	92	597	1194	15
Employee-organisation	62	1667	3334	6
Chief of state	55	1989	3978	8
Soccer player-team	194	4259	8518	39
Soccer team-city	185	180	360	0
Soccer team-manager	43	994	1988	9
Country/region-capital city	222	4533	9066	107
Country/region-area	226	762	1524	2
Country/region-population	288	318	636	3
Country-bordering country	157	6828	13656	240
Country-inhabitant	228	2711	5422	17
Country-continent	197	1606	3212	21
Hyperonymy	0	270	1131	22

Table 3. Number of seed pairs for each relation, and number of unique patterns in each step.

Concerning the computational complexity, for each pair $(hook, target)$ we need a corpus for each hook and a corpus for each target. In the case of birth year, that means 488 corpora in total. In our experiments we have downloaded a maximum of 500 documents per corpus, so the total number of documents is less or equal than 244,000, on which the 4,748 patterns obtained after the generalisation step are evaluated. This procedure required less than two days for each relationship on a Pentium IV laptop computer (1 GHz, 2 GB RAM).

It is interesting to see that for most relations the reduction of the pruning is very drastic. This is because of two reasons. Firstly, most patterns are far too specific, as they include up to 5 words at each side of the hook and the target, and all the words in between. Only those patterns that have generalised very much, substituting large portions with wildcards or disjunctions are likely to apply to the sentences in the hook and target corpora. Secondly, the samples of the hook and target corpora used are too small for some of the relations to apply, so few patterns apply more than twice.

Concerning the precision estimates, a full evaluation is provided for the *birth-year* relation. Table 4 shows in detail the thirty patterns obtained. It can also be seen that some of the patterns with good precision contain the wildcard *. For instance, the first pattern indicates that the presence of any of the words *biography*, *poetry*, etc. anywhere in a sentence before a person name and a date or number between parenthesis is a strong indication that the target is a birth year.

The last columns in the table indicate the number of times that each rule applied in the hook and target corpora, and the precision of the rule in each of the following cases:

- As estimated by the modified precision calculation, complete with target and hook corpora, cardinality and web queries (Prec1).
- As estimated by the traditional hook corpus approach (Prec2). Here, cardinality is not taken into account, patterns are evaluated only on the hook corpora from the same relation, and those pairs whose hook is not in the seed list are ignored.
- The real precision of the rule (real). In order to obtain this metric, two different annotators evaluated the pairs applied independently, and the precision was

No.	Pattern	Applied	Prec1	Prec2	Real
1	Biography|Hymns|Infography|Life|Love|POETRY|Poetry| Quotations|Search|Sketch|Woolf|charts|genius|kindness| poets/NN */* OF|Of|about|by|for|from|like|of/IN <hook> (/(<target> -/-	6	1.00	1.00	1.00
2	"/" <hook> (/(<target> -/-	4	1.00	1.00	1.00
3	[BOS]/[BOS] <hook> was/VBD born/VBN about|around|in/IN <target> B.C.|B.C.E|BC/NNP at|in/IN	3	1.00	1.00	1.00
4	[BOS]/[BOS] <hook> was/VBD born/VBN about|around|in/IN <target> B.C.|B.C.E|BC/NNP at|in/IN location/entity	3	1.00	1.00	1.00
5	[BOS]/[BOS] <hook> was/VBD born/VBN around/IN <target> B.C.E/NNP at/IN location/entity ,/, a/DT	3	1.00	1.00	1.00
6	[BOS]/[BOS] <hook> was/VBD born/VBN around|in/IN <target> B.C.|B.C.E/NNP at|in/IN location/entity ,/,	3	1.00	1.00	1.00
7	[BOS]/[BOS] */* ATTRIBUTION|Artist|Author|Authors| Composer|Details|Email|Extractions|Myth|PAL|Person| Quotes|Title|Topic/NNP :/, <hook> (/(<target> -/-	3	1.00	1.00	1.00
8	classical/JJ playwrights/NNS of/IN organisation/entity ,/, <hook> was/VBD born/VBN near/IN location/entity in/IN <target> BCE/NNP ,/, in/IN the/DT village/NN	3	1.00	1.00	1.00
9	[BOS]/[BOS] <hook> (/(<target> -/-)/)	2	1.00	1.00	1.00
10	[BOS]/[BOS] <hook> (/(<target> -|-/-)/)	2	1.00	1.00	1.00
11	[BOS]/[BOS] <hook> (/(<target> person/entity BC/NNP ;/, Greek/NNP :/,	2	1.00	1.00	1.00
12	ACCESS|AND|Alice|Author|Authors|BY|Biography|CARL| Dame|Don|ELIZABETH|(...)|web|writer|writerMuriel| years/NNP <hook> (/(<target> -|- -/-	8	0.75	1.00	
13	-/- <hook> (/(<target> -/-	3	0.67	1.00	0.67
14	-|-/- <hook> (/(<target> -/-	3	0.67	1.00	0.67
15	[BOS]/[BOS] <hook> (/(<target> -/-	60	0.62	1.00	0.81
16	[BOS]/[BOS] <hook> (/(<target> -/- */*)/)	60	0.62	1.00	0.81
17	[BOS]/[BOS] <hook> (/(<target> -|-/-	60	0.62	1.00	0.81
18	,|:/, <hook> (/(<target> -/-	32	0.41	0.67	0.28
19	[BOS]/[BOS] <hook> ,/, */* (/(<target> -|-/-	15	0.40	1.00	0.67
20	,|:|;/, <hook> (/(<target> -|-/-	34	0.38	0.67	0.29
21	AND|Alice|Authors|Biography|Dame|Don|ELIZABETH|Email| Fiction|Frances|GEORGE|Home|I.|Introduction|Jean|L|Neben| PAL|PAULA|Percy|Playwrights|Poets|Sir|Stanisaw|Stanislaw| W.|WILLIAM|feedback|history|writer/NNP <hook> (/(<target> -/-	3	0.33	n/a	0.67
22	AND|Frances|Percy|Sir/NNP <hook> (/(<target> -/- Alice|Authors|Biography|Dame|Don|ELIZABETH|Email| Fiction|Frances|GEORGE|Home|I.|Introduction|Jean|	3	0.33	n/a	0.67
23	L|Neben|PAL|PAULA|Percy|Playwrights|Poets|Sir|Stanisaw| Stanislaw|W.|WILLIAM|feedback|history|writer/NN <hook> (/(<target> -/-	3	0.33	n/a	0.67
24	[BOS]/[BOS] <hook> ,|:/, */* ,|:/, <target> -/-	7	0.28	0.67	0.43
25	[BOS]/[BOS] <hook> ,|:/, <target> -/-	36	0.19	1.00	0.11
26	[BOS]/[BOS] <hook> ,/, */* (/(<target>)/)	20	0.15	0.33	0.10
27	[BOS]/[BOS] <target> <hook> ,/,	18	0.00	n/a	0.00
28	In|On|on/IN <target> ,/, <hook> grew|was/VBD	17	0.00	0.00	0.00
29	In|On|on/IN <target> ,/, <hook> grew|was|went/VBD	17	0.00	0.00	0.00
30	[BOS]/[BOS] <hook> ,/, */* DE|SARAH|VON|dramatist|novelist| playwright|poet/NNP (/(<target> -/-	3	0.00	n/a	1.0
	TOTAL	436	0.46	0.84	0.54

Table 4. Patterns for the relation *birth year*, results extracted by each, precision estimated with this procedure and with the traditional hook corpus approach, and precision evaluated by hand.

estimated from the pairs in which they agreed (there was a 96.29% agreement, Kappa=0.926).

As can be seen, in most of the cases our procedure produces lower precision estimates. If we calculate the total precision of all the rules altogether, shown in the last row of the table, we can see that, with the hook corpus approach, the whole set of rules would be considered to have a total precision of 0.84, while that estimate decreases sharply

Relation	Prec1	Prec2	Real
Birth year	0.46 [0.41,0.51]	0.84 [0.81,0.87]	0.54 [0.49,0.59]
Death year	0.29 [0.24,0.34]	0.55 [0.41,0.69]	0.38 [0.31,0.44]
Birth place	0.65 [0.62,0.69]	0.36 [0.29,0.43]	0.84 [0.79,0.89]
Death place	0.82 [0.73,0.91]	1.00 [1.00,1.00]	0.96 [0.93,0.99]
Author-book	0.07 [0.07,0.07]	0.26 [0.19,0.33]	0.03 [0.00,0.05]
Actor-film	0.07 [0.01,0.13]	1.00 [1.00,1.00]	0.02 [0.00,0.03]
Director-film	0.03 [0.03,0.03]	0.26 [0.18,0.34]	0.01 [0.00,0.01]
Painter-painting	0.10 [0.07,0.12]	0.35 [0.23,0.47]	0.17 [0.12,0.22]
Employee-organisation	**0.31 [0.22,0.40]**	1.00 [1.00,1.00]	0.33 [0.26,0.40]
Chief of state	**0.00 [0.00,0.00]**	-	0.00 [0.00,0.00]
Soccer player-team	**0.07 [0.06,0.08]**	1.00 [1.00,1.00]	0.08 [0.04,0.12]
Soccer team-city	-	-	-
Soccer team-manager	0.61 [0.53,0.69]	1.00 [1.00,1.00]	0.83 [0.77,0.88]
Country/region-capital city	**0.12 [0.11,0.13]**	0.23 [0.22,0.24]	0.12 [0.07,0.16]
Country/region-area	**0.09 [0.00,0.19]**	1.00 [1.00,1.00]	0.06 [0.02,0.09]
Country/region-population	**1.00 [1.00,1.00]**	**1.00 [1.00,1.00]**	1.00 [1.00,1.00]
Country-bordering country	**0.17 [0.17,0.17]**	1.00 [1.00,1.00]	0.15 [0.10,0.20]
Country-inhabitant	**0.01 [0.00,0.01]**	0.80 [0.67,0.93]	0.01 [0.00,0.01]
Country-continent	0.16 [0.14,0.18]	0.07 [0.04,0.10]	0.00 [0.00,0.01]

Table 5. Precision estimates for the whole set of extracted pairs by *all* rules and all relations.

to 0.46 with our modified precision estimate. This value is nearer the precision of 0.54 evaluated by hand. Note that the precision estimated by the new procedure is even lower than the real precision of the patterns, as measured by hand, due to the fact that the web queries consider unknown pairs as incorrect unless they appear in the web exactly in the format of the query in the input table. Specially for not very well-known people, we cannot expect that all of them will appear in the web following the pattern *"X was born in date"*, so the web estimates tend to be over-conservative.

Table 5 shows the precision estimates for every pair extracted with all the rules using both procedures, with 0.95 confidence intervals, for the non-taxonomic relations. The real precision has been estimated by sampling randomly 200 pairs and evaluating them by hand, as explained above for the *birth year* relation. As can be observed, the precision estimate of the whole set of rules for each relation is not statistically dissimilar to the real precision in many more cases than using the hook corpus approach. Please note as well that the precisions indicated in the table refer to all the pairs extracted by all the rules, some of which are very precise, but some of which are very imprecise. If the rules are to be applied in an annotation system, only those with a high precision estimate would be used, and expectedly much better overall results would be obtained.

It is important to mention that the patterns with low precision estimates are discarded in the current system, even though they also extract many correct relations. We leave for future work the application of those patterns, maybe with a post-processing step in which all the extracted instances are evaluated and ranked.

2.4.6. Evaluation of all the patterns on the Wikipedia

As indicated above, the experiments on the whole English Wikipedia are still work in progress, but this section aims at providing some preliminary results on a subset of the same that includes 20,075 entries [5]. Table 6 shows the number of results (pairs of related terms) that the whole set of the pattern sets has extracted, and the precision attained. This precision has been estimated by correcting manually at least 50 results from each relationship. Note that, in this preliminary experiment, we have included all the pat-

Relation	No. of results	Precision
Birth-year	15746	74.14%
Death-year	5660	90.20%
Birth-place	154	27.27%
Actor-film	4	50.00%
Country-Chief of state	272	50.00%
Writer-book	179	37.29%
Country-capital	825	11.45%
Player-team	315	7.75%

Table 6. Number of patterns obtained for each relationship, number of results extracted by each pattern set, and precision.

terns, regardless of the precision estimated for them, so very general patterns such as the genitive construction are included.

As can be seen, the precision for birth year and death year is very good, because they are usually expressed with very fixed patterns, and years and dates are entities that are very easily recognised. The few errors are mainly due to the following two cases:

- Named Entity tagging mistakes, e.g. a TV series mistagged as a person, where the years in which it has been shown are taken as birth and death date.
- Names of persons that held a title (e.g. king or president) during a period of time, that is mistakenly considered their life span.

On the other hand, as expected, the other examples have proven more difficult to identify. We have observed the problem, mentioned in the previous sections, that some patterns are applicable for many kinds of relationships at the same time. This phenomenon is specially relevant in the case of the *player-team* relation. The precision of the patterns is 92% when they are applied only to the entries about soccer players, but the figure falls down to 7.75% when applied to the whole Wikipedia corpus collected. This means that they are patterns that, in the domain of soccer, usually indicate the relationship between the player and its club, but in other contexts they may be conveying a different meaning. One of these patterns is the already mentioned genitive construction. In sports articles, when this construction is found between an organisation and a person is usually expressing the *player-team* relation, as in *Liverpool's Fowler*. But it also extracted many wrong pairs from documents belonging to different topics.

The same also applies to the case of countries and capitals. During training, from phrases such as *Spain's Madrid*, the system extracted the genitive construction as indicating a relationship of capitality, but it is a source of errors because it can also express a part-of relationship between a country and any of its cities.

In the case of *actor-film*, we have observed that in the actors' entries in the Wikipedia, there is usually a section containing all the filmography, expressed as an HTML bullet list. In this way, because the information is already semi-structured, the textual patterns cannot apply. It should be easier to extract that data using other simpler procedures that take benefit of the structure of the entry.

2.5. Classification of subconcepts and instances

Following [73], the terms identified in the Wikipedia and classified inside WordNet are classified as either instances or subclasses of each of their hyperonyms. In order to do that, we apply a Maximum Entropy model using features such as the Named Entity type of the term, whether it is used or not with determiners or whether it can be seen in plural or in singular number throughout the Wikipedia. Because the current version of WordNet already contains this information for the existing synsets, this step is aimed at completing the information of the newly learnt concepts according to the structure of the lexical semantic network. This work is still ongoing.

3. Discussion and conclusions

We have described here a new procedure for the automatic semantic annotation of the Wikipedia, which is underpinned by previous works concerning: (a) the automatic association of Wikipedia entries with nodes in a lexical semantic network such as WordNet [58], (b) the automatic generation of patterns for extracting taxonomic [74] and non-taxonomic relations [71], and (c) the classification of instances and concepts based on surface linguistic clues [73]. The architecture described here shows a concrete application of all the previous techniques in order to attain a common goal: the semantic annotation of a large textual corpus such as the Wikipedia.

Concerning the automatic disambiguation of the Wikipedia entries, we are currently disambiguating the whole of the English Wikipedia (as of June 2006) with respect to WordNet. We believe that the output of this process may have many applications apart from a direct semantic annotation of the Wikipedia, so when it is finished, we intend to make public the results to the research community.

Concerning the pattern learning procedures for relation extractions, we show that (a) the procedure described is able to generate generalised patterns containing wildcards; (b) it makes use of PoS and Named Entity tags during the generalisation process; and (c) several relations are learnt and evaluated at the same time, in order to test each one on the test corpora built for the others. The precision estimates of the generated patterns are not statistically significantly dissimilar to the real precision of the patterns, which allows us to tune the accuracy of the system according to the user preferences.

Concerning future work, we intend to finish the full processing of the Wikipedia, and, in particular, the application of the patterns for the relationships that have not been evaluated yet on it, including the patterns for hyponymy. We are also trying to improve the estimation of the patterns accuracy for the pruning step. We plan explore ways in which to use the pairs extracted by low precision patterns (which are not used in the current implementation), as some of the results that this kind of rules extracts are also useful [41,42].

Acknowledgements

This work has been partially funded by the Spanish Ministry of Education, project number TIN-2005-06885.

References

[1] S. Sekine. On-demand information extraction. In *Proceedings of the twenty-first International Conference on Computational Linguistics and forty-fourth Annual Meeting of the association of computational linguistics (ACL-2006)*, pages 731–738, 2006.

[2] M. E. Califf, M. A. Greenwood, M. Stevenson, and R. Yangarber. *Information Extraction Beyond the Document, Proceedings of the ACL06 workshop*. The Association of Computational Linguistics, 2006.

[3] R. Bunescu and M. Pasca. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL06)*, pages 9–16, Trento, Italy, 2006.

[4] Jussi Karkgren. *New text–Wikis and blogs and other dynamic text sources. Proceedings of the EACL06 workshop*. The Association of Computational Linguistics, Trento, Italy, 2006.

[5] M. Ruiz-Casado, E. Alfonseca, and P. Castells. From wikipedia to semantic relationships: a semi-automated annotation approach. In *Proceedings of the First Workshop on Semantic Wikis – From Wiki To Semantics*, 2006.

[6] J. Kim, E. Shaw, D. Feng, C. Beal, and E.H. Hovy. Modeling and assessing student activities in online discussions. In *Proceedings of the Workshop on Educational Data Mining at the conference of the American Association of Artificial Intelligence (AAAI-06)*, Boston, MA., 2006.

[7] P. Castells, M. Fernandez, and D. Vallet. An adaptation of the vector-space model for ontology-based information retrieval. *IEEE Transactions on Knowledge and Data Engineering*, 19(2):261–272, 2007.

[8] J. Rech, B. Decker, and E. Ras. *Emerging Technologies for Semantic Work Environments: Techniques, Methods, and Applications*. Idea Group Inc., in press, 2008.

[9] K.C. Litkowski. Question-Answering Using Semantic Relation Triples. In *Proceedings of the 8th Text Retrieval Conference (TREC-8)*, pages 349–356, 1999.

[10] D. Ravichandran and E. Hovy. Learning surface text patterns for a question answering system. In *Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL-2002)*, pages 41–47, 2002.

[11] B. Katz and J. Lin. Selectively using relations to improve precision in question answering. In *Proceedings of the EACL-2003 Workshop on Natural Language Processing for Question Answering*, 2003.

[12] M. Völkel, M. Krötzsch, D. Vrandecic, H. Haller, and R. Studer. Semantic wikipedia. In *Proceedings of the 15th International Conference on World Wide Web*, pages 585–594, Edinburgh, Scotland, 2006. ACM Press.

[13] N. Kushmerick. *Wrapper Induction for Information Extraction*. PhD thesis, University of Washington, 1997.

[14] D. Florescu, A. Levy, and A. Mendelzon. Database techniques for the World-Wide Web: a survey. *ACM SIGMOD Record*, 27(3):59–74, 1998.

[15] F. Viégas, M. Wattenberg, and D. Kushal. Studying cooperation and conflict between authors with history flow visualization. In *Proceedings of the 2004 conference on Human factors in computing systems*, pages 575–582. ACM Press New York, NY, USA, 2004.

[16] A. Lih. Wikipedia as participatory journalism: Reliable sources? Metrics for evaluating collaborative media as a news resource. In *Proceedings of the 5th International Symposium on Online Journalism*, 2004.

[17] A. Toral and R. Muñoz. A proposal to automatically build and maintain gazetteers for named entity recognition by using wikipedia. In *Proceedings of the workshop on New Text—wikis and blogs and other dynamic text sources*, pages 56–61, 2006.

[18] S. F. Adafre and M. de Rijke. Finding similar sentences across multiple languages in wikipedia. In *Proceedings of the workshop on New Text—wikis and blogs and other dynamic text sources*, pages 56–61, 2006.

[19] D. Ahn, V. Jijkoun, G. Mishne, K. Muller, M. de Rijke, and S. Schlobach. Using wikipedia at the TREC QA track. In *E.M. Voorhees and L.P. Buckland (eds.), The Thirteenth Text Retrieval Conference (TREC 2004)*, 2004.

[20] M. Strube and S.P. Ponzetto. WikiRelate! Computing semantic relatedness using Wikipedia. *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, pages 1419–1424, 2006.

[21] A. Herbelot and A. Copestake. Acquiring ontological relationships from wikipedia using RMRS. In *Proceedings of the Web Content Mining with Human Language Technologies workshop*, Athens, Georgia, U.S.A., 2006.

[22] F. M. Suchanek, G. Ifrim, and G. Weikum. LEILA: Learning to extract information by linguistic analysis. In *Proceedings of the 2nd Workshop on Ontology Learning and Population: Bridging the Gap between Text and Knowledge*, pages 18–25. Association for Computational Linguistics, 2006.

[23] S. Suh, H. Halpin, and E. Klein. Extracting common sense knowledge from wikipedia. In *Proceedings of the Web Content Mining with Human Language Technologies workshop*, Athens, Georgia, U.S.A., 2006.

[24] Y. Wilks, D. C. Fass, C. Ming Guo, J. E. McDonald, T. Place, and B. M. Slator. Providing machine tractable dictionary tools. In *James Pustejovsky (ed.) Semantics and the Lexicon*, pages 341–401. Kluwer Academic Publisher, 1993.

[25] Y. A. Wilks, B. M. Slator, and L. M. Guthrie. *Electric words: Dictionaries, computers and meanings*. Cambridge, MA: MIT Press, 1996.

[26] G. Rigau. *Automatic Acquisition of Lexical Knowledge from MRDs*. PhD Thesis, Departament de Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya, 1998.

[27] S. Harabagiu and D. I. Moldovan. Knowledge processing on an extended wordnet. In *WordNet: An Electronic Lexical Database*, pages 379–405. MIT Press, 1998.

[28] A. Novischi. Accurate semantic annotation via pattern matching. In *Proceedings of the Florida Artificial Intelligence Research Society (FLAIRS-2002)*, pages 375–379, 2002.

[29] M. DeBoni and S. Manandhar. Automated discovery of telic relations for wordnet. In *Poceedings of the First International Conference on General WordNet*, Mysore, India, january 2002.

[30] D. Faure and C. Nédellec. A corpus-based conceptual clustering method for verb frames and ontology acquisition. In *LREC workshop on Adapting lexical and corpus resources to sublanguages and applications*, pages 707–728, Granada, Spain, 1998.

[31] S. A. Caraballo. Automatic construction of a hypernym-labeled noun hierarchy from text. In *Proceedings of the thirty-seventh Annual Meeting of the Association for Computational Linguistics*, pages 120–126, University of Maryland, College Park, Maryland, U.S.A., 1999.

[32] F. Pereira, N. Tishby, and L. Lee. Distributional clustering of english words. In *Proceedings of the thirty-first Annual Meeting of the Association for Computational Linguistics*, pages 183–190, Columbus, Ohio, 1999.

[33] E. Alfonseca and S. Manandhar. Extending a lexical ontology by a combination of distributional semantics signatures. In *Knowledge Engineering and Knowledge Management*, volume 2473 of *Lecture Notes in Artificial Intelligence*, pages 1–7. Springer Verlag, 2002.

[34] M.A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics*, volume 2, pages 539–545. Association for Computational Linguistics Morristown, NJ, USA, 1992.

[35] M. A. Hearst. *Automated Discovery of WordNet Relations. In Christiane Fellbaum (Ed.) WordNet: An Electronic Lexical Database*, pages 132–152. MIT Press, 1998.

[36] M. Berland and E. Charniak. Finding parts in very large corpora. In *Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL-99)*, 1999.

[37] J. Kietz, A. Maedche, and R. Volz. A method for semi-automatic ontology acquisition from a corporate intranet. In *Workshop "Ontologies and text"*, 2000.

[38] M. Finkelstein-Landau and E. Morin. Extracting semantic relationships between terms: supervised vs. unsupervised methods. In *Workshop on Ontologial Engineering on the Global Info. Infrastructure*, 1999.

[39] S. Brin. Extracting patterns and relations from the World Wide Web. In *Proceedings of the WebDB Workshop at the 6th International Conference on Extending Database Technology EDBT'98*, pages 172–183, 1998.

[40] E. Agichtein and L. Gravano. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the fifth ACM Conference on Digital Libraries*, pages 85–94, 2000.

[41] R. Girju, A. Badulescu, and D. Moldovan. Learning semantic constraints for the automatic discovery of part-whole relations. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, volume 1, pages 1–8, 2003.

[42] P. Pantel and M. Pennacchiotti. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of Conference on Computational Linguistics / Association for Computational Linguistics (COLING/ACL-06)*, pages 113–120, Sydney, Australia, 2006.

[43] G. S. Mann and D. Yarowsky. Unsupervised personal name disambiguation. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, volume 4, pages 33–40. Association for Computational Linguistics Morristown, NJ, USA, 2003.

[44] G. S. Mann and D. Yarowsky. Multi-field information extraction and cross-document fusion. In *Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL 2005)*, 2005.

[45] P. Pantel and M. Pennacchiotti. Automatically harvesting and ontologizing semantic relations. In P. Buitelaar and P. Cimiano, editors, *Bridging the Gap between Text and Knowledge - Selected Contributions to Ontology Learning and Population from Text*. IOS Press, 2007. this volume.

[46] S. Blohm, P. Cimiano, and E. Stemle. Harvesting relations from the web -quantifiying the impact of filtering functions. In *Proceedings of the 22nd Conference on Artificial Intelligence (AAAI-07)*, pages 1316–1323, 2007.

[47] V. S. Uren, P. Cimiano, J. Iria, S. Handschuh, M. Vargas-Vera, E. Motta, and F. Ciravegna. Semantic annotation for knowledge management: Requirements and a survey of the state of the art. *Journal of Web Semantics*, 4(1):14–28, 2006.

[48] A. Maedche and S. Staab. Semi-automatic engineering of ontologies from text. In *Proceedings of the 12th Internal Conference on Software and Knowledge Engineering*, Chicago, USA, 2000.

[49] P. Buitelaar and Michael M. Sintek. OntoLT version 1.0: Middleware for ontology extraction from text. In *Proc. of the Demo Session at the International Semantic Web Conference*, 2004.

[50] B. Popov, A. Kiryakov, D. Ognyanoff, D. Manov, and A. Kirilov. Kim - a semantic platform for information extaction and retrieval. *Journal of Natural Language Engineering*, 10(3–4):375–392, 2004.

[51] P. Cimiano, G. Ladwig, and S. Staab. Gimme' the context: Context-driven semantic annotation with c-pankow. In *Proceedings of the 14th. International World Wide Web Conference*, pages 332–341, Chiba, Japan, 2005.

[52] P. Cimiano and J. Völker. A framework for ontology learning and data-driven change discovery. *Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems (NLDB 2005), Alicante*, pages 15–17, 2005.

[53] G. A. Miller. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995.

[54] E. Alfonseca, A. Moreno-Sandoval, J. M. Guirao, and M. Ruiz-Casado. The wraetlic NLP suite. In *Proceedings of the Language Resources and Evaluation Conference (LREC-2006)*, 2006.

[55] J. S. Justeson and S. M. Katz. Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, 1(1):9–27, 1995.

[56] P. Pantel and D. Lin. A statistical corpus-based term extractor. In *Proceedings of the 14th Biennial Conference of the Canadian Society on Computational Studies of Intelligence*, pages 36–46, London, UK, 2001. Springer-Verlag.

[57] M. T. Cabré, R. Estopá, and J. Vivaldi. Automatic term detection: a review of current systems. In *Recent advances in computational terminology, volume 2 of Natural Language Processing*, pages 53–87. John Benjamins, 2001.

[58] M. Ruiz-Casado, E. Alfonseca, and P. Castells. Automatic assignment of wikipedia encyclopedic entries to wordnet synsets. In *Proceedings of the Atlantic Web Intelligence Conference, AWIC*, pages 380–386. Springer, 2005.

[59] N. Ide and J. Véronis. Introduction to the special issue on word sense disambiguation: the state of the art. *Computational Linguistics*, 24:1–40, 1998.

[60] C. D. Manning and H. Schütze. *Foundations of statistical Natural Language Processing*. MIT Press, 2001.

[61] G. Hirst and D. St-Onge. Lexical chains as representations of context for the detection and correction of malapropisms. In *WordNet: an electronic lexical database*. MIT Press, 1998.

[62] P. K. Resnik. Disambiguating noun groupings with respect to wordnet senses. In *Proceedings of the Third Workshop on Very Large Corpora*, pages 54–68, Somerset, 1995. ACL.

[63] R. Mihalcea and D. Moldovan. A method for word sense disambiguation of unrestricted text. In *Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL'99)*, Maryland, NY, 1999.

[64] C. Grozea. Finding optimal parameter settings for high performance word sense disambiguation. In *Proceedings of the Senseval-3: The third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, Barcelona, Spain, 2004.

[65] Y. Keok Lee, H. Tou Ng, and T. Kiah Chia. Supervised word sense disambiguation with support vector machines and multiple knowledge sources. In *Proceedings of the Senseval-3: The third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 137–140, Barcelona, Spain, 2004.

[66] C. Strappavara, A. Gliozzo, and C. Giuliano. Pattern abstraction and term similarity for word sense disambiaguation: IRST at senseval-3. In *Proceedings of the Senseval-3: The third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 229–234, Barcelona, Spain, 2004.

[67] M. Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th International Conference on Systems Documentation*, pages 24–26, 1986.

[68] M. Marcus, B. Santorini, and M.A. Marcinkiewicz. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[69] P. Pantel, D. Ravichandran, and E. Hovy. Towards terascale knowledge acquisition. In *Proceedings of Conference on Computational Linguistics (COLING-04)*, pages 771–777, Geneva, Switzerland, 2004.

[70] R. Wagner and M. Fischer. The string-to-string correction problem. *Journal of Association for Computing Machinery*, 21(1):168–173, 1974.

[71] E. Alfonseca, P. Castells, M. Okumura, and M. Ruiz-Casado. A rote extractor with edit distance-based generalisation and multi-corpora precision calculation. In *Poster session of the Annual Meeting of the Association of Computational Linguistics ACL-2006*, 2006.

[72] E. Alfonseca, M. Ruiz-Casado, M. Okumura, and P. Castells. Towards large-scale non-taxonomic relation extraction: Estimating the precision of rote extractors. In *Proceedings of the Ontology Learning and Population workshop at ACL-2006*, 2006.

[73] E. Alfonseca and S. Manandhar. Distinguishing instances and concepts in wordnet. In *Poceedings of the First International Conference on General WordNet*, Mysore, India, 2002.

[74] M. Ruiz-Casado, E. Alfonseca, and P. Castells. Automatic extraction of semantic relationships for wordnet by means of pattern learning from wikipedia. In *Natural Language Processing and Information Systems*, volume 3513 of *LNCS*, pages 67–79. Springer, 2005.

Automatically Harvesting and Ontologizing Semantic Relations

Patrick PANTEL [a] and Marco PENNACCHIOTTI [b]
[a] *Information Sciences Institute,University of Southern California, 4676 Admiralty Way, Marina del Rey, CA 90292 pantel@isi.edu*
[b] *Dept. of Computational Linguistics, Saarland University, Germany pennacchiotti@coli.uni-sb.de*

Abstract. With the advent of the Web and the explosion of available textual data, it is key for modern natural language processing systems to access, represent and reason over large amounts of knowledge in semantic repositories. Separately, the knowledge representation and natural language processing communities have been developing representations/engines for reasoning over knowledge and algorithms for automatically harvesting knowledge from textual data, respectively. There is a pressing need for collaboration between the two communities to provide large-scale robust reasoning capabilities for knowledge rich applications like question answering. In this chapter, we propose one small step by presenting algorithms for harvesting semantic relations from text and then automatically linking the knowledge into existing semantic repositories. Experimental results show better than state of the art performance on both relation harvesting and ontologizing tasks.

Keywords. knowledge acquisition, relation extraction, ontology learning

1. Introduction

With the advent of the Web and the explosion of available textual data, it is key for modern Natural Language Processing (NLP) systems to access, represent and reason over large amounts of knowledge contained in semantic repositories.

Separately, the knowledge representation (KR) community has developed many formal ontologies for use in various reasoning tasks such as planning and theorem proving, and the natural language processing (NLP) community has developed several algorithms for automatically harvesting knowledge from textual resources. Most mined resources from NLP consist of very large but noisy and unstructured knowledge, making their use in KR reasoning engines futile. Knowledge rich applications such as question answering and information extraction would benefit greatly from the reasoning power of the KR community and the breadth of knowledge extracted from the NLP community. There is therefore a pressing need for the NLP community to not only harvest knowledge from text, but also to link this knowledge into semantic repositories over which KR reasoning engines can execute.

In this chapter, we present algorithms for both extracting semantic relations from textual resources and for linking, or *ontologizing*, them into a semantic repository.

1.1. Exploiting Knowledge Resources

Recent attention to knowledge-rich problems such as question answering [1] and textual entailment [2] has encouraged natural language processing researchers to develop algorithms for automatically harvesting semantic resources. With seemingly endless amounts of textual data at our disposal, we have a tremendous opportunity to automatically grow semantic term banks and ontological resources.

Knowledge resources can be mainly divided in two types: *textual resources* and *structured resources*. *Textual resources* include linguistic text collections, ranging from large generic repositories such as the Web to specific domain texts such as collections of texts or books on specific subjects. These repositories contain a large and ever growing amount of information expressed *implicitly* in natural language texts. These resources greatly vary in size, from the terabytes of data on the Web to the kilobytes of textual material in electronic books. Structured resources consist of repositories in which knowledge is *explicit* and organized in lists or graphs of entities. In contrast with textual resources, structured resources are used to explicitly represent domain and generic knowledge, making their inherent knowledge directly usable in applications. *Structured resources* vary largely on their degree of internal structuring, and can be accordingly divided in two different classes: *semantic repositories* and *lexical resources*. The first class is formed by highly structured resources that usually organize knowledge at a *conceptual* level (e.g., concepts, relations among concepts, situation types) or at a *sense* level (word senses and relations among senses). Ontologies such as Mikrokosmos [3,4], DOLCE [5] and SUMO [6], and situation repositories such as FrameNet [7] are good examples of the former, while WordNet [8] is an example of the latter. *Lexical resources* are less structured resources such as thesauri, lists of facts, lexical relation instances, lists of paraphrases, and other flat lists of lexical objects. These resources usually organize knowledge at a pure *lexical* level, and are in most cases built by using automatic or semi-automatic techniques.

Two main issues must be addressed in order to use knowledge resources in applications: extract the implicit knowledge in textual resources (*knowledge harvesting*), and make the knowledge of both textual and structured resource usable (*knowledge exploitation*).

Regarding *knowledge harvesting*, harvesting algorithms are used to analyze textual repositories and extract knowledge in the form of lexical resources. NLP researchers have developed many algorithms for mining knowledge from text and the Web, including facts [9], semantic lexicons [10], concept lists [11], and word similarity lists [12]. Many recent efforts have also focused on extracting binary semantic relations between entities, such as entailments [13], *is-a* [14], *part-of* [15], and other relations. Relational knowledge is in fact crucial in many applications. Unfortunately, most relation extraction algorithms suffer from many limitations. First, they require a high degree of supervision. Secondly, they are usually limited in breadth (they cannot be easily applied to different corpus sizes and domains) and generality (they can harvest only specific types of relations).

So far, little attention has been spent on the issue of *knowledge exploitation*. As Bos [16] outlined, whilst lexical resources are potentially useful, their successful use in applications has been very limited due to a variety of problems. For example, question answering (QA) systems based on logical proving could in theory improve their performance by simply exploring knowledge in lexical resources which have been acquired in-

dipendently and semantic repositories. For instance, suppose a QA system must answer the following question:

"When did James Dean die?"

Suppose the system could rely on a lexical resource formed by a list of entailment rules. The lexical resource could contain the entailment $kill(X, Y) \rightarrow die(Y)$. The system could then answer *"1955"*, by examining the Web and finding the snippet *"In 1955, actor James Dean was killed in a two-car collision near Cholame, Calif"*.

Consider the following question:

"Who was Horus' father?"

A system could answer *"Osiris"* from the snippet *"It also hosted statues of Amon's wife, Mut, the goddess Isis, her husband, Osiris, and their son Horus"*, by using a generic world knowledge ontology containing the fact:

$$\forall x(husband(x) \rightarrow male(x))$$
$$\forall x \forall y(son(x) \wedge of(x, y) \wedge male(y) \rightarrow father(y) \wedge of(y, x))$$

The main reasons that limit the exploitation of existing resources stem from the nature of semantic repositories and lexical resources. Although rich in structure and precision, semantic repositories are difficult to use since they are built by hand and are therefore limited in size and scope. In contrast, lexical resources represent a very large amount of knowledge, but they suffer from low precision and structure.

1.2. Harvesting and Ontologizing Knowledge Desiderata

In order to leverage knowledge resources in NLP applications, it is necessary to improve knowledge harvesting algorithms and to integrate the different types of resources in a coherent framework (e.g., a semantic repository such as an ontology or term bank). An ideal framework for knowledge harvesting and exploitation should then guarantee the following desired properties:

- *Generality*. Knowledge harvesting should be able to extract as many relation types as possible.
- *Minimal supervision*. Knowledge harvesting should be carried out using little or no human intervention.
- *Breadth*. Harvesting algorithms should be adaptable to different corpus sizes, in order to successfully extract knowledge from both large textual resources such as the Web and small ones.
- *Precision*. Harvested knowledge must be precise. As lexical resources are usually very noisy, they can be made more precise by both improving the harvesting algorithms and by filtering erroneous information during the linking process to a semantic repository.
- *Domain knowledge coverage*. Harvested knowledge must cover all the domain knowledge.
- *Closeness to language*. As applications work on linguistic expressions, it is crucial to map conceptual/sense knowledge to language. This can be achieved by linking concepts/senses and relations in a semantic repository to terms and term relations in a lexical resource.

- *Structure*. The structure of a semantic repository is a key aspect to expand the lexical knowledge embedded in lexical resources and make them usable in applications. For example a simple list of *part-of* relation instances can be expanded by using generalizations or synonymy information enclosed in a semantic repository such a WordNet.

1.3. Harvesting and Ontologizing Knowledge in Practice

In this chapter, we present a pipeline of two systems which form a complete and coherent framework for knowledge harvesting and exploitation, by addressing the above mentioned desired properties. In particular, our systems aim at addressing the issue of extracting and ontologizing *relational knowledge*, which is an important component of many NLP applications, as outlined in Section 1.1. The first of these two systems is called *Espresso* and is described in Section 3. *Espresso* is a general-purpose, broad, and accurate corpus harvesting algorithm requiring minimal supervision. The main algorithmic contribution is a novel method for exploiting *generic patterns*, which are broad coverage noisy extraction patterns - i.e., patterns with high recall and low precision. Insofar, difficulties in using these patterns have been a major impediment for minimally supervised algorithms resulting in either very low precision or very low recall. We propose a method to automatically detect generic patterns and to separate their correct and incorrect instances. The key intuition behind the algorithm is that given a set of *reliable* (high precision) patterns on a corpus, correct instances of a generic pattern will fire more with reliable patterns on a very large corpus, like the Web, than incorrect ones. Previous work like Girju et Al. [15] that has made use of generic patterns through filtering has shown both high precision and high recall, at the expensive cost of much manual semantic annotation. Minimally supervised algorithms, like [17,18], typically ignore generic patterns since system precision dramatically decreases from the introduced noise and bootstrapping quickly spins out of control.

Secondly, is Section 4, we propose a system which adopts two alternative algorithms for ontologizing binary semantic relations into WordNet. Formally, given an instance (x, r, y) of a binary relation r between terms x and y, the ontologizing task is to identify the WordNet senses of x and y where r holds. For example, the instance *(proton, PART-OF, element)* ontologizes into WordNet as *(proton#1, PART-OF, element#2)*. The first algorithm that we explore, called the *anchoring approach*, was suggested as a promising avenue of future work by Pantel [19]. This bottom up algorithm is based on the intuition that x can be disambiguated by retrieving the set of terms that occur in the same relation r with y and then finding the senses of x that are most similar to this set. The assumption is that terms occurring in the same relation will tend to have similar meaning. We here propose a measure of similarity to capture this intuition. In contrast to anchoring, our second algorithm, called the *clustering approach*, takes a top-down view. Given a relation r, suppose that we are given every conceptual instance of r, i.e., instances of r in the upper ontology like *(particles#1, PART-OF, substances#1)*. An instance (x, r, y) can then be ontologized easily by finding the senses of x and y that are subsumed by ancestors linked by a conceptual instance of r. For example, the instance *(proton, PART-OF, element)* ontologizes to *(proton#1, PART-OF, element#2)* since $proton\#1$ is subsumed by *particles* and *element#2* is subsumed by *substances*. The problem then is to automatically infer the set of conceptual instances. For this purpose, we develop a clustering al-

gorithm for generalizing a set of relation instances to conceptual instances by looking up the WordNet hypernymy hierarchy for common ancestors, as specific as possible, that subsume as many instances as possible. An instance is then attached to those senses that are subsumed by the highest scoring conceptual instances.

In Section 5 we report a complete experimental analysis of both systems. Experimental evidence demonstrates that our two systems are successful in harvesting and ontologizing relational knowledge by outperforming similar state of the art approaches.

2. Relevant Work

In this section, we review previous work in both relational knowledge harvesting and ontologizing.

2.1. Relational Knowledge Harvesting

To date, most research on relation harvesting has focused on *is-a* and *part-of*. Approaches fall into two categories: pattern- and clustering-based.

Most common are *pattern-based approaches*. Hearst [17] pioneered using patterns to extract hyponym (*is-a*) relations. Manually building three lexico-syntactic patterns, Hearst sketched a bootstrapping algorithm to learn more patterns from instances, which has served as the model for most subsequent pattern-based algorithms.

Berland and Charniank [20] proposed a system for *part-of* relation extraction, based on the Hearst [17] approach. Seed instances are used to infer linguistic patterns that are used to extract new instances. While this study introduces statistical measures to evaluate instance quality, it remains vulnerable to data sparseness and has the limitation of considering only one-word terms.

Improving upon Berland and Charniank [20], Girju et Al. [15] employ machine learning algorithms and WordNet [8] to disambiguate *part-of* generic patterns like *"X's Y"* and *"X of Y"*. This study is the first extensive attempt to make use of generic patterns. In order to discard incorrect instances, they learn WordNet-based selectional restrictions, like *"X(scene#4)'s Y(movie#1)"*. While making huge grounds on improving precision/recall, heavy supervision is required through manual semantic annotations.

Ravichandran and Hovy [14] focus on scaling relation extraction to the Web. A simple and effective algorithm is proposed to infer surface patterns from a small set of instance seeds by extracting substrings relating seeds in corpus sentences. The approach gives good results on specific relations such as *birthdates*, however it has low precision on generic ones like *is-a* and *part-of*. Pantel and et Al. [21] proposed a similar, highly scalable approach, based on an edit-distance technique, to learn lexico-syntactic patterns, showing both good performance and computational efficiency. *Espresso* uses a similar approach to infer patterns, but we make use of generic patterns and apply refining techniques to deal with a wide variety of relations.

Other pattern-based algorithms have been proposed by Riloff and Shepherd [10], who used a semi-automatic method for discovering similar words using a few seed examples, in KnowItAll [9] that performs large-scale extraction of facts from the Web, by Mann [22] who used part of speech patterns to extract a subset of *is-a* relations involving proper nouns, by Downey et Al. [23] who formalized the problem of relation extrac-

tion in a coherent and effective combinatorial model that is shown to outperform previous probabilistic frameworks, by Snow et Al. [24], and in co-occurrence approaches such as in Roark and Charniak [25]. Ciaramita et al.'s chapter in this book presents a very nice approach to learning structured arbitrary binary semantic relations which is fully unsupervised, domain independent and quite efficient since it ultimately relies on named-entity tagging and dependency parsing which can be both solved in linear time.

Clustering approaches have so far been applied only to *is-a* extraction. These methods use clustering algorithms to group words according to their meanings in text, label the clusters using its members' lexical or syntactic dependencies, and then extract an *is-a* relation between each cluster member and the cluster label. Caraballo [26] proposed the first attempt which used conjunction and apposition features to build noun clusters. Recently, Pantel and Ravichandran [18] extended this approach by making use of all syntactic dependency features for each noun. The advantage of clustering approaches is that they permit algorithms to identify *is-a* relations that do not explicitly appear in text, however they generally fail to produce coherent clusters from fewer than 100 million words; hence they are unreliable for small corpora.

2.2. Ontologizing Knowledge

Several researchers have worked on ontologizing semantic resources. Most recently, Pantel [19] defined the task of ontologizing a lexical semantic resource as linking its terms to the concepts in a WordNet-like hierarchy. He developed a method to propagate lexical co-occurrence vectors to WordNet synsets, forming ontological co-occurrence vectors. Adopting an extension of the distributional hypothesis [27], the co-occurrence vectors are used to compute the similarity between synset/synset and between lexical term/synset. An unknown term is then attached to the WordNet synset whose co-occurrence vector is most similar to the term's co-occurrence vector. Though the author suggests a method for attaching more complex lexical structures like binary semantic relations, he focuses only on attaching terms.

Basili et Al. [28] proposed an unsupervised method to infer semantic classes (WordNet synsets) for terms in domain-specific verb relations. These relations, such as *(x, EXPAND, y)* are first automatically learnt from a corpus. The semantic classes of x and y are then inferred using *conceptual density* [29], a WordNet-based measure applied to all instantiations of x and y in the corpus. Semantic classes represent possible common generalizations of the verb arguments. At the end of the process, a set of syntactic-semantic patterns are available for each verb, such as:

(social_group#1, expand, act#2)
(instrumentality#2, expand, act#2)

The method is successful on specific relations with few instances (such as domain verb relations) while its value on generic and frequent relations, such as *part-of*, was untested.

Girju et Al. [15] presented a highly supervised machine learning algorithm to infer semantic constraints on *part-of* relations, such as *(object#1, PART-OF, social_event#1)*. These constraints are then used as selectional restrictions in harvesting *part-of* instances from ambiguous lexical patterns, like *"X of Y"*. The approach shows high performance in terms of precision and recall, but, as the authors acknowledge, it requires large human effort during the training phase.

Others have also made significant additions to WordNet. For example, in eXtended WordNet [30], the glosses in WordNet are enriched by disambiguating the nouns, verbs, adverbs, and adjectives with synsets. Another work has enriched WordNet synsets with topically related words extracted from the Web [31]. Finally, the general task of word sense disambiguation [32] is relevant since there the task is to ontologize each term in a passage into a WordNet-like sense inventory. If we had a large collection of sense-tagged text, then our mining algorithms could directly discover WordNet attachment points at harvest time. However, since there is little high precision sense-tagged corpora, methods are required to ontologize semantic resources without fully disambiguating text.

3. Knowledge Harvesting: The Espresso Algorithm

Espresso is based on the framework adopted by Hearst [17]. It is a minimally supervised bootstrapping algorithm that takes as input a few seed instances of a particular relation and iteratively learns surface patterns to extract more instances. The key to *Espresso* lies in its use of *generic patterns*, i.e., those broad coverage noisy patterns that extract both many correct and incorrect relation instances. For example, for *part-of* relations, the pattern *"X of Y"* extracts many correct relation instances like *"wheel of the car"* but also many incorrect ones like *"house of representatives"*.

The key assumption behind *Espresso* is that in very large corpora, like the Web, correct instances generated by a generic pattern will be instantiated by some *reliable patterns*, where reliable patterns are patterns that have high precision but often very low recall (e.g., *"X consists of Y"* for *part-of* relations). In this section, we describe the overall architecture of *Espresso*, propose a principled measure of reliability, and give an algorithm for exploiting generic patterns.

3.1. System Architecture

Espresso iterates between the following three phases: *pattern induction*, *pattern ranking/selection*, and *instance extraction*. The algorithm begins with seed instances of a particular binary relation (e.g., *is-a*) and then iterates through the phases until it extracts τ_1 patterns or the average pattern score decreases by more than τ_2 from the previous iteration. In our experiments, we set $\tau_1 = 5$ and $\tau_2 = 50\%$.

For our tokenization, in order to harvest multi-word terms as relation instances, we adopt a slightly modified version of the term definition given by Justeson and Katz [33], as it is one of the most commonly used in the NLP literature:

((Adj|Noun)+|((Adj|Noun)(NounPrep)?)(Adj|Noun)*)Noun*

This term defintion allows to capture both simple expressions like *underground economy*, and more complex ones like *Iraqi National Joint Action Committee for Reforms*.

3.1.1. Pattern Induction

In the *pattern induction* phase, *Espresso* infers a set of surface patterns P that connects as many of the seed instances as possible in a given corpus. Any pattern learning algorithm would do. We chose the state of the art algorithm described by Ravichandran and

Hovy [14] with the following slight modification. For each input instance $\{x, y\}$, we first retrieve all sentences containing the two terms x and y. The sentences are then generalized into a set of new sentences $S_{x,y}$ by replacing all terminological expressions by a terminological label, *TR*. For example:

"Because/IN HF/NNP is/VBZ a/DT weak/JJ acid/NN and/CC x is/VBZ a/DT y"

is generalized as:

"Because/IN TR is/VBZ a/DT TR and/CC x is/VBZ a/DT y"

Term generalization is useful for small corpora to reduce data sparseness. Generalized patterns are naturally less precise, but this is ameliorated by our filtering step described in Section 3.2.

As in the original algorithm, all substrings linking terms x and y are then extracted from $S_{x,y}$, and overall frequencies are computed to form P.

3.1.2. Pattern Ranking/Selection

In Ravichandran and Hovy [14], a frequency threshold on the patterns in P is set to select the final patterns. However, low frequency patterns may in fact be very good. In this work, instead of frequency, we propose a novel measure of pattern reliability, r_π, which is described in detail in Section 3.1.4. *Espresso* ranks all patterns in P according to reliability r_π and discards all but the top-*k*, where k is set to the number of patterns from the previous iteration plus one. In general, we expect that the set of patterns is formed by those of the previous iteration plus a new one. Yet, new statistical evidence can lead the algorithm to discard a pattern that was previously discovered.

3.1.3. Instance Extraction

In this phase, *Espresso* retrieves from the corpus the set of instances I that match any of the patterns in P. In Section 3.1.4, we propose a principled measure of instance reliability r_ι for ranking instances. Next, *Espresso* filters incorrect instances using the algorithm proposed in Section 3.2 and then selects the highest scoring m instances according to r_ι as input for the subsequent iteration. We experimentally set $m = 200$.

In small corpora, the number of extracted instances can be too low to guarantee sufficient statistical evidence for the pattern discovery phase of the next iteration. In such cases, the system enters an *expansion phase*, where instances are expanded as follows.

Web expansion: New instances of the patterns in P are retrieved from the Web, using the Google search engine. Specifically, for each instance $\{x, y\} \in I$, the system creates a set of queries, using each pattern in P instantiated with y. For example, given the instance *"Italy, country"* and the pattern *"Y such as X"*, the resulting Google query will be *"country such as *"*. New instances are then created from the retrieved Web results (e.g. *"Canada, country"*) and added to I. The noise generated from this expansion is attenuated by the filtering algorithm described in Section 3.2.

Syntactic expansion: New instances are created from each instance $\{x, y\} \in I$ by extracting sub-terminological expressions from x corresponding to the syntactic head of terms. For example, the relation *"new record of a criminal conviction part-of FBI report"* expands to: *"new record part-of FBI report"*, and *"record part-of FBI report"*.

3.1.4. Pattern and Instance Reliability

Intuitively, a reliable pattern is one that is both highly precise and one that extracts many instances. The recall of a pattern p can be approximated by the fraction of input instances that are extracted by p. Since it is non-trivial to estimate automatically the precision of a pattern, we are wary of keeping patterns that generate many instances (i.e., patterns that generate high recall but potentially disastrous precision). Hence, we desire patterns that are highly associated with the input instances. Pointwise mutual information [34] is a commonly used metric for measuring this strength of association between two events x and y:

$$pmi(x, y) = log \frac{P(x, y)}{P(x)P(y)}$$

We define the reliability $r_\pi(p)$ of a pattern p, as its average strength of association across each input instance $i \in I$, weighted by the reliability of each instance i:

$$r_\pi(p) = \frac{\sum_{i \in I} \frac{pmi(i,p)}{max_{pmi}} * r_\iota(i)}{|I|}$$

where $r_\iota(i)$ is the reliability of instance i (defined below) and max_{pmi} is the maximum pointwise mutual information between all patterns and all instances. $r_\pi(p)$ ranges from $[0, 1]$. The reliability of the manually supplied seed instances is $r_\iota(i) = 1$. The pointwise mutual information between instance $i = \{x, y\}$ and pattern p is estimated using the following formula:

$$pmi(i, p) = log \frac{|x, p, y|}{|x, *, y||*, p*|}$$

where $|x, p, y|$ is the frequency of pattern p instantiated with terms x and y and where the asterisk (*) represents a wildcard. A well-known problem is that pointwise mutual information is biased towards infrequent events. We thus multiply $pmi(i, p)$ with the discounting factor suggested by Pantel and Ravichandran [18].

Estimating the reliability of an instance is similar to estimating the reliability of a pattern. Intuitively, a reliable instance is one that is highly associated with as many reliable patterns as possible (i.e., we have more confidence in an instance when multiple reliable patterns instantiate it). Hence, analogous to our pattern reliability measure, we define the reliability $r_\iota(i)$ of an instance i as:

$$r_\iota(i) = \frac{\sum_{p \in P'} \frac{pmi(i,p)}{max_{pmi}} * r_\pi(p)}{|P'|}$$

where $r_\pi(p)$ is the reliability of pattern p (defined earlier) and max_{pmi} is as before. Note that $r_\iota(i)$ and $r_\pi(p)$ are recursively defined, where $r_\iota(i) = 1$ for the manually supplied seed instances.

3.2. Exploiting Generic Patterns

Generic patterns are high recall / low precision patterns (e.g, the pattern *"X of Y"* can ambiguously refer to a *part-of*, *is-a* and *possession* relations). Using them blindly increases system recall while dramatically reducing precision. Minimally supervised algorithms have typically ignored them for this reason. Only heavily supervised approaches, like Girju et Al. [15] have successfully exploited them.

Espresso's recall can be significantly increased by automatically separating correct instances extracted by generic patterns from incorrect ones. The challenge is to harness the expressive power of the generic patterns while remaining minimally supervised.

The intuition behind our method is that in a very large corpus, like the Web, correct instances of a generic pattern will be instantiated by many of *Espresso*'s reliable patterns accepted in P. Recall that, by definition, *Espresso*'s reliable patterns extract instances with high precision (yet often low recall). In a very large corpus, like the Web, we assume that a correct instance will occur in at least one of *Espresso*'s reliable pattern even though the patterns' recall is low. Intuitively, our confidence in a correct instance increases when, i) the instance is associated with many reliable patterns; and ii) its association with the reliable patterns is high. At a given Espresso iteration, where P_R represents the set of previously selected reliable patterns, this intuition is captured by the following measure of confidence in an instance $i = \{x, y\}$:

$$S(i) = \sum_{p \in P_R} S_p(i) \times \frac{r_\pi(p)}{T}$$

where T is the sum of the reliability scores $r_\pi(p)$ for each pattern $p \in P_R$, and

$$S_p(i) = pmi(i, p) = log \frac{|x, p, y|}{|x, *, y| \times |*, p*|}$$

where pointwise mutual information between instance i and pattern p is estimated with Google as follows:

$$S_p(i) \approx \frac{|x, p, y|}{|x| \times |y| \times |p|}$$

An instance i is rejected if $S(i)$ is smaller than some threshold τ. Although this filtering may also be applied to reliable patterns, we found this to be detrimental in our experiments since most instances generated by reliable patterns are correct. In *Espresso*, we classify a pattern as generic when it generates more than 10 times the instances of previously accepted reliable patterns.

4. Ontologizing Semantic Relations

The output of most relation harvesting algorithms, such as *Espresso* described in Section 3, consists of flat lists of lexical semantic knowledge such as *"Italy is-a country"* and *"orange similar-to blue"*. However, using this knowledge beyond simple keyword matching, for example in inferences, requires it to be linked, or *ontologized*, into semantic repositories such as ontologies or term banks like WordNet.

Given an instance (x, r, y) of a binary relation r between terms x and y, the ontologizing task is to identify the senses of x and y where r holds. In this work, we focus on WordNet 2.0 senses, though any similar term bank would apply.

Let S_x and S_y be the sets of all WordNet senses of x and y. A *sense pair*, s_{xy}, is defined as any pair of senses of x and y: $s_{xy} = \{s_x, s_y\}$ where $s_x \in S_x$ and $s_y \in S_y$. The set of all sense pairs S_{xy} consists of all pairings between senses in S_x and S_y.

In order to attach a relation instance (x, r, y) into WordNet, one must:

- *Disambiguate* x and y, that is, find the subsets $S'_x \subseteq S_x$ and $S'_y \subseteq S_y$ for which the relation r holds; and
- *Instantiate* the relation in WordNet, using the synsets corresponding to all correct pairings between the senses in S'_x and S'_y. We denote this set of attachment points as S'_{xy}.

If S_x or S_y is empty, no attachments are produced.

For example, the instance *(study, PART-OF, report)* is ontologized into WordNet through the senses $S'_x = \{survey\#1, study\#2\}$ and $S'_y = \{report\#1\}$. The final attachment points S'_{xy} are:

(survey#1, PART-OF, report#1)
(study#2, PART-OF, report#1)

Unlike common algorithms for word sense disambiguation, here it is important to take into consideration the semantic dependency between the two terms x and y. For example, an entity that is *part-of* a study has to be some kind of information. This knowledge about mutual selectional preference (the preferred semantic class that fills a certain relation role, as x or y) can be exploited to ontologize the instance.

In the following sections, we propose two algorithms for ontologizing binary semantic relations.

4.1. Method 1: Anchor Approach

Given an instance (x, r, y), this approach fixes the term y, called the anchor, and then disambiguates x by looking at all other terms that occur in the relation r with y. Based on the principle of distributional similarity [27], the algorithm assumes that the words that occur in the same relation r with y will be more similar to the correct sense(s) of x than the incorrect ones. After disambiguating x, the process is then inverted with x as the anchor to disambiguate y.

In the first step, y is fixed and the algorithm retrieves the set of all other terms X' that occur in an instance (x', r, y), $x' \in X'$[1]. For example, given the instance *(reflections, PART-OF, book)*, and a resource containing the following relations:

(false allegations, PART-OF, book)
(stories, PART-OF, book)
(expert analysis, PART-OF, book)
(conclusions, PART-OF, book)

[1]For semantic relations between complex terms, like *(expert analysis, PART-OF, book)*, only the head noun of terms are recorded, like *"analysis"*. As a future work, we plan to use the whole term if it is present in WordNet.

the resulting set X' would be: *{allegations, stories, analysis, conclusions}*. All possible pairings, $S_{xx'}$, between the senses of x and the senses of each term in X', called $S_{x'}$, are computed. For each sense pair $\{s_x, s_{x'}\}$ in $S_{xx'}$, a similarity score $r(s_x, s_{x'})$ is calculated using WordNet:

$$r(s_x, s_{x'}) = \frac{1}{d(s_x, s_{x'}) + 1} \times f(s_{x'})$$

where the distance $d(s_x, s_{x'})$ is the length of the shortest path connecting the two synsets in the hypernymy hierarchy of WordNet, and $f(s_{x'})$ is the number of times sense $s_{x'}$ occurs in any of the instances of X'. Note that if no connection between two synsets exists, then $r(s_x, s_{x'}) = 0$. The overall sense score for each sense s_x of x is calculated as:

$$r(s_x) = \sum_{s_{x'} \in S_{x'}} r(s_x, s_{x'})$$

Finally, the algorithm inverts the process by setting x as the anchor and computes $r(s_y)$ for each sense of y. All possible pairings of senses are computed and scored by averaging $r(s_x)$ and $r(s_y)$. Pairings scoring higher than a threshold τ_1 are selected as the attachment points in WordNet. We experimentally set $\tau_1 = 0.02$.

4.2. Method 2: Clustering Approach

The main idea of the clustering approach is to leverage the lexical behaviors of the two terms in an instance as a whole. The assumption is that the general meaning of the relation is derived from the combination of the two terms.

The algorithm is divided in two main phases. In the first phase, *semantic clusters* are built using the WordNet senses of all instances. A semantic cluster is defined by the set of instances that have a common semantic generalization. We denote the *conceptual instance* of the semantic cluster as the pair of WordNet synsets that represents this generalization. For example the following two *part-of* instances:

(second section, PART-OF, Los Angeles-area news)
(Sandag study, PART-OF, report)

are in a common cluster represented by the following conceptual instance:

[writing#2, PART-OF, message#2]

since *writing#2* is a hypernym of both *section* and *study*, and *message#2* is a hypernym of *news* and *report*[2].

In the second phase, the algorithm attaches an instance into WordNet by using WordNet distance metrics and frequency scores to select the best cluster for each instance. A good cluster is one that:

- achieves a good trade-off between generality and specificity; and

[2] Again, here, we use the syntactic head of each term for generalization since we assume that it drives the meaning of the term itself.

- disambiguates among the senses of x and y using the other instances' senses as support.

For example, given the instance *(second section, PART-OF, Los Angeles-area news)* and the following conceptual instances:

[writing#2, PART-OF, message#2]
[object#1, PART-OF, message#2]
[writing#2, PART-OF, communication#2]
[social_group#1, PART-OF, broadcast#2]
[organization#, PART-OF, message#2]

the first conceptual instance should be scored highest since it is both not too generic nor too specific and is supported by the instance *(Sandag study, PART-OF, report)*, i.e., the conceptual instance subsumes both instances. The second and the third conceptual instances should be scored lower since they are too generic, while the last two should be scored lower since the sense for *section* and *news* are not supported by other instances. The system then outputs, for each instance, the set of sense pairs that are subsumed by the highest scoring conceptual instance. In the previous example:

(section#1, PART-OF, news#1)
(section#1, PART-OF, news#2)
(section#1, PART-OF, news#3)

are selected, as they are subsumed by *[writing#2, PART-OF, message#2]*. These sense pairs are then retained as attachment points into WordNet.

Below, we describe each phase in more detail.

4.2.1. Phase 1: Cluster Building

Given an instance (x, r, y), all sense pair pairings $s_{xy} = \{s_x, s_y\}$ are retrieved from WordNet. A set of *candidate conceptual instances*, C_{xy}, is formed for each instance from the pairing of each WordNet ancestor of s_x and s_y, following the hypernymy link, up to degree τ_2.

Each candidate conceptual instance, $c = \{c_x, c_y\}$, is scored by its degree of generalization as follows:

$$r(c) = \frac{1}{(n_x + 1) \times (n_y + 1)}$$

where n_i is the number of hypernymy links needed to go from s_i to c_i, for $i \in \{x, y\}$. $r(c)$ ranges from $[0, 1]$ and is highest when little generalization is needed.

For example, the instance *(Sandag study, PART-OF, report)* produces 70 sense pairs since *study* has 10 senses and *report* has 7 senses. Assuming $\tau_2 = 1$, the instance sense *(survey#1, PART-OF, report#1)* has the set of candidate conceptual instances reported in Table 1.

Finally, each candidate conceptual instance c forms a cluster of all instances (x, r, y) that have some sense pair s_x and s_y as hyponyms of c. Note also that candidate conceptual instances may be subsumed by other candidate conceptual instances. Let G_c refer to the set of all candidate conceptual instances subsumed by candidate conceptual instance c.

C_{xy}	n_x	n_y	$r(c)$
(survey#1, PART-OF,report#1)	0	0	1
(survey#1, PART-OF,document#1)	0	1	0.5
(examination#1, PART-OF,report#1)	1	0	0.5
(examination#1, PART-OF,document#1)	1	1	0.25

Table 1. Example of conceptual instances for the instance sense *(survey#1, PART-OF, report#1)*.

Intuitively, better candidate conceptual instances are those that subsume both many instances and other candidate conceptual instances, but at the same time that have the least distance from subsumed instances. We capture this intuition with the following score of c:

$$score(c) = \frac{\sum_{g \in G_c} r(g)}{|G_c|} \times log|I_C| \times log|G_C|$$

where I_c is the set of instances subsumed by c. We experimented with different variations of this score and found that it is important to put more weight on the distance between subsumed conceptual instances than the actual number of subsumed instances. Without the log terms, the highest scoring conceptual instances are too generic (i.e., they are too high up in the ontology).

4.2.2. Phase 2: Attachment Points Selection

In this phase, we utilize the conceptual instances of the previous phase to attach each instance (x, r, y) into WordNet.

At the end of Phase 1, an instance can be clustered in different conceptual instances. In order to select an attachment, the algorithm selects the sense pair of x and y that is subsumed by the highest scoring candidate conceptual instance. It and all other sense pairs that are subsumed by this conceptual instance are then retained as the final attachment points.

As a side effect, a final set of conceptual instances is obtained by deleting from each candidate those instances that are subsumed by a higher scoring conceptual instance. Remaining conceptual instances are then rescored using $score(c)$. The final set of conceptual instances thus contains unambiguous sense pairs.

5. Experimental Results

In this section, we evaluate both the harvesting algorithm *Espresso* and our two algorithms for ontologizing semantic relations.

5.1. Espresso Evaluation

Here, we present an empirical comparison of *Espresso* with three state of the art systems on the task of extracting various semantic relations.

5.1.1. Experimental Setup

We perform our experiments using the following two datasets:

- **TREC**: This dataset consists of a sample of articles from the Aquaint (TREC-9) newswire text collection. The sample consists of 5,951,432 words extracted from the following data files: AP890101 - AP890131, AP890201 - AP890228, and AP890310 - AP890319.
- **CHEM**: This small dataset of 313,590 words consists of a college level textbook of introductory chemistry [35].

Each corpus is pre-processed using the Alembic Workbench POS-tagger [36].

Below we describe the systems used in our empirical evaluation of *Espresso*.

- **RH02**: The algorithm by Ravichandran and Hovy [14] described in Section 2.
- **GI06**: The algorithm by Girju et Al. [15] described in Section 2.
- **PR04**: The algorithm by Pantel [18] described in Section 2.
- **ESP-**: The *Espresso* algorithm using the pattern and instance reliability measures, but without using generic patterns.
- **ESP+**: The full *Espresso* algorithm described in this work exploiting generic patterns.

For *ESP+*, we experimentally set τ from Section 3.2 to $\tau = 0.4$ for TREC, and $\tau = 0.3$ for CHEM by manually inspecting a small set of instances.

Espresso is designed to extract various semantic relations exemplified by a given small set of seed instances. We consider the standard *is-a* and *part-of* relations as well as the following more specific relations:

- *succession*. This relation indicates that a person succeeds another in a position or title. For example, *George Bush* succeeded *Bill Clinton* and *Pope Benedict XVI* succeeded *Pope John Paul II*. We evaluate this relation on the TREC-9 corpus.
- *reaction*. This relation occurs between chemical elements/molecules that can be combined in a chemical reaction. For example, *hydrogen gas* reacts with *oxygen gas* and *zinc* reacts-with *hydrochloric acid*. We evaluate this relation on the CHEM corpus.
- *production*. This relation occurs when a process or element/object produces a result[3]. For example, *ammonia* produces *nitric oxide*. We evaluate this relation on the CHEM corpus.

For each semantic relation, we manually extracted a small set of seed examples. The seeds were used for both *Espresso* as well as *RH02*. Table 2 lists a sample of the seeds as well as sample outputs from *Espresso*.

5.1.2. Precision and Recall

We implemented the systems outlined in Section 5.1.1, except for *GI06*, and applied them to the TREC and CHEM datasets. For each output set, per relation, we evaluate the precision of the system by extracting a random sample of instances (50 for the TREC corpus and 20 for the CHEM corpus) and evaluating their quality manually using two

[3] *Production* is an ambiguous relation; it is intended to be a causation relation in the context of chemical reactions.

	Is-a (12)	*Part-Of (12)*	*Succession (12)*	*Reaction (13)*	*Production (14)*
Seeds	wheat :: crop	leader :: panel	Khrushchev :: Stalin	magnesium :: oxygen	bright flame :: flares
	George Wendt :: star	city :: region	Carla Hills :: Yeutter	hydrazine :: water	hydrogen :: metal hydrides
	nitrogen :: element	ion :: matter	Bush :: Reagan	aluminum metal :: oxygen	ammonia :: nitric oxide
	diborane :: substance	oxygen :: water	Julio Barbosa :: Mendes	lithium metal :: fluorine gas	copper :: brown gas
Espresso	Picasso :: artist	trees :: land	Ford :: Nixon	hydrogen :: oxygen	electron :: ions
	tax :: charge	material :: FBI report	Setrakian :: John Griesemer	Ni :: HCl	glycerin :: nitroglycerin
	protein :: biopolymer	oxygen :: air	Camero Cardiel :: Camacho	carbon dioxide :: methane	kidneys :: kidney stones
	HCl :: strong acid	atom :: molecule	Susan Weiss :: editor	boron :: fluorine	ions :: charge

Table 2. Sample seeds used for each semantic relation and sample outputs from *Espresso*. The number in the parentheses for each relation denotes the total number of seeds used as input for the system.

human judges (a total of 680 instances were annotated per judge). For each instance, judges may assign a score of 1 for correct, 0 for incorrect, and 1/2 for partially correct. Example instances that were judged partially correct include *"analyst is-a manager"* and *"pilot is-a teacher"*. The kappa statistic [37] on this task was $K = 0.69$[4]. The precision for a given set of instances is the sum of the judges' scores divided by the total instances.

Although knowing the total number of correct instances of a particular relation in any non-trivial corpus is impossible, it is possible to compute the recall of a system relative to another system's recall. Following Pantel et Al. [21], we define the relative recall of system A given system B, $R_{A|B}$, as:

$$R_{A|B} = \frac{R_A}{R_B} = \frac{\frac{C_A}{C}}{\frac{C_B}{C}} = \frac{C_A}{C_B} = \frac{P_A \times |A|}{P_B \times |B|}$$

where R_A is the recall of A, C_A is the number of correct instances extracted by A, C is the (unknown) total number of correct instances in the corpus, P_A is A's precision in our experiments, and $|A|$ is the total number of instances discovered by A.

Tables 3-9 report the total number of instances, precision[5], and relative recall[6] of each system on the TREC-9 and CHEM corpora. The relative recall is always given in relation to the *ESP-* system. For example, in Table 3, *RH02* has a relative recall of 5.31 with *ESP-*, which means that the *RH02* system outputs 5.31 times more correct relations than *ESP-* (at a cost of much lower precision). Similarly, *PR04* has a relative recall of 0.23 with *ESP-*, which means that *PR04* outputs 4.35 fewer correct relations than *ESP-* (also with a smaller precision). We did not include the results from *GI06* in the tables since the system is only applicable to *part-of* relations and we did not reproduce it. However, the authors evaluated their system on a sample of the TREC-9 dataset and reported 83% precision and 72% recall (this algorithm is heavily supervised.)

In all tables, *RH02* extracts many more relations than *ESP-*, but with a much lower precision, because it uses generic patterns without filtering. The high precision of *ESP-* is due to the effective reliability measures presented in Section 3.1.4.

5.1.3. Effect of Generic Patterns

Experimental results, for all relations and the two different corpus sizes, show that *ESP-* greatly outperforms the other methods on precision. However, without the use of generic patterns, the *ESP-* system shows lower recall in all but the *production* relation.

[4] The kappa statistic jumps to K = 0.79 if we treat partially correct classifications as correct.

[5] Because of the small evaluation sets, we estimate the 95% confidence intervals using bootstrap resampling to be in the order of ± 10-15% (absolute numbers).

[6] Relative recall is given in relation to ESP-.

System	Instances	Precision	Rel Recall
RH02	57,525	28.0%	5.31
PR04	1,504	47.0%	0.23
ESP-	4,154	**73%**	1.00
ESP+	69,156	36.2%	**8.26**

Table 3. System performance: TREC/*is-a*.

System	Instances	Precision	Rel Recall
RH02	2556	25.0%	3.76
PR04	108	40.0%	0.25
ESP-	200	**85.0%**	1.00
ESP+	1490	76.0%	**6.66**

Table 4. System performance: CHEM/*is-a*.

System	Instances	Precision	Rel Recall
RH02	12,828	35.0%	42.52
ESP-	132	**80.0%**	1.00
ESP+	87,203	69.9%	**577.22**

Table 5. System performance: TREC/*part-of*.

System	Instances	Precision	Rel Recall
RH02	11,582	33.8%	**58.78**
ESP-	111	**60.0%**	1.00
ESP+	5973	50.7%	45.47

Table 6. System performance: CHEM/*part-of*.

System	Instances	Precision	Rel Recall
RH02	49,798	2.0%	**36.96**
ESP-	55	**49.0%**	1.00
ESP+	55	**49.0%**	1.00

Table 7. System performance: TREC/*succession*.

System	Instances	Precision	Rel Recall
RH02	6,083	30%	53.67
ESP-	40	85%	1.00
ESP+	3102	**91.4%**	**89.39**

Table 8. System performance: CHEM/*reaction*.

System	Instances	Precision	Rel Recall
RH02	197	57.5%	0.80
ESP-	196	**72.5%**	1.00
ESP+	1676	55.8%	**6.58**

Table 9. System performance: CHEM/*production*.

As hypothesized, exploiting generic patterns using the algorithm from Section 3.2 substantially improves recall without much deterioration in precision. *ESP+* shows one to two orders of magnitude improvement on recall while losing on average below 10% precision. The *succession* relation in Table 7 was the only relation where *Espresso* found no generic pattern. For other relations, *Espresso* found from one to five generic patterns. Table 5 shows the power of generic patterns where system recall increases by 577 times with only a 10% drop in precision. In Table 8, we see a case where the combination of filtering with a large increase in retrieved instances resulted in both higher precision and recall.

In order to better analyze our use of generic patterns, we performed the following experiment. For each relation, we randomly sampled 100 instances for each generic pattern and built a gold standard for them (by manually tagging each instance as correct or incorrect). We then sorted the 100 instances according to the scoring formula $S(i)$ derived in Section 3.2 and computed the average precision, recall, and F-score of each top-K ranked instances for each pattern[7]. Due to lack of space, we only present the graphs for four of the 22 generic patterns: *"X is a Y"* for the is-a relation of Table 3, *"X in the Y"* for the *part-of* relation of Table 5, *"X in Y"* for the *part-of* relation of Table 6, and *"X and Y"* for the reaction relation of Table 8. Figure 1 illustrates the results.

In each figure, notice that recall climbs at a much faster rate than precision decreases. This indicates that the scoring function of Section 3.2 effectively separates correct and

[7] We can directly compute recall here since we built a gold standard for each set of 100 samples.

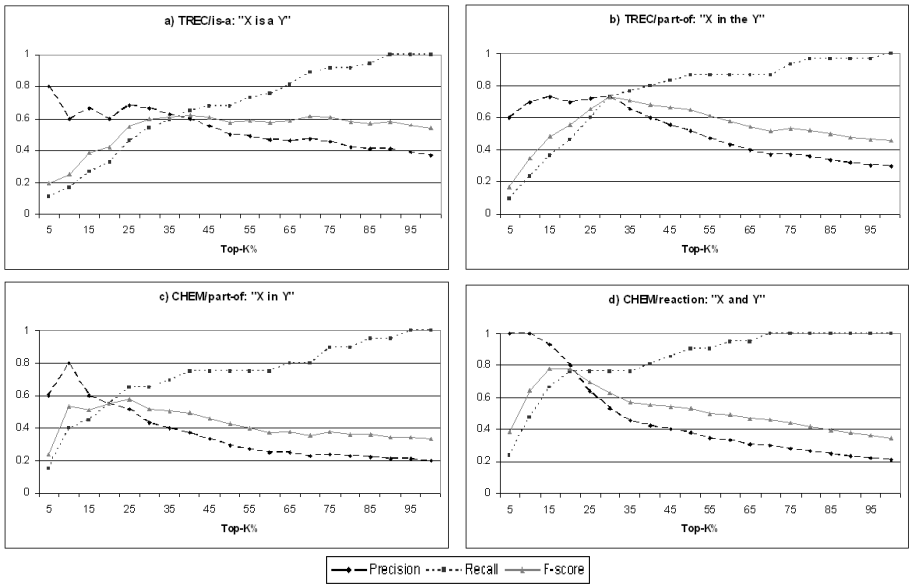

Figure 1. Precision, recall and F-score curves of the Top-$K\%$ ranking instances of patterns *"X is a Y"* (TREC/*is-a*), *"X in Y"* (TREC/*part-of*), *"X in the Y"* (CHEM/*part-of*), and *"X and Y"* (CHEM/*reaction*).

incorrect instances. In Figure 1.a), there is a big initial drop in precision that accounts for the poor precision reported in Table 3.

Recall that the cutoff points on $S(i)$ were set to $\tau = 0.4$ for TREC and $\tau = 0.3$ for CHEM. The figures show that this cutoff is far from the maximum F-score. An interesting avenue of future work would be to automatically determine the proper threshold for each individual generic pattern instead of setting a uniform threshold.

5.2. Ontologizing Evaluation

Here, we present an empirical evaluation of our two methods for ontologizing binary semantic relations, presented in Section 4.

5.2.1. Experimental Setup

Researchers have developed many algorithms for harvesting semantic relations from corpora and the Web. For the purposes of this work, we may choose any one of them and manually validate its mined relations. We choose *Espresso*, the harvesting algorithm described in Section 3.

Test Sets

We experiment with two relations: *part-of* and *causation*. The *causation* relation occurs when an entity produces an effect or is responsible for events or results, for example *(virus, CAUSE, influenza)* and *(burning fuel, CAUSE, pollution)*. We manually built five seed relation instances for both relations and apply *Espresso* to a dataset consisting of a sample of articles from the Aquaint (TREC-9) newswire text collection. The sample consists of 55.7 million words extracted from the Los Angeles Times data files. *Espresso*

extracted 1,468 *part-of* instances and 1,129 *causation* instances. We manually validated the output and randomly selected 200 correct relation instances of each relation for ontologizing into WordNet 2.0.

Gold Standard

We manually built a gold standard of all correct attachments of the test sets in WordNet. For each relation instance (x, r, y), two human annotators selected from all sense pairings of x and y the correct attachment points in WordNet. For example, for *(synthetic material, PART-OF, filter)*, the judges selected the following attachment points: *(synthetic material#1, PART-OF, filter#1)* and *(synthetic material#1, PART-OF, filter#2)*. The kappa statistic [37] on the two relations together was $K = 0.73$.

Systems

The following three systems are evaluated:

- **BL**: the baseline system that attaches each relation instance to the first (most common) Word-Net sense of both terms;
- **AN**: the anchor approach described in Section 4.1.
- **CL**: the clustering approach described in Section 4.2.

5.2.2. Precision, Recall and F-*score*

For both the *part-of* and *causation* relations, we apply the three systems described above and compare their attachment performance using precision, recall, and F-score. Using the manually built gold standard, the precision of a system on a given relation instance is measured as the percentage of correct attachments and recall is measured as the percentage of correct attachments retrieved by the system. Overall system precision and recall are then computed by averaging the precision and recall of each relation instance.

Table 10 and Table 11 report the results on the *part-of* and *causation* relations. We experimentally set the CL generalization parameter τ_2 to 5 and the τ_1 parameter for AN to 0.02.

SYSTEM	PRECISION	RECALL	F-SCORE
BL	54.0%	31.3%	39.6%
AN	40.7%	47.3%	43.8%
CL	57.4%	49.6%	53.2%

Table 10. System performance on the *part-of* relation.

SYSTEM	PRECISION	RECALL	F-SCORE
BL	45.0%	25.0%	32.1%
AN	41.7%	32.4%	36.5%
CL	40.0%	32.6%	35.9%

Table 11. System performance on the *causation* relation.

5.2.3. Discussion

For both relations, CL and AN outperform the baseline in overall F-score. For *part-of*, Table 10 shows that CL outperforms BL by 13.6% in F-score and AN by 9.4%. For causation, Table 11 shows that AN outperforms BL by 4.4% on F-score and CL by 0.6%.

The good results of the CL method on the *part-of* relation suggest that instances of this relation are particularly amenable to be clustered. The generality of the *part-of* relation in fact allows the creation of fairly natural clusters, corresponding to different sub-types of *part-of*, as those proposed by Winston et Al. [38]. The *causation* relation,

however, being more difficult to define at a semantic level [39], is less easy to cluster and thus to disambiguate.

Both CL and AN have better recall than BL, but precision results vary with CL beating BL only on the *part-of* relation. Overall, the system performances suggest that ontologizing semantic relations into WordNet is in general not easy.

The better results of CL and AN with respect to BL suggest that the use of comparative semantic analysis among corpus instances is a good way to carry out disambiguation. Yet, the BL method shows surprisingly good results. This indicates that also a simple method based on word sense usage in language can be valuable. An interesting avenue of future work is to better combine these two different views in a single system.

The low recall results for CL are mostly attributed to the fact that in Phase 2 only the best scoring cluster is retained for each instance. This means that instances with multiple senses that do not have a common generalization are not captured. For example the *part-of* instance *(wings, PART-OF, chicken)* should cluster both in *[body_part#1, PART-OF, animal#1]* and *[body_part#1, PART-OF, food#2]*, but only the best scoring one is retained.

5.2.4. Conceptual Instances: Other Uses

Our *clustering approach* from section 4.2 is enabled by learning conceptual instances - relations between mid-level ontological concepts. Beyond the ontologizing task, conceptual instances may be useful for several other tasks. In this Section, we discuss some of these opportunities and present small qualitative evaluations.

Conceptual instances represent common semantic generalizations of a particular relation. For example, below are two possible conceptual instances for the *part-of* relation:

[person#1, PART-OF, organization#1]
[act#1, PART-OF, plan#1]

The first conceptual instance in the example subsumes all the *part-of* instances in which one or more persons are part of an organization, such as:

(president Brown, PART-OF, executive council)
(representatives, PART-OF, organization)
(students, PART-OF, orchestra)
(players, PART-OF, Metro League)

Below, we present three possible ways of exploiting these conceptual instances.

Support to Relation Extraction Tools

Conceptual instances may be used to support relation extraction algorithms such as *Espresso*.

Most minimally supervised harvesting algorithm do not exploit *generic patterns*, i.e. those patterns with high recall but low precision, since they cannot separate correct and incorrect relation instances. For example, the pattern *"X of Y"* extracts many correct relation instances like *"wheel of the car"* but also many incorrect ones like *"house of representatives"*.

Girju et Al. [15] described a highly supervised algorithm for learning semantic constraints on *generic patterns*, leading to a very significant increase in system recall with-

CONCEPTUAL INSTANCES	SCORE	# INSTANCES	INSTANCES
[multitude#3, PART-OF, group#1]	2.04	10	(ordinary people, PART-OF, Democratic Revolutionary Party) (unlicensed people, PART-OF, underground economy) (young people, PART-OF, commission) (air mass, PART-OF, cold front)
[person#1, PART-OF, organization#1]	1.71	43	(foreign ministers, PART-OF, council) (students, PART-OF, orchestra) (socialists, PART-OF, Iraqi National Joint Action Committee) (players, PART-OF, Metro League)
[act#2, PART-OF, plan#1]	1.60	16	(major concessions, PART-OF, new plan) (attacks, PART-OF, coordinated terrorist plan) (visit, PART-OF, exchange program) (survey, PART-OF, project)
[communication#2, PART-OF, book#1]	1.14	10	(hints, PART-OF, booklet) (soup recipes, PART-OF, book) (information, PART-OF, instruction manual) (extensive expert analysis, PART-OF, book)
[compound#2, PART-OF, waste#1]	0.57	3	(salts, PART-OF, powdery white waste) (lime, PART-OF, powdery white waste) (resin, PART-OF, waste)

Table 12. Sample of the highest scoring conceptual instances learned for the *part-of* relation. For each conceptual instance, we report score(c) , the number of instances, and some example instances.

CONCEPTUAL INSTANCES	SCORE	# INSTANCES	INSTANCES
[change#3, CAUSE, state#4]	1.49	17	(separation, CAUSE, anxiety) (demotion, CAUSE, roster vacancy) (budget cuts, CAUSE, enrollment declines) (reduced flow, CAUSE, vacuum)
[act#2, CAUSE, state#3]	0.81	20	(oil drilling, CAUSE, air pollution) (workplace exposure, CAUSE, genetic injury) (industrial emissions, CAUSE, air pollution) (long recovery, CAUSE, great stress)
[person#1, CAUSE, act#2]	0.64	12	(homeowners, CAUSE, water waste) (needlelike puncture, CAUSE, physician) (group member, CAUSE, controversy) (children, CAUSE, property damage)
[organism#1, CAUSE, disease#1]	0.03	4	(parasites, CAUSE, pneumonia) (virus, CAUSE, influenza) (chemical agents, CAUSE, pneumonia) (genetic mutation, CAUSE, Dwarfism)

Table 13. Sample of the highest scoring conceptual instances learned for the *causation* relation. For each conceptual instance, we report score(c) , the number of instances, and some example instances.

out deteriorating precision. Conceptual instances can be used to automatically learn such semantic constraints by acting as a filter for generic patterns, retaining only those instances that are subsumed by high scoring conceptual instances. Effectively, conceptual instances are used as *selectional restrictions* for the relation. For example, our system discards the following incorrect instances:

(week, CAUSE, coalition)
(demeanor, CAUSE, vacuum)

as they are both part of the very low scoring conceptual instance *[abstraction#6, CAUSE, state#1]*.

Ontology Learning from Text

Each conceptual instance can be viewed as a formal specification of the relation at hand.

For example, Winston et Al. [38] manually identified six sub-types of the *part-of* relation: *member-collection*, *component-integral object*, *portion-mass*, *stuff-object*, *feature-activity* and *place-area*. Such classifications are useful in applications and tasks where a semantically rich organization of knowledge is required. Conceptual instances can be viewed as an automatic derivation of such a classification based on corpus usage. Moreover, conceptual instances can be used to improve the ontology learning process itself. For example, our *clustering approach* can be seen as an *inductive* step producing conceptual instances that are then used in a *deductive* step to learn new instances. An algorithm could iterate between the induction/deduction cycle until no new relation instances and conceptual instances can be inferred.

Word Sense Disambiguation

Word Sense Disambiguation (WSD) systems can exploit the selectional restrictions identified by conceptual instances to disambiguate ambiguous terms occurring in particular contexts. For example, given the sentence:

"the board is composed by members of different countries"

and a harvesting algorithm that extracts the *part-of* relation *(members, PART-OF, board)*, the system could infer the correct senses for board and members by looking at their closest conceptual instance. In our system, we would infer the attachment *(member#1, PART-OF, board#1)* since it is part of the highest scoring conceptual instance *[person#1, PART-OF, organization#1]*.

Qualitative Evaluation

Table 12 and Table 13 list samples of the highest ranking conceptual instances obtained by our system for the *part-of* and *causation* relations. Below we provide a small evaluation to verify:

- the *correctness* of the conceptual instances. Incorrect conceptual instances such as *[attribute#2, CAUSE, state#4]*, discovered by our system, can impede WSD and extraction tools where precise selectional restrictions are needed; and
- the *accuracy* of the conceptual instances. Sometimes, an instance is incorrectly attached to a correct conceptual instance. For example, the instance *(air mass, PART-OF, cold front)* is incorrectly clustered in *[group#1, PART-OF, multitude#3]* since mass and front both have a sense that is descendant of *group#1* and *multitude#3*. However, these are not the correct senses of *mass* and *front* for which the *part-of* relation holds.

For evaluating *correctness*, we manually verify how many correct conceptual instances are produced by Phase 2 of the clustering approach described in Section 4.2. The claim is that a *correct* conceptual instance is one for which the relation holds for all possible subsumed senses. For example, the conceptual instance *[group#1, PART-OF, multitude#3]* is correct, as the relation holds for every semantic subsumption of the two senses. An example of an incorrect conceptual instance is *[state#4, CAUSE, abstraction#6]* since it subsumes the incorrect instance *(audience, CAUSE, new context)*. A manual evaluation of the highest scoring 200 conceptual instances, generated on our test sets described in Section 5.2.1, showed 82% correctness for the *part-of* relation and 86% for *causation*.

For estimating the overall clustering *accuracy*, we evaluated the number of correctly clustered instances in each conceptual instance. For example, the instance *(business people, PART-OF, committee)* is correctly clustered in *[multitude#3, PART-OF, group#1]* and the instance *(law, PART-OF, constitutional pitfalls)* is incorrectly clustered in *[group#1, PART-OF, artifact#1]*. We estimated the overall accuracy by manually judging the instances attached to 10 randomly sampled conceptual instances. The accuracy for *part-of* is 84% and for causation it is 76.6%.

6. Conclusions

In this chapter, we presented algorithms for both extracting semantic relations from textual resources and for linking, or *ontologizing*, them into a semantic repository. We proposed a weakly-supervised, general-purpose, and accurate algorithm, called *Espresso*, for harvesting binary semantic relations from raw text. The main contributions are: i) a method for exploiting generic patterns by filtering incorrect instances using the Web; and ii) a principled measure of pattern and instance reliability enabling the filtering algorithm. We have empirically compared *Espresso*'s precision and recall with other systems on both a small domain-specific textbook and on a larger corpus of general news, and have extracted several standard and specific semantic relations: *is-a*, *part-of*, *succession*, *reaction*, and *production*. *Espresso* achieves higher and more balanced performance than other state of the art systems. By exploiting generic patterns, system recall substantially increases with little effect on precision.

We then proposed two algorithms for automatically ontologizing binary semantic relations into WordNet: an *anchoring approach* and a *clustering approach*. Experiments on the *part-of* and *causation* relations showed promising results. Both algorithms outperformed the baseline on F-score. Our best results were on the *part-of* relation where the clustering approach achieved 13.6% higher F-score than the baseline. The induction of conceptual instances has opened the way for many avenues of future work. We intend to pursue the ideas presented in Section 5.2.4 for using conceptual instances to: i) support knowledge acquisition tools by learning semantic constraints on extracting patterns; ii) support ontology learning from text; and iii) improve word sense disambiguation through selectional restrictions. Also, we will try different similarity score functions for both the clustering and the anchoring approaches, as those surveyed in Corley and Mihalcea [40].

The algorithms described in this chapter may be applied to ontologize many lexical resources of semantic relations, no matter the harvesting algorithm used to mine them. In doing so, we have the potential to quickly enrich our ontologies, like WordNet, thus reducing the knowledge acquisition bottleneck. It is our hope that we will be able to leverage these enriched resources, albeit with some noisy additions, to improve performance on knowledge rich problems such as question answering and information extraction.

References

[1] M. Pasca and S. Harabagiu. The informative role of wordnet in open-domain question answering. In *Proceedings of the NAACL-2001 Workshop on WordNet and Other Lexical Resources: Applications, Extensions and Customizations*, pages 138–143, Pittsburgh, PA, 2001.

[2] M. Geffet and I. Dagan. The distributional inclusion hypotheses and lexical entailment. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005)*, Ann Arbor, MI, 2005.

[3] K. Mahesh. Ontology development for machine translation: Ideology and methodology. Rl report mccs-96-292, New Mexico State University, 1996.

[4] Mahesh K. O'Hara, T and S. Nirenburg. Lexical acquisition with wordnet and the mikrokosmos ontology. In *Proceedings of the COLING/ACL Workshop on Usage of WordNet in Natural Language Processing Systems*, Montreal, Canada, 1998.

[5] Guarino N. Gangemi, A., C. Masolo, A. Oltramari, and L. Schneider. Sweetening ontologies with dolce. In *Proceedings of Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 13th International Conference, EKAW 2002*, pages 166–181, Siguenza, Spain, 2002.

[6] I. Niles and A. Pease. Towards a standard upper ontology. In *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001)*, pages 2–9, Ogunquit, Maine, 2001.

[7] C. Baker, C. Fillmore, and J. Lowe. The berkeley framenet project. In *Proceedings of the Joint Conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics (COLING/ACL-98)*, pages 86–90, Montreal, Canada, 1998.

[8] C. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[9] O. Etzioni, M.J. Cafarella, D. Downey, A.-M. Popescu, T. Shaked, S. Soderland, D.S. Weld, and A. Yates. Unsupervised named-entity extraction from the web: An experimental study. *Artificial Intelligence*, (165(1)):91–134, 2005.

[10] E. Riloff and J. Shepherd. A corpus-based approach for building semantic lexicons. In *Proceedings of 2nd Conference on Empirical Methods in Natural Language Processing (EMNLP-2007))*, pages 117–124, Somerset, NJ, 1997.

[11] D. Lin and P. Pantel. Concept discovery from text. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-02)*, pages 577–583, Taipei, Taiwan, 2002.

[12] D. Hindle. Noun classification from predicate-argument structures. In *Proceedings of the 28rd Annual Meeting of the Association for Computational Linguistics (ACL-1990)*, pages 268–275, Pittsburgh, PA, 1990.

[13] H.; Dagan I.; Szpektor, I.; Tanev and B. Coppola. Scaling web-based acquisition of entailment relations. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 41–48, Barcelona, Spain, 2004.

[14] D. Ravichandran and E.H. Hovy. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*, pages 41–47, Philadelphia, PA, 2002.

[15] R. Girju, A. Badulescu, and D. Moldovan. Automatic discovery of part-whole relations. *Computational Linguistics*, (32(1)):83–135, 2006.

[16] J. Bos. Invited talk. In *2nd Workshop on Ontology Learning and Population: Bridging the Gap between Text and Knowledge*, Sydney, Australia, 2006. Association for Computational Linguistics.

[17] M. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING-92)*, pages 539–545, Nantes, France, 1992.

[18] P. Pantel and D. Ravichandran. Automatically labeling semantic classes. In *Proceedings of Human Language Technology conference / North American chapter of the Association for Computational Linguistics annual meeting (HLT/NAACL-04)*, pages 321–328, Boston, MA, 2004.

[19] P. Pantel. Inducing ontological co-occurrence vectors. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005)*, pages 125–132, Ann Arbor, MI, 2005.

[20] M. Berland and E. Charniak. Finding parts in very large corpora. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics (ACL-1999)*, pages 57–64, College Park, MD, 1999.

[21] P. Pantel, D. Ravichandran, and E.H. Hovy. Towards terascale knowledge acquisition. In *Proceedings of the 21st International Conference on Computational Linguistics (COLING-04)*, pages 771–777, Geneva, Switzerland, 2004.

[22] G. S. Mann. Fine-grained proper noun ontologies for question answering. In *Proceedings of SemaNet' 02: Building and Using Semantic Networks*, pages 1–7, Taipei, Taiwan, 2002.

[23] D. Downey, O. Etzioni, and S. Soderland. A probabilistic model of redundancy in information extraction.

In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05)*, pages 1034–1041, Edinburgh, Scotland, 2005.

[24] R. Snow, D. Jurafsky, and A. Y. Ng. Learning syntactic patterns for automatic hypernym discovery. In *Proceedings of the 7th Neural Information Processing System Conference (NIPS-05)*, Vancouver, Canada, 2005.

[25] B. Roark and E. Charniak. Noun-phrase co-occurrence statistics for semi-automatic semantic lexicon construction. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING-98)*, pages 1110–1116, Montreal, Canada, 1998.

[26] S. Caraballo. Automatic acquisition of a hypernym-labeled noun hierarchy from text. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL-1999)*, pages 57–64, College Park, MD, 1999.

[27] Z. Harris. *Distributional structure*, pages 26–47. New York: Oxford University Press, 1985.

[28] R. Basili, M.T. Pazienza, and M. Vindigni. Corpus-driven learning of event recognition rules. In *Proceedings of Workshop on Machine Learning for Information Extraction workshop held in conjunction with the 14th European Conference on Artificial Intelligence (ECAI-00)*, Berlin, Germany, 2000.

[29] E. Agirre and G. Rigau. Word sense disambiguation using conceptual density. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, pages 16–22, Copenhagen, Danmark, 1996.

[30] S. Harabagiu, G. Miller, and D. Moldovan. Wordnet 2 - a morphologically and semantically enhanced resource. In *Proceedings of SIGLEX-99*, pages 1–8, University of Maryland, 1999.

[31] E. Agirre, G. Rigau, D. Martinez, and E.H. Hovy. Enriching wordnet concepts with topic signatures. In *Proceedings of the NAACL-2001 Workshop on WordNet and Other Lexical Resources: Applications, Extensions and Customizations*, Pittsburgh, PA, 2001.

[32] W. Gale, K. Church, and D. Yarowsky. A method for disambiguating word senses in a large corpus. *Computers and Humanities*, (26):415–439, 1992.

[33] J. Justeson and S. Katz. Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, (1):9–27, 1995.

[34] T.M. Cover and J.A. Thomas. *Elements of Information Theory*. John Wiley and Sons, 1991.

[35] T.L. Brown, H.E. LeMay, and B.E. Bursten. *Chemistry: The Central Science*. Prentice Hall, 2003.

[36] D. Day, J. Aberdeen, L. Hirschman, R. Kozierok, P. Robinson, , and M. Vilain. Mixed-initiative development of language processing systems. In *Proceedings of Fifth Conference on Applied Natural Language Processing (ANLP-97)*, pages 348–355, Washington D.C., 1997.

[37] S. Siegel and N. J. Castellan Jr. *Nonparametric Statistics for the Behavioral Sciences*. McGraw-Hill, 1998.

[38] M. Winston, R. Chaffin, and D. Hermann. A taxonomy of part-whole relations. *Cognitive Science*, (11):417–444, 1987.

[39] R. Girju. Automatic detection of causal relations for question answering. In *Proceedings of ACL Workshop on Multilingual Summarization and Question Answering*, pages 107–114, Sapporo, Japan, 2003.

[40] C. Corley and R. Mihalcea. Measuring the semantic similarity of texts. In *Proceedings of the ACL Workshop on Empirical Modelling of Semantic Equivalence and Entailment*, pages 13–18, Ann Arbor, MI, 2005.

Part V: Methodology

The TERMINAE Method and Platform for Ontology Engineering from Texts

Nathalie AUSSENAC-GILLES [a,1], Sylvie DESPRES [b] and Sylvie SZULMAN [b]

[a] *IRIT, UMR5505 CNRS, Univ. Toulouse, France*

[b] *LIPN, UMR7030 CNRS, Univ. Paris 13, France*

Abstract. Designed about ten years ago, the TERMINAE method and workbench for ontology engineering from texts have been going on evolving since then. Our investigations integrate the experience gained through its use in industrial and academic projects, the progress of natural language processing as well as the evolution of the ontology engineering. Several new methodological guidelines, such as the reuse of core ontologies, have been added to the method and implemented in the workbench. It has also been modified in order to be compliant to some recent standards such as the OWL knowledge representation.

The paper recalls the terminology engineering principles underlying TERMINAE and comments its originality. Then it presents the kind of conceptual model that is built with this method, and its knowledge representation. The method and the support provided by the workbench are detailed and illustrated with a case-study in law. With regard to the state of the art, TERMINAE is one of the most supervised methods in the trend of ontology learning. This option raises epistemological issues about how language and knowledge can be articulated and the distance that separate formal ontologies from learned conceptual models.

Keywords. Ontology engineering, natural language processing, knowledge acquisition from text, conceptual modeling method, modeling software, terminology.

1. Introduction

In knowledge engineering, early motivations for the definition of ontologies mainly were the reuse of domain knowledge models and interoperability across knowledge-based systems. This goal dates back to the mid-80's. Then, very few applications using ontologies were related to texts. Among these few cases was the Menelas project, where an ontology is used as a semantic resource for multi-lingual natural language processing [1]. In the 90's, the increasing success of ontologies justified the definition of methodological processes for building them, like the MethOntology approach [2]. Still, existing ontologies and human expertise were the main knowledge sources, and texts were scarcely mentionned. These methods insisted rather on engineering guidelines inspired from software engineering, but did not pay much attention on the difficulty to identify, structure and then validate the knowledge represented in the ontology. Neither did they worry about

[1]Corresponding Author: Nathalie Aussenac-Gilles, IC3 team, IRIT, UPS, 118, route de Narbonne, 31062 Toulouse Cedex 4, France; E-mail: aussenac@irit.fr

the problem of listing concept instances in order to form a knowledge base described according to the ontology.

The gap between texts and knowledge models has been bridged after the convergence of research in knowledge representation and semantic networks for natural language processing (i.e. Skuce [3] or Sowa [4]) and a critic of Wuster's constructivism by terminologists [5]. This cross-disciplinary work benefited from recent progress in natural language processing (NLP) and information extraction. These systems could reduce the time spent on knowledge acquisition with domain experts thanks to the mining of texts as knowledge sources. This evolution prepared the ground for ontology and terminology engineering from texts.

The TERMINAE method for ontology engineering from texts refers to this trend which has been promoted in France by the French TIA[2] special interest group. Like for Simperl et al., the main motivation for formalizing a method was to give precise hints to knowledge engineers [6]. The main options of ontology engineering from texts were stated right from the start: the importance of text selection, the gain brought by NLP for text analysis, the necessary human post-processing to check ontological criteria, the change of scope required for an effective formalization. At first, the method insisted on term extraction and concept identification, providing little hints about relations and axioms [7]. It has been refined over the years after several experimental evaluations.

Since its definition, the TERMINAE method is supported by a specific tool, the TERMINAE platform[3], which provides guidance to conceptualization activities. This tool integrates functions for linguistic analyzes and conceptual modeling, it imports results from term and relation extractors, and makes it possible to reuse existing ontologies. One of its key features is to trace knowledge items from their linguistic form in texts to their modeling in an ontology, and back. The intended users are knowledge engineers and terminologists.

Designed about ten years ago, the TERMINAE method and plaform have been going on evolving since then [7]. This paper reports on recent advances related to both the method and the software. Our methodological investigation takes its roots in the experience gained through its use in industrial and academic projects or case studies, in the advances of NLP technologies as well as in the evolution of ontology engineering. It has led to several new methodological guidelines, such as the reuse of core ontologies. The platform has also been modified in order to be compliant to some recent standards such as the OWL [8] representation language.

The paper first develops most of the statements that originated the TERMINAE method (section 2). In particular, we underline the necessity of an epistemological discussion about how language and knowledge can be linked when building knowledge models from text. Then we list and comment some of the method principles, which refer to terminology and semantic analysis (section 3). Section 4 is dedicated to a presentation of TERMINAE's knowledge representation at various levels. Then we detail the method and the use of the platform in section 5, focusing on the conceptualization stage. We stress recent advances and compare TERMINAE with other methods and tools from the state of the art in section 6. In the conclusion, we sum up TERMINAE's strengths and limitations, and draw some perspectives for a better guidance to get knowledge models that actually are ontologies.

[2] http://tia.loria.fr/ , April 2007

[3] The TERMINAE software is written in Java. It is developed at LIPN by S. Szulman

2. Knowledge Modeling from Texts

2.1. Early Knowledge Acquisition from Texts

Knowledge modeling from texts emerged as a promising way to save time and to improve the quality of domain models in the early 80's. In France, early works were the KOD method and the K-station workbench [9], where texts in French could be browsed to identify knowledge fragments. They were the first attempts to consider texts as major knowledge repositories, and approaches in linguistics and terminology as the proper way to explore them before knowledge modeling. Knowledge acquisition from texts used to include two major trends [10]: natural language processing tools for the automatic "translation" of texts into knowledge bases; hypertext editors that guided text browsing and fragmentation into meaningful sequences (like those available for CommonKADS).

Even more than these practical reasons, new statements about the grounding of knowledge in language motivated a new direction for investigations in the early 90's [3]. In France, researchers from the fields of terminology, natural language processing (NLP), corpus linguistics, knowledge engineering and philosophy collaborated in the TIA group to share both theoretical issues and experiments in building thesaurus, terminologies and ontologies. This convergence was made possible thanks to the maturity of shallow and robust NLP together with the availability of large amounts of digital documents. Moreover, the evolution of terminology and corpus linguistics brought a new insight on the possible connection between concepts and their linguistic realization [5].

2.2. Anchoring Models in Language

A part of terminology as a discipline has progressively abandoned the normative view of the Vienna Circle as stated by Wüster in his General Theory of Terminology in the 30's. According to this paradigm, terms are monosemic within specialized domains and correspond to the linguistic form of pre-existing and stable concepts. The on-going dynamics of knowledge and language, the linguistic cross fertilization of specialized domains and the increasing number of available terminologies with diverging conceptualizations in each specific domain break down most of the hypotheses of this theory [11]. Various conceptual models may be defined in one domain according to their purpose use, and they may evolve over time.

Although the traditional view still remains active, an alternative position emerged that promotes a reverse process where linguistic expressions are the root from which sets of concepts can be identified and defined [12]. This view of terminology stresses that domain knowledge lies in the sum of all the discourses produced by the domain members, whether written texts, handbooks, technical notes, terminologies ... or oral expert definitions. The semantics of terms is then considered both as textual (the meaning of terms results from combining the interpretations of all its certified uses in texts) and differential (terms are defined in contrast with the meaning of other terms) [13]. These semantics meet the analyzes and theories conveyed by corpus linguistics [14]. If terms are defined from the interpretation of the sentences where they occur, concepts then are normalized meanings. Their definition results from a restriction process: among all possible interpretations, some are selected according to a specific purpose and domain. This definition is neither definitive nor rigid: it could evolve over time since domain knowledge, termi-

nologies and even domain frontiers may be modified. Because the nature of the terminological networks obtained from this *terminology engineering* process are very close to ontologies, this trend has soon be considered as a possible contribution to ontology engineering [15].

2.3. A New Paradigm for Text Analysis

Within this new perspective, many approaches suggested to carry out a shallow text parsing, such as lexical, syntactic or semantic analyzes of a selected corpus of texts, to build up semantic resources likes terminologies [16]. The NLP tools used in this context are much less ambitious than those required for text understanding. They provide the knowledge engineer with means to identify terms, to compare the contexts in which they are used and to define concepts, or even to set relationships between them. These tools are all the more efficient as they can be adapted to each application. For instance, Harris suggests that terms that appear in similar syntactical contexts may share similar meanings. It relies on several statistical measures, the so-called distributional analysis, to compare contexts and propose term clusters that lead to define concepts [17].

3. TERMINAE: Principles

The TERMINAE method has been defined within the TIA group and belongs to this research stream. We report here some of the grounding principles that influenced it. These options formed an innovative paradigm for ontology engineering at the time when the method was proposed, around 1999.

3.1. Characteristics of the Models Built with TERMINAE

Models based on the language in use One major claim of TERMINAE is that language as it is used in texts can provide relevant information on domain concepts. The practical incidence of this choice is a tight connection between the models under construction and linguistic data and even texts. The first connection is established through the lexicon. Term extraction software tools produce candidate terms, that additional information (statistic criteria like frequency, productivity, or linguistic criteria like contexts of use, ...) helps to select as labeling concepts. Not all nouns are concept labels. Additional information is required to state that a noun phrase or a verb phrase can form a term. The second means used to anchor the conceptual model in language relies on predicates (mainly verbs). They connect or define concepts, or they are part of patterns that reveal lexical relations.

Domain and task specific models The models built with TERMINAE should integrate, in their lexicon, the diversity of the domain terminology identified in texts and, as conceptual models, they should take into account the application that will use them. As a consequence, these models are domain- and task-specific: concept definitions result from the selection of a single interpretation context that reflects the application requirements; they are intended to reflect one of the ways knowledge can be perceived through the use of language in documents. These models will not contain universal and unique definitions of concepts in reference to some semantic primitives unless these concepts are reused from a generic formal ontology [18].

Models that reflect a compromise among knowledge sources Depending on the user community, concept definitions can be the result of negotiations among domain experts or a compromise between various text sources. Thus the ontology is supposed to be the most consensual knowledge among the user community, and, at the same time, the most relevant one for the application. It is useful to extend the ontological commitment as long as this does not conflict with the perspective needed for the application and the user community.

Conceptual models ranging from terminologies to ontologies Terminology engineering may produce terminologies, kernels of light-weigh ontologies, taxonomies or conceptual networks. If no human expertise or general knowledge is added, bottom-up text analysis leads to a draft semantic network or a taxonomy of concepts related to terms. TERMINAE proposes a supervised validation of this draft and syntactical checking in order to get a relevant terminology or semantic network. If the application requires a precise and formal model, the model can be refined to get an ontology. TERMINAE suggests to apply ontological criteria for concept definition borrowed from other methods, such as Archonte differentiation criterion [19] or OntoClean formal properties [18].

Models with a terminological component The conceptual models built with TERMINAE comprise a rich terminological component which is closely connected to the domain concepts on the one hand, to the texts from the corpus on the other hand. A rich terminology (called lexicon in [20]) associated to an ontology is an advantage for the interaction with users, for document management, meta-data extraction or document annotation.

Weakly-formal models With TERMINAE, the most precise form that a model can take is a *weakly-formal ontology* that can be represented in a language like OWL. By *weakly-formal ontology* we mean a light-weigh ontology (without any rule or axiom) where most of the relations have no other formal semantics than the type of their range and domain. The output model may be represented with a more or less formal and expressive language. Nevertheless, concepts are rather less specified compared with those defined by formal ontology.

3.2. Characteristics of the approach

Given these features of the model that can be built with TERMINAE, here are some of the main options of the method.

TERMINAE uses texts as knowledge sources The way terms are used reveals a lot of information about their actual meaning and related knowledge, which is complementary and sometimes orthogonal with the standard definitions given in dictionaries or terminologies. Keeping connections from the model to the source texts improves its documentation and makes it easier to maintain.

It reuses additional resources to overcome text limitations We have experimented the limitations of text-based ontology engineering. Texts may be of very different nature, content and genre [14], and their processing may lead to non-homogeneous knowledge. Most of the ontological knowledge is often missing. For instance, only pedagogical handbooks or documents that popularize technical information contain a lot of explicit definitions [21] [22]. On the contrary, texts present counterexam-

ples that may not be relevant. TERMINAE suggests to refer to a domain expert for validation and as a source of knowledge when it is missing in texts. Additional resources may also be used such as existing thesaurus or terminologies, and available ontologies. The gain brought by core-ontologies is significant for structuring concepts.

It makes use of various NLP tools Learning relevant knowledge from texts requires application of NLP tools: what can be identified are terms, indications about their use (frequency, distributions, collocations, dependencies, ...), lexical or semantic relations between noun or verb phrases, semantically rich contexts, lexical or syntactic regularity, term variations, definitions, etc. NLP provide better results if they run on tokenized or tagged texts. So most of the tools suggested by TERMINAE use texts tagged with TreeTagger[4]. Up to now, experiments have been carried out using specific tools (two term extractors, a concordancer, a pattern matching system and a synonymy relation extractor). The TERMINAE platform proposes specific import facilities and interfaces for browsing their results.

It promotes a supervised process TERMINAE insists on the need for human intervention to supervise the process of building a semantic resource. Considering the definition of ontology and the obvious limitations of texts as knowledge sources, involving a knowledge engineer is the only way to take into account the application needs and other knowledge selection criteria. Supervision takes place all along the modeling process, when selecting or preprocessing the results of NLP tools, when defining and formalizing concepts, when judging their relevance with respect to the application needs. These evaluations have an important impact on the validity and quality of the ontology. They lead to a more consensual representation.

Concepts result from a normalization process Concepts are the result of a normalization process: only some features of domain objects or ideas are represented in concepts. These features depend on a particular insight on the domain that the knowledge engineer should have in mind. The selected features are those necessary (and may be sufficient) to define the concept in a way that is relevant for the application. Normalization results in setting each concept in the subsumption hierarchy, in defining its relations with other concepts or its properties. Normalization is one of the conditions to reach a motivated concept description that reflects a consensual view and anticipates further uses of this ontology in close contexts.

It supports the early stages of ontology engineering TERMINAE provides guidelines for the early stages of ontology engineering from texts: resource gathering, linguistic analysis and conceptual modeling. The focus is conceptual modeling and normalization. This stage is carried out following a cyclic process where alternatively linguistic data are mined and the conceptual model is enriched. Although formalisation can be done with OWL language as an export of TERMINAE models, only a part of the OWL primitives can be used. So formalisation should be better carried out in another platform like Protégé-OWL and following a method like OntoClean [18] or Völker's method presented in this book [23].

[4] http://www.ims.unistuttgart.de/projekte/corplex/TreeTagger/

4. TERMINAE: Knowledge Representation

Two main requirements influenced the choice of knowledge representation in TERMINAE. Firstly, knowledge should be represented in a way that makes it easier to go from linguistic data to the conceptual model and back. Secondly, this representation must be compliant with the formalization in description logics, which was supported by TERMINAE right from its early version. So some specific structures have been defined at three different layers: terminological, conceptual and formal layers.

4.1. Terminological Representation

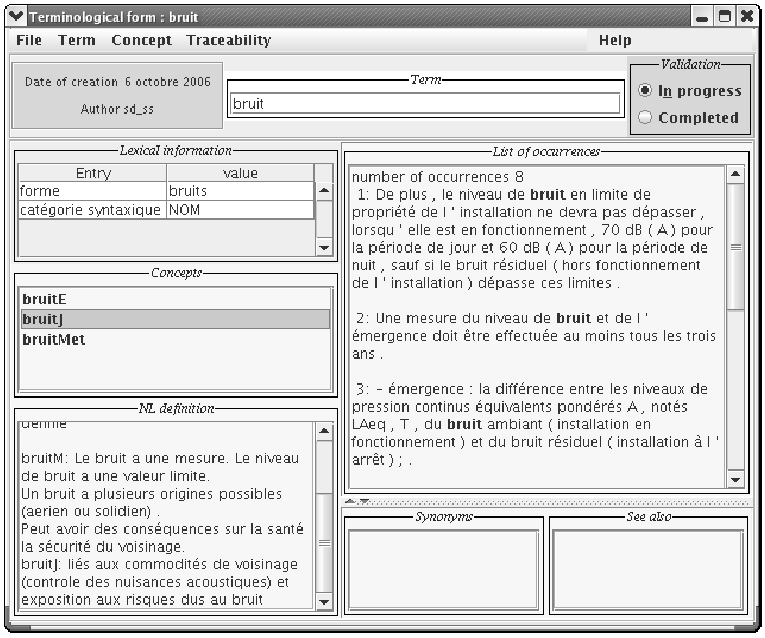

Figure 1. Terminological form of "*bruit* (noise)"

Terminological forms are specific data structures defined to store all the available information related to terms and extracted using NLP techniques. Inspired from those used by terminographers, they present grammatical information and documentary metadata (date of creation, author). For each meaning of the term, the form can be filled with a definition in natural language that makes this meaning explicit (in general, in keeping with the way the term is used in the corpus), a concept, all the term occurrences where it has this meaning and its synonyms, as shown in Figure 1. Terminological forms can be reused for new applications. They can be imported and lead to the definition of the concepts corresponding to the terms in the forms. Several concepts may appear in the terminological form of a polysemic term, but a concept generally appears only in one terminological form that gathers all its synonym terms.

4.2. Conceptual Representation

The semantics of concepts proceeds first of all from their label but also from their situation in the model and in partiular their relations to other concepts: their situation in the hierarchy of classes and sub-classes, inherited relations and specific relations to this concept. Relations are represented as roles. Their meaning results of the human interpretation of their label and the formal verification of the concepts that can be their domain and range. Some roles may be defined as restrictions of inherited roles.

To help the understanding and maintenance of the model, concepts have specific features to express why they have been introduced in this model. Concepts may be defined because related terms have been found in texts : this feature is stored in the *linguistic dimension* attribute of a concept (which may have as value terminological or not-terminological). Concepts may also have been defined while expanding the model by abstracting (bottom-up value) or refining already defined concepts (top-down value); this information is stored in a feature called *structuring dimension*.

These two features take independent values. They qualify the concept before its formalization. So they affect neither the classification nor any other inferential process. These features are used as hints by the knowledge engineer when building or maintaining the model, and they reflect his/her own experience in knowledge engineering. They emphasize a modeling point of view on the concept, following two methodological dimensions: the usual structuring dimension, and the linguistic one that we have introduced.

4.3. Formal Representation

Knowledge representation formalisms, like conceptual graphs or description logics, describe knowledge at an epistemological level, in the sense of [24]. They provide logical primitives, the meaning of which is domain-independent, and syntactic rules to build language formulae. They also provide rules for semantic composition, which allows the sense of a formula to be computed from the sense of the primitives which compose it. But the definition of the non-logical primitives of the domain, the interpretation of which is given by the domain, is left to the knowledge engineer. Hence the task of building a formal ontology of a domain comes down to defining the non-logical primitives of the representation [25].

The representation language used in TERMINAE belongs to the family of description logics. The description logics describe concepts by their necessary and sufficient conditions in order to organize them into a taxonomy of subsumption along which the properties are inherited, and to classify new concepts in this hierarchy according to their properties. Formal representation makes it possible to check some validation criteria for the concepts and relations.

4.4. OWL export of the model

Current knowledge representations used for ontology engineering are somehow too weak to account for the complexity of term-concept relationships [26]. For instance, in OWL [8], concepts are represented by classes that may have several *label* properties. Each label is a term that may be used to refer to this concept. These terms cannot be described by independent classes. Terms are accessed through concepts and no information can be

added to them. On the other hand, the data model defined for terminological knowledge bases proposes that terms are specific classes with properties like use indication, possible variations or grammatical information. These structures are related to concepts, which makes it a powerful way to represent term variations, synonymy, and homonymy [5], [27]. New standards able to represent both conceptual and terminological information are currently being defined for semantic web applications, like Skos [5].

TERMINAE exports a conceptual model built with the tool in the OWL representation language. In order to keep the benefits of a rich terminological component, the terminological files are taken into account. To overcome the limitations of OWL labels, the terms and the sentences containing their occurrences are stored as concept annotations (in the OWL meaning) [26].

5. TERMINAE: Method and Tool

5.1. Overview of the Method

Most of the ontology engineering methods distinguish at least the following phases: feasibility study, requirement analysis, conceptualization and finally deployment [2]. Deployment typically consists in a loop of application, evaluation and maintenance of the ontology [28]. The TERMINAE method covers only the conceptualization phase and suggests to split it up into four steps: domain resource (including a corpus) gathering, linguistic analysis, conceptual modeling and formalization. The last three steps are performed as many times as required during a cyclic process. They include various tasks as shown on figure 2 and that we will detail in the next sections. A tight loop includes (1) the exploration or mining of the results of some NLP tools on the corpus and (2) manual modeling with the definition and structuring of concepts and relations.

TERMINAE has been applied successfully in many projects connected with different application purposes such as the detection of inconsistencies in telecommunication services [29] or the elaboration of conceptual models used for document classification or annotation in various domains:

- glass-fiber making [30]
- break-down diagnosis of car electronic components [31]
- road safety from accident reports [32]
- the "worker" concept through European directives [33]
- legislation related to hygiene, security and environment (HSE) [34]
- archaeology of techniques [35]

Therefore, the design of TERMINAE integrated the feed-back brought by all these practitioners for whom it tends to become more and more useful. The models built with TERMINAE up to now have ranged from 200 to 1000 concepts and consisted of about 50 non-taxonomic relations.

The TERMINAE tool is modular and consists of specific components that perform the tasks suggested by the method and lead to a draft ontology in a formal language. A more precise and powerful formalization would require to export this draft ontology in

[5] http://www.w3.org/2004/02/skos/

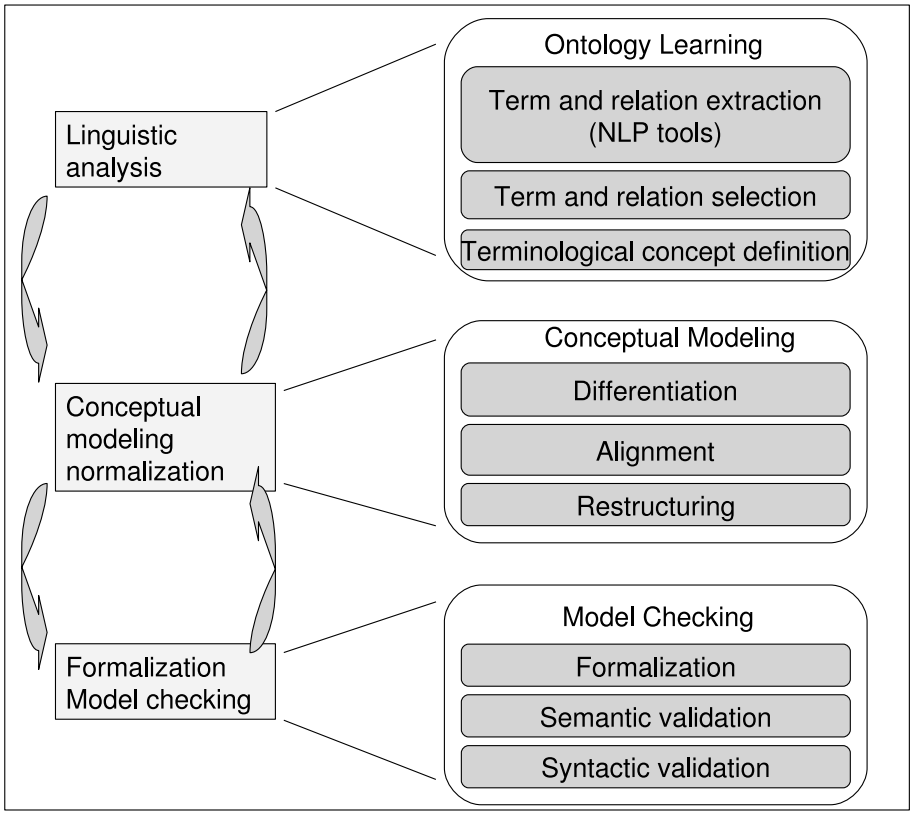

Figure 2. Overview TERMINAE main tasks

an editor like Protégé OWL. A global data structure, mainly a set of XML files, is used to facilitate the exchange of data and results between the components. Although these components are integrated into the platform and they the may be used separately.

In this section, we detail how to carry out the three first steps of the method and how the TERMINAE platform can be used all along them. The method is exemplified through a use case from the legal domain. The goal was to build an ontology of the HSE domain from French regulations.

5.2. Gathering Domain Resources

The first step of TERMINAE aims at gathering most of the knowledge sources that will be used to build a model. In [7], the method used to insist only on texts and this first step resulted in collecting texts to form a corpus. To improve the quality of the resulting models and save time in identifying or defining concepts, the current method also promotes the reuse of existing resources. We provide guidelines for these two tasks.

5.2.1. Collecting a Corpus

A corpus gathers texts carefully selected among the available domain documentation according to various criteria and guidelines. Compilation of a suitable corpus is in itself labor-intensive.

Soergel suggests that text selection could be automated or at least computer-assisted by testing the meta-data in the documents or by classifying them on the basis of their content [36]. In the case of ontology or terminology engineering, TERMINAE proposes the following criteria:

- texts should cover the domain specified in the application requirements that explain the objectives underlying the model development;
- they should share common properties: similar content, same or complementary production periods, authors belonging to the same group, similar or complementary content to cover a given domain, digital format ...;
- they can be trusted and reflect a valid opinion in the domain;
- their content will provide results after being processed by NLP tools;
- they are rich in ontological information to extract.

Text selection may be easier with the help of a domain expert or a librarian. As specialists of the documentation that they use, produce or organize for their activity, they know which documents could be selected. They also can give precise information about them, such as size, authors, date, genre, type and content. In the early stages of linguistic analysis, the corpus may be considered as not relevant and modified.

In our case-study, the corpus consists of five textual documents selected by the experts among 500 available HSE regulation texts. Their sizes comprise of between 2 and 150 pages. They contain terms about HSE trades, environmental conservation and social welfare. These five documents are the following ones :

- a decree related to classified installations for the environment protection : *Décret n^o 77- 1133 du 21 septembre 1977 pris pour l'application de la loi n^o 76-663 du 19 juillet 1976 relative aux Installations Classées pour la Protection de l'Environnement*;
- a decree related to the introduction of new batteries on the market and to their disposal: *Décret n^o 99-374 du 12 mai 1999 relatif à la mise sur le marché des piles et accumulateurs et à leur élimination*;
- a decree related to the functioning outcome provided by another decree: *Arrêté du 29 juin 2004 relatif au bilan de fonctionnement prévu par le décret n^o 77-1133 du 21 septembre 1977 modifié*;
- a decree related to combustion: *Arrêté type - Rubrique n^o 2910 : Combustion*;
- a decree related to the control of disposal recycling circuits: *Décret n^o 2005-635 du 30 mai 2005 relatif au contrôle des circuits de traitement des déchets*.

Each text is characterized by a heading defining the type of text (law, application decree, European directive, ...), a title and a list of references to other texts. It is structured in articles describing the application domain of the text with definitions, the applicability conditions of the articles, the application date and the person in charge of its application.

5.2.2. Reusing Existing Ontologies and Terminologies

Existing terminologies, thesaurus or ontologies are valuable resources to guide concept structuring, to improve reusability or to add new domain terms. TERMINAE makes it possible to import various kinds of resources:

- *ontologies* represented with OWL or RDFs may form the kernel of the domain model of a new project; they can be added to a model under construction;
- *thesauri* can be imported either as linear lists of terms or as hierarchies to be integrated in the current model;
- *conceptual models* built with the TERMONTO user interface of the SYNTEX term extractor can also form the root for an ontology [30].

Adding an existing ontology assumes that its concepts and roles have a precise and formal meaning. It may require to align the reused ontology with the existing model in order to recognize similar concepts, to avoid duplication, to connect both representations and to correctly share roles. In the case of reusing a high-level or core ontology, it is important to identify in the reused ontology which concepts will play the role of *anchor concept* as proposed in [37]. Such concepts have direct children concepts in the domain conceptual model.

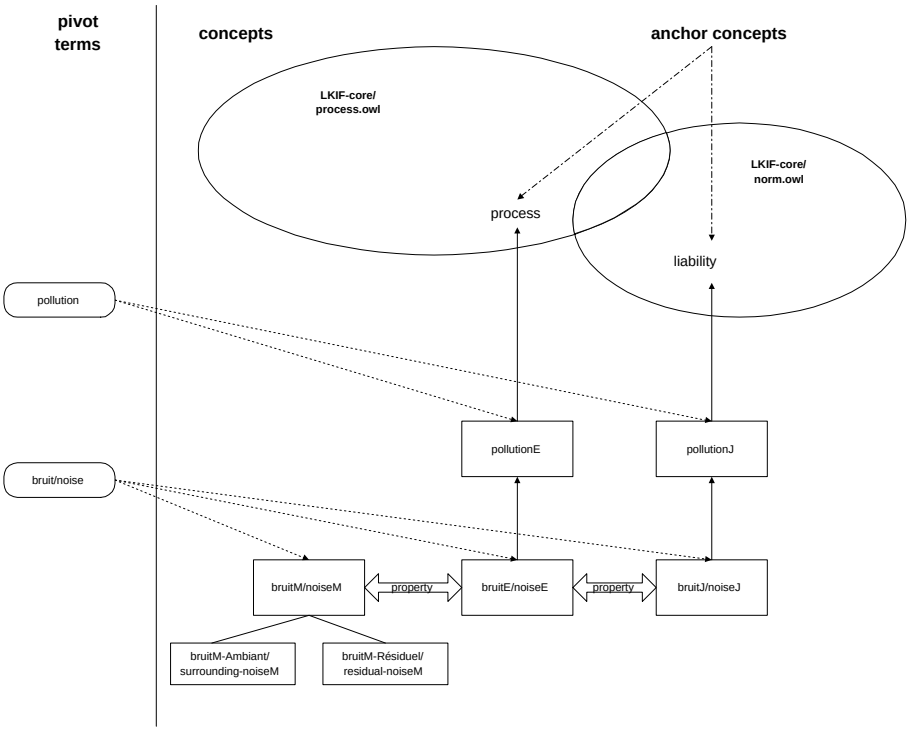

Figure 3. From terms to concepts

In our case-study, we used the LKIF-core[6] ontology as a bootstrap. It provided high level concepts to which we anchored more specific domain concepts. The definition of LKIF-core concepts helps to refine and formalize the definition of domain concepts. PROCESS and LIABILITY were identified as anchor concepts as shown on Figure 3.

[6]http://www.estrellaproject.org/lkif-core/doc/index.html , April 2007

5.3. Linguistic Analysis

This step consists in (1) running several NLP tools on the corpus, (2) collecting results of corpus processing like terms and lexical relations and (3) if required, cleaning these results. They will be further refined and used to enrich the model. The input of this stage are digitalized corpus documents in the format required by the NLP tools. The output are linguistic data identified in texts as relevant because they potentially refer to some piece of domain knowledge.

Linguistic analysis is carried out several times. At the beginning of the modeling process, it provides a bootstrap of linguistic data that will be explored according to some criteria in order to get domain knowledge. Later on, once a draft of the model has been built, it helps searching for some specific information in the corpus. We will call this process *focused text mining* in the following. We will give guidelines for this model-driven process at the conceptual modeling step.

5.3.1. Bootstrap Linguistic Analysis

The purpose of bootstrap linguistic analysis is to gather all potential terms (the so-called term candidates [30]), syntactic and semantic relations, statistics about terms, occurrences or term clusters that could be relevant for knowledge modeling. Another purpose may be to reduce the noise in the results by getting rid of errors or marginal results. But this validation is quite time consuming. It may be avoided if the list of lexical entries is not to be kept as a result in itself. It is easier to choose relevant term candidates than to eliminate useless or wrong ones.

5.3.2. How the Platform Supports Linguistic Analysis

Recommended NLP tools

Several NLP tools with different goals may be used to get this input data. For instance, generic tool suites like GATE [38] or dedicated algorithms for relation extraction like Espresso and Macchiato [39] can be useful. But systems defined for the extraction of relations between instances, generally for the purpose of semantic annotation (like Rote Extractors [40]), are not appropriate at that time. We mention here bellow the software tools for which the TERMINAE platform provides interfaces to import, browse and filter their results through the "linguistic study" menu of the platform main frame. Most of them may be used on French or English corpora:

- *term extractors* that also provide statistics and syntactic relations about terms: LEXTER, SYNTEX [41] or YATEA [42];
- *synonym extractor*: SYNOTERM [43] detects synonymy relations between the extracted terms from a corpus, or between extracted terms and terms for a thesaurus;
- *concordancers*: TERMINAE has its own integrated concordancer, LINGUAE;
- *lexical-relation extractor*: it will be possible soon to import results from CAMÉLÉON [44] and MFD [45].

Gathering terms

The TERMINAE platform provides facilities to browse the results of term extractors and clean them from noise by removing or gathering term candidates. Term candidates that

are erroneous or not suitable for the application can be selected by hand and deleted from the lists. Term variants can be selected and clustered as one unique term candidate that refer to the union of their occurrences. Synonymy should not lead to term gathering, but to an explicit synonymy relation between terms in the terminological forms.

We illustrate the support brought by TERMINAE with the visualization of results obtained from YATEA. For each term candidate ("*bruit* (noise)" on the left part of Figure 4), the graphical user interface shows its occurrences (right part of Figure 4). Some rules also are available for massive deletion of terms on syntactical bases, like one letter terms, numbers or so, that can be fired on demand.

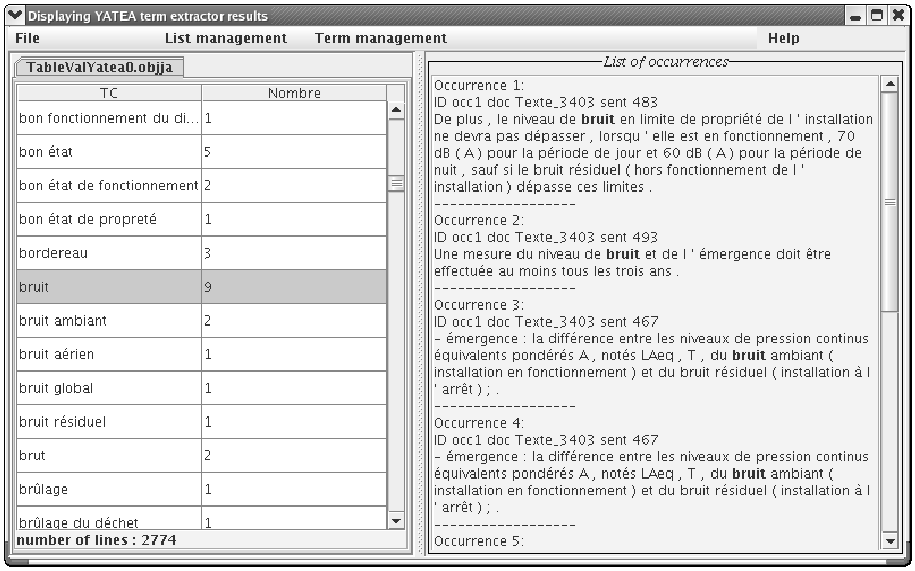

Figure 4. TERMINAE user interface for the evaluation of YATEA results

Collecting relations

Paradigmatic lexical relations between terms are one of the means to decide which terms could be relevant to define domain concepts, to get useful information for defining concepts and semantic relations.

Detailed reviews of existing techniques and tools for relation extraction can be found [47] or in the introduction paper of [16]. Endogeneous approaches try to learn relations only from linguistic regularity inside the corpus, without any additional resource. Even relation patterns can be learned from tagged corpora. They may rely on statistical measures, the measure of co-occurring terms or distributional analysis [41]. Exogeneous systems use linguistic resources, like lexical data bases or generic relation patterns. Pattern-based approaches require human interpretation of their results. Because it is complex and time consuming, some systems use lexical resources or statistics to select the best results, identify the possible relation meaning and related concepts. They automatically learn semantic relations between concepts which later require human validation.

So far, CAMÉLÉON [44], LINGUAE and MFD [45] have been used with TERMINAE for this task. MFD relies on association rules to find lexical relations. CAMÉLÉON implements pattern matching and proposes a collection of validated patterns for French. They can be adapted to any new corpus. LINGUAE can be used to encode and match domain-specific patterns, which can be capitalized for each project. Nevertheless, no default generic-pattern list is given in LINGUAE. The matched sentences on a given corpus can be stored and browsed later on.

The knowledge engineer can use either TERMINAE interface for LINGUAE results, or CAMÉLÉON or MFD own interfaces. From reading the linguistic data, he defines concepts and relations in the conceptual model. For each sentence matched with a pattern, he must decide the precise relation meaning and which are the related concepts.

Looking for specific linguistic data or phenomenon

Concordancers (like LINGUAE or KESKYA [44]) and information extraction platforms like GATE [38] may prove very conveniant for focused text mining. When looking for some knowledge related to one or several concepts, the knowledge engineer is generally able to characterize the linguistic form of this knowledge. This characterization defines a search pattern in a concordancer. Patterns can be based on lexical entries, semantic classes, syntactical categories or even punctuation, document structure ..., depending on the tags available in the documents that have been processed with NLP tools.

The LINGUAE concordancer included in TERMINAE allows pattern definition and recognition. In the HSE corpus the "*bruit* (noise)" term is very often associated with "*vibration* (vibration) ". By looking at their linguistic contexts obtained with LINGUAE, we have had confirmation that these two terms were used identically in the environmental and legal documents.

5.4. Conceptual Modeling and Normalization

During this stage, a conceptual model documented with a terminological component is built up from the interpretation of the lexical data found by NLP tools (particularly terms), and from reused resources. As shown in Figure 5, the *terminological forms* play here a major role: not only they constitute the terminological component associated to the conceptual model, but also do they bridge the gap between the linguistic data and the conceptual model.

5.4.1. The Process of Conceptual Modeling and Normalization

Tasks

Conceptual modeling and normalization refer to different criteria to guide the definition of new concepts:

- During *conceptual modeling*, concepts are defined and organized according to how knowledge is expressed through language in the text corpus and in the application needs.
- During *normalization*, concept definition is guided by ontological guidelines; such guidelines may be the concepts reused from a core-ontology or they may

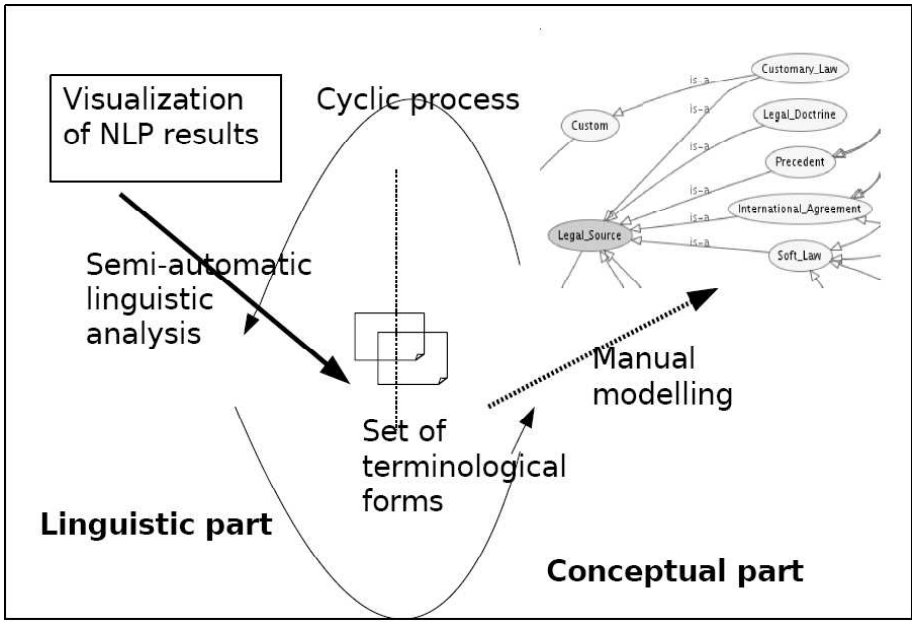

Figure 5. Overview of the TERMINAE modeling stage

be ontological principles like those proposed in OntoClean [18] or Archonte's differentiation criteria [19].

Normalization generally starts with criticizing the results of language-driven conceptual modeling. Subsets of the corpus are analyzed so that the model be progressively enriched. Linguistic criteria and ontological criteria are alternatively applied. We will detail these two tasks in the following.

In the legal case study, concept definitions were both drawn from the corpus and specific to the application. Their properties (expressed with roles) were classified in two categories: structural properties (definitional roles inherited from core-ontology concepts) and functional properties (domain specific roles added to define more precisely properties required for our specific application).

Conceptual modeling

Conceptual modeling starts with a bootstrap stage where pieces of conceptual model are built up, and goes on with the model consolidation phase, where these pieces are connected, reorganized and enriched.

The *bootstrap stage* goes from texts to a first partial model. It consists in identifying central domain terms, in defining relevant concepts and organizing them into local hierarchies. Additional linguistic indices are searched to define relations involving each of these concepts. We propose a list of guidelines for focusing on relevant terms, for defining related concepts and setting them in the hierarchy of concepts.

Model consolidation aims at connecting these local hierarchies and at enriching the model. This stage is model-driven: according to the model current content, the knowledge engineer tries to better define, detail or organize concepts and roles. So he looks

for some particular knowledge in all the available linguistic data. From existing concepts, expanding the model may consist in one of the following three processes that all contribute to improve concept definitions:

- generalization: new ancestor concepts of existing ones are searched; they define more abstract concepts or refer to high-level categories;
- specialization: for each existing concept, the goal is to make sure that its more specific concepts have been defined (all those required for the application);
- clustering consists in creating new concepts that share some properties; this may lead to the definition of non-terminological concepts.

Possible methods

In addition to the modeling criteria, another dimension influences the modeling process. We distinguish two opposite ways to carry out the conceptual modeling and normalization tasks:

- *from the model back to the texts*: the knowledge engineer may (a) browse the conceptual model and feel the need to add some knowledge ; for this purpose, he will (b) mine the linguistic data, (c) define or edit a terminological form and then (d) modify the model ;
- *from the texts towards the model*: the knowledge engineer may (a) browse some linguistic data, feel the need to take it into account in the model, (b) check the model to decide of the data relevance, (c) define or edit a terminological form and (d) modify the model.

The originality of TERMINAE is to promote this traceability in the method, to keep track of it in the model and to support it in the platform. In the conceptual model, traceability is achieved through the links that connect the corpus, the various terminological forms and the ontology terminological concepts. Each terminological concept is linked to its terminological form in which terms and their occurrences are saved. This facility should simplify the use and maintenance of an ontology.

How the platform supports conceptualization

At each step, similar tasks are carried out repeatedly but with a different perspective. These basic tasks are supported by corresponding functions on the platform :

- browsing linguistic data and selected terms to define terminological forms is possible from the linguistic analysis evaluation interface;
- browsing the conceptual model is possible through the conceptual model management frame shown in Figure 6;
- adding, modifying, deleting items in the model requires either the use of this same interface or the definition of new concepts from the terminological form editor;

5.4.2. Guidelines for Conceptual Modeling

Conceptual modeling requires to identify and structure at the terminological and conceptual levels some information selected among the results of NLP tools. TERMINAE provides some

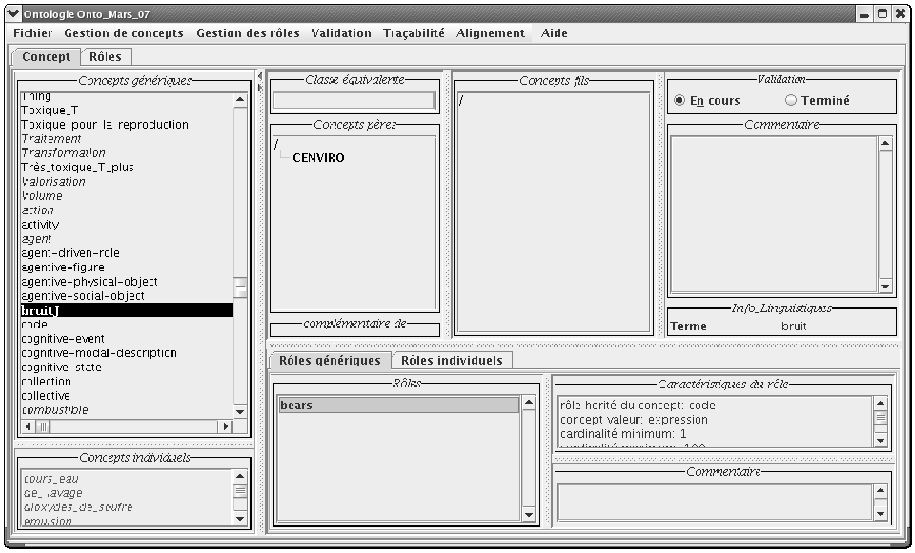

Figure 6. Defining the BRUITJ Concept through the Conceptual Model Management Frame

guidelines to select the most relevant linguistic data, to define concepts and to organize them in a model.

Guidelines for selecting linguistic data

In a model-driven process, the selected terms are those required to feed the model with concepts. These guidelines explain how to identify terms in a bottom-up process. The knowledge engineer should start with the candidate terms that are more likely to be selected to define terminological forms. TERMINAE suggests to study (a) the most frequent and productive noun phrases (phrase that are used in many more complex terms); (b) terms consisting of only one word and sharing numerous syntactic contexts with other terms; (c) noun phrases made of very frequent single terms; (d) terms that appear in titles or headlines; (e) terms suggested by a domain expert or by other semantic resources.

Later on, term selection can be guided by relation extraction: given a term and a relation pattern, by reading all the pattern occurrences, the knowledge engineer may identify new terms and lexical relations.

Selecting a term leads to the definition of a terminological form. The lexical and syntactic part of the terminological form are automatically filled in with the term label and its occurrences (obtained with YATEA or SYNTEX term extractors) and with synonyms found by SYNOTERM if any. Synonyms can also be added by hand. Then the knowledge engineer has to study the term occurrences to find its meaning or its different meanings if several are available. Each concept defined from a term is a *terminological concept*.

In the example, the "*bruit* (noise)" term is polysemic. It is used in many codes in French Law, with a different meaning in each of them. We are interested in the following three codes: CENVIRO (Environment code), CURBANI (Town-planning code) and CTRAVAI (work code). So we distinguish three meanings for this term. We call this kind of polysemic term a *pivot term* because it is common to several sub-domains or points of views on a domain. *Pivot terms* facilitate understanding across sub-domains, but they

have a slightly different meaning in each of them, which may lead to some misunderstandings. The knowledge engineer binds a distinct concept to each meaning with the corresponding set of occurrences. In the example on Figure 3 , the "*bruit* (noise)" term gives birth to three concepts : the BRUITJ concept which represents noise in the law sub-domain, the BRUITE concept which represents noise as the environmental sub-domain, the BRUITMET concept which represents noise in a technical sub-domain.

Guidelines for concept definition

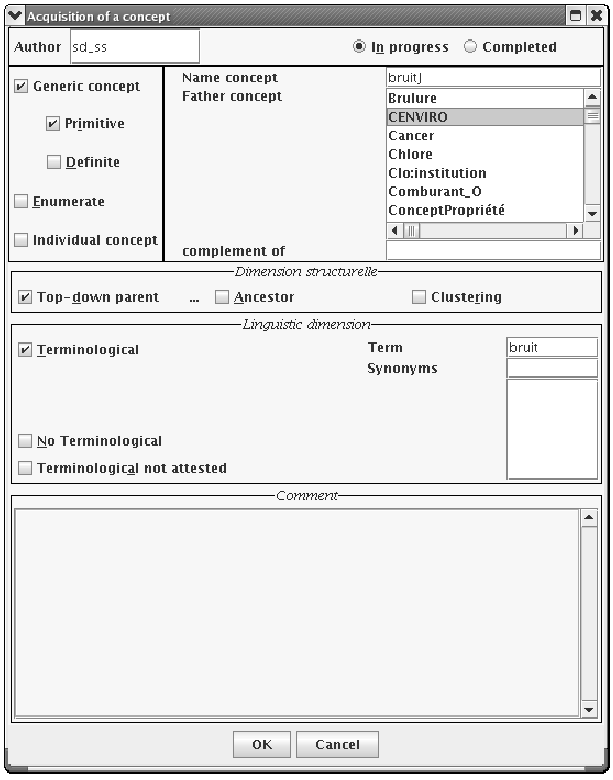

Figure 7. BRUITJ terminological concept definition

Defining a concept opens up a concept definition form like the one shown in Figure 7 where the legal concept BRUITJ is described. If several terms correspond to a terminological concept, one preferred term (the "vedette") is chosen as the concept label. The next choice is to determine the father of this concept in the hierarchy of concepts. It can be selected in the list of available concepts, or, if the desired concept has not been defined yet, TOP can be selected until this new concept is created. Then some of the concept features (its structural and linguistic dimensions) should be given a value.

For instance, the BRUITJ terminological concept defined on Figure 7 is a sub-class of CENVIRO, its father-concept which is subsumed by the CODE concept. it means that BRUITJ refers to all the codes that contain regulations about the environmental issues

related to noise. CODE is an *anchor concept*: it belongs to the LKIF-core ontology.

Top-down structuring type concepts are defined when refining high-level or abstract concepts into more precise sub-concepts. High-level concepts are good indices to partition off the domain into large sub-domains which are easier to manage. This decomposition allows the knowledge engineer to organize the conceptualization process.

In the legal case study, such top-down structuring concepts are PHYSICAL OBJECT or MENTAL OBJECT. They have been introduced in order to start a top-down organization of the domain concepts according to some fundamental classes. Many such concepts are added when reusing a core-ontology, some of them playing the role of *anchor concepts*.

Bottom-up structuring concepts are those abstracted from low level concepts or rather from terminological concepts that need to be better located in the hierarchy. Bottom-up concepts may be classes added for the purpose of a relevant structuring, for the gathering or the differentiation of some concepts with relations. These concepts may not correspond to natural domain classes or terms. They often are not terminological. Their definition requires a deeper study of the specialized domain.

Guidelines to incrementally feed the conceptual model

After the bootstrap stage, conceptual modeling is rather model driven. Defining concepts from terms leads to "local" hierarchies of concepts with natural language definitions and related terms, but concepts have no formal meaning. A deeper linguistic analysis is required to connect concepts with relations and to clean the hierarchy. The next steps suggested by the method are rather classical:

- identifying additional hierarchical relations and, with them, new concepts, to structure "local hierarchies";
- identify hierarchical relations that connect these hierarchies;
- identify other types of relations that contribute to both define concepts and check the hierarchy validity.

Hierarchical relations may be identified from the structure of noun phrases, or with specific patterns that identify definitions or hypernymy relations. Other types of relations may be identified with domain specific patterns, by exploring co-occurrent terms or by exploring terms that share similar syntactical contexts according to distributional analysis. Details on relations extraction can be found in [15], [16], [44].

Normalization

The goal of normalization is to make explicit the meaning of each concept and role, to check that concepts are unique, that definitions are precise enough and that concept decomposition in sub-classes is homogeneous. Modeling options should be explicit too.

If the objective is to build an ontology with TERMINAE, several means can be used to turn a conceptual model into a valid ontology: reusing core-ontologies and defining domain concepts with respect to those of the core ontology; applying some ontological principles. TERMINAE suggests to follow the differentiation principles proposed by B. Bachimont in Archonte and derived from early work on Ontology [19]. According to

these principles, a concept should differ from its father concept and siblings according to some criteria, eventually represented with roles. A concept should also share common properties with its father concept and siblings, also possibly leading to defining roles.

Used as a way to validate the conceptual model, these criteria suggest to check, for each concept, if it is worth being defined, if it is really different from its father concept and siblings, if these differences have been made explicit with roles, how it will be interpreted. Differentiation also assumes a shared point of view or some common features for each list of sibling concepts. These common features may be explicit or not, and may correspond to inherited roles or similar values of a common role. Applying such criteria leads to more valid and readable ontologies that can be better used, maintained and reused.

In TERMINAE, concepts are neither defined as sets of formal axioms nor as a formal conjunction of necessary and sufficient conditions including their roles, like in DOLCE [18]. In the ONTOSPEC method by G. Kassel, terms and concepts are validated according to OntoClean principles, and ONTOSPEC suggests to reuse DOLCE. An attempt has been made to integrate TERMINAE and ONTOSPEC: ONTOSPEC provided support to formalize TERMINAE conceptual models and to build formal ontologies.

Syntactic and semantic model checking

Two kinds of validation are possible on the models built with TERMINAE. On the flow semantic checking is performed every time a concept or a role is introduced in the ontology. The tool verifies the compatibility of ranges between concept roles when they are related in the same hierarchy. Moreover, an overall verification is proposed. Concepts and roles are translated into their normal form. After comparing normal forms, some suggestions are made to better classify concepts. The concept definitions which only differ by their labels are detected as errors. Either these concepts should form one concept, or should they have at least a differenciating role.

6. Discussion

Since the moment when TERMINAE was defined (1999), ontology learning and population from texts has become a very active research field [46] [20] [47] [48]. Most of the works carried out refer to similar trends as those grounding TERMINAE, but the scope of automation is generally stronger (even more for ontology population, like in the systems proposed by Navigli and Verlardi [49] or Tanev and Magnini [50] in this book). As a consequence, the role of human interpretation in the exploration of linguistic data is much less emphasized and delayed further in the process. Validation bears on pieces of conceptual model. For further information, read [36] and [16] where most of the statements and challenges of terminology engineering for ontology engineering are developed.

Although we may wish to introduce more automatic post-processing of extracted linguistic data in TERMINAE, this could not always be feasible because we use rather small corpora that cover very specific domains. Selection thresholds could be computed on the output of term extractors by comparing corpus specific results with the one obtained on other corpora. Among other difficulties with learning algorithms are their applicability to small corpora, the cost of providing training data prepared by hand compared with the

gain actually brought by the learning tool, and error propagation when various tools are combined. A significant effort is still needed before getting to an optimized collection of relevant tools that could be efficiently combined. A further step would be then to make explicit some selection criteria to sort results and to reuse the model under construction as a semantic resource for a more powerful text procesing.

In more advanced frameworks such as Text-to-Onto [46] or Ahmad's method [15] various tools are combined. But, as underlined in [36] neither has the contribution of each component been thoroughly evaluated nor has a systematic state of the art survey been compiled. We need still to capitalize on available methods, tools, and results, and the current book is a significant step towards this aim. A listing of existing techniques, their properties and possible combinations would provide guidance for using the most appropriate combination of techniques and tools for a given application.

Nevertheless, it is not trivial to imagine a cyclic process that would involve the successive application of various NLP tools and learning algorithms. This is one of the research challenges of the Semantic Web for the years to come. If such continuous processes can be defined, even for simplified cases, semantic resources will be much more easily available and document annotation will be facilitated. Let's emphasize that a toolkit of processes and methods dedicated to specific applications is needed. Indeed, because each kind of application requires a particular kind of semantic resource [16] [51], the design process will be slightly different, including more or less sophisticated text analyses [30].

7. Conclusion

Many methodologies and even more tools have been defined to support ontology engineering from texts, which is now known as *Ontology Learning*. This label reveals the strong willingness to automate the process as much as possible. The goal is rather to suggest "noisy" but reasonably large pieces of model than thousands of raw linguistic data. In order to reduce human interpretation, many of the resources required for text analysis are learned form training sets of tagged documents. Moreover, validation is delayed as much as possible in the process, and it bears on pieces of model. Another way to reduce the validation effort is to rely on additional semantic and lexical resources, sometimes on information extracted from the web, to provide an automatic selection of extracted results: these results will be considered as valid if they can be found in one of the resources.

In this context, TERMINAE may look like an old fashioned pioneer. By promoting a supervised process and leaving an important part to the knowledge engineer, by providing little automatic processing of NLP results and a weak integration of the various NLP tools, knowledge modeling from texts remains time consuming and costly. Nevertheless, we would like to go on carrying out experiments in order to test some of its principles, and to formalize the experience that we have gained over the last 10 years. We identify three possible directions for future works, one of which being currently on the way :

Integrating more unsupervised processes We would like to reduce early human validation and to make validation bear on pieces of conceptual model; we plan to better use available statistics about NLP results and other linguistic or semantic resources;

Improving methodological guidance by implementing heuristics Most of the method guidelines for identifying relevant terms and relations, for extending the conceptual model or focusing on some concepts could be implemented with rules. By firing these rules, the method could be more active and suggest a list of tasks to the knowledge engineer. In a similar way, rules could implement differential principals so that the model could be automatically diagnosed and criticized during the normalization stage.

Reusing our experience to define a new platform We currently are two of the partners of a French project, DAFOE4App[7]. The aim of DAFOE4App is to define a new platform to support the early stages of conceptual modeling from texts. The tool should integrate various NLP and text-mining facilities with an editor for reusing, editing and building a conceptual model or an ontology. The method will include most of TERMINAE and ARCHONTE methodological guidelines.

References

[1] Zweigenbaum P., Bachimont, Bouaud, Charlet J., Boisvieux J.F., Issues in the Structuring and Acquisition of an Ontology for Medical Language Understanding. *Methods Inf Med*, 34(1-2), 1995.

[2] Fernandez M., Gómez-Pérez A., Juristo N., MethOntology : From Ontological Arts Towards Ontological Engineering, in *AAAI 97 Springs Symposium Series on Ontological Engineering*, Stanford USA, (1997), 33–40.

[3] Skuce D, Meyer I., Terminology and knowledge acquisition: exploring a symbiotic relationship. In *Proc. 6th Knowledge Acquisition for Knowledge Based Systems Workshop*, Banff, Canada. (1991), 29/1–29/21.

[4] Sowa J. (Ed.), *Principles of Semantic Networks*. Morgan Kaufmann Publishers. 1991.

[5] Meyer I., Skuce D., Bowker L. & Eck K., Toward a new generation of terminological resources: an experiment in building a terminological knowledge base. In *Proc. 13th International Conference on Computational Linguistics (COLING). Nantes, France*, (1992), 956–960.

[6] Simperl E., Tempich C., Vrandecic D., A methodology for ontology learning, in Buitelaar P. & Cimiano P. (Eds.), *Bridging the Gap between Text and Knowledge - Selected Contributions to Ontology Learning and Population from Text*, IOS Press, (2007).

[7] Aussenac-Gilles N., Biébow B. & Szulman S., Revisiting Ontology Design: a methodology based on corpus analysis, in *Knowledge Engineering and Knowledge Management: Methods, Models, and Tools, Proc. of the 12th EKAW Conference*, Dieng, R. & Corby, O. (ed.), Springer-Verlag, (2000), 172–188.

[8] W3C, *Web Ontology Language OWL*. (2004)

[9] Vogel, C. *Génie cognitif*, Masson: Paris, 1988.

[10] Aussenac-Gilles N., Bourigault D., Condamines A., Gros C., How can knowledge acquisition benefit from terminology ? in *Proc. of the 9th Knowledge Acquisition Workshop KAW'95*, Banff, Univ. of Calgary (CAN). 1995.

[11] Meyer, I., Extracting Knowledge-rich Contexts for Terminography : A Conceptual and methodological Framework , in *Recent Advances in Computational Terminology*, D. Bourigault, M.-C. L'Homme & C. Jacquemin (eds), John Benjamins. 2000.

[12] Slodzian M., L'émergence d'une terminologie textuelle et le retour du sens, in *Le sens en terminologie*, publication du Centre de Recherche en Terminologie et Traduction de l'Université Lyon 2. 2000.

[13] Rastier F., Le terme, entre ontologie et linguistique. *Banque des mots*. Special issue on Terminologie et Intelligence Artificielle. **7**, (1995), 35–64.

[14] Condamines A. (ed.), *Sémantique et Corpus*, Hermes: Paris, 2005.

[15] Gillam L., Tariq M., Ahmad K, Terminology and the construction of ontology. in [16] (2007). 49-74.

[16] Ibekwe-SanJuan F., Condamines A. & Cabré M.-T. (eds.), *Application-Driven Terminology Engineering*, John Benjamins Current Topics 2: Amsterdam, 2007.

[7]DAFOE4App is a project funded by the RNTL (software technologies) program of the French National Agency for Research, from 2007 to 2010. It involves 8 partners.

[17] Harris Z., *Mathematical Structures of Language*, Interscience Publishers, 1968.
[18] Guarino N., Welty C., A formal Ontology of Properties. In *Proc. of the 12th International Conference on Knowledge Engineering and Knowledge Management EKAW2000*. Juan-Les-Pins (F). Oct 2000. R Dieng and O. Corby (Eds). LNAI Vol 1937. Berlin: Springer Verlag. 97-112, 2000.
[19] Bachimont B., Troncy, R. & Isaac, A. Semantic commitment for designing ontologies: a proposal, in *Proc. of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW)*, Siguenza, Spain, Springer-Verlag LNAI 2473, 2002.
[20] Maedche A., *Ontology learning for the Semantic Web*. Kluwer Academic Publisher. 2002.
[21] Baneyx A., Charlet J., Jaulent M.-C., Building an ontology of pulmonary diseases with natural language processing tools using textual corpora. *Int. Journal of Medical Informatics (IJMI)*. Elsevier. 2006.
[22] Malaisé V., Zweigenbaum P. & Bachimont B., Mining defining contexts to help structuring differential ontologies, in [16] (2007).
[23] Völker J., Haase P., Hitzler P., Learning Expressive Ontologies, in Buitelaar P. & Cimiano P. (Eds.), *Bridging the Gap between Text and Knowledge - Selected Contributions to Ontology Learning and Population from Text*, IOS Press, (2007).
[24] Brachman R. J., On the epistemological status of semantic networks. In N. V. Findler (ed.), *Associative Networks: Representation and Use of Knowledge by Computers*, Academic Press (1979) 3–50.
[25] Bachimont B., Engagement sémantique et engagement ontologique : conception et réalisation d'ontologies en ingéniérie des connaissances, in *Ingénierie des Connaissances, évolutions récentes et nouveaux défis*, Charlet J., Zacklad M., Kassel G. & Bourigault D. (eds.), Eyrolles, (2000), 305–323.
[26] Szulman S. & Biébow B., OWL et TERMINAE, *actes de la 15e conférence d'Ingénierie des Connaissances IC 2004, Lyon, France, mai 2004*, Presses Universitaires de Grenoble, (2004), 41–52.
[27] Soergel, D., Lauser, B., Liang, A., Fisseha, F., Keizer, J., Katz, S., Reengineering thesauri for new applications: the AGROVOC example, *Journal of Digital Information*, **4** 4, Article No. 257, March 2004.
[28] Cimiano P., Völker J. & Studer R., Ontologies on Demand? *Information Wisenschaft and Praxis 57*, (2006), **6-7**, 315-320.
[29] Biébow B. & Szulman S., TERMINAE: A Linguistics-Based Tool for the Building of a Domain Ontology, in *Proc. of the 11th European Workshop on Knowledge Acquisition, Modelling and Management (EKAW'99)*,D. Fensel and R. Studer (Eds.), Springer-Verlag, LNAI 1621 (1999), 49–66.
[30] Bourigault D., Aussenac-Gilles N. & Charlet J., Construction de ressources terminologiques ou ontologiques à partir de textes : un cadre unificateur pour trois études de cas, *Revue d'Intelligence Artificielle (RIA)* **18**(1), Techniques Informatiques et Structuration de Terminologies, Pierrel J.-M. & Slodzian M. (Ed.), Hermès: Paris, (2004), 87–110.
[31] Reymonet A., Aussenac-Gilles N., Thomas J., Tâche, Domaine et Application : Influences sur le processus de modélisation de connaissances. *17èmes journées d'Ingénierie des Connaissances (IC 2006)*. Nantes (France), Juin 2006.
[32] Ceausu, V. & Després, S. Ontology based term to concept mapping. in *Proc. of Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU 2006)*, 2006.
[33] Despres, S. & Szulman, S. Merging of Legal Micro-Ontologies from European Directives, *Artificial Intelligence and Law*, Springer-Verlag, 2007.
[34] Despres S., Furst F. & Szulman S., Construction d'une ontologie du domaine HSE, in *Actes de la 18èmes journées francophones d'Ingénierie des connaissances, IC 2007, Grenoble*, 2007.
[35] Aussenac-Gilles N. Ontology or meta-model for retrieving scientific reasoning in documents: the Arkeotek project. in *Workshop on Exploring the limits of global models for integration and use of historical and scientific information*, Herakleion (Greece), M. Doerr, A. Renear (Eds.), 2006.
[36] Aussenac-Gilles N., Soergel D., Text Analysis for Ontology and Terminology Engineering *Applied Ontology*, Amsterdam: IOS Press. **1**(1), 2005.
[37] Breuker J., Valente A. & Winkels, Use and Reuse of Legal Ontologies, in *Law and the Semantic Web*, Benjamins V. R., Casanovas P., Breuker J. & Gangemi A. (ed.), Springer Verlag, (2005), 36–64.
[38] Bontcheva K., Tablan V., Maynard D. & Cunningham H., Evolving GATE to Meet New Challenges in Language Engineering. *Natural Language Engineering*, **10**(3//4), (2004) 349–373.
[39] Pantel P., Pennacchiotti M., Harvesting and Ontologizing Semantic relations, in Buitelaar P. & Cimiano P. (Eds.), *Bridging the Gap between Text and Knowledge - Selected Contributions to Ontology Learning and Population from Text*, IOS Press, (2007).
[40] Ruiz-Casado M., Alfonseca E., Okumura M., Castells P., Rote extractors for Automatic annotation of encyclopedic texts, in Buitelaar P. & Cimiano P. (Eds.), *Bridging the Gap between Text and Knowledge*

- *Selected Contributions to Ontology Learning and Population from Text*, IOS Press, (2007).

[41] Bourigault D. & Fabre C., Approche linguistique pour l'analyse de corpus, *Cahiers de Grammaires* **25**, Université Toulouse Le Mirail, (2000) 131–151.

[42] Aubin S. & Hamon T., Improving Term Extraction with Terminological Resources, in *Advances in Natural Language Processing, 5th International Conference on NLP, FinTAL 2006*, Tapio Salakoski, S. P. T. P. (ed.), Springer, (2006), 380–387.

[43] Hamon T. & Nazarenko A., Detection of synonymy links between terms: experiment and results, *Recent Advances in Computational Terminology*. John Benjamins, (2001), 185-208.

[44] Aussenac-Gilles N., Jacques M.-P., Designing and Evaluating patterns for Ontology Enrichment from texts. in *International Conference on Knowledge Engineering and Knowledge Management (EKAW)*, Prague, S. Staab, V. Svatek (Eds.), Springer-Verlag, Lecture Notes in AI, V.4248, (2006), 158–165.

[45] Ceausu V. & Després S., Une approche mixte pour la construction d'une ressource terminologique, *actes de la 15e conférence d'Ingénierie des Connaissances (IC 2004)*, Lyon, France, mai 2004, Presses Universitaires de Grenoble, (2004) 211–223.

[46] Staab S., Maedche A., Ontology Learning for the Semantic Web, in *IEEE Intelligent Systems*, Special Issue on the Semantic Web, (2001), 16(2).

[47] Cimiano P., *Ontology Learning and Population from Text. Algorithms, Evaluation and Applications*, IOS Press. (2007) 347 p.

[48] Ciravegna F., Dingli A., Petrelli D., Wilks Y., User-system cooperation in document annotation based on information extraction. In *proceedings of the 13th International conference in Knowledge Engineering and Knowledge Management (EKAW)*, Gómez-Pérez A., Benjamins V.R., (Eds), LNAI 2473. Berlin: Springer Verlag, 2002.

[49] Navigli R., Velardi P., From Glossaries to Ontologies: Learning the Structure Hidden in the Text, in Buitelaar P. & Cimiano P. (Eds.), *Bridging the Gap between Text and Knowledge - Selected Contributions to Ontology Learning and Population from Text*, IOS Press, (2007).

[50] Tanev H., Magnini B., Weakly Supervised Approaches for Ontology Population, in Buitelaar P. & Cimiano P. (Eds.), *Bridging the Gap between Text and Knowledge - Selected Contributions to Ontology Learning and Population from Text*, IOS Press, (2007).

[51] Ait El Mekki T. and Nazarenko A., Using NLP to build the hypertextuel network of a back-of-the-book index, in *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP) (2005)*, 316-320.

A Methodology for Ontology Learning

Elena Simperl[1], Christoph Tempich[2], and Denny Vrandečić[3]

[1]Free University of Berlin, Takustr. 9, 14195 Berlin, Germany
paslaru@inf.fu-berlin.de
[2]Institute AIFB, University of Karlsruhe, 76128 Karlsruhe, Germany
christoph.tempich@aifb.uni-karlsruhe.de
[3]Institute AIFB, University of Karlsruhe, 76128 Karlsruhe, Germany
denny.vrandecic@aifb.uni-karlsruhe.de

Abstract. Due to the well-known difficulties implied by manually building an ontology, machine-driven knowledge acquisition techniques—in particular in the field of ontology learning—are promoted by many ontology engineering methodologies as a feasible alternative to aid ontology engineers in this challenging process. Though the benefits of ontology learning are widely acknowledged, to date its systematic application is considerably constricted by the lack of adequate methodological support. The advantages of an elaborated ontology learning methodology are twofold; on the one hand it reduces the need for a high expertise level in this field: a detailed description of the process and best practices in operating it in a variety of situations make ontology learning techniques more accessible to large communities of ontology developers and users; on the other hand the methodology clearly formalizes the ways ontology learning results are integrated into a more general ontology engineering framework, thus opening up new application scenarios for these techniques and technologies. In this article we aim at contributing at the operationalization of ontology learning processes by introducing a methodology describing the major coordinates of these processes in terms of *activities*, *actors*, *inputs*, *outputs* and *support tools*. The methodology was employed to build an ontology in the legal domain. We present the lessons learned from the case study, which are used to empirically validate the proposed process model.

1 Introduction

Ontology learning is targeted at acquiring knowledge in form of ontological categories such as concepts, taxonomies, properties or axioms from information sources describing a specific domain of interest. Due to the well-known difficulties related to manually building an ontology, knowledge acquisition and in particular ontology learning are promoted by many ontology engineering methodologies as one of the fundamental *support activities* aiming at aiding ontology engineers in this challenging process [1]. In order to achieve this goal ontology learning approaches combine methods and techniques from a multitude of disciplines,

e.g., Machine Learning, Computational Linguistics or Knowledge Representation, with a (still) indispensable amount of human expertise, required to evaluate and refine the results of the information extraction and classification tasks.

The popularity of this emerging research field is indicated by the rich inventory of tools and methods for (automatically) learning ontologies from semi-structured knowledge sources or textual documents. However, the issue of integrating these methods into a generic methodology, and furthermore into the overall ontology engineering process, has been marginally explored by the Semantic Web community yet. Maedche [2] has presented a high-level ontology learning architecture. The proposed ontology learning process model builds on the idea of data mining as a process (e.g. [3]) with the phases of *business and data understanding*, *data preparation*, *modeling*, *evaluation* and *deployment*. However, no holistic approach similar to or extending the aforementioned one is known so far. Similarly, Aussenac-Gilles et al. [4, 5] propose a process model to support the generation of ontological categories from text based on a specific learning algorithm, but does not integrate the process model into a generic ontology engineering methodology. As a consequence current ontology learning methods and tools, though producing valuable results, can not be optimally applied to arbitrary real-world ontology engineering scenarios. On the one hand, in absence of a user-friendly, fine grained process description, ontology learning can not be performed by domain experts without a considerable technical assistance. On the other hand, the integration of the knowledge acquisition results into the ontology development task is still performed in an ad-hoc, unsupervised manner, thus deteriorating the quality and limiting the flexibility of the overall engineering process.

This chapter aims at contributing to the alleviation of this problem by proposing a fine-grained ontology learning process model which explicitly addresses the aforementioned issues. The methodology was applied and empirically validated in an ontology-related case study in the legal domain.

The remaining of this chapter is organized as follows: we give a high-level overview of the general ontology engineering process in Section 2 and situate the ontology learning process model within the general process (cf. Section 3). The main components of the model are elaborated in Section 4. Section 5 describes the setting in which the methodology was applied and the findings of this case study evaluation. The conclusions of our work and some directions for future development are discussed in Section 6.

2 Ontology Engineering in a Nutshell

Ontology Engineering (OE) is formally defined as "the set of activities that concern the ontology development process, the ontology life cycle, and the methodologies, tools and languages for building ontologies"[1]. This section summarizes the most important of these activities.

Ontology engineering methodologies support ontology building for centralized ontology applications.[1] [7, 8] focus on the consensus building process in collaborative ontology engineering. Methodologies guiding the ontology reuse process, e.g., [9, 10] or the ontology learning process, the focus of this book, complete the picture.

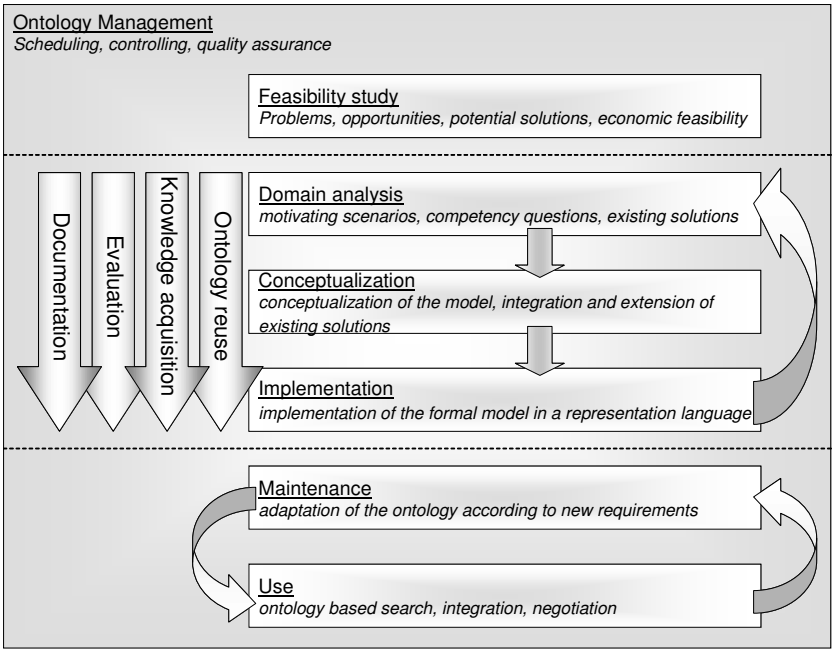

Fig. 1. Ontology Engineering Activities

Methodologies divide the ontology building process in a varying number of stages, and propose a number of activities for each stage. The importance of a particular activity within a methodology primarily depends on, e.g., the characteristics of the ontology-based application, the complexity of the ontology to be built, the availability of information sources, and the experience of the ontology engineers.

[1] differentiates among *management*, *development-oriented* and *support* activities within an ontology engineering process (cf. Fig. 1). The organizational setting of the overall process is covered by so-called ontology management activities. In the pre-development phase the *feasibility study* examines if an ontology-based application or the use of an ontology in a given context is the right way to solve the problem at hand. *Domain analysis*, *conceptualization* and *implementation* are classical ontology development activities. The *maintenance* and

[1] Refer for example to [1, 6] for recent overviews.

the *use* of the ontology are post-development activities. Ontology support activities, e.g., *knowledge acquisition* (KA), *evaluation*, *reuse*, and *documentation* are performed in parallel to the core development activities.

Methodologies additionally define the roles of the individuals involved in the ontology building process. They primarily differentiate between *domain experts* providing knowledge w.r.t. the domain to be modeled, *ontology engineers* with expertise in fields such as knowledge representation or ontology tools, and *users* applying the ontology for a particular purpose.

3 General process

Ontology learning is intended to be performed in conjunction with the main ontology development process: during the domain analysis phase ontology engineers and domain experts analyze the feasibility of a learning approach and specify the excerpts of the target ontology which could be developed in this way. The results of the learning process are integrated into the final ontology in a subsequent phase, such as conceptualization or implementation (see Figure 1). During the ontology learning process ontology engineers *and ontology learning experts* collaborate with domain experts in order to extract relevant knowledge from information sources, such as text documents, Web sites, databases or tables. In order to achieve this goal, they have to specify the main parameters of the learning process: the corpora used as input for the learning tools, the most suitable tools and methods, as well as the desired outcomes and the way they should be evaluated. Depending on the particularities of the application scenario, the learning process can be arbitrarily complex: multiple ontologies might be acquired from external sources, and need to be integrated into a final result; various tools might be used on different corpora to generate (parts of) the same ontology, some phases of the learning procedure need to be repeated in order to improve their outcomes etc.

Our methodology distinguishes among the following eight process stages:

1. Feasibility study
2. Requirements specification
3. Selection of information sources
4. Selection of ontology learning methods and tools
5. Learning preparation
6. Learning execution
7. Ontology evaluation
8. Ontology integration

An overview of process model including roles of the participants, main process stages and the associated activities, decisions, input and output parameters, as well as support tools is depicted in Figure 2. Depending on the outcomes of a particular process stage the learning workflow can be executed in a linear or iterative manner.

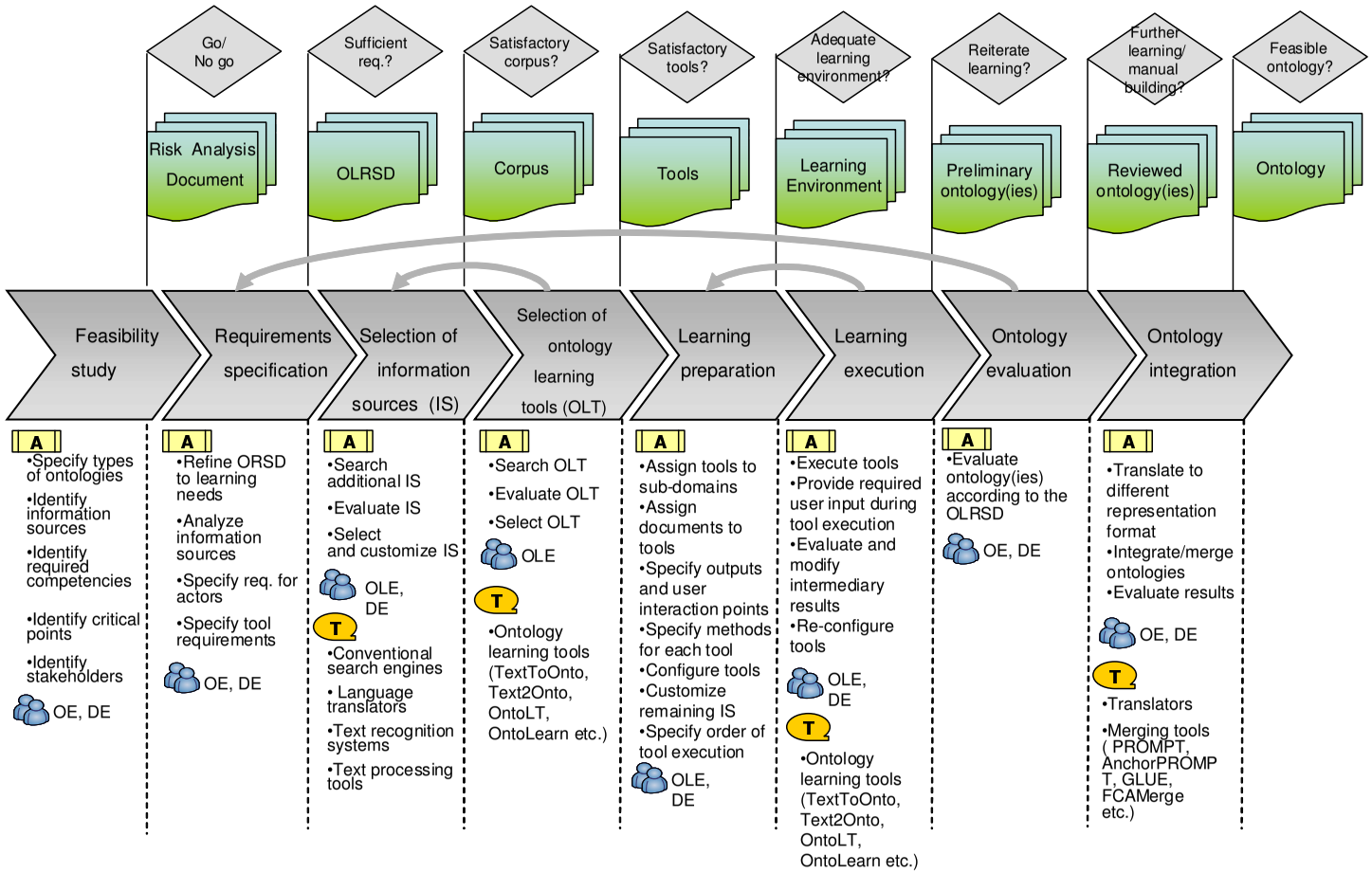

Fig. 2. Overview of a learning-driven ontology engineering process

The process model has been defined after extensively surveying recent ontology learning literature, studying the ontology learning activities implicitly supported by the tools currently available, and from empirical findings of several case studies in the field. We collected the activities covered by these sources and mapped them to established equivalents in ontology engineering. Identifying the communalities between the activities across the various approaches and renaming the activities using common ontology engineering terminology was rather straightforward. None of the analyzed approaches covered the entire process model; most of them implicitly supported a simplified version of the process model, while others had a different focus.

4 Process stages

In the following we elaborate the eight process stages introduced above in terms of their low-level activities, the roles involved in these activities, the input and output parameters, the decisions which have to be taken in order to produce the desired outputs and potential technological support.

4.1 Feasibility study

The feasibility study is targeted at assessing the general feasibility of an ontology learning approach in the context of building an ontology in a particular application setting. Provided that a positive feasibility study with respect to the possibility of solving a specific issue by means of ontologies has been carried out, the goal of this phase is to approximate the risks and problems which might endanger the knowledge acquisition task. The link to the global ontology engineering process is provided by the input parameter of this phase i.e. the *ontology requirements specification document* (ORSD).

Roles As part of an ontology engineering process, this process step is mainly operated by the same key actors, *ontology engineers (OE)* and *domain experts (DE). Ontology learning experts (OLE)* are additionally required to realistically assess the impact of technical issues on the success of the learning process.

Input factors In order to approximate the appropriateness of a learning approach the participants analyze the *ontology requirements specification document (ORSD).* An excerpt of a typical ORSD is depicted in Table 1 [11].

Output factors The result of this process step is a *risk analysis document (RAD)* analyzing potential risk factors and proposing risk management strategies. An example for a risk analysis document is provided in Table 2 below.

Activities During the feasibility study the ontology learning team performs the following activities:

- *Specify types of ontologies:* The activity is necessary as particular types of ontologies (e.g. domain ontology, task ontology, upper-level ontology) optimally require different methods and techniques for knowledge acquisition. This should be done considering the sub-domains already indicated in the ontology requirements specification document.
  Identify information sources: In this activity ontology learning and domain experts collect a set of information sources potentially useful for the learning process.
- *Identify required competencies:* The ontology learning team should identify the types of competencies required in the operation of the process. Typical examples might be
  - *Linguistic expertise:* This is required to select appropriate methods and tools and to supervise their usage in the concrete setting.
  - *Domain expertise:* Knowledge of the domain is crucial for the selection of appropriate information sources and for the evaluation of the resulting ontology with respect to domain-specific criteria such as correctness or completeness [12].
  - *Tool know-how:* Experience with ontology learning tools as well as other auxiliary ontology management environments is important so as to configure and execute the learning task.
  - *Language know-how:* This is required for selecting and eventually customizing the domain-related information sources, which might be available in a particular natural language.
- *Identify stakeholders:* In this activity the ontology learning team should analyze and identify the stakeholders of the learning process in order to be able

Item	**Description**
Goal, domain and scope	Tourist information about activities, attractions,
	environmental information in European cities
Size	200 concepts
Granularity level	Major activities
Terminology	Concept labels in English and Spanish
Knowledge sources	Domain experts, databases, Lonely planet Web site
Users	Tourists
Application scenarios	Selection of travel destination
Supported applications	Web portal
Competency questions	1). What can one do in Barcelona? 2). Where can I sky and bike? 3). Which city with good weather has a museum exhibiting the art of Picasso? ...

Table 1. Excerpt of an ontology requirements specification document (ORSD)

Item	Description
Domain	The domain modelled by the ontology
Purpose	The purpose the ontology is used for
Scope	The application scope
Roles	Roles involved in the setting
	Critical issues
Knowledge-sources	Style of speech Document structure Number of knowledge sources High-level of expertise Translation to further languages or forms Laborious search heuristics Fuzzy evaluation of the utility of the sources
Ontology Learning Tools	Fulfill requirements with respect to language, input and output forms etc.
Domain	Feasible coverage by documents Appropriate sub-domain modularization
Organizational issues	Multi-site development Tight schedule
Technical and technological issues	Appropriate hardware and software
Personnel issues	Resource management Level of expertise

Table 2. Template for the risk analysis document (RAD)

to optimize its operation and outcomes to this target group.

– *Identify critical points:* The central activity in this process stage is concerned with the analysis of risk factors, which might have a significant impact on the overall process and the expected results, and the elaboration of a risk management procedure. Examples of such risk factors could be related to the *quality of the knowledge sources*, the *lack of domain, language or technical expertise*, the *lack of adequate tools* etc.

Decisions It should be decided for or against building (parts of) the target ontology by learning.

Support tools Currently there are no established tools to support the feasibility study.

4.2 Requirements specification

The goal here is to identify and specify requirements for the ontology learning process, whose feasibility has been positively assessed in the previous step. The

task can be seen as an instantiation of the more general ontology requirements specification to the particularities of the learning task. In this case some of the original requirements are to be further revised and adapted to the new context. New requirements, especially those related to the technological infrastructure applicable for knowledge acquisition purposes, may also arise.

Roles This process step is performed collaboratively between *ontology engineers*, *domain experts* and *ontology learning experts*. While the former two are essential for the specification of the ontology-centric requirements, the latter are responsible for the definition of particular knowledge acquisition requirements such as those with respect to tools and techniques.

Input factors The activities in this step are performed on the basis of the *ontology requirements specification document (ORSD)* and the *risk analysis document (RAD)* produced at step 1.

Output factors The result of this phase is an *ontology learning requirements specification document* (ORLSD) covering the requirements regarding the ontology to be learned and the ontology learning task to be carried out for this purpose. A template for an ontology learning requirements specification document (OLRSD) is depicted in Table 3.

Activities The following activities are relevant to complete the requirements specification:

- *Refine ORSD:* The original requirements specification needs to be revised and extended for the purpose of ontology learning.
- *Specify information sources requirements:* The ontology learning experts specify which linguistic and technological criteria need to be fulfilled by the un- or semi-structured information sources used to learn the ontology. Representation-specific issues such as the domain knowledge to be covered by these information sources and the natural language of the domain descriptions are also relevant.
- *Specify tool requirements:* Ontology learning experts identify which tool features with respect to inputs, outputs, language, type of learning method might be required.
- *Specify personnel requirements:* The team also provides details on the level of expertise required for the participants in this process phase and the need for additional experts

Decisions The ontology learning team needs to decide whether and when a feasible set of requirements has been collected and specified so as to allow an efficient and effective process operation.

Support tools Currently this step is not supported by dedicated tools.

Item	Description (source)
Domain	Domain formalized in the ontology (from the ORSD)
Purpose	The purpose the ontology is used for (from the ORSD)
Scope	The application scope (from the ORSD)
Roles	The roles involved in the setting (from the ORSD plus ontology learning specific)
Design Guidelines	
Number of ontological primitives	The size of the ontology to be learned
Granularity level	The granularity of the ontology to be learned
Terminology	The language to be used in the learned ontology
Knowledge Sources	
Domain experts	Experts to be involved in the learning process (from the ORSD)
Data bases	Partially from the ORSD
Documents	The learning corpus (partially from the ORSD)
Further sources	Ontologies, lexis, thesauri (partially from the ORSD)
Users	
Application scenario	Description of the application setting (adapted from the ORSD)
Use cases	Description of the main use cases (adapted from the ORSD)
Supported application	The target application embedding the ontology (from the ORSD)
Ontology	
Competency questions	Adapted from the ORSD
Ontology Learning	
Input	Language, format
Output	Language, format
Learning result	The types of primitives required to be learned
Automatization level	User interaction required or not
Learning quality	Minimal quality measures

Table 3. Template for the ontology learning requirements specification document (OLRSD)

4.3 Selection of the information sources

After specifying the requirements for the ontology learning process, its main coordinates—the information sources to be used as input for acquiring ontological knowledge and the tools and methods involved in achieving this goal—are to be selected and configured. In terms of the process model this is represented in form of two closely related process steps. The execution order of these steps is not dictated by the methodology; due to the interdependencies between their outcomes, the concrete execution workflow strongly depends on the particularities of the ontology engineering setting. In case the information sources involved in the learning process are fixed in advance, the selection step is reduced to the identification of a subset of this corpus, which is responsible for the generation of

the ontology. In case the tools and methods applied during the ontology learning process are known in advance, the selection of the information sources should take into consideration this restriction. Finally, if none of these parameters is pre-defined, the two steps can be executed iteratively, until an optimal configuration between information sources and knowledge acquisition methods and tools is achieved.

Roles The execution of this task requires both domain and technical expertise. The former is needed in order to assess the representativeness of the learning corpus with respect to the domain to be modelled. The latter is related to ontology learning methods and tools being able to deal with a particular corpus and to produce the expected results.

Input factors As input the ontology learning team uses the ontology learning requirements specification document (OLRSD) summarizing the main aspects related to the learning task, the information sources to be selected and possibly the methods and tools applied. The latter holds true if the selection is realized in an iterative manner or if the usage of certain technologies has been decided in advance.

Output factors The result of this process phase is a collection of information sources (*learning corpus*).

Activities In this process stage the following activities need to be performed:

- *Search additional information sources:* First the team might look for information sources in addition to those identified during the previous process steps and mentioned in the ontology learning requirements specification document (OLRSD).
- *Evaluate information sources:* According to the criteria specified in the OLRSD the set of information sources should be evaluated with respect to language, domain, representation, structure etc.
- *Select and customize information sources:* The sources which have been positively assessed might be subject to various customization operations in order to allow their seamless usage in the learning process. Customization includes the translation to different languages or formats, basic parsing operations, the digitalization or other form of processing.

Decisions During this process step the participants need to decide whether they produced a corpus which allows for a feasible execution of the learning task.

Support tools In order to enable or speed up the construction of the ontology learning corpus a wide range of technological support tools can be applied.

Some of them are general-purpose, such as search engines used to discover the domain-specific knowledge sources. Other are closely related to the form of the information sources to be used: language translators, text recognition systems, text processing tools etc.

4.4 Selection of the ontology learning methods and tools

This process step aims at identifying the exact methods and tools which are to be used to extract the target ontology/ontologies. As aforementioned, the outcomes and execution of this step might be influenced by a previously pre-selected learning corpus. Furthermore the decision with respect to specific ontology learning tools and methods might trigger a re-iteration of the selection process: if no feasible methodical or technological support is available for the input information sources, the ontology learning team might consider a revision of the corpus followed by their assignment to the learning tools and methods. An overview of the dependencies between methods and information sources they are typically applied on is provided in Table 4 below. A recent overview of some of the most relevant tools and methods is provided in [13].

Domain	Method	Features used	Prime purpose
Free text	Clustering	Syntax	Extract
	Inductive logic programming	Syntax, logic representation	Extract
	Association rules	Syntax, tokens	Extract
	Frequency-based	Syntax	Prune
	Pattern matching	-	Extract
	Classification	Syntax, Semantics	Refine
Dictionaries	Information extraction	Syntax	Extract
Knowledge base	Concept induction A-Box Mining	Relations	Extract
Semi-structured Schemata	Naive Bayes	Relations	Reverse Engineering
Relational Schemata	Data correlations	Relations	Reverse Engineering

Table 4. Ontology learning methods and sources according to [14]

Roles Compared to the selection of the information sources, this step is mainly technological. Its execution requires expertise in the ontology learning field: with

respect to the particularities of the analyzed methods and the feasibility of the tools implementing these methods.

Input factors Just as in the previous step the activities performed here make use of the ontology learning requirements specification document (OLRSD), the information sources and the learning technology available.

Output factors The process step is completed when the team selected which ontology learning tools and techniques will be used in the process.

Activities The decision upon the technological infrastructure of the process should be based on the results of the following activities:

- *Search ontology learning tools:* Ontology learning experts might need to look for additional ontology learning technology prior to selecting the appropriate one.
- *Evaluate tools usability:* Given an inventory of techniques and tools for extracting ontological knowledge from unstructured and semi-structured resources, ontology learning experts evaluate which technology is appropriate according to the pre-defined requirements. Specific criteria such as the applied linguistic methods, types of ontological knowledge extracted, natural language, input characteristics, additional knowledge sources or components required, but also more general ones like the user friendliness are relevant in this context.
- *Select learning tool(s):* After assessing the usability of the tools in relation to the application setting, ontology learning experts select the ones which will be used in the next step.

Decisions In this step the team needs to decide whether the evaluated technology is likely to be adequate for producing the prospected ontology. A negative decision with this respect might trigger a re-iteration of the process at the previous step, in order to produce a learning corpus which better fits the available knowledge acquisition methods and tools.

Support tools There are no dedicated tools supporting this selection step.

4.5 Learning preparation

This step gives full particulars about the ontology learning process. After identifying the information sources from which the target ontology will be extracted and the methods and tools appropriate for performing this task, ontology learning experts are required to configure the technical infrastructure of the knowledge acquisition procedure which will be executed in the next step. The distinction between the preparation and the execution of the learning task is due to the

complexity of the former, which might imply complicated parameter settings at individual tool and at global workflow level. In particular, besides providing the optimal input parameters to the learning tools applied for each of the sub-domains and sub-corpora, the learning experts are expected to clearly specify and guide the tool execution, the interaction between tools and between domain experts and tools, and the ways intermediary results are to be integrated.

Roles Ontology learning experts are indispensable for the correct and effective preparation of the learning procedure. Domain experts might be involved in this phase in order for them to get familiar with the technology generating the ontology they have to subsequently evaluate.

Input factors This phase makes use of the ontology learning requirements specification document (OLRSD), the learning corpus, and the ontology learning tools and their documentation. All these resources provide input for the configuration of the learning infrastructure.

Output factors The result of this phase is a configured tool environment, including the documentation of the configuration and the execution plan for the learning task.

Activities The preparation of the ontology learning tools includes the following activities:

- *Assign tools to sub-domains and information sources:* if fragments of the final ontology are to be extracted from different information sources using different learning tools, ontology learning experts need to specify these interdependencies.
- *Configure tools:* ontology learning experts select the most appropriate learning method or algorithm to be applied for a particular purpose, and configure the tools in order to produce the desired output.
- *Specify user interaction points:* ontology learning experts define the interaction points and guide domain experts in providing the required input during the tool execution.
- *Specify order of tool execution:* in case several tools are applied for a joint knowledge acquisition task the team needs to define the exact learning workflow.

Decisions The learning preparation phase ends with a positive decision upon the feasibility of the learning environment.

Support tools This phase involves the ontology learning tools to be used in the process. The aforementioned activities are carried out manually.

4.6 Learning execution

This step is dedicated to the actual acquisition of the ontological knowledge on the basis of the configuration specified so far. The learning tools require the domain and learning experts to support the required tool interaction and to evaluate the correct accomplishment of each learning sub-task. The detection of major problems in the configuration of the tools imposes a re-iteration of the process at the previous step.

Roles Domain experts and ontology learning experts supervise the execution of the tools.

Input factors In order to feasibly supervise the execution of the learning tools, the team needs access to various resources. General information and requirements are provided by the ontology learning requirements specification document (OLRSD). The learning corpus and tools are implicitly required for the generation of the ontology. Further on, the configuration and execution plan contains further details on the way the tools are to be used to create the desired output.

Output factors At the end of this phase, a preliminary ontology/ontologies is/are available.

Activities The learning execution phase can be further divided into the subsequent activities:

- *Execute tools:* This activity directly refers to the actual acquisition of the planned ontology/ontologies with the help of the ontology learning tools.
- *Provide user input:* If the ontological knowledge is not extracted in a fully automatic manner, the participants should provide the input required to continue this task.
- *Evaluate intermediary results:* Intermediary outputs returned during this process phase should be evaluated immediately in order to detect potential errors in the learning process at an early stage.
- *Re-iterate learning execution:* If the results are not satisfactory, the ontology engineering team should decide whether a re-iteration of this process step or of the entire process is feasible and should optimize the learning environment according to experiences gained so far.

Decisions Either the learning process is completed, or a reiteration of the learning process with revised parameters is performed.

Support tools This phase involves the ontology learning tools to be used to generate the target ontological content. The aforementioned activities are primarily accomplished manually. Support can be provided in terms of automatic means to evaluate the intermediary results on the basis of pre-defined metrics.

4.7 Ontology evaluation

Given a correct execution of the ontology learning tools, ontology engineers and domain experts jointly assess the quality of the achieved results, i.e. the learned ontologies. The evaluation is guided by the requirements specified for each of the ontologies to be generated in the ontology learning requirements specification document OLRSD. Methodologically, the process leans at existing general-purpose, as well as at more specific, NLP-based evaluation techniques (cf. for instance [15] for a recent overview of the methods, and [16] for specific evaluation techniques for NLP). In case the assessment is negative, the engineering teams decides upon the activities which are to be performed in order to improve the quality of the learning outcomes, be that a different ontology building strategy, a complete re-iteration of the process or (manual) revisions at ontological level. Ontology learning experts are needed to assess the feasibility of a re-vised ontology learning cycle given the experiences gained so far.

Roles Domain experts and ontology engineers evaluate the results of the learning process. Ontology learning experts need to be involved in the decision upon a re-iteration of the learning task.

Input factors The evaluation step is based on the requirements contained in the ontology learning requirements specification document and of course on the learned ontological sources.

Output factors The evaluation step produces a reviewed and revised version of the input ontology which can be integrated with the results achieved in parallel engineering approaches.

Activities Activities in this step depend on the evaluation methodology chosen for the particular application scenario.

Decisions The engineering team needs to decide whether the learning results are feasible for the application scenario, or whether they need to be improved either through a re-iteration of the learning process or through other means such as manual ontology building.

Support tools Ontology editors can be used to visualize the input ontology in a language-independent manner. Further on, this step can take advantage of automatic test environments which evaluate the learning process from a method-specific perspective.

4.8 Ontology Integration

Feasible learning outputs need to be integrated into the final ontology. This applies for both learned ontologies and for the ones built using other methods (e.g. manually built, translated from other formats). This step, which is not specific to ontology learning processes, is to be performed according existing integration methodologies, methods and tools in ontology engineering (cf. for instance). [17, 18].

Roles Ontology engineers are the main actors for the implementation of the integration task. Domain experts might be required to provide additional input.

Input factors The engineering team resorts to the original ontology requirements specification document ORSD and the ontologies developed in the process.

Output factors A preliminary application ontology is generated at the end of this process phase.

Activities Integration of ontologies includes the following activities:

- *Translate:* Ontologies created using different engineering methods might be represented in different languages. Translators help in overcoming this heterogeneity.
- *Integrate ontologies:* This activity refers to the execution of a particular merging algorithm and on the resolution of possible conflicts.
- *Evaluate results:* This refers to the evaluation of the integration tools with respect to the results produced.

Decisions The integration phase is finalized with the decision upon the feasibility of the produced application ontology.

Support tools Various ontology management tools for translating, partitioning, merging and integrating ontological content are useful to speed-up this process phase.

5 Case study

In this section we describe a case study in the legal domain in which five domain experts followed the ontology learning methodology introduced in the previous sections with the purpose of developing ontologies for the Iuriservice prototype II (see below). The case study was embedded into a project aiming at methodological and technological support for a better knowledge transfer of applied legal knowledge to young judges in Spain.

5.1 Objectives of the Case Study

The main objective of the case study was to validate the proposed methodology from an operational point of view. In particular we were interested in the understandability of the process description, the coverage of the decision support, and the required training effort for non-experts in order to apply the methodology to concrete scenarios. As a secondary goal we aimed at identifying shortcomings of currently available ontology learning applications as regarding their usability.

5.2 Data collection and analysis techniques

Three experienced ontology engineers were involved in the case study, being responsible for guiding the case study participants and for collecting and analyzing the data. They conducted structured interviews with the five domain experts on a face-to-face basis. The interviews contained pre-defined questions related to the objectives of the case study, and open questions in which the domain experts could express their experiences with and their view on the methodology. In addition to the personal interviews the ontology engineers directly observed the domain experts performing some of the activities covered by the methodology. Furthermore, as an additional information source, they had access to all specification documents proposed by the methodology and filled out by the domain experts during the operation of the case study. At the end of the data collection procedure, the results of the interviews, individual observations and the documentation available were analyzed manually by the evaluators in order to obtain a holistic view on the experiences of the case study participants in applying the ontology learning methodology.

5.3 Organization setting

The case study was situated in the context of the European research project SEKT, *Semantically Enabled Knowledge Technologies*, which developed and deployed technologies for knowledge management. The project had a duration of three years while the aforementioned case study had a duration of four months. The organization conducting the case study was responsible for the development of a system which should provide Spanish judges with access to frequently asked questions (FAQ) related to the application of legal text book knowledge in real life situations through a natural language interface. The system, called Iuriservice, should respond with a list of question-answer pairs, which offer solutions to the problem specified by the judge, and a set of related, up-to-date and relevant case rulings, which are stored in the La Ley case base from CENDOJ (Figure 3). Thus, the software should be capable of clearing-up doubts concerning judicial practice and case resolution by providing justified and uniform answers to the raised questions (Figure 4). Within the system, ontologies represent the existing FAQs, the content of the case rulings and support the mapping between the user questions and the pre-defined FAQs. A detailed description of the technical

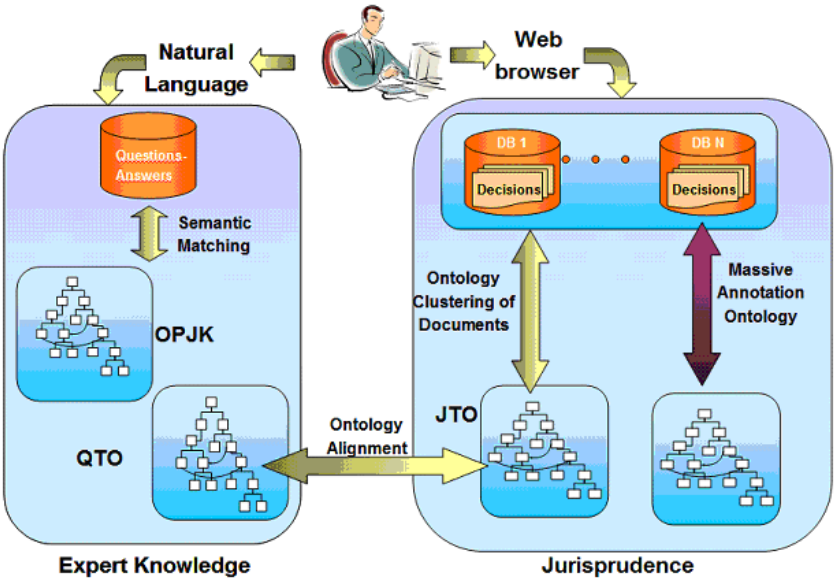

Fig. 3. A part of the Iuriservice architecture. The left side shows the FAQ system, whereas the right part shows how supporting cases are retrieved from the case base.

Fig. 4. Screenshot of the Iuriservice II system giving an answer. The user question can be seen on the top of the screenshot, while the bottom part shows the answer [20]

realization of the system is beyond the scope of this work and can be found in V. R. Benjamins et al. [19].

Ontology learning techniques were used in two ways for developing, extending, and maintaining the application ontologies. First, they were applied for supporting the creation of the initial ontology that was used to model the professional knowledge in the system to represent the FAQs. This was done prior to the development of the methodology described in this chapter, and so the experiences gained there were used as an input for the development of the ontology learning methodology. Later, ontology learning was applied to the case base in order to describe them by means of ontologies, and to raise the accuracy in retrieving the most relevant cases for a given problem. The following case study description focuses on this second ontology learning experiment, in which the participants made use of our methodology.

5.4 Case study description

In this section we describe the operation and the main results of the case study in the legal domain. We first give an overview of the general ontology engineering process which the ontology learning activity was an integral part of. Then we elaborate on the experiences made with the ontology learning methodology. The description of the case study is based on [20].

Integration into Ontology Engineering Process The project started with a feasibility study followed by a requirements engineering phase. The feasibility study showed that ontologies can support the retrieval of FAQs and that an ontology-based system can increase the quality of the retrieval results to a feasible extent. As a consequence, a team of ontology engineers and experts in the legal domain proceeded with the development of the application ontology. As part of the domain analysis the legal experts conducted questionnaire-based interviews extracting nearly 800 competency questions for the areas of domestic violence, on-duty period, procedural doubts and imprisonments. This knowledge acquisition activity ensured that the ontology to be built truly models professional legal knowledge which refers to the core of professional work that contains the experience of the daily treatment of cases. From the competency questions the ontology development team created the *Ontology of Professional Judicial Knowledge* (OPJK) which was subsequently used as a basis to retrieve correct question-answer pairs for a given user-specified problem. Additionally to this functionality, the system should select and provide the judge with the most relevant precedence cases for the formulated query. Due to the large amount of precedence cases available, a manual approach to formalizing the knowledge in this corpus was not pursued. Thus ontology learning tools were applied on the case corpus in order to extract and formalize domain knowledge, and to allow for the retrieval of the most relevant cases based on the ontological similarity between the specified question and the cases.

The Ontology Learning Activity In order to develop the ontology capturing the knowledge implicitly contained in the corpus of precedence cases, the legal experts followed the eight steps proposed by our ontology learning methodology.

Feasibility study As part of the feasibility study the participants analyzed the available case rulings, and learned that they are semi-structured, containing metadata as well as textual descriptions of the cases. The metadata could be used to pre-categorize the text modules as summary, case descriptions and ruling. The word choice, as well as the phrasing were found to be lengthy and significantly different from the competency questions. From a technical point of view the documents required pre-processing and preparation to be usable as input for an ontology learning tool. As the case rulings were in Spanish some of the available ontology learning tools needed further customization to provide the required language support.

Requirements specification In order to extend the ORSD and obtain the OLRSD, the participants selected the competency questions which directly referred to case rulings. It was decided to concentrate on the sub-domain *domestic violence* for which the domain experts expected to extract 700 concepts. The scope for tool support was restricted to the identification of relevant terms in Spanish and the semi-automatic generation of a class hierarchy. Regarding the integration of the learned ontology with the OPJK it was decided to map equivalent concepts in a subsequent step.

Selection of information sources The case rulings were stored in the database La Ley from CENDOJ, which is a government supported organization to archive juridical knowledge in Spain. From this database 1000 documents were selected covering the sub-domain domestic violence.

Selection of ontology learning methods and tools In the context of the project only two tools, namely OntoGen [21] and Text2Onto [22] were considered for the task. While the former is better suited to visually cluster the selected documents according to domains and to identify relevant terms, the latter has its strength in building hierarchies and select multi-word terms. A preliminary test in automatically building hierarchies was not satisfactory and therefore it was decided to build the hierarchy manually. Major drawbacks of the evaluated tools were the missing interaction possibilities, requiring many unnecessary interactions with the user interface in order to apply manual corrections to the built hierarchy and extracted terms.

Learning preparation The case rulings were extracted from the data base and the different parts of the rulings were prepared in order to comply with the input format of OntoGen and Text2Onto. It was decided to extract relevant terms from the case ruling summary, the case description and the ruling separately. The learning preparation required several pre-runs on the selected texts in order to fine-tune the extraction parameters. For example the performance of Text2Onto could be improved by filtering the text with specific linguistic patterns. Several different linguistic patterns were tested in order to achieve better results. Again, this step required deep understanding of the underling algorithms, a situation which might hinder its wider adoption.

The users asked for an interactive mechanism and support to select the best extraction patterns.

Learning execution We run the different algorithms on the extracted texts (Figure 5). Although we could export and import the extracted terms into the ontology editor, it was difficult to choose relevant terms from the extraction list. It was therefore decided to build the ontology from the extracted terms manually, while taking the outcomes of the learning tools into account. At this point ontology engineers would require a functionality of a tool supporting the comparision between the requirements listed in the OLRSD and the learned ontology. Due to the number of learned ontology entities, it is tedious and errorprone to manually evaluate all entities again after each rerun of the algorithm. Since the building of the ontology involved the manual transcription of the learned entities into the ontology editor the evaluation of the intermediary results was performed iteratively until the built ontology covered all requirements.

Another difficulty was that there was no version of TextToOnto available that could deal with Spanish text, thus only a very small and basic selection of the algorithms implemented in this tool could be applied to the Spanish corpus. These results were still helpful for building the ontology, but they could not be used directly.

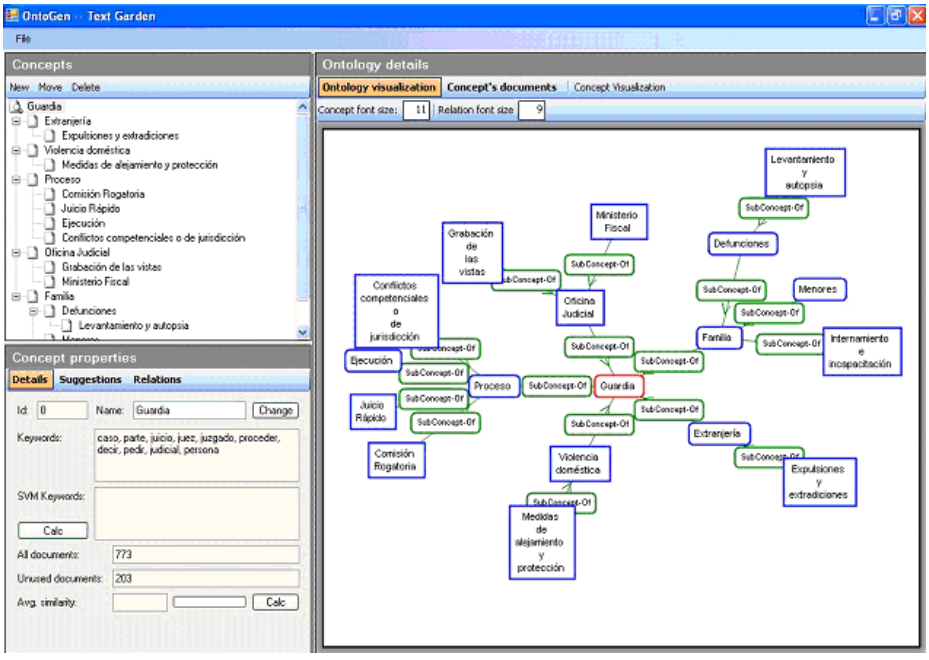

Fig. 5. The ontology learned from the cases in the Ontogen system

Ontology evaluation The ontology was manually built from the automatically extracted terms. Therefore, there was no need to evaluate the learned ontology separately.

Ontology integration We integrated the OPJK with the learned ontology by mapping similar concepts. This allowed to calculate similarity measures between the FAQs and the concepts of the learned ontology. The final evaluation of the learned ontology was achieved by assessing the ranking of the cases.

The resulting ontologies were used, as shown in Fig. 3, in order to find the most similar cases to a given user question.

5.5 Lessons Learned and Open Issues

Our lessons learned relate to the operation of the ontology learning process and to the tools used for this purpose. The domain experts appreciated the process description as it guided them through the ontology learning effort. They were satisfied with the level of detail of the description, especially the fine-grained elaboration of the single tasks and the clear definition of the results of each phase were particularly helpful. The direct interaction with the learning software required, however, additional efforts. We hope to establish this process model as a guideline for tool developers to implement process support accordingly.

A detailed report on the user interface requirements for ontology learning tools is out of the scope of this paper. Our case study showed that ontology learning tools require explicit process support in order to allow for their application in a real life ontology engineering context. Currently non-ontology learning experts cannot achieve good results as the customization of the tools requires a deeper knowledge of the applied algorithms. We thus suggest to develop methods to support the selection of these algorithms and their customization as the algorithms can be optimized for different scenarios. In this context we encourage developers to spend more effort in good user interface design. Moreover, the integration of manual and automatic ontology building needs improvement. For example the integration with competency questions would help the ontology engineer to decide whether the learned ontology is complete or not.

Ontology learning tools proved to be particularly useful in visualizing the available knowledge and in showing that it is organized within distinct subdomains, which could then be considered separately.

Beyond the scope of the case study we are aware of several open issues in our work which require further investigation. A number of activities covered by the process model are particularly challenging even for experts in the field of knowledge acquisition and ontology learning if they are not familiar with the functionality and the methods embedded in each of the tools potentially to be utilized. Such activities need to be further elaborated in order to ease their operation in real-world scenarios. The best examples are the evaluation of the learning results and the selection of tools for (partially) building the ontology. For the former we ideally should extend our process model with in-depth considerations of the possible evaluation approaches and relate these to

the tools and methods employed. For the latter the methodology would surely benefit from further examples complemented by explanations of the trade-offs associated with each method and their appropriateness with respect to various properties of the target ontology, the application using it or the learning corpus.

6 Conclusions

In this chapter we have presented an ontology learning methodology which is integrated into the overall ontology engineering process. The proposed methodology has eight phases. It starts with a feasibility study and requirements specification phase and ends with the integration of the learned ontologies with other ontologies developed in a given context. The methodology describes in detail input and output factors, activities which should be performed in each of the phases and decision criteria. It thus provides domain experts with detailed guidance for selecting and preparing information sources, applying ontology learning tools and evaluating the learned ontology.

Furthermore, we have described a case study in which we have applied the methodology. In the case study, legal experts built an ontology with the help of ontology learning tools for case rulings in the domain of domestic violence and integrated it with a manually build ontology capturing professional legal knowledge. As a result of our observations we could validate our hypothesis that in order for domain experts to effectively and efficiently apply ontology learning methods they require a detailed methodology guiding them through the respective process. However, we found that current ontology learning tools are by far too complicated to be applied by domain experts on their own, and that such tools need at least major improvements from a usability point of view. We envision that the proposed methodology will contribute to the wider adoption of ontology learning as a viable method to develop ontologies and will help tool developers to implement their tools in a more user-oriented way.

References

1. Gómez-Pérez, A., Fernández-López, M., Corcho, O.: Ontological Engineering. Springer (2003)
2. Maedche, A.: Ontology Learning for the Semantic Web. Kluwer Academic Publisher (2002)
3. Chapman, P., Kerber, R., Clinton, J., Khabaza, T., Reinartz, T., Wirth, R.: The CRISP-DM process model. Discussion Paper. http://www.crisp-dm.org (1999)
4. Aussenac-Gilles, N., Biebow, B., Szulman, S.: Revisiting ontology design: A methodology based on corpus analysis. In Dieng, R., Corby, O., eds.: Proceedings of the 12th International Conference onKnowledge Acquisition, Modeling and Management (EKAW 2000). Volume 1937 of Lecture Notes in Computer Science., Springer (2000) 172–188
5. Aussenac-Gilles, N., Despres, S., Szulman, S.: The TERMINAE method and platform for ontology engineering from text. In Buitelaar, P., Cimiano, P., eds.: Bridging the Gap between Text and Knowledge - Selected Contributions to Ontology Learning and Population from Text. IOS Press (2007) THIS VOLUME.

6. Sure, Y., Tempich, C., Vrandecic, D.: Ontology engineering methodologies. In: Semantic Web Technologies: Trends and Research in Ontology-based Systems. Wiley, UK (2006)
7. Euzenat, J.: Building consensual knowledge bases: Context and architecture. In: Proc. of the 2nd Int. Conference on Building and Sharing Very Large-Scale Knowledge Bases (KBKS), Enschede the Netherlands (1995) 143–155
8. Pinto, H.S., Tempich, C., Staab, S.: Diligent: Towards a fine-grained methodology for distributed, loosely-controlled and evolving engingeering of ontologies. In: Proc. of the 16th European Conference on Artificial Intelligence (ECAI 2004). (2004) 393–397
9. Gangemi, A., Pisanelli, D., Steve, G.: Ontology integration: Experiences with medical terminologies. In: Formal Ontology in Information Systems, IOS Press (1998)
10. Pinto, H.S., Martins, J.: Reusing ontologies. In: AAAI 2000 Spring Symposium on Bringing Knowledge to Business Processes. (2000) 77–84
11. Sure, Y., Studer, R.: On-To-Knowledge methodology. In: On-To-Knowledge: Semantic Web enabled Knowledge Management. J. Wiley and Sons (2002)
12. Gómez-Pérez, A.: Evaluation of ontologies. International Journal of Intelligent Systems **16** (2001) 391–409
13. Cimiano, P., Maedche, A., Staab, S., Völker, J.: Ontology Learning. In Staab, S., Studer, R., eds.: Handbook of Ontologies. Volume 2nd Edition. Springer (2008) to appear.
14. Maedche, A., Staab, S.: Ontology Learning for the Semantic Web. IEEE Intelligent Systems **16** (2001) 72 – 79
15. KnowledgeWeb European Project: Methods for ontology evaluation (Deliverable D1.2.3 KnoweldgeWeb FP6-507482) (2004)
16. Dellschaft, K., Staab, S.: Strategies for the evaluation of ontology learning. In Buitelaar, P., Cimiano, P., eds.: Bridging the Gap between Text and Knowledge - Selected Contributions to Ontology Learning and Population from Text. IOS Press (2007) THIS VOLUME.
17. Pinto, H.S., Martins, J.P.: A methodology for ontology integration. In: Proceedings of the international conference on Knowledge capture (K-CAP), ACM Press (2001) 131–138
18. Gangemi, A., Pisanelli, D.M., Steve, G.: An overview of the ONIONS project: Applying ontologies to the integration of medical terminologies. Data Knowledge Engineering **31** (1999) 183–220
19. Benjamins, V.R., Casanovas, P., Contreras, J., Lopez-Cobo, J.M., Lemus, L.: Iuriservice: An intelligent frequently asked questions system to assist newly appointed judges. In: Law and the Semantic Web. Springer (2005) 201–217
20. Casanovas, P., Vallbe, J.J., Casellas, N., Poblet, M., Blazquez, M., Contreras, J., Benjamins, V.R., Lopez-Cobo, J.M.: D10.4.1 after analysis. implementation of iuriservice at the spanish judicial school. SEKT official deliverable 10.4.1, Institute AIFB, University of Karlsruhe (2006)
21. Fortuna, B., Grobelnik, M., Mladenič, D.: Background knowledge for ontology construction. In: WWW '06: Proceedings of the 15th international conference on World Wide Web, New York, NY, USA, ACM Press (2006) 949–950
22. Cimiano, P., Völker, J.: Text2onto - a framework for ontology learning and data-driven change discovery. In: Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems (NLDB), Springer (2005) 227–238

Part VI: Evaluation

Strategies for the Evaluation of Ontology Learning

Klaas DELLSCHAFT [a] and Steffen STAAB [a]
[a] *Universität Koblenz-Landau, Koblenz, Germany*

Abstract. An important aspect of ontology learning is a proper evaluation. Generally, one can distinguish between two scenarios: (i) quality assurance during an ontology engineering project in which also ontology learning techniques may be used and (ii) evaluating and comparing ontology learning algorithms in the laboratory during their development. This paper gives an overview of different evaluation approaches and matches them against the requirements of the scenarios. It will be shown that different evaluation approaches have to be applied depending on the scenario. Special attention will be paid to the second scenario and the gold standard based evaluation of ontology learning for which concrete measures for the lexical and taxonomic layer will be presented.

Keywords. ontology learning, evaluation

1. Introduction

When it comes to the evaluation of ontology learning one has to distinguish between two different scenarios: On the one hand there is the scenario where an ontology learning algorithm is used in the context of an automatic or semi-automatic approach to ontology engineering (cf. [1] and [2]). On the other hand there is the evaluation of the ontology learning algorithm itself (cf. [3] and [4]). Both scenarios differ in their requirements with regard to their evaluation thus leading to different evaluation approaches.

In Fig. 1 one can see the general approach of ontology learning: It takes as input a domain centered corpus and tries to learn an ontology which conceptualizes the information implicitly available in the corpus. Depending on the scenario, different aspects of ontology learning have to be evaluated. In the first scenario not only the learning algorithm influences the results but another important aspect is the choice of the correct corpus which has to contain information relevant for the task. In the second scenario one is only interested in the quality of the learning algorithm itself.

We argue in this paper that in the first scenario the functional dimension of an ontology should be evaluated by means of an extrinsic, task-based evaluation, i. e. in the running application for which the ontology is engineered. This kind of evaluation ensures that the objective of using an ontology, improving a certain task, is really achieved. But this approach is not feasible in the second scenario where the aim is to compare ontology learning algorithms. Its objective is the assessment of the quality of different learning algorithms. In this scenario, an intrinsic or task-neutral evaluation by means of a gold-standard based evaluation is usually the better choice.

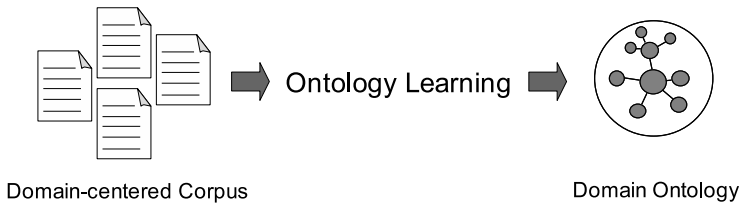

Figure 1. General Approach of Ontology Learning

After describing the two scenarios and their requirements with regard to the evaluation in more detail we will focus in the remainder of this paper on the second scenario and especially on the gold standard based evaluation of the lexical and taxonomic layer of ontologies. It will be shown that existing measures have been faulty and that a well-founded evaluation model is largely missing. Therefore, we describe a new framework for gold standard-based evaluation of ontology learning which includes measures for the lexical and the taxonomic layer of an ontology. The framework avoids common mistakes and we show by analytical considerations and by some experiments that it fulfills crucial evaluation criteria that other frameworks do not meet.

2. Evaluation Scenarios and Approaches

In the following, we will present the two scenarios in more detail and list several applicable evaluation approaches. We will distinguish between approaches which try to measure the functional dimension of a learned ontology and the ones which measure the structural dimension (cf. [5]). The functional dimension of an ontology is related to its conceptualization while the structural dimension is related to the representation of an ontology as a graph. It will be shown that, depending on the scenario, different evaluation approaches for the functional dimension should be used.

2.1. Scenario 1: Quality Assurance During Ontology Engineering

In this scenario ontologies are evaluated during the ontology engineering process as part of the quality assurance. Typical questions for evaluating an ontology are whether it is consistent, complete, concise and expandable (see [6]). For this purpose, in [5] and [7] it is proposed to measure the structural and functional dimension of ontologies and their usability profile. The requirements which should be fulfilled by an ontology with regard to the dimensions will be usually defined during the start phase of an ontology engineering project.

For example, one may check whether the target domain of the ontology is sufficiently modeled to fulfill the functional requirements and/or whether the ontology helps to improve the performance in the task for which it is designed. As a consequence of such an evaluation one may e. g. decide to further extend some aspects of the ontology. During the structural evaluation it is checked whether certain criteria are fulfilled which are related to the design principles of good ontologies and which help to improve an ontology's overall quality. In [5] it is additionally proposed to evaluate the usability pro-

file of an ontology. During such an evaluation the quantity and quality of the ontology's metadata is checked which address the communication context of an ontology.

In this scenario, the evaluation of ontologies is seen as an important part of the quality assurance process. In many cases the ontologies will be manually engineered. But in case of semi-automatic approaches to ontology engineering (cf. [1] and [2]) also the output of ontology learning algorithms is contained in the engineered ontology and it is thus also evaluated during the quality assurance process.

The following approaches to a functional and structural evaluation can be identified in this scenario:

Task-based Approaches Task-based approaches try to measure in how far an ontology helps to improve the results of a certain task. They will usually measure the functional dimension of an ontology but also the structural dimension may influence the outcome of task-based evaluations. For example, if one designs an ontology for improving the performance of a web search engine (cf. [8]) one may collect several example queries and compare whether the search results contain more relevant documents if a certain ontology is used. A task-based evaluation is influenced by many aspects which have to be kept constant during all evaluations so that changes in the results can be put down to the changes in the used ontologies. The choice of concrete measures for such an evaluation is dependent on the task, e. g. for the web search engine example one may adapt measures known from information retrieval but also other success criteria may be defined.

Because every task-based evaluation is individual, no finite set of well-suited measures can be defined. Nevertheless, some principles can be identified: Usually, it is not enough to know whether an ontology is better or worse than another but one wants to conclude on concrete shortcomings in its conceptualization. Thus, in [9] it is demanded that a task-based evaluation allows for concluding on insertion, deletion and substitution errors in the ontology, i. e. whether there are superfluous, missing or off-target concepts and/or relations. But again, there is no universally valid way how the principles can be realized in a concrete task-based evaluation. In [9] it is only demonstrated for one example task.

Corpus-based Approaches Corpus-based approaches are used for checking in how far an ontology sufficiently covers a given domain. They address the functional dimension of an ontology. For this purpose, the ontology is compared with the content of a text corpus which is representative for the domain. The content of the corpus is analyzed with natural language techniques, e. g. in [10] Latent Semantic Analysis and a clustering method were applied for identifying terms in the corpus. The list of identified terms was then compared with the terms in the evaluated ontology. Similar approaches for evaluating the lexical layer of an ontology are described in [11] and [12] while [13] contains a preliminary method applicable for evaluating triples in ontologies.

All the corpus-based approaches have in common that they involve information extraction and/or ontology learning techniques in the evaluation. Thus they are only partially suitable for evaluating ontologies which were learned with other ontology learning algorithms because the information extraction and/or ontology learning techniques from the corpus-based approaches are like a benchmark which can not be outperformed by other ontology learning algorithms. In [11] it is proposed to

evaluate and extend ontologies at the same time with such an approach, e. g. by suggesting terms which are currently missing in the ontology and which would improve the evaluation results.

Criteria-based Approaches In this category fall a wide variety of evaluation measures which all have in common that they measure in how far an ontology or taxonomy adheres to certain desirable criteria. One can distinguish between measures related to the structure of an ontology, e. g. if it is represented as a graph, and more sophisticated measures which e. g. evaluate a taxonomy based on philosophical notions.

Structural measures are quite straightforward and easy to understand: For example, one may measure the average depth of paths from root to leaf nodes in a directed graph, how many nodes have more than one ingoing arc (i. e. multi-hierarchical nodes) or whether there are cycles in the directed graph (cf. [5] and [7]). But also for ontologies based on frame logic or description logic one may define structural measures, e. g. for detecting potential inconsistencies in the partitioning of a taxonomy (cf. [6]). Such a partitioning error measure may for example find instances belonging to more than one class where two or more of the classes are defined as disjoint.

For the structural measures it is usually no problem to have a fully automatic evaluation. This is not the case for the more sophisticated measures like OntoClean [14] which evaluates taxonomies based on philosophical notions like the essence, identity and unity which should be taken into account during modeling an ontology in order to avoid common pitfalls. For example, a property is essential for an entity if it holds for that entity in every possible world. Furthermore, a property is rigid if it is essential for all its possible instances. In [14], this is explained with the example relations *having a brain* and *being a student*. In this example, the *having a brain* relation is essential for all human beings thus it is a rigid property. In contrast the *being a student* relation, which is not essential for any human being as everyone can become a student or cease to be a student at any time. Thus it would be an anti-rigid property. (For more examples and explanations see [14].) Because of this high complexity, OntoClean is designed for manually analyzing ontologies although an approach for partially automating this process was recently proposed (see [15]).

The most important success criterion for an ontology engineering project is whether the final ontology helps to improve the task for which it was engineered. Thus, improving the results during a task-based evaluation can be seen as the most important goal. Corpus-based and criteria-based evaluation approaches only help to pinpoint the remaining problems which should be addressed in an improved version of the ontology. The main assumption behind corpus-based and criteria-based evaluation measures is that an improvement with regard to the measures correlates with an improvement in the task-based evaluation (see [8] where the correlation was shown for OntoClean [14]).

2.2. Scenario 2: Comparing Ontology Learning Algorithms

In this scenario one tries to assess and compare ontology learning algorithms with each other. It can be used by researchers to improve an existing learning algorithm or to find out how changing the values of input parameters affects the results. An example how

such an evaluation may look like is available in [4]. It is the goal in this scenario to measure the quality of an ontology learning algorithm. This should ideally be done by looking at the output (i. e. the learned ontology) and comparing it with the input (i. e. the content of the corpus). As we will see below, there basically exist two approaches how one can approximate the comparison with the input by either making a manual evaluation by human experts or a gold-standard based evaluation where the gold-standard covers the content of the corpus.

With regard to the functional dimension of the learned ontology one is interested in measuring in how far the learning algorithm is able to conceptualize the information from a given corpus (e. g. whether it extracts isA-relations between relevant concepts) and which fraction is found. This corresponds to measuring the precision and recall (see 4.1). But also from evaluating the structural dimension of the learned ontology one may draw interesting conclusions on the qualities of a learning algorithm.

In the following, two approaches to measuring the functional dimension will be presented which are specific for the needs in this scenario and which are different to the approaches from the first scenario. In contrast, it is possible to re-use a subset of the structural measures described in 2.1. Thus, we will concentrate here on the evaluation of the functional dimension.

In the previous scenario, a task-based evaluation was considered as ideal for evaluating the functional dimension of an ontology. This is not the case for the evaluation and comparison of ontology learning algorithms. Here, it would be necessary to filter out the influence of the task on the evaluation results in order to make valid conclusions on the strength and weaknesses of the learning algorithm itself. For example, the results of a task-based evaluation would be influenced by many other factors like the choice of the corpus, the task itself or the algorithm used for performing the task. Additionally, it is very difficult to conclude from the results of a task-based evaluation on the concrete precision and recall values achieved by the learning algorithm. Instead, it would be valuable to have a more direct approach to measuring those dimensions of interest. All in all, the following list of criteria should be fulfilled by an evaluation in this scenario:

- The evaluation should be task neutral and allow developers to easily pinpoint the advantages and disadvantages of a learning algorithm. Weighing the different advantages and disadvantages of a learning algorithm is then up to the ontology engineer who has a concrete task in mind. This weighing can be based on his experience or even on a task-based evaluation where it was shown that certain aspects of an ontology are more important than others.
- All influencing factors of the evaluation have to be sufficiently described so that its results can be reproduced at another time and place. This is important for having a proper scientific evaluation in general and also applies for other approaches and scenarios like task-based evaluation approaches.
- It should be possible to do additional evaluation runs at low cost because frequent and large-scale evaluations are required during developing ontology learning algorithms, e. g. in order to find the parameter values of the learning algorithm for which the best results are achieved. It has to be ensured that all evaluation runs are performed under the same conditions in order to have comparable results.

By looking at the literature, one can identify the following two approaches for measuring the functional dimension in this scenario:

Manual Evaluation by Human Experts This evaluation approach can be found in several papers about ontology learning algorithms like in [16] and [17] where the learned ontology is presented to one or more human experts which have to judge in how far the extracted information is correct (i. e. the precision is measured). But the approach has several downsides: First of all, the extracted information is not compared with the information found in the corpus but with the knowledge of the human expert. While this is not so problematic for measuring the precision of the learning algorithm it makes a reliable measurement of the recall nearly impossible. Furthermore, the most important influencing factor of the evaluation is the choice of the human experts. Because they may not be available at another time and place the last two criteria outlined before are not fulfilled. This problem can only be avoided by asking a sufficiently large number of experts. Additionally, every evaluation run comes with the same high costs as the first run thus making frequent and large-scale evaluations unfeasible.

Gold Standard Based Approaches Gold standard based approaches compare the learned ontology with a previously created gold standard which represents an idealized outcome of the learning algorithm. A learning algorithm is considered to be better when the learned ontology has a high similarity with the gold standard. Examples for this kind of evaluation can be found in papers like [3], [12] and [18]. The gold standard based evaluation fulfills all the criteria from above: It can be used for directly measuring the precision and recall of the learned ontology compared to the gold standard. Furthermore, the evaluation results can be reproduced and are comparable if the same corpus, learning algorithm and gold standard are used. Additionally, only for the first run of the evaluation the high costs of creating the gold standard exist. Subsequent runs of the evaluation are then fully automatic.

Although the gold standard based evaluation seems to be ideal in this scenario there remains one big issue: Where to get or how to create such a gold standard? On the one hand, one may ask a human expert to create a gold standard based on the information in the used corpus. Depending on the size of the corpus, this can constitute a very work intensive approach. Another approach might be to take an already existing ontology and choose the corpus accordingly so that it can be assumed that most of the information of the gold standard is available in the corpus. An example of the latter approach is available in [4].

Independent from this decision, the term “gold standard” may be misleading as there exists not only one gold standard but, depending on who is asked, one may get several gold standards which differ in their details. This is due to the different conceptualization humans may have of a domain (cf. [5]). The same problem exists for the manual evaluation by human experts. There it is typically addressed by measuring the consensus between several experts (cf. [19]). A similar way may be used for the creation of the gold standard. For example, one may involve several experts in the creation of the gold standard and measure their consensus or one may compare with several gold standards (and measuring the agreement between those gold standards). But regardless of this decision, the main advantage of gold standard based evaluation remains that the conceptualizations of the experts become explicitly available in form of the gold standard. This ensures that every learning algorithm is compared against the same standard and that everyone can control how thoroughly the gold standard was created.

There exist many measures for the gold standard based evaluation of ontologies. They can be distinguished between measures which only evaluate the lexical layer of an ontology, the ones which also take the concept hierarchy or taxonomic layer into account and the ones which evaluate the non-taxonomic relations contained in an ontology. In this paper we will concentrate on the measures for evaluating the lexical and the taxonomic layer.

On the lexical layer "binary" measures are often used that compare the terms from the reference and the learned ontology based on an exact match of strings. Examples for this kind of measure are the *Term Precision and Term Recall* as they are presented in [18]. There exist several other names for these measures like *Lexical Precision and Recall* or simply *precision and recall* (see [20] and [21]). Another example of a lexical evaluation measure is the *String Matching* measure presented in [22] and [19]. This measure is based on the edit distance between two strings. It is therefore more robust with regard to slightly different spellings and typing errors (e. g. "center" and "centre").

The comparison of concept hierarchies or taxonomies is more complicated than the comparison of the lexical layer of ontologies. Such concept hierarchy measures are often divided into kinds of local and global measures. The local measure compares the similarity of the positions of two concepts in the learned and the reference hierarchy. The global measure is then computed by averaging the results of the local measure for concept pairs from the reference and the learned ontology.

Furthermore, we have to distinguish between different learning approaches. An example for such an approach is the *General Named Entity Identification* (GNE) where the algorithm has to find for previously unknown concepts their maximally specific generalization from a given ontology, i. e. it adds them as leaf nodes to the ontology. Examples of measures suitable for evaluating GNE algorithms are available in [23]. They partially depend on the assumption that the compared ontologies only differ in their leaf nodes.

But in this paper we will concentrate on another, more general approach where ontologies are learned from scratch, i. e. without a seed ontology which is extended. In the case of concept hierarchies, it leads to the fact that not only the positions of leaf nodes may differ between the learned and the reference hierarchy but also the position of inner concepts. Thus, the evaluation measures can not depend on the assumption that large portions of the two compared hierarchies (i. e. the seed hierarchy) match exactly.

One of the first examples of such a concept hierarchy evaluation measure is the *Taxonomic Overlap* (TO) presented in [22] and [19]. The local taxonomic overlap compares two concepts based on the set of all their super- and sub concepts. In opposite to the local overlap, which is a symmetric measure, this is not the case for the global taxonomic overlap measures proposed in [22], [19] and [4], i. e. they can be computed into two directions. In [4] this asymmetry is interpreted as a kind of precision and recall. But in section 4.5 we will show that this is a misinterpretation of the asymmetry, as local taxonomic overlap already constitutes a kind of combination of precision and recall.

Another example is the *Augmented Precision and Recall* (AP & AR) presented in [24] and [25]. It is also divided into a global and a local part of the measure. For the local part two alternatives may be used: The *Learning Accuracy* (LA) and the *Balanced Distance Metric* (BDM). LA was proposed by [26]. It compares two concepts based on their distance in the tree (e. g. the length of the shortest path between the root and their most specific common abstraction). BDM further develops the idea of LA by taking further types of paths and a branching factor of the concepts into account (see [24]).

Table 1. Rating of concept hierarchy measures

	multi dimensionality	proportional error effect	usage of interval
TO	−	+	?
AP & AR	∘	+	?
LA	−	∘	?
OntoRand[1]	−	+	+
TP_{csc} (cf. section 4.3)	+	+	+

The latest measure for comparing concept hierarchies is the *OntoRand* index proposed in [27]. It is a symmetric measure which extends techniques used in the clustering community for comparing two partitions of the same set of instances. A concept hierarchy is seen as a hierarchical partitioning of instances. For OntoRand two alternatives exist to measure the similarity of concepts. The first alternative is based on the set of common ancestors. The second alternative is based on the distance between two concepts in the tree (like LA and BDM). An important constraint imposed on the concept hierarchy is that both compared hierarchies must contain the same set of instances.

3. Criteria for Good Evaluation Measures

Given this variety of evaluation measures for doing a gold standard based evaluation of concept hierarchies it is now the question what is a "good" measure and can we give some criteria according to which to evaluate the different measures. Measures fulfilling the following criteria will help to avoid misinterpreting evaluation results and ease drawing the right conclusions for the improvement of the evaluated ontology learning algorithm.

The most important criterion is that an ontology is evaluated along **multiple dimensions**. This criterion is formulated in several papers like [24] and [28]. But instead of having a measure which aggregates the evaluation of all those dimensions into a single value one should use separate measures for each of the dimensions. Thus a user can weight different kinds of errors based on his own preferences. This enables to better analyze the strengths and weaknesses of a learned ontology.

As we will show in 5, it is very important that a measure is only influenced by exactly one dimension and/or type of error. For example, if one uses measures for evaluating the lexical layer of an ontology (e.g the lexical precision and recall) and one also wants to evaluate the quality of the learned concept hierarchy (e. g. with the taxonomic overlap), then a dependency between those measures should be avoided.

The second criterion is that the effect of an error onto the measure should be **proportional** to the distance between the correct and the given result. For example, an error near the root of a concept hierarchy should have a stronger effect on the evaluation measure than an error nearer to the leafs (see also [28]).

The third criterion is closely related to the previous one. For measures with a closed scale **interval** (e. g. $[0..1]$), a gradual increase in the error rate should also lead to a gradual decrease in the evaluation results. For example, if a measure has the interval $[0..1]$ as its scale but already slight errors lead to a decrease of the returned results from 1 to 0.2 then it is difficult to distinguish between slight and severe errors (see [27]).

In Tab. 1 it is shown in how far the measures for the functional dimension described in section 2.2 meet the criteria listed in this section. The rating is based on the descriptions in [19], [24] and [27]. Additionally, the new findings from section 4.5 were used for rating the taxonomic overlap. A measure can improve its multi dimensionality by two factors: either by removing the influence of the lexical layer on the evaluation of the concept hierarchy or by separately measuring different aspects of the hierarchy (e. g. precision and recall). None of the measures removes the influence of the lexical layer and only the augmented precision and recall distinguishes between two aspects of the hierarchy. The Learning Accuracy does not achieve the best score for the proportional error effect because it considers the distance between the correct and the given answer only to some small extent (see [24]). In the following a truly multi dimensional approach for evaluating an ontology will be presented, thus overcoming the problems of the current measures.

4. Comparing Learned Ontologies with Gold Standards

In this section measures will be presented which can be used for an evaluation of the lexical layer and the concept hierarchy of an ontology. The measures extend the idea of precision and recall to the gold standard based evaluation of ontologies. The lexical layer of an ontology will be evaluated with lexical precision and recall (see section 4.2). For the concept hierarchy a framework of building blocks will be defined in section 4.3. This framework defines a family of measures and it will be used for systematically constructing a measure which fulfills the criteria from section 3.

In the following the simplified definition of a core ontology will be used. This definition of an ontology only contains the lexical layer and the concept hierarchy. Similarly to [4], we define a core ontology as follows:

Definition 1 *The structure $\mathcal{O} := (\mathcal{C}, root, \leq_{\mathcal{C}})$ is called a core ontology. $\mathcal{C}$ is a set of concept identifiers and $root$ is a designated root concept for the partial order $\leq_{\mathcal{C}}$ on $\mathcal{C}$. This partial order is called concept hierarchy or taxonomy. The equation $\forall c \in \mathcal{C} : c \leq_{\mathcal{C}} root$ holds for this concept hierarchy.*

In this definition of a core ontology the relation between terms on the lexical layer and their associated concept is a bijection, i. e. each term is associated with exactly one concept and each concept with exactly one term. Thus it is possible to use the a term as the identifier of a concept. This restriction simplifies the following formulas. Nevertheless it would be possible to generalize them to the case where an $n : m$ relation between concepts and terms exists (in analogy to [22] and [19]).

4.1. Precision & Recall

This section gives a short overview of precision, recall and F-measure, as they are known from information retrieval (see [29]). They are used for comparing a reference retrieval

[1] In [27] two different variants of OntoRand are presented. One variant is based on a tree distance while the other is based on finding the common ancestor in the concept hierarchy. For the comparison in Tab. 1 the measure based on the common ancestors was used because it was shown in [27] that it is superior to the tree distance based measure.

(Ref) with a computed retrieval ($Comp$) returned by a system. Precision and recall are defined as follows:

$$P(Ref, Comp) = \frac{|Comp \cap Ref|}{|Comp|} \qquad R(Ref, Comp) = \frac{|Comp \cap Ref|}{|Ref|} \tag{1}$$

It is interesting that precision and recall are the inverse of each other:

$$P(Ref, Comp) = \frac{|Comp \cap Ref|}{|Comp|} = R(Comp, Ref) \tag{2}$$

The F_1-measure is used for giving a summarizing overview and for balancing the precision and recall values. The F_1-measure is the harmonic mean of P and R.

$$F_1(Ref, Comp) = \frac{2 \cdot P(Ref, Comp) \cdot R(Ref, Comp)}{P(Ref, Comp) + R(Ref, Comp)} \tag{3}$$

4.2. Lexical Precision & Recall

There exist several measures sufficient for evaluating the lexical layer of an ontology (see section 2.2). In this subsection the lexical precision and recall measures, as they are described in [20], will be explained in a bit more detail. Later on they will be used in conjunction with the measures for evaluating concept hierarchies, as they are presented in section 4.3. Given a computed core ontology $\mathcal{O}_C$ and a reference ontology $\mathcal{O}_R$, the lexical precision (LP) and lexical recall (LR) are defined as follows:

$$LP(\mathcal{O}_C, \mathcal{O}_R) = \frac{|\mathcal{C}_C \cap \mathcal{C}_R|}{|\mathcal{C}_C|} \qquad LR(\mathcal{O}_C, \mathcal{O}_R) = \frac{|\mathcal{C}_C \cap \mathcal{C}_R|}{|\mathcal{C}_R|} \tag{4}$$

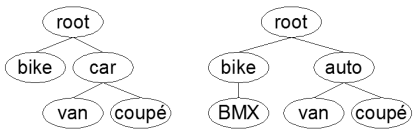

Figure 2. Example reference ontology ($\mathcal{O}_{R1}$, left) and computed ontology ($\mathcal{O}_{C1}$, right)

The lexical precision and recall reflect how good the learned terms cover the target domain. For example, if one compares $\mathcal{O}_{C1}$ and $\mathcal{O}_{R1}$ in Fig. 2 with each other, one gets $LP(\mathcal{O}_{C1}, \mathcal{O}_{R1}) = \frac{4}{6} = 0.67$ and $LR(\mathcal{O}_{C1}, \mathcal{O}_{R1}) = \frac{4}{5} = 0.8$.

4.3. Taxonomic Precision & Recall

In this subsection a framework of building blocks is described. It defines a family of taxonomic precision and recall measures from which two concrete measures will be selected afterward. Only the equations for the taxonomic precision measures will be presented. The corresponding equations for the taxonomic recall measures can be easily derived from them because of equation (2). This framework extends and improves the framework used for the taxonomic overlap measures in [19]. It especially replaces the previously used equation for comparing the position of two concepts with each other *leading to a completely different behavior of the measure* (see also section 4.5).

4.3.1. Comparing Concepts

As mentioned before, measures for comparing two concept hierarchies with each other are usually divided into a kind of local and a global measure (cf. section 2.2). The local measure compares the positions of two concepts and the global measure is used for comparing two whole concept hierarchies. We start with describing the framework's local measure. It is then used in the definition of the global measure.

For the local taxonomic precision the similarity of two concepts will be computed based on extracts from the concept hierarchy, which are characteristic for the position of a concept in the hierarchy. That is,. two extracts should contain many common objects if the characterized objects are at similar positions in the hierarchy. The proportion of common objects in the extracts should decrease with increasing dissimilarity of the characterized concepts. Given such a characteristic extract ce, the local taxonomic precision tp_{ce} of two concepts $c_1 \in \mathcal{O}_C$ and $c_2 \in \mathcal{O}_R$ is defined as

$$tp_{ce}(c_1, c_2, \mathcal{O}_C, \mathcal{O}_R) := \frac{|ce(c_1, \mathcal{O}_C) \cap ce(c_2, \mathcal{O}_R)|}{|ce(c_1, \mathcal{O}_C)|} \tag{5}$$

The characteristic extract from the concept hierarchy is an important building block of the local taxonomic measure and several alternative instantiations exist. As we will see below, they have a major influence on the properties of the corresponding global measure. For the taxonomic overlap measure described in [19] it was suggested to characterize a concept by its semantic cotopy, i. e. all its super- and subconcepts. Given the concept $c \in \mathcal{C}$ and the ontology $\mathcal{O}$, the semantic cotopy sc is defined as follows:

$$sc(c, \mathcal{O}) := \{c_i | c_i \in \mathcal{C} \wedge (c_i \leq c \vee c \leq c_i)\} \tag{6}$$

If one uses the semantic cotopy for defining the local taxonomic precision measure tp_{sc}, the results will be heavily influenced by the lexical precision of $\mathcal{O}_C$ because with decreasing lexical precision more and more concepts of $sc(c, \mathcal{O}_C)$ are not contained in $\mathcal{O}_R$ and $sc(c, \mathcal{O}_R)$. This increases the probability that $sc(c, \mathcal{O}_C)$ contains such concepts, leading to a direct dependency between the lexical and the taxonomic precision. But according to section 3, evaluation measures should be judged by whether the different measures are independent of each other. So taxonomic measures based on the semantic cotopy shouldn't be used in conjunction with the lexical precision and recall.

This influence of lexical precision and recall on the taxonomic measures can be avoided if one uses the common semantic cotopy csc as the characteristic extract. The common semantic cotopy excludes all concepts which are not also available in the other ontology's set of concepts:

$$csc(c, \mathcal{O}_1, \mathcal{O}_2) := \{c_i | c_i \in \mathcal{C}_1 \cap \mathcal{C}_2 \wedge (c_i <_1 c \vee c <_1 c_i)\} \tag{7}$$

In Tab. 2 and 3 one can see the influence of inserting and replacing concepts in a hierarchy. The tables contain the sets sc and csc for the ontologies $\mathcal{O}_{R1}$ and $\mathcal{O}_{C1}$ which were already used as an example for lexical precision and recall (see Fig. 2). One can see that inserting and replacing concepts without actually changing the hierarchy has no effect on the common semantic cotopy while the semantic cotopy is heavily influenced by these changes on the lexical layer of an ontology.

Table 2. Semantic cotopies for the ontologies in Fig. 2.

c	$sc(c, \mathcal{O}_{R1})$	$sc(c, \mathcal{O}_{C1})$
root	{root, bike, car, van, coupé}	{root, bike, BMX, auto, van, coupé}
car	{root, car, van, coupé}	–
auto	–	{root, auto, van, coupé}
van	{root, car, van}	{root, auto, van}
coupé	{root, car, coupé}	{root, auto, coupé}
bike	{root, bike}	{root, bike, BMX}
BMX	–	{root, bike, BMX}

Table 3. Common semantic cotopies for the ontologies in Fig. 2.

c	$csc(c, \mathcal{O}_{R1}, \mathcal{O}_{C1})$	$csc(c, \mathcal{O}_{C1}, \mathcal{O}_{R1})$
root	{bike, van, coupé}	{bike, van, coupé}
car	{root, van, coupé}	–
auto	–	{root, van, coupé}
van	{root}	{root}
coupé	{root}	{root}
bike	{root}	{root}
BMX	–	{root, bike}

Besides the previously described extracts of the concept hierarchy, further extracts are imaginable. For example, the upwards cotopy (see [19]) or the set of all direct subconcepts might be used. In [30] also measures based on the direct subconcepts were evaluated. But [30] shows also that measures based on the semantic cotopy meet more of the criteria from section 3.

4.3.2. Comparing Concept Hierarchies

It is now possible to define a framework for constructing a global taxonomic precision measure. Fig. 3 shows the building blocks used in this framework for a global taxonomic precision measure.

$$TP(\mathcal{O}_C, \mathcal{O}_R) := \frac{1}{|\mathcal{C}_C|} \sum_{c \in \mathcal{C}_C} \begin{cases} tp(c, c, \mathcal{O}_C, \mathcal{O}_R) & \text{if } c \in \mathcal{C}_R \\ \max_{c' \notin \mathcal{C}_R} tp(c, c', \mathcal{O}_C, \mathcal{O}_R) & \text{if } c \notin \mathcal{C}_R \end{cases}$$

local taxonomic precision

concept set

estimation

Figure 3. Building blocks of the global taxonomic precision measure

The *set of concepts* whose local taxonomic precision values are summed up is the first building block. Two alternatives may be used. The first alternative is to use the set of concepts $\mathcal{C}_C$ from the learned ontology. If one chooses this alternative, the global taxonomic precision is influenced by the lexical precision. For example, if the lexical precision of a learned ontology is approximately 5% (like in the empirical evaluation in section 5.2) then for 95% of the concepts a local taxonomic precision value has to be estimated because there doesn't exist a corresponding concept in the reference ontology (see below). If such an influence of the lexical precision should be avoided then the set

of common concepts $\mathcal{C}_C \cap \mathcal{C}_R$ should be preferred. It especially makes sense if one also uses a local taxonomic precision value based on the common semantic cotopy.

The *local taxonomic precision* is the next building block. It is used for comparing the position of a concept in the learned hierarchy with the position of the same concept in the reference hierarchy. Thus the current concept has to exist in both hierarchies.

An *estimation* of a local taxonomic precision value is the last building block. It is only used if the current concept isn't contained in both ontologies. Its usage is therefore influenced by the chosen set of concepts (see above). In [19] it is suggested to make an optimistic estimation by comparing the current concept with all concepts from the reference ontology and choose the highest local taxonomic precision value. This ensures that concepts which do not match on the lexical layer (e. g. "auto" and "car" in Fig. 2) will nonetheless match in the concept hierarchy and thus return a high local taxonomic precision value. The optimistic estimation reduces the influence of lexical precision but it may also cause misleading results.

In opposite to that, assuming a local taxonomic precision value of 0% if no match on the lexical layer can be found maximizes the influence of the lexical precision. But if one wants to completely eliminate the influence of lexical precision one should avoid this estimation building block anyway. This is done by only averaging the local taxonomic precision values of the common concepts.

4.3.3. Concrete Measures

In the following the previously presented building blocks will be combined to concrete measures fulfilling the criteria from section 3. The measures will be evaluated in section 5. In [30] further measures are described and evaluated. This paper only contains the best two pairs of measures.

The first pair of measures consists of TP_{sc} and TR_{sc}. They are based on the semantic cotopy and are thus influenced by the lexical layer. In the evaluation in section 5 they will be used for demonstrating the disadvantages of mixing the evaluation of lexical layer and concept hierarchy. The other building blocks are selected so that they further increase this influence. This is achieved by computing the local taxonomic precision for all learned concepts and by estimating the local taxonomic precision as 0 if the current concept isn't also contained in the reference ontology.

$$TP_{sc}(\mathcal{O}_C, \mathcal{O}_R) := \frac{1}{|\mathcal{C}_C|} \sum_{c \in \mathcal{C}_C} \begin{cases} tp_{sc}(c, c, \mathcal{O}_C, \mathcal{O}_R) & \text{if } c \in \mathcal{C}_R \\ 0 & \text{if } c \notin \mathcal{C}_R \end{cases} \tag{8}$$

$$TR_{sc}(\mathcal{O}_C, \mathcal{O}_R) := TP_{sc}(\mathcal{O}_R, \mathcal{O}_C) \tag{9}$$

All in all, the measures TP_{sc} and TR_{sc} do not allow a separate evaluation of lexical layer and concept hierarchy. For evaluation scenarios where a thorough analysis of the learned ontologies is needed the measures TP_{csc} and TR_{csc} are better suited. Here the building blocks will be selected so that the influence of the lexical layer is minimized. This is achieved by using the common semantic cotopy and by computing the taxonomic precision values only for the common concepts of both ontologies. The latter makes the estimation of local taxonomic precision values unnecessary.

$$TP_{csc}(\mathcal{O}_C, \mathcal{O}_R) := \frac{1}{|\mathcal{C}_C \cap \mathcal{C}_R|} \sum_{c \in \mathcal{C}_C \cap \mathcal{C}_R} tp_{csc}(c, c, \mathcal{O}_C, \mathcal{O}_R) \tag{10}$$

$$TR_{csc}(\mathcal{O}_C, \mathcal{O}_R) := TP_{csc}(\mathcal{O}_R, \mathcal{O}_C) \tag{11}$$

4.4. Taxonomic F- and F'-Measure

Like it is the case for precision and recall in information retrieval, also the taxonomic precision and recall have to be balanced if one wants to output a combined measure. Therefore the taxonomic F-measure is introduced, which is the harmonic mean of the global taxonomic precision and recall.

$$TF(\mathcal{O}_C, \mathcal{O}_R) := \frac{2 \cdot TP(\mathcal{O}_C, \mathcal{O}_R) \cdot TR(\mathcal{O}_C, \mathcal{O}_R)}{TP(\mathcal{O}_C, \mathcal{O}_R) + TR(\mathcal{O}_C, \mathcal{O}_R)} \tag{12}$$

A higher taxonomic F-measure corresponds to a better quality of the concept hierarchy. The meaningfulness with regard to the overall quality of the ontology (lexical level + taxxonomy) depends on the chosen building blocks. If TF is not influenced by the lexical level then the taxonomic F'-measure (see [4]) may additionally be computed. It is the harmonic mean of LR and TF:

$$TF'(\mathcal{O}_C, \mathcal{O}_R) := \frac{2 \cdot LR(\mathcal{O}_C, \mathcal{O}_R) \cdot TF(\mathcal{O}_C, \mathcal{O}_R)}{LR(\mathcal{O}_C, \mathcal{O}_R) + TF(\mathcal{O}_C, \mathcal{O}_R)} \tag{13}$$

4.5. Taxonomic Overlap

In [22] and [4] the taxonomic overlap measure is defined. It is also divided into a global and a local part of the measure. The global taxonomic overlap TO has the same building blocks like TP but instead of the local taxonomic precision it uses the local overlap to:

$$to_{sc}(c_1, c_2, \mathcal{O}_1, \mathcal{O}_2) := \frac{|sc(c_1, \mathcal{O}_1) \cap sc(c_2, \mathcal{O}_2)|}{|sc(c_1, \mathcal{O}_1) \cup sc(c_2, \mathcal{O}_2)|} \tag{14}$$

Because to is a symmetric measure, it depends on the other building blocks (concept set and estimation component) whether the global taxonomic overlap is symmetric or asymmetric. We have shown the following lemma (cf. [30] for its proof):

Lemma 1 *Symmetric global taxonomic overlap measures can be solely derived from taxonomic F-measures. The equation $TO = TF/(2 - TF)$ holds.*

This lemma implies that symmetric TO measures behave like TF measures (see [30] for a symmetric TO measure). In [22] and [4] an asymmetric overlap measure is defined. There, this asymmetry is interpreted like a kind of precision and recall. But in [30] it was shown that no strictly monotonic dependency exists between that asymmetric TO measure and corresponding TP and TR measures. Thus the asymmetry can not be interpreted like precision and recall. It should be avoided to use asymmetric TO measures until the unclarity with regard to their interpretation is resolved. Instead corresponding taxonomic precision and recall measures should be used.

Table 4. Evaluation of the ontologies in Fig. 4 with a semantic cotopy based measure

Compare $\mathcal{O}_{R2}$ with	LP	LR	TP_{sc}	TR_{sc}	TF_{sc}	TF'_{sc}
$\mathcal{O}_{C2}$	100.00%	57.14%	100.00%	51.02%	67.57%	61.92%
$\mathcal{O}_{C3}$	71.43%	71.43%	54.25%	54.25%	54.25%	61.67%

Table 5. Evaluation of the ontologies in Fig. 4 with a common semantic cotopy based measure

Compare $\mathcal{O}_{R2}$ with	LP	LR	TP_{csc}	TR_{csc}	TF_{csc}	TF'_{csc}
$\mathcal{O}_{C2}$	100.00%	57.14%	100.00%	100.00%	100.00%	72.73%
$\mathcal{O}_{C3}$	71.43%	71.43%	100.00%	100.00%	100.00%	83.33%

5. Evaluation

In this section the measures presented in 4.3.3 will be analytically and empirically evaluated. In the analytical evaluation it will be checked in how far they fulfill the criteria defined in section 3. Subsequently in the empirical evaluation, it will be shown in how far the choice of the measure influences the outcome of the evaluation of an ontology learning task.

5.1. Analytical Evaluation

First, it will be checked in how far the taxonomic measures are independent of the measures for the lexical layer. This corresponds to the first criterion that a good set of measures allows for evaluating along multiple dimensions. Closely related to this criterion is the objective that each measure is independent of the other measures. The ontologies in Fig. 4 will be used for this purpose. Compared to $\mathcal{O}_{R2}$ there are three concepts missing in $\mathcal{O}_{C2}$, but the hierarchy of the remaining concepts is not changed. Also in $\mathcal{O}_{C3}$ the hierarchy is not changed but the natural language identifier of two concepts is changed (e. g. "car" is renamed to "auto"). Thus the hierarchy of both ontologies is perfectly learned but there are errors on the lexical layer. This has to be reflected by taxonomy measures which are not influenced by errors on the lexical layer.

As one can see in Tab. 4 and 5 only the measures TP_{csc} and TR_{csc} are independent of the lexical precision and recall. But this was already expected from the properties of the single building blocks of the taxonomic measures. It is more surprising to which extent the lexical precision and recall influence TP_{sc} and TR_{sc}. The errors on the lexical layer of both learned ontologies lead to a higher decrease of the taxonomic measures than of the lexical measures. This can be seen by comparing the values of the taxonomic measures and of the lexical measures in Tab. 4. The values of the taxonomic measures are lower than the corresponding values of the lexical measures although the evaluated ontologies only contain errors on the lexical layer.

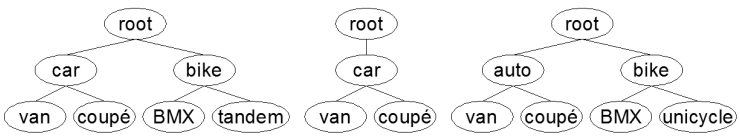

Figure 4. Reference ontology ($\mathcal{O}_{R2}$, left) and two learned ontologies ($\mathcal{O}_{C2}$, middle; $\mathcal{O}_{C3}$, right)

Table 6. Evaluation of the ontologies in Fig. 5 with a semantic cotopy based measure

Compare $\mathcal{O}_{R3}$ with	LP	LR	TP_{sc}	TR_{sc}	TF_{sc}	TF'_{sc}
$\mathcal{O}_{C4}$	100.00%	100.00%	66.67%	66.67%	66.67%	80.00%
$\mathcal{O}_{C5}$	100.00%	100.00%	83.33%	83.33%	83.33%	90.91%

Table 7. Evaluation of the ontologies in Fig. 5 with a common semantic cotopy based measure

Compare $\mathcal{O}_{R3}$ with	LP	LR	TP_{csc}	TR_{csc}	TF_{csc}	TF'_{csc}
$\mathcal{O}_{C4}$	100.00%	100.00%	52.38%	52.38%	52.38%	68.75%
$\mathcal{O}_{C5}$	100.00%	100.00%	76.19%	76.19%	76.19%	84.49%

The second criterion of good evaluation measures was that the effect of an error onto the measure should be proportional to the distance between the correct and the given result. This criterion will be checked with the ontologies in Fig. 5. There, in $\mathcal{O}_{C4}$, the two concepts "car" and "bike" are interchanged, corresponding to an error near the root of the hierarchy. In $\mathcal{O}_{C5}$ the two leaf concepts "coupé" and "BMX" are interchanged. Altogether the errors in $\mathcal{O}_{C4}$ are more serious than the errors in $\mathcal{O}_{C5}$. Thus measures which fulfill this second criterion should rate $\mathcal{O}_{C4}$ worse than $\mathcal{O}_{C5}$. In Tab. 6 and 7 one can see that both pairs of measures fulfill this criterion.

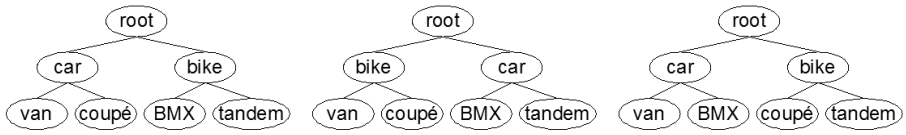

Figure 5. Reference ontology ($\mathcal{O}_{R3}$, left) and two learned ontologies ($\mathcal{O}_{C4}$, middle; $\mathcal{O}_{C5}$, right)

The third and last criterion of good evaluation measures was that a gradual increase in the error rate should lead to a more or less gradual decrease in the evaluation results. One can see from the previously given examples that TP_{csc} and TR_{csc} fulfill this criterion. Especially for the ontologies in Fig. 4 it returned perfect evaluation results. The opposite is true for TP_{sc} and TR_{sc}: Because these measures are influenced by errors in the lexical layer as well as by errors in the concept hierarchy they will drop very fast if both kinds of errors occur in an ontology. Additionally it was shown that they are more strongly influenced by errors in the lexical layer than the lexical precision and recall measure itself.

TP_{csc} and TR_{csc} are all in all better suited for evaluating a concept hierarchy and drawing conclusions about the strengths and weaknesses of the used learning procedure.

5.2. Empirical Evaluation

In this section the previously described measures will be used in a real evaluation of concept hierarchies learned with Hearst patterns (cf. [31], [3]). In this evaluation it will be shown in how far the choice of the measure influences the nature of the results and subsequently the conclusions which are drawn from the evaluation of a learning algorithm. For the evaluation, several ontologies for the tourism domain were learned from a corpus of 4596 tourism related Wikipedia articles with 6.54 million tokens. The reference ontology was created by an experienced ontology engineer within the GETESS project (see [32] and Tab. 8 for more details about the ontology).

If the Hearst patterns are applied on a collection of texts, it is very likely that the same relation is extracted more than once. This information can be used for defining a confidence value in the extracted relation. The confidence is increased, with the number of occurrences. The most often extracted relation gets a confidence value of 1.0. It drops to 0.0 with descending occurrences. Four different thresholds θ were applied to the confidence value for filtering the taxonomic relations. For more details on the experiment and further results for other learning algorithms and document corpora see [30].

In Fig. 6, 7 and 8 one can see the evaluation results for the taxonomic and the lexical layer of the learned ontologies. These raw evaluation results should now be used for deciding for which threshold the best results were achieved. Fig. 7 and 8 contain the evaluation of the taxonomic layer of the same ontologies but evaluated with the two different measures from section 4.3.3.

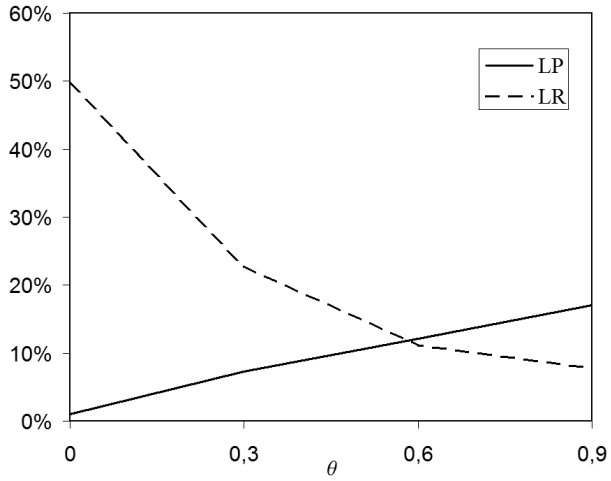

Figure 6. Evaluation of the lexical layer depending on threshold θ

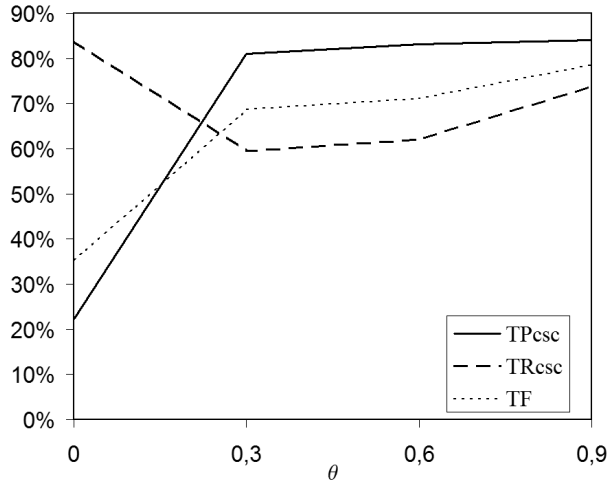

Figure 7. Evaluation of learned ontologies with TP_{csc} depending on threshold θ

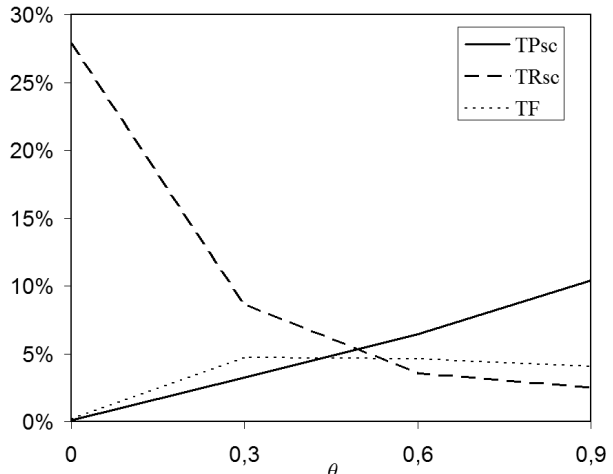

Figure 8. Evaluation of learned ontologies with TP_{sc} depending on threshold θ

Table 8. Structural evaluation of the reference ontology and the learned ontologies

θ	concepts	circles	avg. depth	avg. sub	sub. dev.	avg. super	super dev.
ref	294	1	5.14	5.22	4.42	1.03	0.17
0.0	14569	4973	119.29	3.57	53.2	1.52	2.2
0.3	893	97	3.8	2.81	14.89	1.22	0.87
0.6	246	24	3.29	2.68	8.39	1.16	0.78
0.9	116	2	3.17	2.76	6.06	1.08	0.35

Looking at the results in Fig. 7 one can see that there is a major improvement of the quality on the taxonomic layer if θ is increased from 0.0 to 0.3. But this improvement on the taxonomic layer is accompanied by a decrease of the lexical recall (see Fig. 6) thus it isn't so clear whether the ontologies with $\theta = 0.0$ or 0.3 are better. But from the low lexical and taxonomic precision for a $\theta = 0.0$ one may also conclude that this ontology more or less "accidentally" contains correct terms and taxonomic relations (which lead to the high recall values). So after a deeper analysis of the evaluation results one may come to the conclusion that learning taxonomic relations with Hearst patterns works best if the output ontology is moderately filtered based on the threshold values.

The conclusion based on the functional evaluation of the lexical and taxonomic layer is also supported by the structural evaluation in Tab. 8. The first row of the table contains the values of the reference ontology against which the learned ontologies are compared. The following rows contain the values of the learned ontologies. One can see that the unfiltered concept hierarchy contains 4,973 circularity errors in the concept hierarchy (i. e. a concept is also one of its superconcepts) and that the average cardinality of the paths from the root to the leaf nodes (i. e. the average depth of the hierarchy) is 119. Additionally, it is interesting to look at the branching factor of the hierarchy: The concepts have 3.57 direct subconcepts in average with a very high deviation of 53.2. The average number of direct superconcepts is also quite high with 1.52 and a deviation of 2.2 (i. e. there exist many multi-hierarchical concepts). All these structural measures show that the hierarchy of the unfiltered ontology is more or less degenerated while the values for the ontology with $\theta = 0.3$ are close to the values of the manually built reference ontology.

The exemplary evaluation with TP_{csc} and TR_{csc} shows that they allow for separately evaluating the taxonomic and lexical layer of an ontology. The different evaluation measures have to be weighed and prioritized thus forming an overall picture of the advantages and disadvantages of the ontologies and thus the used learning algorithm.

The separate evaluation of the functional dimension of the taxonomic and lexical layer is not possible if TP_{sc} and TR_{sc} are used instead. In constructing the measures in section 4.3 as well as in the analytical evaluation in 5.1 it was predicted that there is a strong dependency of TP_{sc} and TR_{sc} on the respective measure from the lexical layer. This dependency also becomes obvious by comparing Fig. 6 and Fig. 8. Both graphs show more or less the same information, i. e. the evaluation of the taxonomic layer is superimposed by the influence of the lexical layer. Thus drawing conclusions about the taxonomic layer and making a truly multidimensional evaluation is impossible because the used measures are not independent of each other.

6. Conclusions

In this chapter we presented an overview of several existing approaches to the evaluation of ontologies. It was shown that in the scenario of evaluating ontology learning algorithms a gold standard based evaluation approach is the best choice while for the quality assurance during an ontology engineering project a combination of task-, corpus- and criteria-based evaluation approaches should be used.

Focusing on the scenario of evaluating ontology learning algorithms, we presented a framework for gold standard based evaluations. It was used for creating a measure for the taxonomic layer. It was shown by means of an analytical and empirical evaluation that it fulfills the three basic criteria for gold standard based evaluations: (i) allowing for evaluating along multiple dimensions, (ii) taking the distance between correct and given answer into account and (iii) the scale interval of the measure is used more evenly.

Acknowledgments

This work has been supported by the European projects *Lifecycle Support for Networked Ontologies* (NeOn, IST-2006-027595) and *Semiotic Dynamics in Online Social Communities* (Tagora, FP6-2005-34721).

References

[1] J.-U. Kietz, A. Maedche, and R. Volz, "A method for semi-automatic ontology acquisition from a corporate intranet," in *Proc. of the EKAW'2000 Workshop "Ontologies and Texts"*, 2000.

[2] E. Simperl, C. Tempich, and D. Vrandecic, "A methodology for ontology learning," in *Bridging the Gap between Text and Knowledge – Selected Contributions to Ontology Learning and Population from Text* (P. Buitelaar and P. Cimiano, eds.), IOS Press, 2007. THIS VOLUME.

[3] P. Cimiano, A. Pivk, L. Schmidt-Thieme, and S. Staab, "Learning taxonomic relations from heterogenous sources of evidence," in *Ontology Learning from Text: Methods, Applications, Evaluation* (P. Buitelaar, P. Cimiano, and B. Magnini, eds.), Amsterdam: IOS Press, 2005.

[4] P. Cimiano, A. Hotho, and S. Staab, "Learning concept hierarchies from text corpora using formal concept analysis," *JAIR – Journal of AI Research*, vol. 24, pp. 305–339, 2005.

[5] A. Gangemi, C. Catenacci, M. Ciaramita, and J. Lehmann, "Modelling ontology evaluation and validation," in *Proceedings of the 3rd European Semantic Web Conference (ESWC)*, 2006.
[6] A. Gómez-Pérez, "Ontology evaluation," in *Handbook on Ontologies* (S. Staab and R. Studer, eds.), Heidelberg: Springer Verlag, 2003.
[7] A. Gangemi, C. Catenacci, M. Ciaramita, and J. Lehmann, "Ontology evaluation and validation: An integrated formal model for the quality diagnostic task," tech. rep., ISTC-CNR, 2005.
[8] C. Welty, R. Kalra, and J. Chu-Carrol, "Evaluating ontological analysis," in *Proceedings of the ISWC-03 Workshop on Semantic Integration*, 2003.
[9] R. Porzel and R. Malaka, "A task-based approach for ontology evaluation," in *Proc. of the ECAI Workshop on Ontology Learning and Population*, 2004.
[10] C. Brewster, H. Alani, S. Dasmahapatra, and Y. Wilks, "Data-driven ontology evaluation," in *Proceedings of International Conference on Language Resources and Evaluation*, 2004.
[11] W. Daelemans and M.-L. Reinberger, "Shallow text understanding for ontology content evaluation," *IEEE Intelligent Systems*, vol. 19, no. 4, pp. 74–81, 2004.
[12] P. Spyns and M.-L. Reinberger, "Lexically evaluating ontology triples generated automatically from texts," in *Proc. of the second European Conference on the Semantic Web*, 2005.
[13] P. Spyns, "EvaLexon: Assessing triples mined from texts," Tech. Rep. STAR-2005-09, STAR Lab, 2005.
[14] N. Guarino and C. Welty, "An overview of OntoClean," in *Handbook on Ontologies* (S. Staab and R. Studer, eds.), Heidelberg: Springer Verlag, 2003.
[15] J. Völker, D. Vrandecic, and Y. Sure, "Automatic evaluation of ontologies (aeon)," in *Proceedings of the 4th International Semantic Web Conference (ISWC2005)*, 2005.
[16] M. Berland and E. Charniak, "Finding parts in very large corpora," in *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, 1999.
[17] R. Girju and D. Moldovan, "Text mining for causal relations," in *Proc. of the FLAIRS Conference*, 2002.
[18] M. Sabou, C. Wroe, C. Goble, and H. Stuckenschmidt, "Learning domain ontologies for semantic web service descriptions," *Journal of Web Semantics*, vol. 3, no. 4, 2005.
[19] A. Maedche and S. Staab, "Measuring similarity between ontologies," in *Proc. of the European Conference on Knowledge Acquisition and Management (EKAW-2002)*, 2002.
[20] M. Sabou, C. Wroe, C. Goble, and G. Mishne, "Learning domain ontologies for web service descriptions: an experiment in bioinformatics," in *Proc. of WWW05*, 2005.
[21] M.-L. Reinberger and P. Spyns, "Unsupervised text mining for the learning of dogma-inspired ontologies," in *Ontology Learning from Text: Methods, Applications and Evaluation* (P. Buitelaar, P. Cimiano, and B. Magnini, eds.), IOS Press, 2005.
[22] A. Maedche, *Ontology Learning for the Semantic Web*. Boston: Kluwer, 2002.
[23] E. Alfonseca and S. Manandhar, "Proposal for evaluating ontology refinement methods," in *Proceedings of the Language Resources and Evaluation Conference (LREC-2002)*, 2002.
[24] D. Maynard, W. Peters, and Y. Li, "Metrics for evaluation of ontology-based information extraction," in *Proc. of the EON 2006 Workshop*, 2006.
[25] D. Maynard, Y. Li, and W. Peters, "Using contextual information for term extraction and ontology population," in *Bridging the Gap between Text and Knowledge - Selected Contributions to Ontology Learning and Population from Text* (P. Buitelaar and P. Cimiano, eds.), IOS Press, 2007. THIS VOLUME.
[26] U. Hahn and K. Schnattinger, "Towards text knowledge engineering," in *Proc. of the 15th National Conference on Artificial Intelligence (AAAI-98)*, 1998.
[27] J. Brank, D. Mladenic, and M. Grobelnik, "Gold standard based ontology evaluation using instance assignment," in *Proc. of the EON 2006 Workshop*, 2006.
[28] J. Hartmann, P. Spyns, D. Maynard, R. Cuel, M. Carmen Suarez de Figueroa, and Y. Sure, "Methods for ontology evaluation," Deliverable D1.2.3, Knowledge Web, 2004.
[29] C. van Rijsbergen, *Information Retrieval*. London: Butterworths, 1979.
[30] K. Dellschaft, "Measuring the similiarity of concept hierarchies and its influence on the evaluation of learning procedures," diploma thesis, Universität Koblenz-Landau, December 2005. http://www.uni-koblenz.de/FB4/Institutes/IFI/AGStaab/Theses/2005/DADellschaft.pdf.
[31] M. Hearst, "Automatic aquisition of hyponyms from large text corpora," in *Proc. of the 14th International Conference on Computational Linguistics*, 1992.
[32] S. Staab, C. Braun, I. Bruder, A. Dusterhoft, A. Heuer, M. Klettke, G. Neumann, B. Prager, J. Pretzel, H. Schnurr, R. Studer, H. Uszkoreit, and B. Wrenger, "Getess - searching the web exploiting german texts," in *Proc. of the 3rd Workshop on Cooperative Information Agents*, 1999.