The Knowledge Engineering Review, Vol. 18:4, 293–316. © 2003, Cambridge University Press
DOI: 10.1017/S0269888903000687 Printed in the United Kingdom

The state of the art in ontology learning: a framework for comparison

MEHRNOUSH SHAMSFARD[1] and AHMAD ABDOLLAHZADEH BARFOROUSH[2]

[1] *Intelligent Systems Laboratory, Computer Engineering Dept., Amir Kabir University of Technology, Hafez ave., Tehran, Iran; e-mail: shams@pnu.ac.ir*

[2] *Intelligent Systems Laboratory, Computer Engineering Dept., Amir Kabir University of Technology, Hafez ave., Tehran, Iran; e-mail: ahmad@ce.aku.ac.ir*

Abstract

In recent years there have been some efforts to automate the ontology acquisition and construction process. The proposed systems differ from each other in some factors and have many features in common. This paper presents the state of the art in Ontology Learning (OL) and introduces a framework for classifying and comparing OL systems. The dimensions of the framework concern what to learn, from where to learn it and how it may be learnt. They include features of the input, the methods of learning and knowledge acquisition, the elements learned, the resulting ontology and also the evaluation process. To extract this framework, over 50 OL systems or modules thereof that have been described in recent articles are studied here and seven prominent ones, which illustrate the greatest differences, are selected for analysis according to our framework. In this paper after a brief description of the seven selected systems we describe the dimensions of the framework. Then we place the representative ontology learning systems into our framework. Finally, we describe the differences, strengths and weaknesses of various values for our dimensions in order to present a guideline for researchers to choose the appropriate features to create or use an OL system for their own domain or application.

1 Introduction

Ontologies are means of knowledge-sharing and reuse. They are *semantic containers*. The term *ontology* has various definitions in various texts, domains and applications. In philosophy and linguistics, ontology is the study of existence, a theory of what there is in the world, or a taxonomy of the world-concepts. The most popular definition of ontology in information technology and the AI community from a theoretical point of view is "a formal explicit specification of a shared conceptualization" or "an abstract view of the world we are modeling, describing the concepts and their relationships" (Gruber, 1993).

In practical terms, an ontology may be defined as $O = (C, R, A, Top)$, in which C is the non-empty set of concepts (including relation concepts and the *Top*), R is the set of all assertions in which two or more concepts are related to each other, A is the set of axioms and *Top* is the highest-level concept in the hierarchy. R itself is partitioned to two subsets, H and N. H is the set of all assertions in which the relation is a taxonomic relation and N is the set of all assertions in which the relation is a non-taxonomic relation (Shamsfard & Barforoush, 2002b). There may also be bidirectional functions that relate the members of C and their motivating elements in the real world (for example words in a natural language).

Ontologies are widely used in information systems, and ontology construction has been addressed in several research areas. The major problems in building and using ontologies are the bottleneck of knowledge acquisition and time-consuming construction and integration of various ontologies for various domains and applications. In recent years two approaches have been concerned with solving these problems:

1. The development of methods, methodologies, tools and algorithms to *integrate* existing ontologies. Many disparate source ontologies have been built for various domains or applications. There are different approaches to bringing these sources together and reusing them. The integration process finds commonalities between source ontologies and from them derives a new ontology that facilitates interoperability between computer systems that are based on the source ontologies (Sowa, 2000). The integration may be done in the following ways (adapted from Noy & Musen, 1999):
   a. by merging the ontologies to create a single coherent ontology,
   b. by aligning the ontologies by establishing links between them and allowing them to reuse information from one another and
   c. by mapping the ontologies by finding correspondence elements in each one.

   As an example of a merged ontology we can mention the project of merging the top-most levels of two general commonsense knowledge ontologies – SENSUS[1] (Knight & Luk, 1994) and Cyc (Lenat, 1995) – to create a single top-level ontology of world knowledge (Chapulsky *et al.*, 1997). There are also some research projects working on general methods to merge and align ontologies such as the work by Noy and Musen (2000) introducing PROMPT, an algorithm for semi-automatic merging and alignment of ontologies or Ryutaro *et al.* (2001), proposing a concept alignment method used to induce appropriate align rules for concept hierarchies. There is also some research on ontology mapping such as the proposed approach by Lacher and Groh (2001) using supervised classification.
2. Development of methods, methodologies, tools and algorithms to *acquire* and *learn* ontologies (semi-)automatically.

In this paper we focus on the second approach and introduce a framework for classifying and comparing ontology learning systems. The framework dimensions answer to questions about what to learn, from where to learn and how to learn. They focus on features of the inputs, the elements learned, the built ontology and the methods of learning and knowledge acquisition.

In the following sections we will first give an overview of some existing ontology learning systems and then describe the dimensions of the framework. The references for this study are papers presented in the last three workshops on ontology learning held in the last three years (Staab *et al.*, 2000; 2001; OLT'2002) and other journal and conference papers, technical reports and books recently published on this topic. From these articles, which number over 50, seven prominent ones are selected to be described and compared explicitly in the paper and others are also referred to while discussing our framework. The Conclusion will make the differences, strengths and weaknesses of various values for the dimensions clearer and give a guideline for researchers to choose the appropriate features (the values along dimensions) to build ontologies for their domain or application of interest, or to choose an existing OL system.

2 Ontology learning systems

Ontology learning refers to extracting ontological elements (conceptual knowledge) from input and building an ontology from them. Manual building of ontologies is a costly and time-consuming, tedious and error-prone task and manually built ontologies are expensive, biased towards their developer, inflexible and specific to the purpose that motivated their construction. Automation of ontology construction not only reduces costs, but also results in an ontology that better matches its application.

[1] The SENSUS ontology itself resulted from the manual merging of the PENMAN upper model, WordNet and several other ontologies.

Ontology learning uses methods from a diverse spectrum of fields such as machine learning, knowledge acquisition, natural-language processing, information retrieval, artificial intelligence, reasoning and database management.

During the last decade, several ontology learning approaches and systems have been proposed. Some of them are autonomous ontology learning systems while some others are supporting tools to build ontologies. In this section we will discuss a selection of both. Our criteria for selecting systems for the study are (1) to select from well-developed autonomous systems and supporting tools which are described from end to end in the documents, (2) to select the most recent systems (since 1997), (3) to select systems with the greatest differences, each as a representative of its group, having some distinguishing features and (4) to select well-documented systems to be able to answer the questions we are asking about the dimensions.

For example we chose ASIUM as the representative of the category of systems on learning verb subcategorisations, HASTI for learning axioms besides words, concepts and relations from scratch, TEXT-TO-ONTO for learning from structured, semi-structured and unstructured data using a multi-strategy method and WEB → KB for combining statistical and symbolic methods to learn instances and rules from Web documents. Table 1 shows a summary of the selected systems in alphabetical order with their references. In this table we mention the most distinguishing features of the selected system for which it is selected.

Table 1 Summary of the selected systems

System Name	References	Distinguishing features
Asium	(Faure *et al*., 1998; Faure & Poibeau, 2000)	Learning verb frames and taxonomic knowledge, based on statistical analysis of syntactic parsing of French texts.
DODDLE II	(Yamaguchi, 2001)	Supporting tool to learn taxonomic and non-taxonomic relations using statistical methods (co-occurrence analysis), exploiting a machine-readable dictionary (WordNet) and domain-specific texts.
HASTI	(Shamsfard, 2003; Shamsfard & Barforoush, 2000; 2002a; b)	Learning words, concepts, relations and axioms in both incremental and non-incremental modes, starting from a small kernel (learning from scratch), using a hybrid symbolic approach, a combination of logical, linguistic-based, template-driven and heuristic methods.
SVETLAN'	(Chalendar & Grau, 2000)	Supporting tool to build an ontology by learning noun hierarchies, receiving semantic domains with thematic units, building structured domains to classify nouns according to same relations to same verbs.
SYNDIKATE	(Hahn & Schnattinger, 1998; Hahn & Romacker, 2001; Hahn & Marko, 2002)	Incremental learning of words, concepts and relations, based on text understanding on both sentence level and text level, using the linguistic and conceptual "quality" of various forms of evidence underlying the generation and refinement of concept hypotheses.
TEXT-TO-ONTO	(Maedche & Staab, 2000a; b; 2001)	Learning concepts and relations from unstructured, semi-structured and structured data, using a multi-strategy method comprising a combination of association rules, formal concept analysis and clustering.
WEB → KB	(Craven *et al*., 1998; 2000)	Combining statistical (Bayesian learning) and logical (FOL rule-learning) methods to learn instances and instance extraction rules from World Wide Web documents.

As the table shows, among the selected systems two are supporting tools for learning ontologies and five are autonomous OL systems. The supporting tools (SVETLAN' and DODDLE II) extract essential structures from input to make an ontology learning system able to build an ontology.

For the rest of this paper we will first give an overview of the seven selected systems. Then we will describe the extracted feature set (the framework dimensions) to classify and compare ontology learning systems in the next section. There we will bring some examples from over 50 studied works for each dimension, and finally we will compare the selected systems according to the introduced framework.

ASIUM ASIUM learns subcategorization frames of verbs and ontologies from the syntactic parsing of technical texts in natural language (French). The inputs of ASIUM result from syntactic parsing of texts, they are subcategorization examples and basic clusters formed by head words that occur with the same verb after the same preposition (or with the same syntactical role). ASIUM successively aggregates the clusters to form new concepts in the form of a generality graph that represents the ontology of the domain. Subcategorization frames are learned in parallel, so that as concepts are formed, they fill restrictions of selection in the subcategorization frames. The ASIUM method is based on conceptual clustering. ASIUM proposes a cooperative ML method, which provides the user with a global view of the acquisition task and also with acquisition tools like automatic concept splitting, example generation, and an ontology view with attachments to the verbs. Validation steps using these features are intertwined with learning steps so that the user validates the concepts as they are learned.

DODDLE II DODDLE II is a Domain Ontology Rapid Development Environment. It makes an environment to construct domain ontologies with both taxonomic and non-taxonomic conceptual relationships, exploiting a Machine-Readable Dictionary (MRD) and domain-specific texts. It supports a user in constructing domain ontologies. The taxonomic relationships come from WordNet in the interaction with a domain expert, using match-result analysis and trimmed-result analysis. The non-taxonomic relationships come from domain-specific texts with the analysis of lexical co-occurrence statistics, based on WordSpace to represent lexical items according to how semantically close they are to one another. To evaluate the system, some case studies have been done in the field of law.

HASTI HASTI is an automatic ontology-building system, which builds dynamic ontologies from scratch. HASTI learns the lexical and ontological knowledge from natural-language (Persian) texts. Its lexicon is nearly empty initially and will grow gradually by learning new words. The ontology in HASTI is a small kernel at the beginning. HASTI learns concepts, taxonomic and non-taxonomic conceptual relations, and axioms, to build ontologies upon the existing kernel. The learning approach in HASTI is a hybrid symbolic approach, a combination of linguistic, logical, template-driven and semantic analysis methods. It performs online and offline clustering to organise its ontology.

SYNDIKATE SYNDIKATE is a system for automatically acquiring knowledge from real-world texts (German), and for transferring their content to formal representation structures which constitute a corresponding text knowledge base. It integrates requirements from the analysis of single sentences, as well as those of referentially linked sentences forming cohesive texts. Besides centring-based discourse analysis mechanisms for pronominal, nominal and bridging anaphora, SYNDIKATE is supplied with a learning module for automatically boot-strapping its domain knowledge as text analysis proceeds. The approach to learning new concepts as a result of text understanding builds on two different sources of evidence: the prior knowledge of the domain the texts are about and grammatical constructions in which unknown lexical items occur in the texts. A given ontology is incrementally updated as new concepts are acquired from real-world texts. The acquisition process is centred on the linguistic and conceptual "quality" of various forms of evidence underlying the generation and refinement of concept hypotheses. On the basis of the quality of evidence, concept hypotheses are ranked according to credibility and the most credible ones are selected for assimilation into the domain knowledge base.

SVETLAN' SVETLAN' is a system for classifying nouns in context. It is able to learn categories of nouns from texts, whatever their domain is. Words are learned considering the contextual use of them to avoid mixing their meanings. SVETLAN' is a supporting tool. Its input data are semantic domains with the Thematic Units (TUs) automatically learned by Segapsith (Ferret & Grau, 1998) and the output is the learned structured domain containing the noun classifications with their relations to verbs. It is based on a distributional approach: nouns playing the same syntactic role with a verb in sentences related to the same topic, i.e. the same domain, are aggregated in the same class.

TEXT-TO-ONTO TEXT-TO-ONTO is an ontology learning environment, based on a general architecture for discovering conceptual structures and engineering ontologies from text. It supports as well the acquisition of conceptual structures such as mapping linguistic resources to the acquired structures. It creates an environment for discovering conceptual relations that in turn make it possible to build ontologies. The new version of the system, which supports learning ontologies from web documents, allows the import of semi-structured and structured data as input as well as texts. It also has a library of learning methods that may be used on demand. Its learning method is a multi-strategy approach combining various methods for various inputs and tasks.

WEB→KB The goal of this research is to automatically create a computer-understandable worldwide knowledge base whose content mirrors that of the World Wide Web. Its approach is to develop a trainable information extraction system that takes two inputs: (1) a knowledge base consisting of an ontology defining the classes (e.g. person) and relations (e.g. instructor-of) of interest and, optionally, instances of some of these classes and relations; and (2) training examples from the Web that describe instances of these classes and relations. Given these inputs, the system determines general procedures capable of extracting additional instances of these classes and relations by browsing the rest of the Web. The outputs would be the classified instances and rules to extract new instances, rules to classify pages and rules to recognise relations among several pages. WEB→KB uses logical and statistical learning algorithms for these tasks.

3 The dimensions of the framework

During the last decade there have been some efforts to automate ontology construction. Several ontology learning approaches and systems are proposed, which differ from each other in some dimensions. The major distinguishing factors between ontology learning systems are classified in six main categories (dimensions) and some subcategories. For each dimension some features are selected and for each feature there are some references to systems having them.

The framework dimensions are as follows:

1. elements learned (concepts, relations, axioms, rules, instances, syntactic categories and thematic roles);
2. starting point (prior knowledge and the type and language of input);
3. preprocessing (linguistic processing such as deep understanding or shallow text processing);
4. learning method, consisting of
   a. learning category (supervised vs. unsupervised, online vs. offline),
   b. learning approach (statistical vs. symbolic, logical, linguistic-based, pattern matching, template-driven and hybrid methods),
   c. learning task (classification, clustering, rule-learning, concept formation, ontology population),
   d. degree of automation (manual, semi-automatic, cooperative, fully automatic),
      i) Type and amount of user intervention;
5. the result (ontology vs. intermediate structures and in the first case the features of the built ontology such as coverage degree, usage or purpose, content type, structure and topology and representation language); and
6. evaluation methods (evaluating the learning methods or evaluating the resulted ontology)

Figure 1 shows the framework dimensions, their sub-dimensions and some of their possible values. Dimensions and sub-dimensions are shown in boxes and values in ellipses.

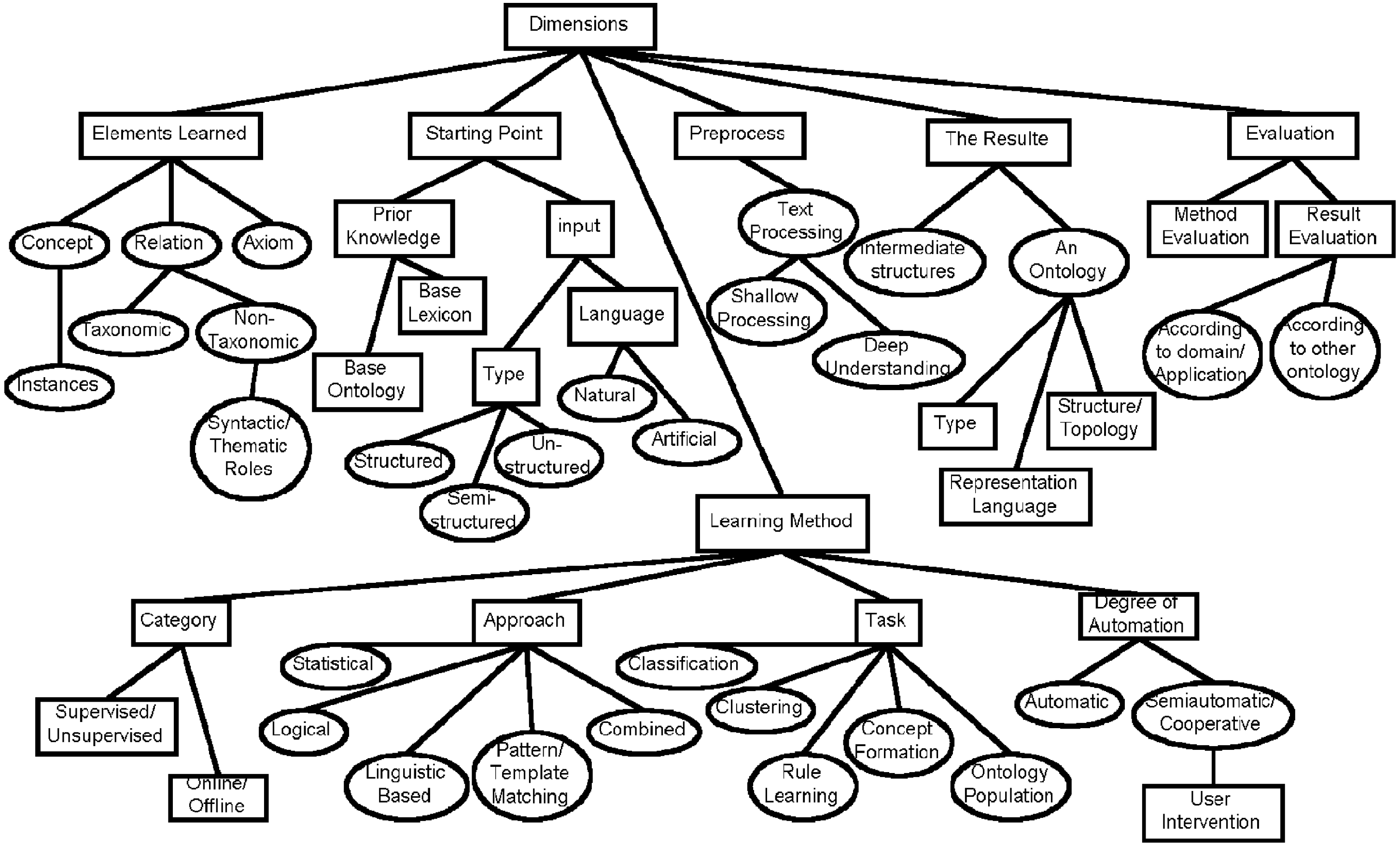

Figure 1 The framework dimensions, sub-dimensions and values

For the rest of this section, the framework dimensions will be described in more detail. Each subsection will describe one of the dimensions. In each subsection, sub-dimensions will be numbered and the values for each will be bulleted.

3.1 The elements learned

In this part we answer the question "what type of conceptual structure is acquired?" The elements that are learnt can be merely ontological knowledge or both lexical and ontological knowledge. The main lexical elements, which the various systems learn, are words and the main ontological elements are concepts, relations and axioms. There are also some systems that learn meta-knowledge about how to extract ontological knowledge from input.

- **Words** Although most ontology learning systems (to mention a few, TEXT-TO-ONTO, DODDLE II; Kietz *et al*., 2000; Borgo *et al*., 1997) use pre-defined lexicons, some of them (Thompson & Mooney, 1999; HASTI, SYNDIKATE) learn lexical knowledge about words too. The method of handling unknown words and the type of lexical knowledge to be learned about words are different in different OL systems. For instance, SYNDIKATE predicts word class for unknown lexical items given a hierarchy that covers all relevant word classes for a particular natural language. So once a word class is hypothesized, grammatical information associated with this word class (such as valency frames, word-order constraints or morpho-syntactic features) comes for free due to the organisation of the grammar as a lexical class hierarchy. HASTI learns the morphological features and syntactic category of unknown words besides their meanings and Wolfie (WOrd Learning From Interpreted Examples, Thompson & Mooney, 1999) acquires a semantic lexicon from a corpus of sentences paired with semantic representations. The lexicon learned consists of words paired with meaning representations.
- **Concepts** A concept can be anything about which something is said and can be abstract or concrete, elementary or composite, real or fictitious, or the description of a task, function, action, strategy, reasoning process etc. (Corcho & Gomez-Perez, 2000). Concepts are represented by nodes in the ontology graphs and may be learned by the ontology learning system (such as HASTI, SYNDIKATE; Roux *et al*., 2000; Soderland *et al*., 1995). They may be extracted from input or be created during the ontology refinement from other concepts. In other words they may or may not have corresponding elements in the input. In *terminological* (or term-based) acquisition of concepts, a concept node will be created corresponding to the extracted term which may be natural-language words or phrases, while in *conceptual* (or semantic-based) concept creation, which is usually done in the refinement phase, the concept will be built according to its features (attributes/values), its functionality and so on and hence may have no corresponding input (no corresponding word or phrase in the input text).
    - **Instances** Some ontology learning systems use an existing ontology and just populate it by instances of classes and relations, like WEB → KB, Suryanto and Compton (2000). Most of these systems do not learn new concepts (classes) and just learn instances of existing classes.
- **Conceptual Relations** Relations may be studied in two ways:
    1. A relation is a node in the ontology, so it is a concept and may be learned like other concepts.
    2. A relation relates two or more concepts and so it should be learned as a subset of a product of n concepts (for $n > 1$).

    In other words relations may be learned *intentionally* independent of what concepts they relate to or *extensionally* by considering the concepts which are being related to each other by it. The first case will be counted in the first type of learned elements (concepts) and the second case in the second type (conceptual relations).

    For example the binary relation *part-of* is a concept under the super-concept "relation" and has its own features (such as transitivity) and may be related to other concepts by other relations too. On the other hand it relates the concepts "hand" and "human" or "door" and "house" which may be shown as (part-of hand human) or (part-of door house) which is its second aspect. In some systems both aspects of relations may be learned (Hasti), while in some others the relations themselves (the first aspect) are predefined and their occurrences (the second aspect) will be learned using some templates (Borgo *et al.*, 1997). In this subsection we mention learning of the extensional definition of relations.

Relations may be *intra-ontology* or *inter-ontology.* Intra-ontology relations are those relating an ontology's elements (structures) to each other while the inter-ontology relations relate the structures in two or more ontologies. Most works on learning relations are from the first kind. But there is some research on learning the relations between ontologies too, such as the proposed work by Williams and Tsatsoulis (2000) on acquiring related concepts within diverse ontologies using an instance-based approach.

Conceptual relations may be taxonomic or non-taxonomic relations.

- ◦ **Taxonomic relations** Taxonomies are widely used to organise ontological knowledge using generalisation/specialisation relationships through which simple/multiple inheritance can be applied (Corcho & Gomez-Perez, 2000). Although some references (Sowa, 2000) refer to hyponymy and meronymy relations as taxonomic relations, most taxonomic relation learning systems (some of them are mentioned below) just learn the hyponymy relations (the ISA hierarchy of concepts). Thus in this part by taxonomic relations we mean the hyponymy relations and leave the meronymy relations to the next part (non-taxonomic relations).

  Taxonomic knowledge is learned by some ontology learning systems. Some, among others, are SynDiKATe, Hasti, Doddle II; Todirascu *et al.* (2000); Agirre *et al.* (2000); Suryanto and Compton (2000); Heyer *et al.* (2001); Caraballo (1999); Delteil *et al.* (2001); Sundblad (2002) and Sporleder (2002). A survey on existing works on learning taxonomic relations from texts is given in Maedche *et al.* (2002).
- ◦ **Non-taxonomic relations** Non-taxonomic conceptual relations refer to any relation between concepts except the ISA relations, such as synonymy, meronymy, antonymy, attribute-of, possession, causality and other relations (learned by systems such as Hasti, Text-to-Onto; Agirre *et al.*, 2000; Gamallo *et al.*, 2002), knowledge about specific words' syntactic categories and thematic roles such as learning subjects and objects of verbs (Pereira *et al.*, 1993), discovering verb frames (Asium; Wagner, 2000), inferring verb meanings (Wiemer-Hastings *et al.*, 1998), classifying adjectives (Assadi, 1997) and nouns (SVETLAN') and identifying names (Bikel *et al.*, 1999).

- • **Axioms** Axioms are used to model sentences that are always true. They can be included in an ontology for several purposes, such as constraining the information contained in the ontology, verifying its correctness or deducing new information (Farquhar *et al.*, 1996).

  Learning axioms (semi-)automatically is an open problem. Hasti is one system that learns axioms in restricted situations. It translates explicit axioms in conditional and quantified natural-language sentences into logically formatted axioms in KIF. There is also ongoing work to extend Hasti to learn implicit axioms from text.
- • **Meta-knowledge** Besides systems that learn ontological knowledge, there are systems that learn how to learn and extract ontological knowledge. They learn meta-knowledge such as rules for extracting instances, relations and specific fields from the Web (WEB→KB) or patterns for extracting knowledge from text (Finkelstein-Landau & Morin, 1999; Soderland *et al.*, 1995) or association rules in a corpus (Cherfi & Toussaint, 2002).

3.2 Starting point

This dimension is concerned with the answer to the question "from where should ontology acquisition start, and from what should it be learnt?" Ontology learning systems use their background knowledge (prior knowledge) and acquire new knowledge elements (or update the existing ones) from their input. The quality and quantity of the prior knowledge and the type, structure and language of the input – from which the system learns ontological knowledge – differ from one system to another.

3.2.1 Background or prior knowledge (base ontology)

Essential background knowledge varies in both type and volume in different projects. The background knowledge may be presented in linguistic (lexical, grammatical, templates etc.) or ontological (base ontology) resources. In many projects there is a predefined lexicon used to process texts (such as Kietz *et al.*, 2000). In some of these projects the lexicon is a semantic lexicon covering ontological knowledge too (such as using (Euro) WordNet in Text-to-Onto, SynDiKATe, Doddle II; Wagner, 2000; Agirre *et al.*, 2000; Termier *et al.*, 2001). The base ontology's size and coverage is another distinguishing factor varying from almost empty (a small kernel of primitives) as in Hasti, a seed ontology sketched by the user as in Brewster *et al.* (2001) and a small number of seed words that represent the high-level concepts as in InfoSleuth (Hwang, 1999), to huge general common-sense ontologies such as Cyc (Lenat & Guha, 1990).

3.2.2 Input

Ontology learning systems extract their knowledge of interest from their input. Input sources differ by type and language.

a) *Type* The type of input from which the system acquires ontological knowledge may be of the following:
   - Structured data. Ontology learning systems may extract ontological knowledge from structured data such as database schemata (Kashyap, 1999), existing ontologies (Williams & Tsatsoulis, 2000), knowledge bases (Suryanto & Compton, 2000) and lexical semantic nets such as WordNet.
   - Semi-structured data. Other sources for ontology learning systems (such as Pernelle *et al.*, 2001) are semi-structured data such as dictionaries, HTML and XML docs and DTDs (Document-Type Definitions). Growing interest in the semantic web leads to increasing interest in building ontologies for the Web. So learning ontologies from semi-structured data in Web documents is a hot topic today. Text-to-Onto, WEB → KB and Kavalec and Svatek (2002) are instances of such systems.
   - Unstructured data. The most difficult type of input to extract knowledge from is the unstructured. Tools that learn ontologies from natural language exploit the interacting constraints on the various language levels (from morphology to pragmatics and background knowledge) in order to discover new concepts and stipulate relationships between concepts (OLT'2002). The unstructured input of OL systems may be natural-language texts (Hasti, SVETLAN', SynDiKATe; Heyer *et al.*, 2001) or Web texts (as in Text-to-Onto; Todirascu *et al.*, 2000 and other systems studied in Omelayenko, 2001).

b) *Language* The input may be texts of natural languages such as English (Doddle II; Wagner, 2000; Termier *et al.*, 2001), German (SynDiKATe, Text-to-Onto), French (Todirascu *et al.*, 2000; Asium; SVETLAN'), Persian (Hasti) and so on or data presented in artificial languages such as XML (Text-to-Onto) or RDF (Delteil *et al.*, 2001).

3.3 Preprocessing

This dimension answers the question, "is there any preprocessing to convert the input to a suitable structure to learn from?" The most popular preprocessing used in learning ontologies from texts is linguistic preprocessing. Deep understanding would provide specific relations among concepts, whereas shallow techniques could provide generic knowledge about the concepts (Agirre *et al.*, 2000). As deep understanding usually decreases the speed of the ontology construction process, most existing systems use shallow text-processing techniques such as tokenising, Part-Of-Speech (POS) tagging, syntactic analysis and so on to extract their essential structures from input texts. For example, Text-to-Onto uses shallow text-processing methods developed at SMES (Saarbrucken Message Extraction System) (Neumann *et al.*, 1997) to process German texts and identify linguistically related pairs of words, which are mapped to concepts using the domain lexicon. InfoSleuth (Hwang, 1999) uses a simple POS tagger to perform superficial syntactic analysis and ASIUM uses Sylex (Constant, 1996) to process French texts. SYNDIKATE uses deep understanding to extract ontological knowledge from text and HASTI exploits Petex (Shamsfard & Barforoush, 2002c), a Persian text-processing system, to extract sentence structures, which indicate thematic roles, from text.

There are also other preprocessing modules to extract special structures from input to allow the learning modules to learn ontological elements such as SVETLAN', extracting noun categories, and Moigno *et al.* (2002), extracting terminology, to help a domain expert build an ontology in the surgical intensive-care field.

3.4 Learning method

With this dimension we answer the question "what kinds of method are used to extract knowledge?" Knowledge extraction methods range from knowledge-poor approaches (statistical techniques) to knowledge-intensive approaches (logical techniques). In this range we may mention different approaches and techniques to extract ontological knowledge and learn ontologies. These methods may be supervised or unsupervised and run online or offline.

3.4.1 Learning approaches

The ontology learning approach may be statistical or symbolic. From symbolic approaches we mention the logical, linguistic-based and template-driven approaches. Heuristic methods may be used to facilitate each approach. There are also hybrid approaches, which combine two or more of the above approaches and employ their benefits and eliminate their limitations.

- **Statistical** In this approach, statistical analysis is performed on data gathered from the input. For instance WEB→KB uses a statistical bag-of-words approach to classify Web pages; Wagner (2000) exploits a modification of the algorithm by Li & Abe (1996) for acquisition of selectional preferences and locating concepts in the ontology at the appropriate generalisation level; TEXT-TO-ONTO, Heyer *et al.* (2001) and DODDLE II use statistical analysis of co-occurrence data to learn conceptual relations from texts; Bikel *et al.* (1999) uses the Hidden Markov Model (HMM) to find and label names and other numerical entities; and Cherfi & Toussaint (2002) uses statistical indices to rank the association rules that are more capable of reflecting the complex semantic relations between terms. Statistical methods may work on isolated words or batches of words together. They differ in size of batches, distribution function and statistical analysis done on input data.

  Models based on isolated words are often called *unigram* or *bag-of-words* models. They ignore the sequence in which the words occur. Since the unigram model naively assumes that the presence of each word in a document is conditionally independent of all other words in the document given its class, this approach, when used with Bayes's rule, is often called *naive Bayes* (Craven *et al.*, 2000). WEB→KB is a system that classifies Web documents using a modification of the naive Bayes method. It builds a probabilistic model of each class using labelled training data, and then classifies newly seen pages by selecting the class that is most probable given the evidence of words describing the new page. Its method for classifying Web pages is naive Bayes, with minor modifications based on Kullback-Leibler divergence.

  Other statistical methods often consider batches of words. The main idea common to these approaches is that the semantic identity of a word is reflected in its distribution over different contexts, so that the meaning of a word is represented in terms of words co-occurring with it and the frequencies of the co-occurrences (Maedche *et al.*, 2002). The occurrence of two or more words within a well-defined unit of information (sentence, document) is called a collocation (Heyer *et al.*, 2001). Learning by collocations and co-occurrences is the most addressed method in statistical learning of ontological knowledge. In learning by collocation and co-occurrence, first, a collocation structure (e.g. matrix) will be created. Then, using statistical analysis of this structure, the conceptual relations between concepts will be discovered. For instance, in Heyer *et al.* (2001) a kind of *average context* for every word *A* is formed by all collocations for *A* with a significance above a certain threshold. This average context of *A* is transferred into a feature vector of *A* using all words as features as usual. The feature vector of word *A* is indeed a description of the meaning of *A*, because the most important words of the contexts of *A* are included. Clustering of feature vectors can be used to investigate the relations between groups of similar words and to figure out whether or not all the relations are of the same kind. Cohyponymy, top-level syntactic relations (such as semantic 'actor-verb'), often-used properties of a noun and instance-of are some relations discovered by this approach.

Another project using collocation analysis is DODDLE II. In this project semantic similarity is calculated by the inner product of (cosine of the angle between) two word vectors in the wordspace, which contains the vectors representing all concepts. After finding similarities, taxonomic and non-taxonomic relations will be extracted using concept specification templates. To build the wordspace, the collocation matrix should be built from 4-grams extracted from the corpus. Using the collocation matrix, the context vectors and the word vectors will be built.

As another example, TEXT-TO-ONTO uses the frequency of word co-occurrences in discovering non-taxonomic relations using background knowledge (a lexicon and a taxonomy) from linguistic processing of text. An algorithm for discovering generalised association rules analyses statistical information and derives correlations at the conceptual level. Thereby it uses the background knowledge from the taxonomy in order to propose relations at the appropriate level of abstraction.

- **Logical** Logical methods such as Inductive Logic Programming (ILP) (Zelle & Mooney, 1993), FOL (First Order Logic)-based clustering (Bisson, 1992), FOL rule-learning (WEB→KB) and propositional learning (Bowers *et al*., 2000) are also used to extract ontological knowledge from input. Logic-based learning methods may discover new knowledge by deduction or induction and represent knowledge by propositions, first-order or higher-order logics. Deduction-based learning systems (such as HASTI) exploit logical deduction and inference rules such as resolution to deduce new knowledge from existing knowledge while induction-based learning systems (such as WEB→KB; Bowers *et al*., 2000) induce hypotheses from observations (examples) and synthesize new knowledge from experience. ILP lies in the intersection of inductive learning and logic programming in which the hypotheses and observations are represented in first-order logic or variants of it (Muggleton & De Raedt, 1994). FOIL (Quinlen & Cameron-Jones, 1993) is one of the best-known and most successful empirical ILP systems and some ontology learning systems such as WEB→KB use variants of it. FOIL is a greedy covering algorithm that induces concept definitions represented as function-free Horn clauses, optionally containing negated body literals. It induces each Horn clause by beginning with an empty tail and using a hill-climbing search to add literals to the tail until the clause covers only (mostly) positive instances.

WEB→KB uses first-order learning algorithms to learn (1) rules for classifying pages, (2) rules to recognise relations among several pages and (3) rules to extract specific text fields within a Web page. Its learning algorithm to classify pages is Quinlan's FOIL algorithm. The learning algorithm to induce relation rules is also similar to FOIL in that it uses a greedy covering approach to learn a set of Horn clauses, but it differs from FOIL in that it merges the hill-climbing search in FOIL with a variant of the *relational pathfinding* method of Richards and Mooney (1992) to be able to learn rules for paths consisting of more than one hyperlink. It also uses a different evaluation function (using m-estimates of a clause's error) to guide the search process. WEB→KB uses the SRV (Sequence Rules with Validation) algorithm to learn rules to extract specific text fields. The SRV algorithm is a first-order learner in the spirit of FOIL. It shares FOIL's top-down approach and gain metric, but is designed with the information extraction problem in mind. Consequently, it is limited to a few pre-defined predicates, and it encompasses search heuristics specific to the information extraction problem. Input to SRV is a set of pages, labelled to identify instances of the field we want to extract, and a set of features defined over tokens. Output is a set of information extraction rules. The extraction process involves examining every possible text fragment of appropriate size to see whether it matches any of the rules.

Other logic-based work is done by Bowers *et al.* (2000), in which a decision-tree learning algorithm is used to learn predicates represented in typed higher-order logic.

- **Linguistic** Linguistic approaches such as syntactic analysis (Asium), morpho-syntactic analysis (Assadi, 1997), lexico-syntactic pattern-parsing (Finkelstein-Landau & Morin, 1999), semantic processing (Hasti) and text understanding (SynDiKATe) are used to extract ontological knowledge from natural-language texts. They are mostly language-dependent and usually perform the preprocessing on the input text to extract essential knowledge to build ontologies from texts. For instance Assadi (1997) performs a partial morpho-syntactic analysis to extract "candidate terms" from technical texts. Then, using these candidate terms, the knowledge engineer, assisted by an automatic clustering tool, builds the conceptual fields of the domain. The result of the morpho-syntactic analysis would be a network of noun phrases which are likely to be terminological units. Any complex term is recursively broken up into two parts, head and expansion, which are both linked to the complex candidate term in a terminological network. The network will then be used by the conceptual analyser to build a classification tree.

  Asium uses syntactic analysis to extract syntactic frames from texts. It only uses head nouns of complements and links with verbs. Adjectives and empty nouns are not used. The learning method relies on the observation of syntactic regularities in the context of words. It does conceptual clustering based on head nouns occurring with the same couple: verb + preposition/syntactic role.

  Hasti exploits both morpho-syntactic and semantic analysis on input texts to extract lexical and ontological knowledge. The morpho-syntactic analysis predicts the features of unknown words and creates sentence structures, which indicate the thematic roles in the sentence. The semantic analysis completes the empty or ambiguous slots of the sentence structures and conducts the process of extracting conceptual knowledge from them using semantic templates.

  SynDiKATe uses text-understanding techniques to acquire knowledge from real-world texts. It integrates requirements from the analysis of single sentences, as well as those of referentially linked sentences forming cohesive texts. The result of its syntactic analysis is captured in a dependency graph in which nodes are words and edges are dependency relations such as specifier, subject, dir-object etc. Then semantic interpretation is performed to find conceptual relations in the knowledge base between conceptual correlates of words. SynDiKATe learns by quality, which means that linguistic and conceptual quality labels are assigned to generated hypotheses and then the higher-ranked hypothesis will be chosen. Linguistic quality labels are Apposition, Exemplification, Case-Frame-Assignment, PP-Attachment, Genitive-Attribution, and they have a higher score than conceptual ones.

  Another linguistic method is lexico-syntactic pattern parsing. In this method the text is scanned for pre-defined lexico-syntactic patterns that indicate a relation of interest, e.g. the taxonomic relation (Maedche *et al.*, 2002). We will discuss these patterns in more detail in the next section.
- **Pattern based/template driven** Keyword/pattern/template-matching approaches are widely used in the information extraction field and are also inherited by the ontology learning domain. In template-driven methods the input (usually the text) will be searched for pre-defined keywords, templates or patterns that indicate some relations (e.g. hyponymy). There are various types of template – syntactic or semantic, and general or special purpose – to extract various ontology elements. As the primary work on pattern-matching we can mention that of Hearst (1992). In his paper, he introduced some lexico-syntactic patterns in the form of regular expressions to extract hyponymy/hyperonymy relations from texts. As examples of these patterns we may mention the following:

$$NPsuchas\ \{NP,\}'(and \mid or)NP$$

$$NP\ \{, NP\}^*\{,\}(or \mid and)otherNP$$

$$NP\ \{,\}including\ \{NP,\}'\{or \mid and\}NP$$

Hasti is another system that uses lexico-syntactic and also semantic patterns (templates) to extract taxonomic and non-taxonomic relations such as hyponymy, meronymy, thematic roles, attribute-values ('has-prop' relation) and other relations and also axioms from texts. An example of its lexico-syntactic pattern is the e*xception template* to extract hyponymies:

$$\{all \mid every\}\ NP_0\ except\ NP_1\ \{(and \mid ,)\ NPi\}^* \ldots (i > 1),\ implies\ (sub\text{-}class\ NP_i\ NP_0)\ (i \geq 1)$$

and an example of its semantic template is the one for modal (copular) sentences with an adjective phrase as the predicate to extract attribute-value ("has-prop") relations:

(⇒ (*and* (*isa* <*subject.head*> *Property*) (*isa* <*adjective*> <*subject.head*>))

(*has-prop* <*subject.modifier.head*> <*subject.head*> <*adjective*>))

or in cases where the predicate is a noun phrase, a template to extract the equality relation is

(*equal* <*subject.head*> <*predicate.head*>)

Symbolic interpretation rules in Gamallo *et al.* (2002) are somehow close to semantic templates in Hasti. They use grammatical patterns to map syntactic dependencies onto semantic relations such as hyperonymy, possession, location, modality, causality, agentivity and so on. A dependency is represented as the binary relation (r: $w1^{\downarrow}$, $w2^{\uparrow}$) where r can be instantiated by specific grammatical markers such as particular prepositions, subject relations, direct object relations etc.; arrows "$^{\downarrow}$" and "$^{\uparrow}$" represent the *head* and *complement* position respectively; w_1 is the word in the head position and w_2 the word in the complement position. The grammatical patterns (markers) indicate syntactic relators (subject, direct object, preposition), morpho-syntactic categories of the two related words (verb and noun), and presence or absence of determinant in the complement. For example, an interpretation rule is following:

$$x = possessed;\ y = possessor \Rightarrow [\lambda \mathrm{x}^{\downarrow} \lambda \mathrm{y}^{\uparrow}(de\ +;\ x^{\downarrow}, y^{\uparrow})]\ or\ [\lambda x^{\downarrow} \lambda y^{\uparrow}(a\ +;\ x^{\downarrow}, y^{\uparrow})]\ or\ [\lambda x^{\downarrow} \lambda y^{\uparrow}(para;\ x^{\downarrow}, y^{\uparrow})]$$

in which for instance the grammatical pattern *de*+ indicates (1) preposition is *de*; (2) the determiner is present before the complement; (3) the head and the complement are both nouns. By means of this rule, the heads (expressed by the variable *x*) of the patterns *de* +, *a*+ and *para* are mapped onto the semantic role "possessed", whereas the complements (expressed by *y*) are mapped onto the role "possession".

Other relevant work is that of Sundblad (2002), in which some linguistic patterns are used to extract hyponymy and meronymy relations from question corpora such as:

Who is/was X?

What is the location of X?

What is/was the X of Y?

How many X are in/on Y?

Heyer *et al.* (2001) proposed two patterns to extract first names and instance-of relations from sentences:

a) **Extraction of first names** A pattern like *(profession) ? (last name)* implies (with high probability) that the unknown category *?* is in fact a *first name* (e.g. actress *Julia* Roberts).
b) **Extraction of *instance-of*-relations given the class name** The pattern *(class name) like ?* implies (with high probability) that the unknown category *?* is in fact an *instance name* (e.g. metals like nickel, arsenic and lead).

The patterns may be general and application-/domain-neutral such as those proposed by Hearst, Hasti and Sundblad or specific to a domain or application such as those used by Assadi (1999) to extract knowledge from electric network planning texts.

On the other hand patterns may be manually defined (Hasti; Sundblad, 2002; Gamallo *et al.*, 2002) or may be extracted (semi-)automatically such as in PROMETHEE (Finkelstein-Landau & Morin, 1999), AutoSlog-TG (Riloff, 1996) and Crystal (Soderland *et al.*, 1995).

- **Heuristic-driven (ad hoc) methods** Heuristics may be used besides any of the other approaches. In other words heuristic-driven methods are not independent and complete methods, rather they should be used to support other approaches. To name a few examples of this approach we may mention Text-to-Onto, Hasti, InfoSleuth; Hwang (1999) and Gamallo *et al.* (2002).

Text-to-Onto uses heuristic rules to increase the recall of the linguistic dependency relations (even for loss of linguistic precision) such as the NP-PP heuristic (which attaches all prepositional phrases to adjacent noun phrases), the sentence heuristic (which relates all concepts contained in one sentence if other criteria fail) and the title heuristic (which links the concepts in the HTML title tags with all the concepts contained in the overall document).

Hasti uses some simplifying heuristics to decrease the size of the hypothesis space, such as the priority-assignment heuristics to assign priorities to ambiguous terms, and candidate-choosing heuristics to choose a merge-set among others in the ontology refinement task (offline clustering).

InfoSleuth uses some heuristic rules to locate new concepts in an appropriate place in the ontology. The heuristic says that (for specific terms) the concept corresponding to a noun phrase should be located under the concept of its head.

Gamallo *et al.* (2002) employs heuristics to select candidate dependencies. They use simple heuristics based on right-association in order to attach basic chunks – a chunk tends to be attached to another chunk immediately to its right. They consider that the word heads of two attached chunks form a candidate syntactic dependency.

- **Multi-strategy learning** Most systems that learn more than one type of ontology element use combined approaches. They apply multi-strategy learning to learn different components of the ontology using different learning algorithms such as Text-to-Onto, which uses association rules, formal concept analysis and clustering techniques; WEB → KB, which combines FOL rule-learning with Bayesian learning; Hasti, which applies a combination of logical, linguistic-based, template-driven and heuristic methods; and Termier *et al.* (2001), which combines statistics and semantics for word- and document-clustering.

3.4.2 Learning task

Learning methods may be categorised based on the task they do. In this category, classification (Suryanto & Compton, 2000; Bowers *et al.*, 2000), clustering (Hasti, Asium), rule learning (WEB → KB; Soderland *et al.*, 1995), formal concept analysis (Text-to-Onto, Richards & Compton, 1997), and ontology population (instance assignment) (WEB → KB, Brewster *et al.*, 2001) are all learning tasks (for each two examples are named), which may be carried out in each of the above approaches (statistical, logical, linguistic-based or template-driven). For example, the Asium system does conceptual clustering using a syntactic parsing approach, while Wagner (2000) uses a statistical approach to clustering and Termier *et al.* (2001) combines statistics and semantics to cluster words and documents.

The learning task may be used to extract knowledge from input or to refine an ontology. Below we will discuss clustering, one of the tasks most commonly carried out in ontology learning, in more detail.

- **Conceptual clustering** Various clustering methods (some are described in Bisson *et al.*, 2000) are distinguished by four factors (adopted from Maedche *et al.*, 2002): clustering mode, clustering direction, similarity measure and computation strategy.

  Clustering mode

  Online vs. offline Clustering may be done in online or offline modes. The online mode does incremental clustering while the offline mode does periodic clustering.

  Hierarchical vs. non-hierarchical In hierarchical clustering the clusters that are built have hierarchical relations with one another while in non-hierarchical clustering they have non-hierarchical relations with one another.

  Single vs. multi-clustering In multi-clustering each concept may be clustered into more than one cluster or, in other words, in the directed graph constructed by clustering, each node may have many parents and/or many children.

  Clustering Direction The hierarchical clustering may be done in any of three directions:

  Top-down In the top-down direction all concepts are first collected in a single cluster. Then the cluster will be split and most similar concepts made into sub-clusters. In other words the hierarchy will be built by incremental specification from top to bottom.

Bottom-up In the bottom-up direction there are first N clusters for N concepts (there is one concept in each cluster). Then the most similar clusters will be merged to make super-clusters. In other words the hierarchy will be built by incremental generalisation from bottom to top.

Middle-out A combination of top-down and bottom-up methods.

Similarity measure In clustering algorithms there is a similarity measure indicating the similarity between two classes. In the literature, two main different types of similarity have been addressed (Eagles, 1996):

"semantic similarity" (so-called paradigmatic or substitutional similarity)

"semantic relatedness" (so called syntagmatic similarity)

Two words can be considered to be semantically similar if they may be substituted for one another in a particular context, and can be considered to be semantically related if they typically or significantly co-occur within the same context.

Semantic similarity may be taxonomy-based (path-length similarity) or distributionally based. In taxonomy-based semantic similarity a shorter path (in the hierarchy) between two concepts means more similar concepts. In distributionally based semantic similarity, less distance between distributions of two concepts means more similar concepts. The formal characterisations of distributions (in a window, in a neighbourhood, in a sentence or in a linguistic-based relation such as verb-object) are different in different methods. There are also different distance metrics to compute the distance of two distributions (for more details see Eagles, 1996).

Semantic relatedness also uses co-occurrence patterns to find similarities. It is somehow similar to distributionally based methods but in semantic relatedness two concepts are related if they occur in the same context which means that they co-occur with each other frequently while in distributional similarity two concepts are similar if their contexts (distributions) are close to each other which means that their co-occurrent words are the same. For example "cut" and "knife" are semantically related as they occur frequently together in a context while "knife" and "scissors" are semantically similar as they occur in the same contexts (e.g. both are instruments for cutting).

Computing Strategy To compute the similarity between two clusters we may use the *single link* strategy, in which the similarity between two clusters is the similarity of the two closest objects in them; *complete link* strategy, in which the similarity of two clusters is the similarity of their two most dissimilar members; or the *group average* similarity, in which the similarity is the average similarity between members.

3.4.3 Degree of automation

The knowledge acquisition phase may be carried out manually, fully automatically or somewhere on a scale between the two. As this paper concerns ontology learning systems, we ignore systems with manual knowledge acquisition. Others use automatic (HASTI, Wagner, 2000), semi-automatic (TEXT-TO-ONTO, Todirascu *et al.*, 2000) and cooperative (HASTI, ASIUM) acquisition tools or methods. In semi-automatic and cooperative systems, the role of the user varies across a wide range. She may propose an initial ontology, validate or change different versions proposed by the system (Brewster *et al.*, 2001), select patterns in the class relations (Suryanto & Compton, 2000), control the generality levels, handle noise and label new concepts (ASIUM), or label concepts, determine weights, validate the artificial sentences built by the system, and confirm the system's decisions (HASTI).

3.5 The result

This dimension is concerned with the result of the learning process and answers the question "what would be built and what would be its features?" The first step to answering this question is to distinguish ontology learning from support for building ontologies. Most of the systems studied here learn ontologies (ontological structures), but some of them just support users, experts or other systems in learning ontologies. In other words some systems are autonomous ontology learning systems while others are modules that perform a task and result in a set of intermediate data that will be used to build the ontology. In these latter kinds of system (such as DODDLE II, SVETLAN', Moigno *et al.*, 2002), initial structures to build ontologies are acquired. For these systems the resulted ontology would not be built unless by the user or other systems. For example, Moigno *et al.* (2002) propose utilising NLP, corpus analysis and distributional analysis tools to help an expert build an ontology in surgical intensive care.

For autonomous systems, which build ontologies, we consider their ontology features in this section. Among several features one can encounter for an ontology, we chose the most distinguishing ones between different OL systems.

3.5.1 Ontology type

Ontology type is made up of several features indicating the nature of the intended ontologies such as coverage degree, usage or purpose, content type, detail level etc. Coverage degree denotes that the ontology is general (such as Cyc by Lenat & Guha, 1990) or special-purpose and domain-specific (such as DODDLE II). The usage and purpose of the ontology shows the potential applications in which it may be used and also the specific domain which it is designed for. For example, the ontology in Moigno *et al.* (2002) is for the surgical intensive care domain while that in Kietz *et al.* (2000) is for the insurance domain and Faure and Poibeau (2000) describe applying ASIUM in the terrorism domain. The applications in which these ontologies are used are different too. The most addressed application of these ontologies is information extraction and retrieval as in Todirascu *et al.* (2000) and Faure & Poibeau (2000). Other applications can be question-answering, information-brokering, search engines and natural-language applications. The content type may be representation ontologies (e.g. frame ontology), natural-language ontologies (e.g. WordNet) or domain ontologies.

3.5.2 Structure and topology

Projects on learning ontologies work on different ontological structure and topology. Strict hierarchy of concepts (just one parent for each node) (Emde & Wettschereck, 1996), pyramid hierarchy (no two links crossing) (ASIUM), directed graphs (TEXT-TO-ONTO) and conjunction of graphs and axioms (HASTI) are some of structures used.

3.5.3 Representation language

Ontological knowledge can be represented by various representation languages, some of which are mentioned below.

- logic-based languages such as KIF (Knowledge Interchange Format) (HASTI), description logics (Todirascu *et al.*, 2000) and the KL-ONE family (SYNDIKATE),
- frame-based languages such as OKBC (Hwang, 1999),
- conceptual graphs (Roux *et al.*, 2000),
- Web-based languages such as XML (TEXT-TO-ONTO) and
- hybrid languages such as F-Logic-based extensions of RDF (TEXT-TO-ONTO).

The representation language of the resulting ontology becomes more important when the output of the ontology learning system is to be used in an existing application or to be merged with other ontologies.

3.6 Evaluation

Finding formal, standard methods to evaluate ontology learning systems is an open problem. To evaluate such systems there are two approaches:

a) evaluating the *learning methods* and
b) evaluating the *resulting ontology.*

As comparing the accuracy of techniques for learning ontologies is not a trivial task, the first approach, concerned with measuring the correctness of the learning techniques, is less addressed. Thus the more popular method for evaluating ontology learning systems is to (partially) evaluate their resulting ontology by means of one of the following approaches:

b1) Comparing two or more ontologies modeled within one domain using cross-evaluation techniques such as Maedche and Staab (2001b) which presents a multi-level approach for cross-evaluating ontologies.

b2) Evaluating domain ontologies through the application in which they are employed. As different systems learn different ontological elements, by applying different methods to different inputs, and the performance of proposed methods depends heavily on the applied methods and textual sources on which they have been applied, comparing their results by a formal method is difficult. So most of the proposed learning systems have their own testing and evaluation environment based on their application and selected domain. Examples include the evaluation in Agirre *et al.* (2000) which is a task-oriented evaluation, via word sense disambiguation using the SemCor corpus, or evaluating Doddle II with small-scale case studies in the law field.

Although it is difficult to establish and prove the quality of results, good indications can be found by the authors themselves. Most OL systems are evaluated by measuring the recall and precision of the learning module. The recall shows the number of correct concepts divided by the total number of concepts in the test set and the precision shows the number of correct concepts divided by the total number of extracted concepts. But still these results are not comparable because these systems work in different domains, on different inputs and using different backgrounds.

4 Comparing the systems in our framework

In Section 2 we selected seven prominent ontology learning systems and described their distinguishing features. Then we introduced a framework for classifying and comparing ontology learning systems in Section 3. There we discussed the framework dimensions, subdimensions and some possible values for them and cited some examples (OL systems) on each value. In our description of each dimension, we mentioned the position of the selected systems besides many other systems cited and discussed their features according to our framework. In this section we summarise these features in a table.

Table 2 shows the features of the systems studied here in the framework introduced above. In this table columns show the dimensions and subdimensions of our framework and rows indicate the systems studied here. As the table shows, some major columns indicating dimensions of the framework are divided into minor columns indicating sub-dimensions (such as dividing “starting point” into “prior knowledge” and “input” in which “input” itself is divided into “type” and “language”). In addition, some major rows indicating systems are divided into minor rows (in part) indicating subsystems or different aspects or capabilities of a system (such as having sub-rows in the second row, showing the different elements learned by Doddle II, each using different learning methods and creating different outputs). Each cell in the intersection of a column (a dimension) and a row (a system) indicates the value(s) of the dimension for the system.

As the table shows the systems are selected to cover a wide range of values and have different values for each dimension.

5 Conclusion

In this paper we introduced a framework for classifying and comparing ontology learning systems and gave an overview of some prominent ontology learning systems according to this framework. The dimensions of this framework help system-developers and knowledge engineers to choose or build their desired OL system with appropriate features according to their requirements. In this section we will first summarise the differences, strengths and weaknesses of various values for the framework

Table 2 Features of the ontology learning systems studied here

		Starting point				Learning		Output			
			Input								
Learning system	Element(s) learned	Prior knowledge	Type	Language	Preprocess	Method (approach and task)	Degree of automation	Type	Topology	Rep. language	Evaluation & testing
Asium	Verb subcat. frames + hierarchies	Linguistic K.	Unstructured (corpora)	French	Syntactic analysis (by Sylex)	Distributional, syntactic analysis + conceptual clustering	Cooperative	Domain specific	Pyramid Hierarchy (special DAG)	Subcat. frames	Tested for cooking and patents domains and terrorism IE
Doddle ii	Taxonomic relations	WordNet	Domain-specific texts	English		Match-result analysis & trimmed-result analysis	User interaction, handmade modification	Hierarchies (sub-trees)			Empirical evaluation in the law field
	Non-taxonomic conceptual relations					Analysis of lexical co-occurrence statistics using 4-grams		Concept pairs, concept specification templates			
Hasti	Words, concepts, taxonomic and non-taxonomic conceptual relations, axioms	Almost empty (small kernel)	Unstructured NL texts	Persian	Morpho-syntactic analysis by PeTex	Linguistic-based + semantic analysis + logical reasoning + using templates	Both automatic and cooperative modes	Lexicon & ontology: general & specific depending on input	Directed graphs + axioms	Subset of KIF	Tests on general and domain-specific texts
SVETLAN’	Noun classes		Structured (semantic domains with thematic units from Segcohlex) + unstructured (newspaper articles) input to SEGAPSITH	French	Syntactic analysis by Sylex + thematic analysis by SEGAPSITH	Clustering based on distributional similarities	Automatic	Structured domains showing small noun classes	Hierarchy (tree)	Special format of structured domains: $v \rightarrow r \rightarrow n_1, n_2, \ldots$	Tested on one month of Agence France Presse wires
SynDiKATe	Words, concepts, taxonomic and non-taxonomic conceptual relations	Generic and domain lexicons and ontologies	Unstructured NL texts	German	Incremental dependency parsing by ParseTalk	Centring-based discourse analysis + quality learning + semantic analysis	Automatic	Text knowledge base & updated input lexicon & ontology	Directed graphs	KL-One-like Description logic language	Evaluating sub-components separately in IT & MED domains
Text-to-Onto	Concepts, Taxonomic and Non-taxonomic conceptual relations	Lexical DB + domain lexicon	NL texts, Web docs, semi-structured (XML, DTD) and structured (DB schema, ontology) data	German, XML, HTML, DTD	Resource processing + shallow text processing by SMES	Association rules + formal concept analysis + clustering	Semi-automatic, interactive, balanced cooperative		Directed graphs	F-Logic based extensions of RDF exportable to OIL, DAML-ONT, . . .	Empirical evaluation by testing for different domains, inputs & methods
WEB → KB	Instances of classes and relations	The ontology for which instances are learnt	An ontology + training examples of instances	HTML		Bayesian learning (modified naïve Bayes)	Automatic	Same as the input ontology			Evaluated by 4-fold cross-validation methodology & tested by a KB of CS Depts
	Rules to recognise instances					FOL rule-learning (FOIL)		First-order rules (logic)			

dimensions to help developers to choose the appropriate features and then give a brief list of open problems to improve the ability and performance of ontology learning systems.

5.1 Choosing an appropriate learning system

To build a suitable ontology for an application (if there is not an existing one) we shall first note to what we have (the starting point) and what we desire to have (the resulting ontology). Then we shall find an appropriate path from the starting point to the desired result. The path may mean using an existing ontology-building system or creating our own.

- *The starting point* Choosing the starting point depends on both the availability and necessity of background knowledge and the input. In environments for which background knowledge is unavailable, e.g. domains for which no base ontology (domain ontology) is developed or languages for which no semantic lexicon is available, we shall build the ontology from scratch. Building ontologies from scratch (with minimal pre-knowledge) has the advantage of avoiding the integrating problems and it also eliminates the knowledge acquisition bottleneck by automating this process too. In this approach the system can acquire what it needs according to its application and ignore superfluities, or irrelevant or intemperate knowledge. Moreover, decreasing the amount of initial knowledge will result in decreasing the degree of developer-bias of the result. This way the result is more flexible and more suited to the special purpose it is applied to. The main disadvantage of this approach is that it is a longer process to learn more knowledge, and that it has difficulties in resolving ambiguities caused by lack of knowledge.

  The necessity and type of background knowledge are also determined by the type of intended ontology and methods to be applied. For example, building general-purpose and domain-specific ontologies may need different background knowledge: general-ontology building relies most of the time on general lexical resources, while in specialised domains the language is more constrained and generally methods are based on computing distributional classes.

  The type of input depends heavily on the application. If we are building ontologies for the semantic web, we should accept semi-structured and unstructured data, while if our system learns ontological knowledge from databases it should accept and process structured data.
- *The resulting ontology* Different applications require different ontologies. Ontologies may differ in their contents and the activities they support such as logical reasoning. For instance scientific, financial or business problem-solving applications usually use small, narrow, deep ontologies with specialised details coded in axioms to solve their specific problems, while for some information retrieval systems (such as web search engines) wide but superficial ontologies containing concept hierarchies with few interrelations, without axioms (such as WordNet), may be enough to improve the performance of their keyword search (Sowa, 2000). NLP applications use various kinds of ontology according to the depth of their language processing and reasoning. Applications that need deep natural-language understanding and reasoning such as automatic programming (translation of natural-language specifications to executable programs), some consulting and decision-making programs with natural-language interfaces and some question-answering systems, require deep, axiomatised ontologies with deep background knowlewdge and reasoning capabilities, while some other NLP applications such as commercial machine translation systems which are used to create a quick draft of a text in the destination language just use semantic lexicons such as WordNet.
- *The learning process*
  - *The degree of automation* At present, although there are some fully automatic systems, they are very restricted, work under limited circumstances and have lower performance (according to their acceptable results) compared to semi-automatic or cooperative systems. In other words cooperative systems give much more acceptable results because some interpretation decisions are left to the user during the learning process. A practical comparison between automatic and cooperative learning is done by some systems (Hasti; Finkelstein-Landau & Morin, 1999). Although this fact is generally valid for any learning system, it is more certain for systems

extracting information from texts. The problem comes from the limitations of linguistic tools (taggers, parsers), together with the particular nature of language. For example, in clustering based on distributional similarities, distributional classes do not correspond exactly with semantic classes, so the results can only give a list of words from which the user decides if this is a class or not and, if it is, then chooses its name and selects the right members. Fully automating this process should exploit multiple (hybrid) approaches to disambiguate results and complement each other.

- *The learning approach* In learning approaches one should choose statistical versus symbolic methods. Statistical methods are blind or knowledge-poor while symbolic methods are knowledge-rich. So for systems in which there is no semantic analysis or no reasoning (such as information retrieval systems), statistical methods are applicable. Statistical methods are more computable, more general, more scalable and easier to implement, while symbolic approaches are more precise, more robust and give more reasonable results. Statistical methods are usually general and can be used for different domains or languages while some symbolic methods such as linguistic-based or pattern-driven methods need more adaptation. Statistical methods need less initial resources than symbolic ones; they do not consider background knowledge. The main disadvantage of statistical methods is that the data is usually sparse, especially for general concepts in technical texts or technical concepts in general or irrelevant texts. Another characteristic of statistical approaches is that they are often used in offline (non-incremental) ontology learning. In other words the OL systems which use these methods usually learn from the whole input (e.g. from fully parsed text) at once.

  In symbolic methods we may learn incrementally, exploit reasoning techniques and use semantic knowledge to extract new knowledge. In symbolic approaches, the depth of processing is usually greater than in statistical approaches while its width is usually less. In other words symbolic methods often have higher accuracy and lower coverage than statistical methods. This can be seen in works such as WEB → KB that use both of them.

  The most knowledge-intensive methods in this category (symbolic approaches) are the logic-based ones. For systems that aim at deep understanding and reasoning and also at extracting meta-rules (rules to extract knowledge), logical methods are appropriate. Another category of symbolic methods, which need prior knowledge are pattern or template-driven methods, which (generally) are linguistic-based as well. They are the most popular symbolic methods for extracting conceptual relations from texts. Although these methods have good performance in particular domains for extracting particular relations, they are limited and inflexible. In other words they extract limited relations and the cost of adapting them to a new domain or extracting new patterns for the new domain may be high.

5.2 Open problems

Although there has been quite a lot of work on different aspects of ontology learning, there are still open problems that need more attention. Much work has been done on extracting taxonomic relations, less work has been done on discovering non-taxonomic relations and little has been done on axiom-learning. On the other hand most of the work done is domain-specific and has built domain-specific ontologies. Automatic building of general ontologies still needs more work. Most proposed systems have been tested in small, limited domains and need enhancement to work in real applications. Some open problems to be considered to improve the field are as follows:

- *Axiom learning* The only report we found on learning axioms is by Hasti, which learns some axioms in restricted circumstances. In this system the explicit axioms in conditional and quantified sentences in input texts are learned. There is work ongoing to extend it to learn implicit axioms from text too.
- *Evaluating ontology learning systems* Currently ontology learning systems are evaluated by evaluating their results in specific domains (for studying some ontology evaluation methodologies see Gomez-Perez, 1999; Gruninger & Fox, 1995; and for comparing ontologies see Maedche &

Staab 2001b; 2002). Finding formal, standard methods to evaluate ontology learning systems by proving their learning methods or proving the accuracy, efficiency and completeness of the built ontology is an open problem.

- *Full automation of the ontology learning process* Most systems use semi-automatic, cooperative or supporting tools to learn ontologies. Moving towards automation and eliminating user intervention needs more research.
- *Integrating successful modules to build complete autonomous systems* There are some successful modules, methods or tools which each carry out a learning task with high performance and leave others. Integrating them may eliminate their weaknesses and intensify their strengths.
- *Flexible, domain-/language-/application-neutral ontology learning* Many proposed ontology learning methods and approaches depend to a great extent on their specific environment, consisting of language, domain, application and input. Moving toward flexible and general methods may eliminate the need for the reconstruction of the learning system for new environments.

References

Agirre, E, Ansa, O, Hovy, E, and Martínez, D, 2000, "Enriching very large ontologies using the WWW" *Proceedings of the ECAI 2000 Workshop on Ontology Learning (OL'2000).*

Assadi, H, 1997, "Knowledge acquisition from texts: using an automatic clustering method based on noun-modifier relationship" *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL'97)* 504–506.

Assadi, H, 1999, "Construction of a regional ontology from text and its use within a documentary system" *Proceedings of the Formal ontologies in Information Systems (FOIS'98)* 236–249.

Bikel, DA, Schwartz, R and Weischedel, R, 1999, "An algorithm that learns what's in a name" *Machine Learning* **34** 211–231.

Bisson, G, 1992, "Learning in FOL with a similarity measure" *Proceedings of the 10th National Conference on Artificial Intelligence (AAAI'92)* 82–87.

Bisson, G, Nedellec, C and Canamero, D, 2000, "Designing clustering methods for ontology building – The Mo'K workbench" *Proceedings of the ECAI 2000 Workshop on Ontology Learning (OL'2000).*

Borgo, S, Guarino, N, Masolo, C and Vetere, G, 1997, "Using a large linguistic ontology for internet based retrieval of object-oriented components" *Proceedings of the Conference on Software Engineering and Knowledge Engineering* 528–534.

Bowers, AF, Giraud-Carrier, C and Lloyd, JW, 2000, "Classification of individuals with complex structure" *Proceedings of the 17th International Conference on Machine Learning (ICML2000)* 81–88.

Brewster, C, Ciravegna F and Wilks, Y, 2001, "Knowledge acquisition for knowledge management" *Proceedings of the IJCAI 2001 Workshop on Ontology Learning (OL'2001)* (position paper).

Caraballo, SA, 1999, "Automatic construction of hypernym labeled noun hierarchy from text" *Proceedings of ACL'99* 120–126.

Chalendar, G and Grau, B, 2000, "SVETLAN': a system to classify nouns in context" *Proceedings of the ECAI 2000 Workshop on Ontology Learning (OL'2000).*

Chapulsky, H, Hovy, E and Russ, T, 1997, "Progress on an automatic ontology alignment methodology" *ANSI Ad Hoc Group on Ontology Standards.*

Cherfi, H and Toussaint, Y, 2002, "How far association rules and statistical indices help structure terminology?" *Proceedings of the ECAI 2002 Workshop on Machine Learning and Natural Language Processing for Ontology Engineering (OLT'2002).*

Constant, P, 1996, *Reducing the Complexity of Encoding Rule-Based Grammars.*

Corcho, O and Gómez-Pérez, A, 2000, "Evaluating knowledge representation and reasoning capabilities of ontology specification languages" *Proceedings of the ECAI 2000 Workshop on Applications of Ontologies and Problem-Solving Methods.*

Craven, M, DiPasquo, D, Freitag, D, McCallum, A, Mitchell, T, Nigam, K and Slattery, S, 1998, "Learning to extract symbolic knowledge from the World Wide Web" *AAAI'98* 509–516.

Craven, M, DiPasquo, D, Freitag, D, McCallum, A, Mitchell, T, Nigam, K and Slattery, S, 2000, "Learning to construct knowledge bases from the World Wide Web" *Artificial Intelligence*, **118** 69–113.

Delteil, A, Faron-Zucker, C and Dieng, R, 2001, "Learning ontologies from RDF annotations" *Proceedings of the IJCAI 2001 Workshop on Ontology Learning (OL'2001).*

Eagles, 1996, homepage at `http://www.ilc.pi.cnr.it/EAGLES96/rep2`

Emde, W and Wettschereck, D, 1996, "Relational instance based learning" *Proceedings of the 13th International Conference on Machine Learning (ICML'96)* 122–130.

Farquhar, A, Fikes, R and Rice, J 1996, “The Ontolingua server: a tool for collaborative ontology construction” *Proceedings of KAW96*, also available as KSL-TR-96-26.

Faure, D, Nedellec, C and Rouveirol, C, 1998, *Acquisition of Semantic Knowledge Using Machine Learning Methods: The System Asium* Technical Report number ICS-TR-88-16, Laboratoire de Recherche en Informatique, Inference and Learning Group, Université Paris Sud.

Faure, D and Poibeau, T, 2000, “First experiments of using semantic knowledge learned by Asium for information extraction task using INTEX” *Proceedings of the ECAI 2000 Workshop on Ontology Learning (OL’2000).*

Ferret, O and Grau, B, 1998, “A thematic segmentation procedure for extracting semantic domains from text” *Proceedings of ECAI’98* 155–159.

Finkelstein-Landau, M and Morin, E, 1999, “Extracting semantic relationships between terms: supervised vs. unsupervised methods” *Proceedings of the International workshop on Ontological Engineering on the Global Information Infrastructure* 71–80.

Gamallo, P, Gonzalez, M, Agustini, A, Lopes, G and de Lima, VS, 2002, “Mapping syntactic dependencies onto semantic relations” *Proceedings of the ECAI 2002 Workshop on Machine Learning and Natural Language Processing for Ontology Engineering (OLT’2002).*

Gómez-Pérez, A, 1999, “Evaluation of taxonomic knowledge in ontologies and knowledge bases” *Proceedings of the 12th Workshop on Knowledge, Acquisition, Modeling and Management (KAW’99).*

Gruber TR, 1993, “A translation approach to portable ontologies” *Knowledge Acquisition* **5**(2) 199–220.

Gruninger, M and Fox, M, 1995, “Methodology for the design and evaluation of ontologies” *Proceedings of the IJCAI 95 Workshop on Basic Ontological Issues in Knowledge Sharing.*

Hahn, U and Schnattinger, K, 1998, “Towards text knowledge engineering” *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI’98)* 524–531.

Hahn, U and Marko, KG, 2002, “Ontology and lexicon evolution by text understanding” *Proceedings of the ECAI 2002 Workshop on Machine Learning and Natural Language Processing for Ontology Engineering (OLT’2002).*

Hahn, U and Romacker, M, 2001, “The SynDiKATe text knowledge base generator” *Proceedings of the 1st International Conference on Human Language Technology Research* 328–333.

Hearst, MA., 1992, “Automatic acquisition of hyponyms from large text corpora” *Proceedings of the 14th International Conference on Computational Linguistics* 539–545.

Heyer, G, Läuter, M, Quasthoff, U, Wittig, T and Wolff, C, 2001, “Learning relations using collocations” *Proceedings of the IJCAI 2001 Workshop on Ontology Learning (OL’2001).*

Hwang, CH, 1999, “Incompletely and imprecisely speaking: using dynamic ontologies for representing and retrieving information” *Proceedings of the 6th International Workshop on Knowledge Representation meets Databases (KRDB’99)* 14–20.

Kashyap, V, 1999, “Design and creation of ontologies for environmental information retrieval” *Proceedings of the 12th Workshop on Knowledge Acquisition, Modeling and Management.*

Kavalec, M and Svatek, V, 2002, “Information extraction and ontology learning guided by Web directory” *Proceedings of the ECAI 2002 Workshop on Machine Learning and Natural Language Processing for Ontology Engineering (OLT’2002).*

Kietz, JU, Maedche, A and Volz, R, 2000, “A method for semi-automatic ontology acquisition from a corporate intranet” *Proceedings of the EKAW 2000 workshop on Ontologies and Texts.*

Knight, K and Luk, SK, 1994, “Building a large-scale knowledge base for machine translation” *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI’94)* 773–778.

Lacher, MS and Groh, G, 2001, “Facilitating the exchange of explicit knowledge through ontology mappings” *Proceedings of FLAIRS’2001* 305–309.

Lenat, DB, 1995, “CYC: a large-scale investment in knowledge infrastructure” *Communications of the ACM* **38**(11) 33–38.

Lenat, DB and Guha, RV, 1990, *Building Large Knowledge Based Systems, Representation and Inference in the Cyc Project, Readings* Addison Wesley.

Li, H and Abe N, 1996, “Learning word association norms using tree cut pair models” *Proceedings of the 13th International Conference on Machine Learning* 3–11.

Maedche, A, Pekar, V and Staab, S, 2002, “Ontology learning part one: learning taxonomic relations (available at `http://wim.fzi.de/wim/publications/entries/1016618323.pdf`).

Maedche, A and Staab, S, 2000a, “Discovering conceptual relations from text” *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI’2000)* 321–325.

Maedche, A and Staab, S, 2000b, “Semi-automatic engineering of ontologies from text” *Proceedings of the 12th International Conference on Software Engineering and Knowledge Engineering (SEKE’2000)* 231–239.

Maedche, A and Staab, S, 2001a, “Ontology learning for the Semantic Web” *IEEE Journal of Intelligent Systems* **16**(2) 72–79.

Maedche, A and Staab, S, 2001b, *Comparing Ontologies – Similarity Measures and a Comparison Study* Internal Report 408, Institute AIFB, University of Karlsruhe.

Maedche, A and Staab, S, 2002, "Measuring similarity between ontologies" *Proceedings of EKAW'02*.

Moigno, S, Charlet, J, Bourigault, D, Degoulet, P and Jaulent, MC, 2002, "Terminology extraction from text to build an ontology in surgical intensive care" *Proceedings of the ECAI 2002 Workshop on Machine Learning and Natural Language Processing for Ontology Engineering (OLT'2002).*

Muggleton, S and De Raedt, L, 1994, "Inductive logic programming: theory and methods" *Journal of Logic Programming* **19**(20) 629–679.

Neumann, G, Backofen, R, Baur, J, Becker, M and Braun, C, 1997, "An information extraction core system for real world German text processing" *Proceedings of ANLP'97* 208–215.

Noy, NF and Musen, MA, 1999, "An algorithm for merging and aligning ontologies: automation and tool support" *Proceedings of the Workshop on Ontology Management at the 16th National Conference on Artificial Intelligence (AAAI-99).*

Noy, NF and Musen, MA, 2000, "PROMPT: algorithm and tool for automated ontology merging and alignment" *Proceedings of Seventeenth National Conference on Artificial Intelligence (AAAI-2000)* 450–455.

OLT'2002, Workshop on Machine Learning and Natural Language Processing for Ontology Engineering, held in conjunction with the ECAI'02 conference, Lyon, France, July 22–23, 2002 (available at `http://www.inria.fr/acacia/OLT2002`).

Omelayenko, B, 2001, "Learning of ontologies for the Web: the analysis of existent approaches" *Proceedings of the International workshop on Web Dynamics*.

Pereira, F, Tishby, N and Lee, L, 1993, "Distributional clustering of English words" *Proceedings of the 31st annual meeting of the Association for Computational Linguistics (ACL'93)* 183–190.

Pernelle, N, Rousset, MC and Ventos, V, 2001, "Automatic construction and refinement of a class hierarchy over semistructured data" *Proceedings of the IJCAI 2001 Workshop on Ontology Learning (OL'2001)* (position paper).

Quinlen, JR and Cameron-Jones, RM, 1993, "FOIL: a midterm report" *Proceedings of 12th European Conference on Machine Learning* 3–20.

Richards, BL and Mooney, RJ, 1992, "Learning relations by pathfinding" *Proceedings of AAAI-92* 50–55.

Richards, D and Compton, P, 1997, "Uncovering the conceptual models in RDR KBS" *Proceedings of the International Conference on Conceptual Structures (ICCS'97)* 198–212.

Riloff, E, 1996, "Automatically generating extraction patterns from untagged text" *Proceedings of the 13th Conference on Artificial Intelligence* 1044–1049.

Roux, C, Proux, D, Rechenmann, F and Julliard, L, 2000, "an ontology enrichment method for a pragmatic information extraction system gathering data on genetic interactions" *Proceedings of the ECAI 2000 Workshop on Ontology Learning (OL'2000).*

Ryutaro, I, Hideaki, T and Shinichi, H, 2001, "Rule induction for concept hierarchy alignment" *Proceedings of the IJCAI 2001 Workshop on Ontology Learning (OL'2001).*

Shamsfard, M, 2003, "Designing the ontology learning model: prototyping in a Persian text understanding system" Ph.D. dissertation, Computer Engineering Dept., AmirKabir University of Technology, Tehran, Iran.

Shamsfard, M and Barforoush, AA, 2000, "A basis for evolutionary ontology construction" *Proceedings of 18th IASTED International Conference on Applied Informatics (AI'2000)* 433–438.

Shamsfard, M and Barforoush, AA, 2002a, "An introduction to HASTI: an ontology learning system" *Proceedings of 6th Conference on Artificial Intelligence and Soft Computing (ASC'2002).*

Shamsfard, M and Barforoush, AA, 2002b, "Ontology learning from natural language texts" *International Journal of Human-Computer Studies (IJHCS)* (to be appeared).

Shamsfard, M and Barforoush, AA, 2002c, "Lexicon acquisition through Persian text processing" Technical report #81-4001-108, Intelligent Systems Lab., Computer Engng. Dept., Amir Kabir University of Technology, Tehran.

Soderland, S, Fisher, D, Aseltine, J and Lehnert, W, 1995, "Issues in inductive learning of domain-specific text extraction rules" *Proceedings of the IJCAI 95 Workshop on Approaches to Learning for Natural Language Processing* 290–301.

Sowa, JF, 2000, *Knowledge Representation: Logical, Philosophical and Computational Foundations* Brooks/Cole.

Sporleder, C, 2002, "A Galois lattice based approach to lexical inheritance hierarchy learning" *Proceedings of the ECAI 2002 Workshop on Machine Learning and Natural Language Processing for Ontology Engineering (OLT'2002).*

Staab, S, Maedche, A, Nedellec, C and Hovy, E (eds), 2001, *Proceedings of the Second Workshop on Ontology Learning, Held in Conjunction with the 17th International Conference on Artificial Intelligence IJCAI'2001 (OL'2001).*

Staab, S, Maedche, A, Nedellec, C and Wiemer-Hastings, P (eds), 2000, *Proceedings of the First Workshop on Ontology Learning, Held in conjunction with the 14th European Conference on Artificial Intelligence ECAI'2000 (OL'2000).*

Stevenson, M, 2002, "Combining disambiguation techniques to enrich an ontology" *Proceedings of the ECAI 2002 Workshop on Machine Learning and Natural Language Processing for Ontology Engineering (OLT'2002).*

Sundblad, H, 2002, "Automatic acquisition of hyponyms and meronyms from question corpora" *Proceedings of the ECAI 2002 Workshop on Machine Learning and Natural Language Processing for Ontology Engineering (OLT'2002).*

Suryanto, H and Compton, P, 2000, "Learning classification taxonomies from a classification knowledge based system" *Proceedings of the ECAI 2000 Workshop on Ontology Learning (OL'2000).*

Termier, A, Rousset, MC and Sebag, M, 2001, "Combining statistics and semantics for word and document clustering" *Proceedings of the IJCAI 2001 Workshop on Ontology Learning (OL'2001).*

Thompson, CA and Mooney, RJ, 1999, "Automatic construction of semantic lexicons for learning natural language interfaces" *Proceedings of 16th National Conference on Artificial Intelligence (AAAI'99)* 487–493.

Todirascu, A, de Beuvron, F, Galea, D and Rousselot, F, 2000, "Using description logics for ontology extraction" *Proceedings of the ECAI 2000 Workshop on Ontology Learning (OL'2000).*

Vargas-Vera, M, Domingue, J, Kalfoglou, Y, Motta, E and Buckingham-Shum, S, "Template-driven information extraction for populating ontologies" *Proceedings of the IJCAI 2001 Workshop on Ontology Learning (OL'2001).*

Wagner, A, 2000, "Enriching a lexical semantic net with selectional preferences by means of statistical corpus analysis" *Proceedings of the ECAI 2000 Workshop on Ontology Learning (OL'2000).*

Wiemer-Hastings, P, Graesser, A and Wiemer-Hastings, K, 1998, "Inferring the meaning of verbs from context" *Proceedings of the 20th Annual Conference of the Cognitive Science Society* 1042–1047.

Williams, AB and Tsatsoulis, C, 2000, "An instance-based approach for identifying candidate ontology relations within a multi-agent system" *Proceedings of the ECAI 2000 Workshop on Ontology Learning (OL'2000).*

Yamaguchi, T, 2001, "Acquiring conceptual relations from domain-specific texts" *Proceedings of the IJCAI 2001 Workshop on Ontology Learning (OL'2001).*

Zelle, JM and Mooney, RJ, 1993, "Learning semantic grammars with constructive inductive logic programming" *Proceedings of the 11th National Conference on Artificial Intelligence (AAAI'93)* 817–822.