Methods and Tools for Automatic Construction of Ontologies from Textual Resources: A Framework for Comparison and Its Application

Toader Gherasim, Mounira Harzallah, Giuseppe Berio, and Pascale Kuntz

Abstract. Over the recent years, several approaches and tools for the automatic construction of ontologies from textual resources have been proposed. This paper provides a comparative analysis of four well known approaches and related tools among existing ones. The selected approaches and related tools indeed cover all the steps of the ontology construction process. In the first part of the paper, we introduce Methontology and related task i.e. a well-known reference methodology designed for the manual construction of ontology; then, according to Methontology, we analyze and classify detailed subtasks required by those approaches. Based on this uniform classification, we provide a very detailed comparison of those approaches: we explain the main techniques and introduce tools used in the various subtasks of each approach and we highlight the main similarities and differences between the techniques used in comparable subtasks belonging to distinct approaches. In the second part of the paper, we introduce various measures for evaluating tools effectiveness wrt a manually constructed ontology. Then, we evaluate and compare the key tools supporting those approaches by using the provided measures and a specific set of textual resources.

1 Introduction

Since the foundational work of Gruber [Gruber, 1993], ontologies are an essential element for knowledge engineering, and the development of the semantic web increased even more their importance. Today, in several domains, ontologies

Toader Gherasim · Mounira Harzallah · Pascale Kuntz
LINA, UMR 6241 CNRS
e-mail: {toader.gherasim,mounira.harzallah}@univ-nantes.fr, pascale.kuntz@polytech.univ-nantes.fr

Giuseppe Berio
LABSTICC, UMR 6285 CNRS
e-mail: giuseppe.berio@univ-ubs.fr

DOI: 10.1007/978-3-642-35855-5_9

are considered the central component of decision support or information retrieval systems (e.g. [Osborne et al., 2009] focuses on ontologies in the medical domain and [Bourigault and Lame, 2002] focuses on ontologies in the legal domain). Earlier, ontology construction was largely based on human expertise. Today, ontologies are more and more used, tend to contain thousands of concepts (as reported in [Aime et al., 2009]) and therefore it is really interesting to massively use automation to better targeting the contribution of human experts. Indeed, as well known, for building an ontology, experts start from reference documents and knowledge by identifying the central concepts and (various kinds of) relations, often extracted from names, verbs and definitions (as also typical in the general areas of software and information system design, especially focusing on requirements). Therefore, it is very interesting to try to extract, prune and filter the huge amount of elements that can be found in input documents and general knowledge.

Accordingly, during the last decade several approaches for automatic ontology construction from text corpus (i.e. reference documents but also general documents) have been proposed [Velardi et al., 2007, Maynard et al., 2009b]. A wide range of techniques has been used to automatically execute the different tasks of ontology construction process. Because most of these approaches have been developed and tested in specific application contexts, sometimes, the employed techniques can be applied only within these contexts and provide interesting outcomes only for specific types of text content.

The aim of this paper is to propose a comparative analysis of some of these approaches and of the associated tools, implementing mentioned techniques. To provide a coherent framework for comparing approaches, we have used Methontology [Fernandez et al., 1997], one of the most known methodologies for ontology construction, supplying a set of reference tasks necessaries to build an ontology. Our interest has been concentrated towards approaches which cover all the steps of the ontology construction process as defined according to Methontology i.e. approaches that take as input texts and that propose as output an ontology in its most basic form – concepts, instances, relations. Tools associated to compared approaches are in turn compared, whenever possible and meaningful, on two distinct plans, technical and experimental, within a coherent test-bed platform.

Through the paper we consider and analyze four approaches and related tools[1]: OntoLearn – *TermExtractor*, *WCL System* [Navigli et al., 2003, Velardi et al., 2007], Alvis [Nedellec, 2006], Text2Onto – *Text2Onto* [Cimiano and Volker, 2005], and SPRAT – *SPRAT* [Maynard et al., 2009a, Maynard et al., 2009b]. These approaches construct domain ontologies that reflects the domain covered by the input texts, and not top level, highly abstract ontologies or lexicalized ontologies (like WordNet).

The paper is organized as follows. In the first part (Section 2) we present the tasks of the conceptualization activity of Methontology: this allows both a deep understanding of each approach and enables a further comparison between the four approaches mentioned above. In the same section, the various techniques employed by tools are also shortly presented. In the second part (Section 3.1), we use again

[1] The name of tools is in italics.

tasks belonging to Methontology and identified subtasks specific to each approach to precisely describe supporting tools and related supported tasks, subtasks and techniques. Then (Section 3.2) we provide an analysis of supporting tools using some of the general and extraction criteria of the framework proposed in [Park et al., 2011]. In Section 3.3 we describe our experimental tests on selected tools mentioned above (on one medium-size corpus), analyze and compare the obtained results. Section 3.4 is devoted to a critical discussion and the last section provides the reader with concluding remarks.

2 Comparison of Four Relevant Approaches for Automatic Construction of Ontologies from Textual Resources

2.1 *Methotology*

Methontology [Fernandez et al., 1997, Corcho et al., 2005] is one of the most known methodologies for ontology construction. It is a general, domain independent methodology, which defines the main activities of the ontology construction process and specifies the steps for performing them. The most important activity is the conceptualization, where informal knowledge is converted into semi-formal specifications which enable the identification of the main ontology components. We have reviewed the literature on Methontology and below we present the seven tasks belonging to the conceptualization activity.

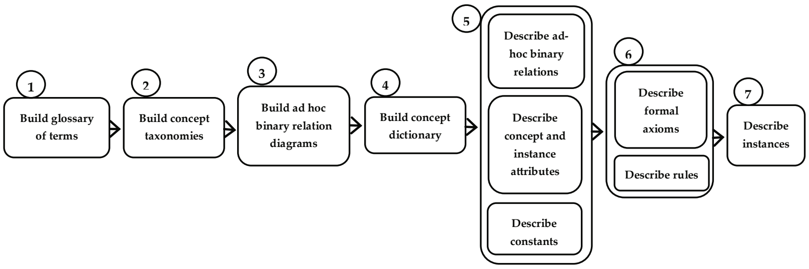

Fig. 1 Tasks of the conceptualization activity of Methontology (figure adapted from [Corcho et al., 2005])

The glossary of terms built in the **first** task (**build glossary of terms**) contains all the terms and associated definitions corresponding to the different ontological constituents (such as concept, instance, relation and attribute). In the **second** task (**build concept taxonomies**) taxonomies are constructed by selecting from the glossary the terms that are concepts, by grouping similar terms corresponding to a same concept, and by arranging concepts in a concept (*is-a*) hierarchy. The **third** task (**build diagrams for ad-hoc binary relations**) concerns binary non-taxonomic relations involving previously identified concepts, usually identified by analyzing the verbs presents in the glossary of terms. A **concept dictionary is built** in the **fourth** task. This dictionary also specifies the properties, instances and relations that are linked to each concept of the taxonomy. In the **fifth** task are identified **detailed definitions for relations, attributes and constants**. Additional **formal axioms and rules** can be defined in the **sixth** task. Finally, in the **seventh** task, a **detailed description of instances** must be provided. Tasks 5, 6 and 7 can be considered as tasks for ontology refinement, because all the central structural constituents of the ontology are already identified in precedent tasks.

In Section 2.3, Methontology and its tasks will be used as a framework for comparing, on a common base, the four approaches we mentioned in the Introduction: OntoLearn, Alvis, Text2Onto, SPRAT.

2.2 Overview of Techniques for Automating Ontology Construction from Textual Resources

Generally speaking, any approach for automating, fully or partially, ontology construction starting from textual documents comprises several algorithmic techniques that are based on theoretic and empiric principles.

[Buitelaar et al., 2005] and [Nazarenko and Hamon, 2002] have already proposed classifications of those techniques. Therefore the aim of this section is to focus and shortly present the techniques implemented in the various tools associated to the four approaches mentioned in the Introduction. These relevant techniques are useful for automating, possibly partially, the first three tasks of the Methontology conceptualization activity i.e.: (1) Build glossary of terms (and more specifically focusing on term extraction), (2) Build concept taxonomies, and (3) Identify ad-hoc relations. Other techniques mentioned in [Buitelaar et al., 2005] cover also other tasks (e.g. Describe rules) but they are however not implemented in the context of the four approaches mentioned in the Introduction. Therefore these additional existing techniques are not reported in the remainder.

Within the context of **Build glossary of terms**, term extraction is usually supported by two types of technique used jointly i.e. *linguistic techniques (L)* and *statistical techniques (S). Linguistic techniques* analyze sentences and discourses in terms of grammatical constituents: delimiting terms, morphosyntactic tagging of terms (e.g. by using Noun, Verb, and Adjective), identifying syntactic constituents (e.g. subject, direct object) and relations between them (usually focusing on verbs). *Statistical techniques* are based on the frequency of a term in one document or in all the documents of one corpus. According to [Zouaq and Nkambou, 2010], these techniques include the popular term measure is *TFIDF* (i.e. normalized term frequency, inverse document frequency [Salton et al., 1975]). Definitions of terms, as the base for building a glossary, are usually found by using techniques looking to *external resources (ES)* such as controlled web pages or WordNet.

For **Build concept taxonomies** task two types of techniques are often used: *structural techniques (St)* and *contextual techniques*. Concrete subtypes of this last type of techniques are presented in the remainder. However, there are some approaches that use *external resources* (WordNet, dictionaries, etc.) rather than the text for building the taxonomy.

Structural techniques use the structure of a term representing a concept. They can be based on the syntax of the term (e.g. *domain ontology* subsumes *ontology*), on the morphology of the term (e.g. *blood mononuclear cell* as a variant of *blood cell*), on the lexical structure of the term (when the internal structure of the term serves as support for term clustering [Nazarenko and Hamon, 2002]) or on the meaning of the term (when the meaning of a complex term is built by taking into account the structure of the term and the meaning of the words that compose the term; the meaning of the words is found in external resources such as WordNet or dictionaries).

Contextual techniques are based on the context where appear terms representing concepts. A context is usually defined as a vector representing syntactic dependencies between the term that represents the concept and other surrounding terms (surrounding terms provide the context). Two main families are recognized as part of those techniques: *distributional & clustering techniques (Di & Cl)* and *pattern based techniques (Pa)*.

Distributional & clustering techniques try to cluster together concepts according to the distributions of their associated terms. These techniques are based on the hypothesis that a term has the same meaning when occurring in similar contexts ([Harris, 1968]).

Pattern based techniques try to identify in texts expressions that contain terms representing concepts and whose structure follows the given pattern. The most popular patterns for the subsumption relation are the so-called Hearst patterns ([Hearst, 1992]); for instance, the pattern *NP such as NP, NP, NP … and NP*[2] applied to the sentence *fruits such as orange and apple* extracts that *apple* and *orange* are subsumed by *fruits*.

Patterns can be predefined or learnt. For the latter case, specifically developed *pattern learning techniques (PL)* have been introduced. These techniques are often based on a set of training concept pairs that satisfy a given relationship.

Finally, for **Identify ad-hoc relations** task, *pattern based techniques* are often used. There exists a large variety of patterns corresponding to different ad-hoc relations: *structural patterns* (e.g. *NP part-of NP*), *domain specific patterns* (e.g. *NP caused by NP*, typically referenced in the medical domain), etc. However, *pattern learning techniques*, *external resources* and *distributional & clustering techniques* are sometimes used.

Our tool analysis reveals that relevant tools often combine several techniques (i.e. resulting in kinds of hybrid techniques) for achieving better results for the purpose of the supported tasks.

[2] NP – Noun Phrase.

2.3 Using Methontology as a Conceptual Comparison Framework

In this section we use Methontology as a reference to compare on a common basis the tasks and the subtasks of the four selected approaches. Also, we identify and compare the main techniques employed within each approach to automatically execute the different tasks.

More precisely, we consider that the seven tasks presented in the Section 2.1 define a complete repository of tasks which must be executed in order to construct an ontology. The four approaches are mainly focused on aspects related to the construction of the structure of the ontology (identification of concepts, concepts taxonomies, relations and instances), aspects which correspond to the first four tasks of our repository, and pay little attention to the refinement of the ontology.

Table 1 summarizes the most important correspondences between the Methontology tasks and the four approaches mentioned in the Introduction: it should be noted that for each approach the task partition is based on the Methontology task definition. When possible, we also indicate in Table 1, for each task of each approach, the type of techniques that are used to automate the task. *Man* acronym is used for uniformity, to indicate a manually performed task.

3 Comparative Analysis of Tools Related to the Four Relevant Approaches for Automating Ontology Construction Process from Textual Resources

In the remainder, we focus on the tools that support the four approaches identified in Section 2. Section 3.1 describes the various tools, indicating the types of techniques they implement, and provides a general description of how tools are used within their respective approaches. Each of the next two sections is devoted to a detailed comparison of tools. The first (Section 3.2) is based on the technical features that characterize those tools. The second (Section 3.3) concerns only available tools (because, as pointed in Section 3.2, some tools are not available) and is based on a series of experimental tests.

The technical features of tools refer to characteristics and functionalities such as accepted formats, generated formats, possibilities of configuration, etc. The technical features that we have identified in a previous work ([Gherasim et al., 2011]) can be re-organized according to what in [Park et al., 2011] are named *general* and *extraction criteria*. We are going to use those criteria to present our current work (Section 3.2).

Tests are devoted to compare tools based on their outcomes. For the same purpose, [Park et al., 2011] have introduced various quality criteria, namely *semantic criteria* (comprising *interpretability*, *consistency* and *clarity* sub-criteria) and *pragmatic criteria* (comprising *accuracy*, *completeness* and *coverage* sub-criteria). Those criteria are used to evaluate, for each tool, the automatically built ontology.

Table 1 Correspondences and differences between the tasks of the four approaches and the repository of tasks inspired from Methontology

Methontology tasks	OntoLearn tasks		Alvis tasks	Text2Onto tasks	SPRAT tasks
Build a glossary of terms *Identify and define terms corresponding to the different elements of the ontology: concepts, attributs, instances and relations*	Terminology extraction – *L & S* Terminology filtering – *using L & S filters* Terminology validation – *Man* Identification of definitions for terms *(which will compose a glossary) – using Internet searches – ES* *Terms = { concepts }*		Terminology extraction – *L* Terminology validation – *Man* *Terms = { concepts, instances }*	Terminology extraction – *L & S* *Terms = { concepts, instances }*	Terminology extraction – *L* *Terms = { concepts, instances }* **Iterative execution** Choose a term that is involved in a *'is-a'* pattern that links him to a concept that already exists in the ontology – *Pa*
Build concept taxonomies *Group terms that correspond to the same concept* *Arrange in one or more taxonomies the already identified concepts*	Semantic disambiguation of every complex term by inter-secting semantic nets associated to each word composing the term: *SSI algorithm (specific to OntoLearn) and WordNet – ES & St* Finding taxonomy relations on the synsets associated to each word composing a complex term underlying concept	Hypernym Extraction Star pattern identification and sentence clustering – *Man* Word-Class Lattice (WCL) construction – *PL* Sentences matching with WCL – *Pa*	Build a taxonomy – *Identify the contextual attributs of each term; build a taxonomy using the unsupervised classification (based on terms attributes) of terms and classes of terms (concepts) – Di & Cl*	Identify *'is-a'* relations between concepts *(using WordNet, patterns and heuristics (based on the structure of compound terms) – St & Pa & ES*	*(if the chosen term is a concept)* Add the concept to the ontology *– using rules for inserting concepts in the ontology and rules for resolving conflicts (these rules are specific to SPRAT) – Pa*
Build diagrams for ad hoc binary relations *Group terms that correspond to the same relation* *Define binary relations by identifying, for each relation, the related concepts – by analyzing the pair of terms associated with each verb (its subject and its object)*	Provide domain semantic relations examples (learning set) – *Man* Learn rules to classify the relations that hold between pairs of concepts – *PL* Apply the rules to complex term to identify relations between its components – *Pa*		Identify the syntactic dependencies that can properly characterize a relation – *using ASA (Abstraction of the Syntactic Analysis), a formalism specific to Alvis* Identify some examples – *Man* Learn a set of rules for discovering relations – *using an inductive learning program that uses the ASA formalism – PL* Identify concepts connected by the relations learned from examples – *automatically, using rules – Pa*	Identify *'subtopic-of'* relations between concepts *– (using statistical analysis of cooccurrence of concepts) – Di & Cl* Identify general relations (relations defined by verbs) *– (using a shallow parsing strategy and information about the frequency of the terms) – Di & Cl*	Identify and add to the ontology relations between the new concept and concepts that already exist in the ontology *(using predefined patterns) – Pa*
Build the concept dictionary *(that contains, for each concept, the associated attributes, instances and relations)*	—		—	Identify *'instance-of'* relations – *Di & Cl*	*(if the chosen term is an instance)* Add the instance to the ontology *– using specific SPRAT rules – Pa*

Human-experts are asked to assess, based on the provided criteria definitions, the quality of ontologies according to each criteria. [Park et al., 2011] tried to limit the subjectivity by asking four human-experts to evaluate each ontology. However, after a deep analysis, we consider that some of the *semantic* and *pragmatic criteria* as well as the associated evaluation methods proposed in [Park et al., 2011] are not fully suitable for a precise comparison between tools.

On the one hand, according to our previous work ([Gherasim et al., 2011]), we consider that a more precise comparison among tools outcomes should be performed by using a common reference ontology: the various distinct ontologies built by using tools can then be compared to that single common reference ontology. On the other hand, despite the interest of *semantic* and *pragmatic criteria* introduced in [Park et al., 2011], we consider that some of those criteria/sub-criteria are vague or not suitable for evaluating ontologies automatically built by tools. Specifically, we consider that *consistency* remains vague (indeed the term "consistency" in the context of ontologies may be interpreted as "logical consistency" it should also be noted that the term "consistency" in the context of ontologies may be interpreted as "logical consistency"), *clarity* does not take into account the ontology domain and *interpretability* focuses only on WordNet; *completeness* and *accuracy* are suitable sub- criteria but their associated evaluation methods, only based on expert judgment, are not suitable especially because ontologies are automatically built (so ontologies lack of concept definitions and the number of concepts may growth exponentially with text-size; also, relationships, when available, are very intricate).

Therefore, in our current work, any tool comparison based on tool outcomes is performed as in our previous work ([Gherasim et al., 2011]). Specifically, a common reference ontology has been manually built (a general golden standard is rarely

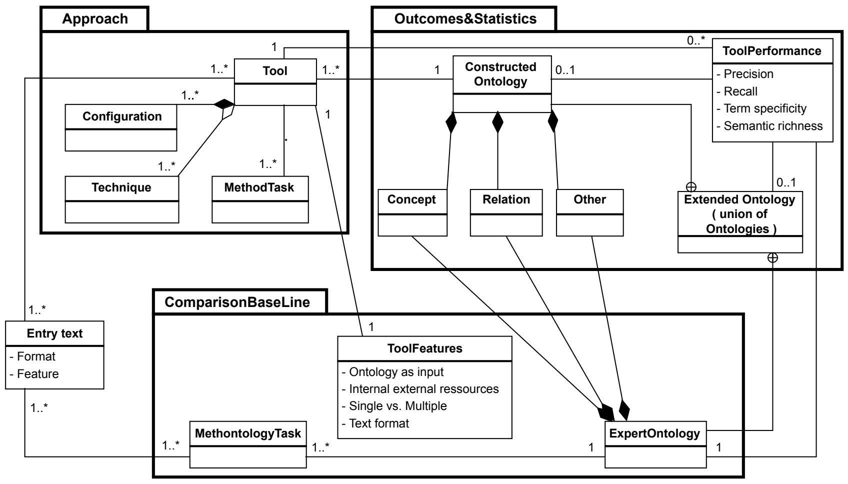

Fig. 2 The UML diagram of the test-bed platform used for performing the different tests

available, see Section 3.3). Standard measures (precision and recall) and additional measures (term specificity and semantic richness) are introduced and used to compare ontologies automatically built by tools to the common reference ontology. The manually built ontology reflects the expert knowledge about the specific domain, as restrained by the texts (a partially view on the domain) submitted to tools. Section 3.3 presents in detail the results of the various tests.

A test-bed platform has been realized for performing tests according to the discussion above. Figure 2 shows the platform packages as a UML diagram. The *Approach* package is used to describe the approach in detail, comprising *Tool*, *Task*, *Technique* and *Configuration* (*Configuration* refers to how a tool is configured and installed within the platform). The *ComparisonBaseLine* package provides the reference ontology (manually built according to Methontology); the package also comprises the technical features of tools. The *Outcomes&Statistics* package provides the various ontologies built by tools and the standard measures used to compare those ontologies to the reference ontology; the package also comprises the extended ontology which is built as the union between the reference ontology and ontologies built by tools. The reason for including this union is discussed in Section 3.3.1. Finally, *EntryText* is used to describe the input corpus. It is characterized by its format (such as plain text, HTML text, etc.) and by some features (such as *dense*, *self-contained*, etc.). Although the text features are extremely important, in this paper we have not fully exploited them because such an analysis requires a large number of additional tests.

3.1 Analysis of the Role of Each Tool for the Corresponding Approach

Each approach is supported by one tool or a set of tools that implements the various techniques presented in Section 2.2. Table 2, built on Table 1, lists those tools and highlights the correspondences between them and the different tasks/subtasks belonging to the identified approaches.

Table 2 is therefore used to partially instantiate the package *Approach* in our test-bed platform (see Section 3 introduction and Figure 2). There are subtasks for which no tool is proposed (e.g. “Provide domain semantic relations examples” subtask of OntoLearn approach), or it remains unclear if some tools are available (e.g. “Identify relations” subtask of Alvis approach).

There are subtasks for which several tools are proposed (e.g. “Terminology extraction” subtask of SPRAT approach). Some tools correspond to only one specific subtask, such as *GlossExtractor*; some tools cover two or several subtasks, or even the entire ontology construction process (e.g. *SPRAT*).

Table 2 also allows to identify comparable tools: for instance, it is not possible to compare *TermExtractor* with the *Relation* module of *Text2Onto*, but it is possible to compare *TermExtractor* with the *Concept* module of *Text2Onto*.

Table 2 Correspondences, for each approach, between its subtasks and its supporting tools. Tested tools are highlighted. For each tool we indicate in parentheses which type of techniqes it implements.

Methontology tasks	OntoLearn tasks		Alvis tasks		Text2Onto tasks		SPRAT tasks	
	Subtask	Tool	Subtask	Tool	Subtask	Tool	Subtask	Tool
Build a glossary of terms	Terminology extraction Terminology filtering	**TermExtractor** – *(L & S)*	Terminology extraction	*YATEA – (L)*	Concept extraction	**Text2Onto** *module concept – (L & S)*	Terminology extraction	*JAPE, NEBOne – (L)* *Term Raider – (L & S)* **SPRAT** – *(Pa)*
	Terminology validation	—	Terminology validation	—	Instance extraction	**Text2Onto** *module Instance – (L & S)*	Choose a term	—
	Identification of definitions for terms	*GlossExtractor – (ES)*						
Build concept taxonomies	Semantic disambiguation of complex terms Hypernym Extraction	*SSI – (ES & St)* **WCL System**– *(PL, Pa)*	Build a taxonomy	*BioLG, ASIUM adapted for Alvis – (Di & Cl)*	Identify *’is-a’* relations	**Text2Onto** *module SubclassOf – (St & Pa & ES)*	Add the concept to the ontology	—
	Find taxonomy relations	—						
Build diagrams for ad hoc binary relations	Provide domain semantic relations examples	—	Abstraction of the Syntactic Analysis	*LINK-PARSER*	Identify *’subtopic-of’* relations	**Text2Onto** *module SubtopicOf – (Di & Cl)*	Identify and add relations to the ontology	—
	Learn rules	*C4.5 – (PL)*	Identify some examples	—				
	Identify relations between the components of complex terms	—	Learn a set of rules	*Propal – (PL)*	Identify general relations	**Text2Onto** *module Relation – (Di & Cl)*		
			Identify relations	—				
Build the concept dictionary	—	—	—	—	Identify *’instance-of’* relations	**Text2Onto** *module InstanceOf – (Di & Cl)*	Add the instance to the ontology	—

We can classify these tools in two groups: in the first group there are generic tools that were developed for another purpose than building ontologies, and have been found useful for building ontologies; in the second one, there are specialized tools especially designed for ontology construction.

The first group includes programs that can manipulate and identify regular expressions in texts (e.g. *JAPE* [Maynard et al., 2009a, Maynard et al., 2009b]), syntactical analyzers (e.g. *LinkParser*, *BioLG* [Nedellec, 2006]), inductive learning programs (e.g. *C4.5* [Navigli et al., 2003], *Propal* [Nedellec, 2006]) and term extractors (e.g. *YATEA* [Nedellec, 2006]).

The second group contains specialized tools that were developed to support the ontology construction process and which are directly associated with the previously presented approaches: *Text2Onto* [Cimiano and Volker, 2005] for the approach with the same name; *TermExtractor*, *GlossExtractor* [Velardi et al., 2008], *SSI* [Navigli and Velardi, 2005] and *WCL System* for OntoLearn; *NEBOnE*, *TermRaider* and *SPRAT* [Maynard et al., 2009a] for SPRAT; *ASIUM* [Nedellec, 2006] for Alvis.

We note that Text2Onto proposes a specialized tool (*Text2Onto*) that covers the entire process of extracting an ontology. SPRAT proposes several tools, but one of them, that has the same name as the approach – *SPRAT*, covers the entire process of extracting an ontology. *SPRAT* as tool uses, in a transparent manner, the results of *JAPE* and *NEBOnE*, but it does not reuse *TermRaider* results. OntoLearn and Alvis do not propose any tool covering the entire process but a set of tools where each tool covers partially the process; a tool can take as input the results of another tool and, eventually, the set of tools covers the entire process of extracting an ontology.

OntoLearn provides specialized tools for building a term glossary (*TermExtractor*, *GlossExtractor*) and the concept taxonomy (*WCL System*), but for the relations extraction most of the works remains to be manually performed. Nevertheless, users can be assisted by a tool (*C4.5*) that learns rules for discovering relationships over compound terms underlying concepts (such as a rule establishing that in a compound term *XY*, if *X* is a type of building material, then *X MATTER Y* holds, with confidence 0.5, being *MATTER* a relation – applying this rule to *stave church*, *Church MATTER Stave* is a possible relation). Alvis proposes a generic tool (*YATEA*) for building a term glossary, a specialized tool (*ASIUM*) for the taxonomy construction, and similarly to OntoLearn, a tool (*Propal*), enables rules learning from examples for ad-hoc relation extraction subtask.

3.2 Tool Comparison Based on Technical Features

This section provides a tool comparison based on technical features, defined as *general* and *extraction criteria* in [Park et al., 2011]. The content of this section is used to partially instantiate the package *ComparisonBaseLine* within the test-bed platform.

The **general features** deal with the exterior features of tools: user interface, availability and time to first use. As in our previous work, we are interested only in the tools' **availability**, defined as *if a tool can be acquired without too much effort*. We adapted this criterion to take into account one of the specificity of the tools we analyze – the fact that some of them are available as web services. So, our definition for tool's availability is: *if a tool can be acquired and installed on local machines or if it is available as a web service or web application and can easily be accessed and tested.*

The tools proposed by OntoLearn, *TermExtractor*, *GlossExtractor* and *SSI*, are available as web applications, on the authors' servers. They can be accessed via a webpage. The development of *WCL System* has just finished, so that *WCL System* is not yet released and it can be tested only by asking the authors to test it. For Alvis, *ASIUM*, the only specialized tool proposed, is not available to be tested. *Text2Onto* can be downloaded and tested on a local computer. *TermRaider* and *SPRAT*, the tools proposed by SPRAT, are available only as web services. The availability of all these tools, except *Text2Onto*, is closely related to the availability of the web servers where they are hosted and we met some access difficulties in our experimentations.

Because *ASIUM* is not available to be tested and *YATEA* is just a generic tool for term extraction we think that it is uninteresting to keep in our analysis Alvis and its tools. So, from now, we will ignore them. For the same reason we ignore *C4.5*, the generic tool used by OntoLearn to learn rules for relation extraction.

The **extraction features** concern the main function used for ontology extraction: (1) preprocessing requirement; (2) ontology reuse; (3) extraction level; (4) degree of automation; (5) algorithm selection; (6) efficiency; (7) reliability. Another extraction feature, not included in [Park et al., 2011] framework, concerns the auxiliary tools and external resources (8) that are used by each tool.

The first criterion, (**preprocessing requirement** (1)), consider whether a tool requires or not additional preprocessing of documents taken as input (e.g. linguistic annotation). This criterion partially corresponds to a criterion from our previous work – the *type of inputs and outputs*. As we here also analyze tools that do not take texts as input, we adapt the *preprocessing requirement* criterion in order to take into account also the type of inputs of each tool, and not only the preprocessing effort.

All the tools covering initial subtasks of each approach (*TermExtractor*, *Text2Onto*, *TermRaider* and *SPRAT*) accept as input simple text files (txt). *Text2Onto* also accepts PDF files, and *TermExtractor* PDF, DOC, HTML files or archives containing this type of files. *SSI* and *GlossExtractor* take as input a list of terms, and *WCL System* a list of definitions. No one of all these tools needs preprocessing efforts.

The second criterion (**ontology reuse** (2)) takes into account the fact that a tool can use concepts from existing ontologies or an entire ontology when constructing a new one. As for the precedent criterion, we extend this criterion to take into account the fact that there are tools that take as input list of terms or definitions. *TermExtractor* and *TermRaider* can enrich an existing list of terms with new terms. *Text2Onto* can selectively update its results when the texts input evolve. *SPRAT* can take as input an ontology and enrich it with new concepts.

The **extraction level** (3) examines whether a tool can automatically or semi-automatically extract concepts or both concepts and their relations. We further refine this criterion by further differentiating between taxonomic relations and the other relations and by taking into account the presence/absence of instance identification.

For the three approaches, the corresponding tools (*TermExtractor*, *Text2Onto* and *SPRAT*) are able to identify concepts. Only *Text2Onto* and *SPRAT* can identify instances. *WCL System*, *Text2Onto* and *SPRAT* can identify taxonomic relations. Only *Text2Onto* and *SPRAT* can automatically identify other (ad-hoc) semantic relations.

The fourth criterion concerns the **degree of automation** (4) for extracting concepts and relations: a tool can be considered automatic if it performs that extraction without any human intervention; a tool can be considered as semi-automatic if the user must supply some extraction rules or whenever any human intervention is required during that extraction.

Text2Onto and *SPRAT* are automatic tools. The four tools proposed by OntoLearn (*TermExtractor*, *GlossExtractor*, *SSI* and *WCL System*) are automatic when considered independently, but, they are not integrated in a fully automatic system for ontology construction.

The fifth criterion, called **algorithm selection** (5), takes into account the possibility for users to select various algorithms/techniques when using one tool. This criterion partially corresponds to the *configurability* criterion of our previous work. We extend it by accounting the possibility to configure different parameters of the proposed algorithms.

Only two tools, *TermExtractor* and *Text2Onto* are configurable. *Text2Onto* provides several algorithms, targeting extraction of concepts, instances, relations; users can select an algorithm or apply a strategy to combine different results. *TermExtractor* proposes several parameters like the maximum number of terms in a compound term, different filtering thresholds and allows to measure and to use the position and the emphasis of the words in the textual analysis.

The sixth criterion is the **efficiency** (6), which measures the convergence speed. *Text2Onto*, *SPRAT* and *TermRaider* seem to be very efficient tools: in our tests (Section 3.3), results have been obtained in less than one minute. *TermExtractor*, *GlossExtractor* and *SSI* are only available through a web interface for submitting inputs and therefore tools are executed depending on server load. For this reason, it is very difficult to precisely evaluate the tool efficiency. *WCL System* is not directly available but you can request authors to process your tests.

The **reliability** (7) criterion reviews if the output remains consistent over repeated tests with the same input data. All the tools, excepting *GlossExtractor*, are reliable. Indeed, as *GlossExtractor* depends on web searches, consistency between results cannot be guaranteed.

Concerning the last criterion – **auxiliary tools and external resources** (8) that are used by each tool – *Text2Onto* requires *GATE*[3] and *TreeTagger*[4] as auxiliary tools and WordNet[5] as auxiliary resource. *SSI* also uses WordNet and *GlossExtractor* use Internet searches. As *SSI*, *TermExtractor* and *GlossExtractor* are already installed on Web servers and ready for use, no auxiliary tool is required. To test

[3] http://gate.ac.uk/, version 4.0.

[4] http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger

[5] http://wordnet.princeton.edu/, version 2.0.

TermRaider and *SPRAT*, which are available as Web services, a specific plugin[6] for *Neon Toolkit*[7] must be installed.

3.3 Tool Experimental Comparison

As defined in the introductory part of Section 3, the experimental comparison focuses on comparing tools by conducting a set of tests within the test-bed platform. According to the test-bed platform, tools outcomes are not only influenced by the implemented techniques but also by features of input texts (or input corpus). However, we do not discuss here in detail these features because requiring additional work. This additional long term work is discussed in the concluding section.

3.3.1 Experimental Settings

The set of tests has been organized as follows. The first phase focuses on concepts and instances, while the second phase focuses on taxonomic relations. Tests for ad-hoc relations have also been performed in a third phase but not presented in this paper because outcomes have been largely meaningless. We have compared the relevant tools outcomes with a common reference ontology (O_{mb}), manually built by following the tasks of Methontology (Fig. 3).

In order to take into account the imperfections of this common reference ontology, we have introduced an adjustment process: the expert (who has manually built the ontology) can validate further ontology elements (concepts, instances and taxonomic relations) that are comprised in one of the automatically built ontologies but are not part of O_{mb}.

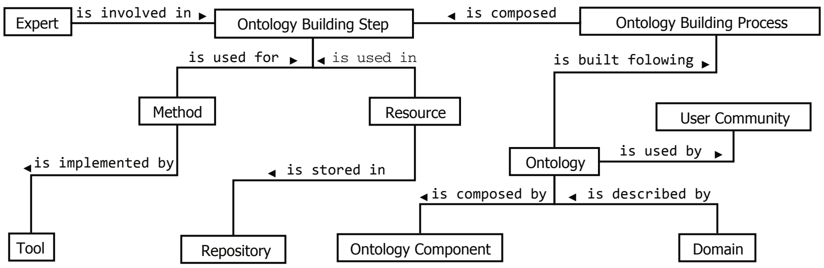

Fig. 3 The core concepts of the manually built ontology and their main relations

The validation of ontology elements by the expert is based on an adapted version of the framework ([Volker and Sure, 2006]) developed to evaluate the results of *Text2Onto*. In the adapted version, to each extracted ontology element the expert

[6] http://neon-toolkit.org/wiki/Gate_Webservice, version 1.1.15.

[7] http://neon-toolkit.org

assigned a score between 1 (fully invalid) to 4 (fully valid) according to the following scale:

- Concepts / Instances
  - 1 - terms that are not concepts or instances (e.g. *non*, *cannot*, *one*, *creating*)
  - 2 - terms that are more like instances than concepts / concepts than instances (e.g. *OntoLearn*, *EuroWordNet project* / *concept*, *ontology*)
  - 3 - terms that identify concepts / instances irrelevant for our domain (e.g. *european project*, *research institution* / *Alfonseca*, *Vossen*)
  - 4 - terms that identify concepts / instances relevant for our domain (e.g. *concept*, *ontology* / *OntoLearn*, *WordNet*)
- Taxonomic relations
  - 1 - fully incorrect: the relation is not correct, or one of the terms it relies is not a concept or an instance (e.g. *term* is a *figure*)
  - 2 - correct to some extent: the two terms/concepts are related but the relation is a true taxonomic relation only in restricted contexts (e.g. *task* is a *project*)
  - 3 - correct: a true taxonomic relation where at least a concept is, referring to the domain, too general or specific (e.g. *tool* is a *object*)
  - 4 - fully valid: a correct relation which relies two domain relevant concepts (e.g. *linguistic processor* is a *tool*)

The validation process has allowed us to account that one tool can provide additional ontology elements that the expert did not include in he first version of O_{mb} but to which he assigned a score of 4. The union ontology (O_u), part of the test-bed platform, is constructed by extending O_{mb} with all these additional valid ontology elements.

As indicated in the test-bed, this adjustment process led us to compare the automatic outputs not only with O_{mb} but also with O_u. Accordingly, we calculated two values for precision and recall: $Precision_m$, $Precision_u$ and $Recall_m$, $Recall_u$. The first value corresponds to the comparison with O_{mb} and the second value to the comparison with O_u. These two measures are computed as following for each type of ontology elements:

$$Precision_m = \frac{|EE \cap O_{mb}|}{|EE|} \qquad Recall_m = \frac{|EE \cap O_{mb}|}{|O_{mb}|}$$

$$Precision_u = \frac{|EE \cap O_u|}{|EE|} \qquad Recall_u = \frac{|EE \cap O_u|}{|O_u|}$$

where $EE = the\, set\, of\, Extracted\, Elements \qquad |X| = the\, number\, of\, elements\, of\, X$

Two other measures of the test-bed platform allows us to compare the results of the different tools with O_{mb}: (1) the *term specificity* and (2) the *semantic richness* of a taxonomic relation.

The first measure, *term specificity*, is based on the idea that complex terms composed of two, three or even more words are more likely to correspond to specific domain concepts than simple terms (one word terms). In our analysis, we consider three levels of specificity, corresponding to the number of words in one term: one word, two words and three or more words.

The second measure, the *semantic richness*, is based on the idea that a taxonomic relation linking a compound term with a simpler term lexically included in the former, is less semantically rich than a taxonomic relation linking two terms that are not lexically included one in the other one. Two values are possible for this measure: semantically rich (e.g. *ontology* is a *conceptualization*, *concept tree* is a *hierarchy*) and semantically poor (e.g. *concept tree* is a *tree*).

The set of tests is performed on a medium-size corpus (about 4000 words) that covers the domain of *ontology construction from texts*. This corpus is composed of just one document – a shrunken version of a scientific paper by Navigli and Velardi titled: *"Learning Domain Ontologies from Document Warehouses and Dedicated Web Sites"* [Navigli and Velardi, 2004]. In fact, as most of the tools take as input plain text files, we have eliminated from the original document all the images, diagrams, tables, mathematical formulas, examples, references, etc. We have finally obtained a corpus that is very *dense* (i.e. contains a high number of concepts when compared to the number of words in the text) (see Table 3), instantiating the *Class EntryText* within the test-bed platform.

In the set of tests, we have tested *TermExtractor*, *Text2Onto* and *SPRAT*. We have excluded *SSI* (the tool proposed by OntoLearn) and *TermRaider* (the tool proposed by SPRAT) from our tests. Indeed, the result of *SSI* is a semantic net (providing the meaning of the identified concepts) that is not exploitable by hands. The required input of *TermRaider* is a corpus containing at least two distinct files and the results are very dependent on the content of the two files so that cutting one file in two (or several) distinct file generates each time very distinct results. In addition, the results of *TermRaider* are not systematically used by the other modules embedded in *SPRAT* ([Gherasim et al., 2011]).

We have decided to keep any default configuration of tools, whenever tools enable various possible configurations (i.e. *Text2Onto* and *TermExtractor*).

This choice instantiates within in the test-bed platform, the *Class Configuration* – part of the *Approach* package. When *Text2Onto* is tested using its default configuration all the proposed extraction algorithms are executed and the final result is the union of results of each algorithm. In the case of TermExtractor we kept the default values for all its parameters.

Moreover, the set of tests showed that *SPRAT* was not well-adapted to the experimental protocol. Consequently, in our current work, as reported in the paper, we propose additional tests for *SPRAT* (Section 3.3.4).

We have also tested *WCL System* on a subset of valid terms extracted by *TermExtractor* (197 terms, see Table 5). As *WCL System* is not publicly available (Section 3.2), we have asked the authors of OntoLearn to test it for us.

According to OntoLearn approach, *WCL System* works on term definitions, provided in some ways or identified by using *GlossExtractor*. For efficiently performing tests, the expert selected a representative subset of them (36 terms). The choice of these 36 terms was basically subjective but trying to keeping a similar distribution of lengths, as for the extracted terms by *TermExtractor* (see Table 4).

3.3.2 Analysis of Test Results: Concepts and Instances

In this section, we present the results of tests performed on the three tools that identify terms corresponding to concepts and instances. *Text2Onto* and *SPRAT* are supposed to identify, in separate lists, concepts and instances. *TermExtractor* identifies only concepts. With the given corpus, *SPRAT* results in a very limited ontology, containing only 9 concepts, and no instances (indeed as said above, larger tests have been performed on *SPRAT* as explained in Section 3.3.4). However, in order to preserve the uniformity of our comparisons, we have kept the results of *SPRAT* in the various analysis tables presented in the remainder.

Table 3 shows the results obtained through the validation of tools outcomes by the expert. We observe that 77% of the concepts extracted by *TermExtractor* have been evaluated as concepts relevant for domain and only 61% of the concepts, and respectively 21% of the instances extracted by *Text2Onto* have been evaluated as relevant for the domain.

Table 3 Automatically extracted concepts and instances: the results of the expert validation

Concepts			
Score	*TermExtractor**	*Text2Onto*	*SPRAT*
All	253	444	9
1	22	30	3
2	4	2	3
3	30	138	0
4	197 (77%)	274 (61%)	3 (33%)

Instances		
Score	*Text2Onto*	*SPRAT*
All	94	0
1	13	0
2	10	0
3	51	0
4	20 (21%)	0 (0%)

* *TermExtractor* extracts only concepts.

Table 4 compares, using the *term specificity* measure, the results of tools to O_{mb}. It can be easily seen that *Text2Onto* results in simpler terms than in the case of *TermExtractor* (simpler terms comprise less words).

Table 5 provides the double comparison of tools outcomes with O_{mb} and O_u. *TermExtractor* obtains a high quality precision: 59% and respectively 78% of the extracted terms correspond to valid concepts of O_{mb}, and respectively of O_u. Referring to O_u, *Text2Onto* has highest recall (47%), but when referring to O_{mb} it has almost the same recall as *TermExtractor* i.e.: 35% vs. 36% respectively. These figures on the medium-size corpus confirm the ones obtained on a smaller corpus ([Gherasim et al., 2011]). However, it is interesting to note that the differences between *TermExtractor* and *Text2Onto* (on precision, recall and term specificity) are smaller when working with a medium-size corpus.

Table 4 Automatically extracted concepts: comparison with O_{mb} based on the *term specificity* measure

Number of terms	O_{mb}	*TermExtractor*	*Text2Onto*	*SPRAT*
All	417	253	444	9
One word	81 (19%)	41 (17%)	311 (70%)	3 (33%)
Two words	220 (53%)	170 (66%)	116 (26%)	3 (33%)
Three or more words	116 (28%)	42 (17%)	17 (4%)	3 (33%)

Table 5 Automatically extracted concepts and instances: comparison with O_{mb} and O_u

	O_{mb}	*TermExtractor*	*Text2Onto*	*SPRAT*	O_u
Extracted concepts	–	253	444	9	–
Valid concepts	417	197	274	3	581
$\cap O_{mb}$	–	149	146	3	417
$Precision_m$	–	59%	33%	30%	–
$Precision_u$	–	78%	62%	30%	–
$Recall_m$	–	36%	35%	0.7%	–
$Recall_u$	69%	34%	47%	0.5%	–
Extracted instances	–	–	94	0	–
Valid instances	38	–	20	0	40
$\cap O_{mb}$	–	–	18	0	38
$Precision_m$	–	–	19%	0	–
$Precision_u$	–	–	21%	0	–
$Recall_m$	–	–	47%	0	–
$Recall_u$	95%	–	50%	0	–

3.3.3 Analysis of Test Results: Taxonomic Relations

In this section, we present the results of tests performed on *Text2Onto*, *SPRAT* and *WCL System* to identify taxonomic relations.

As explained in Section 3.2, these three tools do not work with the same type of input and external resources. *Text2Onto* takes as input the corpus and use WordNet as external resource. *SPRAT* takes as input the corpus and do not use any external resource. *WCL System* takes as input a *list of definitions* corresponding to terms identified by *TermExtractor*. In our tests, this *list of definitions* has been constructed by *GlossExtractor* which use Internet searches as external resource(s).

Table 6 provides the results of the expert validation of the taxonomic relations extracted by *Text2Onto* and *SPRAT*. *Text2Onto* identified 362 taxonomic relations where 111 have been evaluated as fully valid. These valid relations are between 150 concepts, out of 444 concepts already extracted and also out of 274 concepts already validated (see Table 5). *SPRAT* results are very limited: only 5 taxonomic relations are identified, and none of them is fully valid. Consequently, next further analysis presented in this section concerns only *Text2Onto* and *WCL System*.

Table 7 provides the double comparison, based on precision and recall, of *Text2Onto* results with O_{mb} and O_u.

Table 6 Automatically extracted taxonomic relations: the results of the expert validation

Score	*Text2Onto*	*SPRAT*
All	362	5
1	78	2
2	69	3
3	104	0
4	111 (30%)	0 (0%)

Table 7 Automatically extracted taxonomic relations: comparison with O_{mb} and O_u

	O_{mb}	*Text2Onto*	O_u
Extracted relations	–	362	–
Valid relations	407	111	473
$\cap O_{mb}$	–	45	407
$Precision_m$	–	41%	–
$Precision_u$	–	31%	–
$Recall_m$	–	10%	–
$Recall_u$	86%	23%	–

Because *WCL System* has been tested on a subset of valid terms (Section 3.3.1) it is not possible to directly compare it with *Text2Onto*, O_{mb} and O_u by using standard precision and recall.

As test outcomes over the 36 terms used with *WCL System*, the tool has identified 278 taxonomic relations involving the 36 original terms but also 246 additional terms, found by using the definitions provided by *GlossExtractor*: 67 out of 246 belong to O_u. Furthermore, only 25 of these 278 taxonomic relations have been evaluated as fully valid. These 25 relations involve only 38 concepts belonging to O_u.

To perform a comparison standard precision and recall for relations have been slightly adapted: common concepts linked by valid relations (in the case of *Text2Onto* and *WCL System*) or relations (in the case of O_{mb} and O_u) are used in the measures as explained below. Indeed, a comparison performed as above, provides useful insights about the ability of each tool to find-out more or less relations between a set of common concepts as well.

We have observed that there are 21 concepts in common i.e. concepts common among the 38 concepts linked by the 25 valid taxonomic relations found by *WCL System*, the 150 concepts linked by 111 valid taxonomic relations found by *Text2Onto*, the 417 concepts of the 407 taxonomic relations of O_{mb} and the 462 concepts of the 473 taxonomic relations of O_u. Table 8 presents this suggest comparison based on common concepts.

Accordingly, adapation of precision and recall measures are computed as follow:

$$Precision_m = \frac{|Select(EE:CC) \cap O_{mb}|}{|Select(EE:CC)|} \quad Recall_m = \frac{|Select(EE:CC) \cap O_{mb}|}{|Select(O_{mb}:CC)|}$$

$$Precision_u = \frac{|Select(EE:CC) \cap O_u|}{|Select(EE:CC)|} \quad Recall_u = \frac{|Select(EE:CC) \cap O_u|}{|Select(O_u:CC)|}$$

where $CC = the\ subset\ of\ 21\ common\ concepts$
and $Select(A:B) = all\ A\ involving\ only\ concepts\ found\ in\ B$

Table 8 Automatically extracted taxonomic relations: comparison on a subset of 21 common concepts

	O_{mb}	*WCL System*	*Text2Onto*	O_u
(Extracted relations)	–	11	7	–
(Valid relations)	6	9	5	13
$\cap\ O_{mb}$	–	3	3	6
$\cap$ *WCL System*	3	–	3	9
$\cap$ *Text2Onto*	3	3	–	5
$Precision_m$	–	27%	43%	–
$Precision_u$	–	82%	71%	–
$Recall_m$	–	50%	50%	–
$Recall_u$	46%	69%	38%	–
Specific relations*	16%	38%	8%	–

*Relations identified by only one tool.

Table 8 clearly shows the complementarity that may exist between the tools and O_{mb}, and also between the tools. Each tool identifies valid taxonomic relations that are not identified by the expert (in Table 8, we named these relation 'Specific relations'). *WCL System* seems to be really remarkable because 38% of all the relations identified between the 21 common concepts are specific to it.

The results of *WCL System* and *Text2Onto* may also seem complementary when they are analyzed alongside the *semantic richness* measure. In fact, 76% of the 111 fully valid relations identified by *Text2Onto* are semantically poor relations, while 76% of the 25 fully valid relations identified by *WCL System* are semantically rich relations.

3.3.4 Testing SPRAT with a Different Strategy

As said in the previous sections, our experimentations have shown quite limited results for *SPRAT*. To go deeper into the analysis, we have further tested *SPRAT* by the following experimental process. As *SPRAT* can take an ontology as input and enrich this ontology with new concepts, we have tested *SPRAT* with a seed ontology (containing a selection of concepts of the manually built ontology) and on the corpus. This test has been repeated with different seed ontologies for verifying if the content of seed ontology interacts in any manner with the *SPRAT* ontology extraction process. Each time, the built ontology has been a merging of the seed ontology with the ontology automatically built by *SPRAT* directly on the corpus – i.e. without using any content of the seed ontology. This fact makes us to conclude that the seed ontology has no influence on the *SPRAT* ontology extraction process.

Finally, we have also tested *SPRAT* by progressively increasing the corpus size – from 4000 words to 27000 words. Each new corpus has been iteratively constructed by adding a new content to the previous corpus. The results are presented in Table 9. The percentage of domain concepts increases with the size of the corpus. But, the built ontology stays quite flat – with a maximal depth of 2 for all the tests. Moreover, neither instance nor relation – except taxonomic relations – have been identified.

Table 9 The main characteristics of ontologies construted by SPRAT when the size of the texts has gradually increased

Score \ Corpus size	4000*	9000*	15000*	19000*	27000*	8000*
All	9	14	20	27	49	27
1	3	3	4	4	6	5
2	3	3	3	3	3	1
3	0	4	7	10	9	1
4	3 (30%)	4 (29%)	6 (30%)	10 (37%)	31 (63%)	20 (74%)

*The number of words in the corpus.

We have noted much better recall and precision for *SPRAT* when passing from 19000 words corpus to 27000 words corpus. This fact conducted us to test *SPRAT* on a corpus containing only the 8000 words text that has been added to the 19000 words corpus for obtaining the 27000 words corpus. *SPRAT* obtained outstanding results on this 8000 words corpus: it extracted the same number of concepts than when it has been tested on the 19000 words corpus but with better precision (74% compared to 37%).

This performance improvement is explained by the fact that the 8000 words corpus contains more expressions that match *SPRAT* predefined patterns (see Table 2) than the 19000 words corpus. When tested on the 8000 words corpus *SPRAT* extracted terms/concepts that are part of expressions containing keywords like *'such as'* or *'is kind of'*).

3.4 Discussion

Our new experimentations at a medium scale confirm that the different tools are able to scale for medium-size text corpus. Moreover, they show that *SPRAT* significantly improves its results when it takes as input medium text or text containing expressions that match specific patterns. However, *SPRAT* results seem to be of lower quality than those provided by *Text2Onto* and *TermExtractor* + *WCL System*.

More precisely, concerning concept-instance, *Text2Onto* and *TermExtractor* continue to have very good results as already observer in [Gherasim et al., 2011] for smaller text size; specifically, *Text2Onto* has a better recall and *TermExtractor* a better precision but differences are smaller for medium-size corpus.

Together, *TermExtractor* and *Text2Onto* identified 388 terms considered fully valid by an expert independent evaluation and corresponding to 66% of the concepts of a union ontology (the manually built ontology corresponds to 72% of the concepts of the union ontology) defined in Section 3.3.2. Additionally, 83% of the terms identified by *TermExtractor* are complex terms that are likely to correspond to specific domain concepts, while 70% of the terms identified by *Text2Onto* are simple terms underlying general concepts.

Concerning relations, we have restricted ourselves mainly to *WCL System* and *Text2Onto* as we have shown the results of *SPRAT* are very limited under the same test case. Both *WCL System* and *Text2Onto* identify taxonomic relations. Since *WCL System* is not yet released and we have tested it only on a reduced set of terms/concepts, it was difficult to compare its results with the results of *Text2Onto* and with the manually built ontology. However, we have identified a subset of common concepts between the concepts involved in the relations extracted by *WCL System*, the concepts involved in the relations extracted by *Text2Onto* and the concepts involved in taxonomic relations of the manually built ontology. A precise analysis of the relations between the concepts of this subset has underlined strong complementarities between the results of *WCL System* and *Text2Onto*, and between their results and the manually built ontology. As these tools implement very different techniques to extract taxonomic relations, these complementarities make sense to combine their results. In our experimentation on the subset of common concepts *WCL System* and *Text2Onto* have identified together 90% of the taxonomic relations that relate the 21 common concepts in the union ontology.

From a general point of view, the ontologies constructed by the analyzed tools contain instances, concepts and – in the best cases – taxonomic relations only. Despite the good precision of *TermExtractor*, the results are often incomplete but they can be used as a basis to help an expert in the ontology building process, or as additional resources to complete existing ontologies.

4 Conclusion

In this paper we have compared four approaches and related tools among the most common ones to automatically build ontologies from textual resources. Using

Methontology as a framework, we first have proposed a synthetic comparison of the tasks belonging to the four approaches. This synthetic comparison has highlighted a great variability of the used algorithms both in the concept extraction stages and in the relation establishment. Then, we have made an experimental comparison on corpus of about 4000 words.

Our analysis established a major difference between the approaches: some of them (Text2Onto, SPRAT) propose a fully automated ontology construction, while some others propose either separate tools for each task sometimes not fully integrated for ontology extraction (like OntoLearn) or just suggest tools taken from available prototypes and products (like Alvis).

From an experimental point of view, the tests confirm that tools can be used without any major problem on medium-size corpus. However, there are significant differences between those tools. Concerning concept-instance, *Text2Onto* has the best recall and *TermExtractor* the best precision, while *SPRAT* has a low recall. Concerning taxonomic relation, the relations identified by *WCL System* are semantically richer than the relation extracted by *Text2Onto*. Generally speaking, the obtained results allow us to say that *TermExtractor* + *WCL System* and *Text2Onto* seem to have a real potential for automating ontology construction. *SPRAT* is interesting but it needs text corpus which contains a significant number of expressions matching predefined patterns.

Tools automating ontology construction may speed up the ontology construction process (indeed they produce relevant concepts, instances and relationships) and this paper provides few elements for understanding how these tools can be selected and used for specific applications. However, several challenging points remain open. The first is about the relationship between tools performances (especially precision and recall), the implemented techniques and the input text features. A full development of this point should result in a decision support system or in a recommendation system supporting the selections. The second point concerns the integration between manual tasks/subtasks (Table 1) and fully automated tasks/subtasks. The third point is about the correction of automatically built ontologies: how errors can be identified, how ontologies can be improved and so on. Our further work is devoted to investigate these challenging points.

References

[Aime et al., 2009] Aime, X., Furst, F., Kuntz, P., Trichet, F.: Gradients de prototypicalité appliqués à la personnalisation d'ontologies. In: Actes de la Conférence Ingénierie des Connaissances (IC 2009), pp. 241–252 (2009)

[Bourigault and Lame, 2002] Bourigault, D., Lame, G.: Analyse distributionnelle et structuration de terminologie. application á la construction d'une ontologie documentaire du droit. Traitement Automatique des Langues 43(1), 129–150 (2002)

[Buitelaar et al., 2005] Buitelaar, P., Cimiano, P., Magnini, B.: Ontology learning from text: an overview. In: Ontology Learning from Text: Methods, Applications and Evaluation, pp. 3–12. IOS Press (2005)

[Cimiano and Volker, 2005] Cimiano, P., Völker, J.: Text2Onto - a Framework for Ontology Learning and Data-driven Change Discovery. In: Montoyo, A., Muńoz, R., Métais, E. (eds.) NLDB 2005. LNCS, vol. 3513, pp. 227–238. Springer, Heidelberg (2005)

[Corcho et al., 2005] Corcho, Ó., Fernández-López, M., Gómez-Pérez, A., López-Cima, A.: Building Legal Ontologies with METHONTOLOGY and WebODE. In: Benjamins, V.R., Casanovas, P., Breuker, J., Gangemi, A. (eds.) Law and the Semantic Web. LNCS (LNAI), vol. 3369, pp. 142–157. Springer, Heidelberg (2005)

[Fernandez et al., 1997] Fernández-López, M., Gómez-Pérez, A., Juristo, N.: Methontology: From ontological art towards ontological engineering. In: Proc. of the AAA 1997 Spring Symposium Series on Ontological Engineering, pp. 33–40 (1997)

[Gherasim et al., 2011] Gherasim, T., Harzallah, M., Berio, G., Kuntz, P.: Analyse comparative de méthodologies et d'outils de construction automatique d'ontologies á partir de ressources textuelles. In: EGC 2011 (2011)

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

[Harris, 1968] Harris, Z.: Mathematical Structures of Language. John Wiley and Son (1968)

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

[Maynard et al., 2009a] Maynard, D., Funk, A., Peters, W.: Nlp-based support for ontology lifecycle development. In: Proc. of ISWC Workshop on Collaborative Construction, Management and Linking of Ontologies (2009a)

[Maynard et al., 2009b] Maynard, D., Funk, A., Peters, W.: Sprat: a tool for automatic semantic pattern-based ontology population. In: Proc. of the Int. Conf. for Digital Libraries and the Semantic Web (2009b)

[Navigli and Velardi, 2004] Navigli, R., Velardi, P.: Learning domain ontologies from document warehouses and dedicated web sites. Computational Linguistics 30(2), 151–179 (2004)

[Navigli and Velardi, 2005] Navigli, R., Velardi, P.: Structural semantic interconnections: A knowledge-based approach to word sense disambiguation. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 27(7), 1075–1086 (2005)

[Navigli et al., 2003] Navigli, R., Velardi, P., Gangemi, A.: Ontology learning and its application to automated terminology translation. IEEE Intelligent Systems 18(1), 22–31 (2003)

[Nazarenko and Hamon, 2002] Nazarenko, A., Hamon, T.: Structuration de terminologie: quels outils pour quelles pratiques? Traitement Automatique des Langues. Structuration de Terminologie 43(1), 7–18 (2002)

[Nedellec, 2006] Nedellec, C.: Semantic class learning and syntactic resources tuning. Technical report, Deliv. 6.4a for ALVIS (Superpeer semantic Search Engine) Project (2006)

[Osborne et al., 2009] Osborne, J., Flatow, J., Holko, M., Lin, S., Kibbe, W., Zhu, L., Danila, M., Feng, G., Chisholm, R.L.: Annotating the human genome with disease ontology. BMC Genomics 10(supl.1), 63–68 (2009)

[Park et al., 2011] Park, J., Cho, W., Rho, S.: Evaluating ontology extraction tools using a comprehensive evaluation framework. Data Knowl. Eng. 69, 1043–1061 (2011)

[Salton et al., 1975] Salton, G., Yang, C., Yu, C.: A theory of term importance in automatic text analysis. Journal of the American Society for Information Science 26, 33–34 (1975)

[Velardi et al., 2007] Velardi, P., Cucchiarelli, A., Pétit, M.: A taxonomy learning method and its application to characterize a scientific web community. IEEE Trans. on Knowl. and Data Eng. 19(2), 180–191 (2007)

[Velardi et al., 2008] Velardi, P., Navigli, R., D'Amadio, P.: Mining the web to create specialized glossaries. IEEE Intelligent Systems 23(5), 18–25 (2008)

[Volker and Sure, 2006] Volker, J., Sure, Y.: Data-driven change discovery - evaluation. Technical report, Deliv. D3.3.2 for SEKT Project, Instit. AIFB, Univ. of Karlsruhe, SEKT Deliv. (2006)

[Zouaq and Nkambou, 2010] Zouaq, A., Nkambou, R.: A Survey of Domain Ontology Engineering: Methods and Tools. In: Nkambou, R., Bourdeau, J., Mizoguchi, R. (eds.) Advances in Intelligent Tutoring Systems. SCI, vol. 308, pp. 103–119. Springer, Heidelberg (2010)