TIN2009-11005

DAMASK

Data-Mining Algorithms with Semantic Knowledge

PROYECTO DE INVESTIGACIÓN
PROGRAMA NACIONAL DE INVESTIGACIÓN FUNDAMENTAL,
PLAN NACIONAL DE I+D+i 2008-2011
ÁREA TEMÁTICA DE GESTIÓN: Tecnologías informáticas

Deliverable D1

State-of-the-art on Information Extraction from Web resources

Authored by
David Sánchez, Universitat Rovira i Virgili
David Isern, Universitat Rovira i Virgili
Antonio Moreno, Universitat Rovira i Virgili

Document information

project name:	DAMASK	
Project reference:	TIN2009-11005	
type of document:	Deliverable	
file name:	D1.pdf	
version:	Final	
authored by:	D. Sánchez, D. Isern, A. Moreno	15/04/2010
co-authored by		
released by:	A.Moreno	21.04.2010
approved by:	Co-ordinator	A. Moreno

Document history

version	date	reason of modification
1.0	31.March.2010	A preliminary release of the manuscript includes the IE concept and two main families: ontology-based and ontology-driven IE algorithms.
2.0	20.April.2010	Revision of v1.0 plus addition of ontology-driven IE section
3.0	20.April.2010	Internal version, minor changes
4.0	21.April.2010	Some modifications on the sections of v1.0, bibliographic references linked to cites
5.0	21.April.2010	Final version

Table of Contents

1 Introduction

The first task of DAMASK, called T1 - *Semantic integration of the information available in heterogeneous Web resources*, includes two preliminary subtasks, tasks 1.1 and 1.2 (see Figure 1). The former task discusses all related works in information extraction distinguishing algorithms to extract structured, semi-structured and non structured resources. The latter task lists the types of data that can be extracted using the previous algorithms.

This document is the result of the task 1.1, and its aim is to make a state-of-the-art on Information Extraction techniques applied to Web resources.

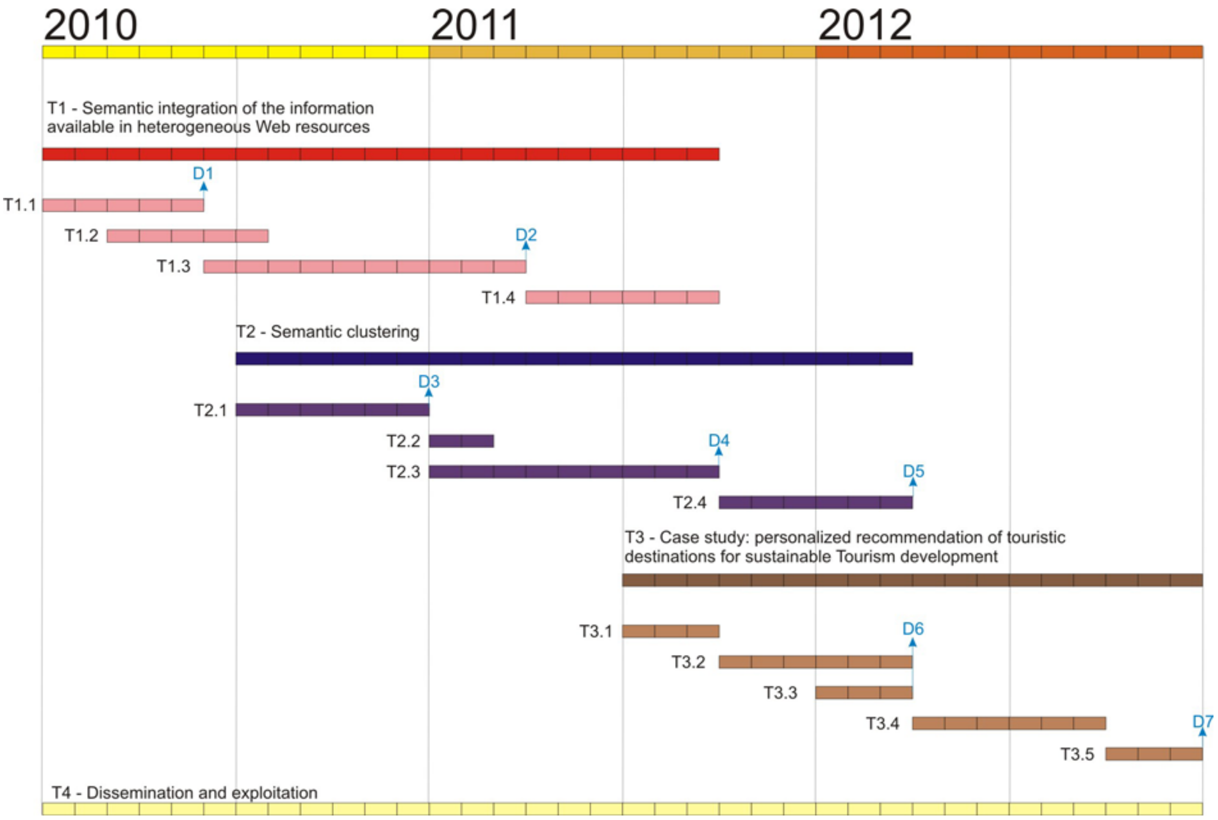

Figure 1: Tasks of DAMASK

2 Information Extraction

There has been an explosive growth in the amount of information available on networked computers around the world, much of it in the form of natural language documents. *Information Extraction (IE)* is the task of locating specific pieces of data within a natural language document [53]. Moreover, the advent of the internet has given IE a particular commercial relevance.

IE is a process which takes unseen texts as input and produces fixed format, unambiguous data as output. At the core of an IE system is an *extractor*, which processes text; it overlooks irrelevant words and phrases and attempts to home in on entities and the relationships between them [19]. These data may be used directly for display to users, or may be stored in a database or spreadsheet for direct integration with a back-office system, or may be used for indexing purposes in search engine/Information Retrieval (IR) applications [53]. If we compare IE and IR, whereas IR simply finds texts and presents them to the user (as classic search engines), IE analyses texts and presents *only* the specific information extracted from the text that is of interest to a user.

In the context of Web resources, a set of *extraction rules* suitable to extract information from a Web site is called a *wrapper* [23]. Two main approaches for wrapper generation tools have been proposed during the last years: one is based on *knowledge engineering* –supervised, traditional IE– and the other on *automatic training* –unsupervised, open IE–. In the first, the domain expert has to manually design the extraction rules or tag some documents, which are used by an algorithm to obtain the appropriate extraction rules. In such an approach the user skills play a crucial role in the successful identification and analysis of relevant information. In the second, open IE exploits AI techniques to induce extraction rules starting from a set of generic information patterns. In Table 1, as stated in [10] the main advantages and disadvantages of both approaches are summarised.

TABLE 1
COMPARISON OF TRADITIONAL IE AND OPEN IE

	Traditional IE	*Open IE*
Input	Corpus + Labelled Data	Corpus + Domain-Independent Methods
Relations	Specified in advance	Discovered automatically
Complexity	*O* (*D* * *R*) *D* documents, *R* relations	*O* (*D*) *D* documents
Precision	Very precise (hand-coded rules)	Reasonable precision (rule induction)
Training	Expensive development & test cycle	Provide training data (expensive)
Patterns	Need to develop grammars	No need for developing grammars

2.1 Traditional IE systems

Traditional methods on IE have focused on the use of supervised learning techniques such as hidden Markov models [25, 45], self-supervised methods [20], rule learning [46], and conditional random fields [38]. These techniques learn a language model or a set of rules from a set of hand-tagged training documents and then apply the model or rules to new texts. Models learned in this manner are effective on documents similar to the set of training documents, but extract quite poorly when applied to documents with a different genre or style. As a result, this approach has difficulty scaling to the Web due to the diversity of text styles and genres on the Web and the prohibitive cost of creating an equally diverse set of hand tagged documents.

The most representative example of this kind of systems is KnowItAll [20]. The KnowItAll Web IE system took the next step in automating IE by learning to label its own training examples using only a small set of domain-independent extraction patterns. KnowItAll was the first published system to carry out extraction from Web pages that was unsupervised, domain-independent, and large-scale. For a given relation, the set of generic patterns was used to automatically instantiate relation-specific extraction rules, which were then used to learn domain-specific extraction rules. The rules were applied to Web pages identified via search-engine queries, and the resulting extractions were assigned a probability using information-theoretic measures derived from search engine hit counts. Next, KnowItAll used frequency statistics computed by querying search engines to identify which instantiations were most likely to be bona fide members of the class. For instance, KnowItAll was able to confidently label China, France, and India as members of the class Country while correctly knowing that the existence of the sentence, “Garth Brooks is a country singer” did not provide sufficient evidence that “Garth Brooks” is the name of a country. KnowItAll is self-supervised; instead of utilizing hand-tagged training data, the system selects and labels its own training examples and iteratively bootstraps its learning process. KnowItAll is relation-specific in the sense that it requires a laborious bootstrapping process for each relation of interest, and the set of relations has to be named by the human user in advance. This is a significant obstacle to open-ended extraction because unanticipated concepts and relations are often encountered while processing text.

2.2 Open IE systems

While most IE work has focused on a small number of relations in specific preselected domains, certain corpora (*e.g.,* encyclopaedias, news stories, email, and the Web itself) are unlikely to be amenable to these methods [19]. Traditional IE requires pre-specifying a set of relations of interest and then providing training examples for each. Open Information Extraction (Open IE) [2] is relation-independent, and instead extracts all relations by learning a set of lexico-syntactic patterns.

The challenge of Web extraction led to the creation of the Open IE field, a novel extraction paradigm that tackles an unbounded number of relations, eschews domain-specific training data, and scales linearly (with low constant factor) to handle Web-scale corpora. For example, an Open IE system might operate in two phases. First, it would learn a general model of how relations are expressed in a particular language. Second, it could utilize this model as the basis of a relation-independent extractor whose sole input is a corpus and whose output is a set of extracted tuples that are instances of a potentially unbounded set of relations. Such an Open IE system would learn a general model of how relations are expressed (in a particular language), based on unlexicalized features such as part-of-speech tags (for example, the identification of a verb in the surrounding context) and domain-independent regular expressions (for example, the presence of capitalization and punctuation). When using the Web as a corpus, the relations of interest are not known prior to extraction, and their number is immense. Thus an Open IE system cannot rely on hand-labelled examples of each relation.

The most representative example of this kind of systems is TextRunner [2, 19]. TextRunner extracts high-quality information from sentences in a scalable and general manner. Instead of requiring relations to be specified in its input, TextRunner learns the relations, classes, and entities from its corpus using its relation-independent extraction model. TextRunner's first phase uses domain-specific examples that have been tagged. With this machine-learning approach, an IE system uses a domain-independent architecture and sentence analyzer. When the examples are fed to machine-learning methods, domain-specific extraction patterns can be automatically learned and used to extract facts from text. Rather than demand hand-tagged corpora, these systems required a user to specify relation-specific knowledge through a small set of seed instances known to satisfy the relation of interest, or a set of hand-constructed extraction patterns to begin the training process. For instance, by specifying the set Bolivia, city, Colombia, district, Nicaragua over a corpus in the terrorism domain, these IE systems learned patterns (for example, headquartered in <x>, to occupy <x>, and shot in <x>) that identified additional names of locations. Nevertheless, the amount of manual effort still scales linearly with the number of relations of interest, and these target relations must be specified in advance.

3 Ontologies and Information Extraction

IE's ultimate goal, which is the detection and extraction of relevant information from textual documents, depends on proper understanding of text resources. Rule-based IE systems are limited by the rigidity and ad-hoc nature of the manually composed extraction rules. As a result, they present a very limited semantic background.

The role of semantics in IE is often reduced to very shallow semantic labelling. Semantic analysis is considered more as a way to disambiguate syntactic steps than as a way to build a conceptual interpretation. Today, most of the IE systems that involve semantic analysis exploit the simplest part of the whole spectrum of domain and task knowledge, that is to say, named entities. However, the growing need for IE application to domains such as functional genomics that require more text understanding pushes towards more sophisticated semantic knowledge resources and thus towards ontologies viewed as conceptual models.

In recent years, ontologies have emerged as a new paradigm to model and formalize domain knowledge in a machine readable way. In [48] an ontology is defined as "a formal, explicit specification of a shared conceptualization". *Conceptualization* refers to an abstract model of some phenomenon in the world by having identified its relevant concepts. *Explicit* means that the type of concepts identified, and the constraints of their use, are explicitly defined. *Formal* refers to the fact that the ontology should be machine-readable. *Shared* reflects the notion that an ontology captures consensual knowledge, that is, not a personal view of the target phenomenon of some particular individual, but one accepted by a group.

Ontologies are designed for being used in applications that need to process the content of information, as well as to reason about it, instead of just presenting information to humans. They permit greater machine interpretability of content than that supported by XML, RDF and RDF Schema (RDF-S), by providing additional vocabulary along with a formal semantics. So, ontologies represent an ideal knowledge background in which to base text understanding and enable the extraction of relevant information. This may enable the development of more flexible and adaptive IE systems than those relying on manually composed extraction rules (both based on linguistic constructions or document structure).

In [55], it is argued that ontologies can assist both manually or semi-automatically constructed rule-based IE systems. On the one hand, the knowledge engineer can commit to the ontology, which would guarantee that the extraction rules are tailored to extract the kind of information represented in the ontology. On the other hand, an annotator can commit to the ontology and annotate only parts of text that are relevant from the ontology's point of view.

Global scale initiatives (*e.g.* the Semantic Web [5]) have brought the development of ontologies for many domains. Nowadays, thousands of domain ontologies are freely available through the Web [16] and big, detailed and consensued general-purpose ontologies (such as WordNet [22]) have been developed.

In this section, we study the ontological paradigm and its possibilities as a knowledge representation formalism, paying special care to modern ontological languages such as OWL. Then, we survey how ontologies have been applied in the process of IE from textual documents, specially focusing on domain independent approaches.

3.1 The ontological paradigm

From a formal point of view [9, 49] an ontology has been defined as:

$$O = (C, \leq_C, R, \sigma_R, \leq_R, A, \sigma_A, T)$$

, where

- *C,R,A* and *T* represent disjoint sets of *concepts, relations, attributes* and *data types.* Concepts are sets of real world entities with common features (such as different types of *diseases*, *treatments*, *actors*, etc.). Relations are binary associations between concepts. There exist inter-concept relations which are common to any domain (such as *hyponymy*, *meronymy*, etc.) and domain-dependant associations (*e.g.* an Actor *performs* an Action, a Disease *is treated with* a certain Treatment, etc.). Attributes represent quantitative and qualitative features of particular concepts (such as the *medical code* of a Disease, the *degree of contagiousness*, etc), which take values in a given scale defined by the *data type* (*e.g.* string, integer, etc.).
- $\leq_C$ represents a concept hierarchy or taxonomy for the set *C*. In this taxonomy, a concept c_1 is a *subclass*, *specialization* or *subsumed concept* of another concept c_2 *if and only if* every instance of c_1 is also and instance of c_2 (which represents its *superclass*, *generalization* or *subsumer*). Concepts are linked by means of transitive *is-a* relationships (e.g. if *respiratory disease is-a disorder* and *bronchitis is-a respiratory disease*, then it can be inferred that *bronchitis is-a disorder*). Multiple inheritance (*i.e.* the fact that a concept may have several hierarchical subsumers) is also supported (for example, *Leukaemia* may be both a subclass of *Cancer* and *Blood disorder*).
- $\leq_R$ represents a hierarchy of relations (e.g. *has primary cause* may be a specialization of the relation *has cause*, which indicates the origin of a Disorder).
- $\sigma_R: R \rightarrow C^+$ refers to the signatures of the relations, defining which concepts are involved in one specific relation of the set R. For example, the signature *σ(is treated with): is treated with -> {Disease, Treatment}* indicates that *is_treated_with* establishes a relation between the two concepts *Disease* and *Treatment*. It is worth to note that some of the concepts in C+ correspond to the *domain* (the origin of the relation) and the rest to the *range* (the destination of the relation). In this example, *Disease* is the domain of the relation *is_treated_with*, and *Treatment* is the range. Those relationships may fulfil properties such as *symmetry* or *transitivity*.
- $\sigma_A: A \rightarrow CxT$ represents the signature describing an attribute of a certain concept *C*, which takes values of a certain data type *T* (e.g. the *number* of the *leukocytes* attribute of the concept *Blood Analysis*, which must be an integer value).

Optionally, an ontology can be populated by instantiating concepts with real world entities (*e.g. Saint John's* is an instance of the concept Hospital). Those are called instances or individuals.

By default, concepts may represent *overlapping* sets of real entities (*i.e.* an individual may be an instance of several concepts, for example a concrete disease may be both a *Disorder* and a *Cause* of another pathology). If necessary, ontology languages permit to specify that two or more concepts are *disjoint* (i.e. individuals cannot be instances of more than one of those concepts).

Some standard languages have been designed to construct ontologies. They are usually declarative languages based on either first-order logic or on description logics. Some examples of such ontology languages are KIF, RDF, KL-ONE, DAML+OIL and OWL [28]. There are some differences between them according to their supported degree of expressiveness. In particular, OWL is the most complete one, allowing to define, in its more expressive forms (OWL-DL and OWL-Full) logical axioms representing restrictions at a class level. They are expressed with a logical language and contribute to define the meaning of the concepts, by means of specifying limitations regarding the concepts to which a given one can be related to. Several restriction types can be defined:

- *Cardinality:* defines that a concept's individual can be related (by means of a concrete relation type) to a *minimum*, *maximum* or *exact* number of other concept's instances. For example, certain types of Disease may have *at minimum* one Symptom.
- *Universality*: indicates that a concept has a local range restriction associated with it (*i.e.* only a given set of concepts can be the range of the relation). For example, all the Symptoms of a certain Disease must be of the same type, the same concept category.
- *Existence*: indicates that at least one concept must be the range of a relation. For example a Disease *always* presents a certain kind of Symptoms, even though other ones may also appear.

All those restrictions can be defined as *Necessary* (*i.e.* an individual should fulfil the restriction in order to be an instance of a particular class) or *Necessary and Sufficient* (*i.e.* in addition to the previous statement, an individual fulfilling the restriction is, by definition, and instance of that class). This is very useful for implementing reasoning mechanisms when dealing with unknown individuals.

In addition, OWL also permits to represent more complex restrictions by combining several axioms using standard logical operators (AND, OR, NOT, etc.). In this manner, it is could be possible to define, for example, a set of Symptoms which co-occur for a particular Disease using the AND operator.

3.2 Ontology exploitation for IE

IE and ontologies are involved in two main and related tasks [41]:

- Ontology is used for Information Extraction: IE needs ontologies as part of the understanding process for extracting the relevant information;
- Information Extraction is used for populating and enhancing the ontology: texts are useful sources of knowledge to design and enrich ontologies.

These two tasks can be combined in a cyclic process: ontologies are used for interpreting the text at the right level for IE and IE extracts new knowledge from text, to be integrated in the ontology.

An ontology identifies the entities that exist in a given domain and specifies their essential properties. It does not describe the spurious properties of these entities. On the contrary, the goal of IE is to extract factual knowledge to instantiate one or several predefined forms. The structure of the form is a matter of the ontology whereas the values of the filled template usually reflect factual knowledge that is not part of the ontology.

Whether one wants to use ontological knowledge to interpret natural language or to exploit written documents to create or update ontologies, in any case, the ontology has to be connected to linguistic phenomena. A large effort has been devoted in traditional IE systems based on local analysis to the definitions of extraction rules that achieve this anchoring. In more powerful IE systems, the ontological knowledge is more explicitly stated in the rules that bridge the gap between the word level and text interpretation. As such, an ontology is not a purely conceptual model, it is a model associated to a domain-specific vocabulary and grammar. In the IE framework, we consider that this vocabulary and grammar are part of the ontology, even when they are embodied in extraction rules.

The complexity of the linguistic anchoring of ontological knowledge is well known. A concept can be expressed by different terms and many words are ambiguous. Rhetoric, such as lexicalized metonymies or elisions, introduces conceptual shortcuts at the linguistic level and must be elicited to be interpreted into domain knowledge. These phenomena, which illustrate the gap between the linguistic and the ontological levels, strongly affect IE performance. This explains why IE rules are so difficult to design.

IE does not require a whole formal ontological system but parts of it only. The ontological knowledge involved in IE can be viewed as a set of interconnected and concept-centered descriptions, or “conceptual nodes”. In conceptual nodes the concept properties and the relations between concepts are explicit. These conceptual nodes should be understood as chunks of a global knowledge model of the domain.

In general, the template or form to be fulfilled by IE is a partial model of world knowledge. IE forms are also classically viewed as a model of a database to be filled by the instances extracted. In [41] different levels of ontological knowledge are distinguished:

- The referential domain entities and their variations are listed in “flat ontologies”. This is mainly used for entity identification and semantic tagging of character strings in documents.
- At a second level, the conceptual hierarchy improves normalization by enabling more general levels of representation.
- More sophisticated IE systems also make use of chunks of a domain model (*i.e.* conceptual nodes), in which the properties and interrelations of entities are described. The projection of these relations on the text both improves the NL processes and guides the instantiation of conceptual frames, scenarios or database tuples. The corresponding rules are based either on lexicosyntactic patterns or on more semantic ones.
- The domain model itself is used for inference. It enables different structures to be merged and the implicit information to be brought to light.

In the following paragraphs those elements are discussed in more detail.

Sets of entities

Recognizing and classifying named entities in texts requires knowledge on the domain entities. Specialized lexical or keyword lists are commonly used to identify the referential entities in documents. Three main objectives of these specialized lexicons can be distinguished: semantic tagging, naming normalization and linguistic normalization.

- **Semantic tagging.** List of entities are used to tag the text entities with the relevant semantic information. In the ontology or lexicon, an entity (*e.g.* Tony Bridge) is described by its type (the semantic class to which it belongs, here PERSON) and by the list of the various textual forms (typographical variants, abbreviations, synonyms) that may refer to it3 (*Mr. Bridge, Tony Bridge, T. Bridge*). However, exact character strings are often not reliable enough for a precise entity identification and semantic tagging. Polysemic words belong to different semantic classes. In the above example, the string "Bridge" could also refer to a bridge named "Tony". The connection between the ontological and the textual levels must therefore be stronger. Identification and disambiguation contextual rules can be attached to named entities.

- **Naming normalization.** As a by-effect, these resources are also used for normalization purposes. For instance, the various forms of Mr. Bridge will be tagged as MAN and associated with its canonical name form: Tony Bridge (<PERSON id=Tony Bridge>). This avoids rule overfitting by enabling specific rules to be abstracted.
- **Linguistic normalization.** Beyond typographical normalization, the semantic tagging of entities contributes to sentence normalization at a linguistic level. It solves some syntactic ambiguities, *e.g.* if *cotA* is tagged as a *gene*, in the sentence "the stimulation of the expression of cotA", knowing that a gene can be "expressed" helps to understand that "cotA" is the patient of the expression rather than its agent or the agent of the stimulating action. Semantic tagging is also traditionally used for anaphora resolution.

Hierarchies

Beyond lists of entities, ontologies are often described as hierarchies of semantic or word classes. Traditionally, IE focuses on the use of word classes rather than on the use of the hierarchical organization. For instance, in WordNet [22], the word classes (*synsets*) are used for the semantic tagging and disambiguation of words but the hyponymy relation that structures the synsets into a hierarchy of semantic or conceptual classes is seldom exploited for ontological generalization inference. Some ML-based experiments have been done to exploit hierarchies of WordNet and of more specific lexicons, such as UMLS [24]. The ML systems learn extraction rules by generalizing from annotated training examples. They relax constraints along two axes, climbing the hyperonym path and dropping conditions. In this way, the difficult choice of the correct level in the hierarchy is left to the systems.

Conceptual nodes

The ontological knowledge is not always explicitly stated as it is in [26], which represents an ontology as a hierarchy of concepts, each concept being associated with an attribute-value structure, or in [17], which describes an ontology as a database relational schema. However, ontological knowledge is reflected by the target form that IE must fill and which represents *the conceptual nodes* to be instantiated. Extraction rules ensure the mapping between a conceptual node and the potentially various linguistic phrasings expressing the relevant elements of information.

The main difficulty arises from the complexity of the text representation once enriched by the multiple linguistic and conceptual levels. The more expressive the representation, the larger is the search space for the IE rule and the more difficult the learning. The extreme alternative consists in either selecting the potentially relevant features before learning, with the risk of excluding the solution from the search space, or leaving the system the entire choice, provided that there is enough representative and annotated data to find the relevant regularities. For instance, the former consists in normalizing by replacing names by category labels whereas the latter consists in tagging without removing the names. The learning complexity can even be increased when the conceptual or semantic classes are learned together with the conceptual node information [54].

3.3 Extraction ontologies

Apart from the exploitation of domain ontologies, recently, there have been proposals for pushing ontologies towards the action extraction process as immediate prior knowledge.

Extraction ontologies [18] define the concepts the instances of which are to be extracted, considering their attributes with their allowed values. Extraction ontologies are assumed to be hand-crafted based on observation of a sample of resources; they allow for rapid start of the actual extraction process, as even a very simple extraction ontology (designed by a competent person) is likely to cover a sensible part of target data and generate meaningful feedback for its own redesign. The clean and rich conceptual structure (allowing partial intra-domain reuse and providing immediate semantics to extracted data) makes extraction ontologies superior to ad-hoc hand-crafted patterns. However, many aspects of their usage still need to be explored.

[34] propose the design of extraction ontologies featuring:

1. The possibility to provide extraction evidence with probability estimates plus other quantitative info such as value distributions, allowing calculating the likelihood for every attribute and instancing candidate using pseudo-probabilistic inference.
2. The effort to combine hand-crafted extraction ontologies with other sources of information

Extraction ontologies are designed so as to extract occurrences of attributes (such as ‘speaker’ or ‘location’), i.e. standalone named entities or values, and occurrences of whole instances of classes (such as ‘seminar’), as groups of attributes that ‘belong together’, from HTML pages or texts in a domain of interest.

Attributes are identified by their name, equipped with a data type (string, long text, integer or float) and accompanied by various forms of extraction evidence relating to the attribute value or to the context it appears in. Attribute value evidence includes (1) textual value patterns; (2) for integer and float types: min/max values, a numeric value distribution and possibly units of measure; (3) value length in tokens: min/max length constraints or a length distribution; (4) axioms expressing more complex constraints on the value and (5) coreference resolution rules. Attribute context evidence includes (1) textual context patterns and (2) formatting constraints.

Extraction patterns (for both the value and the context of an attribute or class) are nested regular patterns defined at the level of tokens (words), characters, formatting tags (HTML) and labels provided by external tools. Patterns may be inlined in the extraction ontology or sourced from (possibly large) external files, and may include e.g. fixed lexical tokens, token wildcards, character-level regexps, formatting tags, labels representing the output of external NLP tools or references to other patterns or attribute candidates. For numeric types, default value patterns for integer/float numbers are provided.

For both attribute and class definitions, axioms can be specified that impose constraints on attribute value(s). For a single attribute, the axiom checks the to-be-extracted value and is either satisfied or not (which may boost or suppress the attribute candidate’s score). For a class, each axiom may refer to all attribute values present in the partially or fully parsed instance. For example, a start time of a seminar must be before the end time. Arbitrarily complex axioms can be authored using JavaScript. Further attribute level evidence includes formatting constraints (such as not allowing the attribute value to cross an HTML element) and coreference resolution scripts.

Each class definition enumerates the attributes which may belong to it, and for each attribute it defines a cardinality range. Extraction knowledge may address both the content and the context of a class. Class content patterns are analogous to the attribute value patterns, however, they may match parts of an instance and must contain at least one reference to a member attribute. Class content patterns may be used e.g. to describe common wordings used between attributes or just to specify attribute ordering. For each attribute, the engagedness parameter may be specified to estimate the *a priori* probability of the attribute joining a class instance (as opposed to standalone occurrence). Regarding class context, analogous class context patterns and similar formatting constraints as for attributes are in effect.

In addition, constraints can be specified that hold over the whole sequence of extracted objects. Currently supported are minimal and maximal instance counts to be extracted from a document for each class.

All the types of extraction knowledge mentioned above are pieces of evidence indicating the presence (or absence) of a certain attribute or class instance. Every piece of evidence may be equipped with two probability estimates: precision and recall. The precision of evidence states how probable it is for the predicted attribute or class instance to occur given the evidence holds, disregarding the truth values of other evidence. The recall of evidence states how abundant the evidence is among the predicted objects, disregarding whether other evidence holds.

3.4 Ontology-based Information Extraction

We consider *ontology-based IE systems* as those approaches relying on predefined ontologies in one or several stages of the extraction process. Those approaches are document driven: they start from a particular document (or set of documents) and they try to identify entities found in that context, trying to annotate them according to the input ontology. So, on the contrary to plain IE systems, ontology-based ones are able to specify their output in terms of a pre-existing formal ontology. These systems almost always use a domain-specific ontology in their operation, but we consider a system to be domain-independent if it can operate without modification on ontologies covering a wide range of domains.

So, the problem is very similar to semantic annotation. Annotations represent a specific sort of metadata that provides references between entities appearing in resources and domain concepts modelled in an ontology. Semantic annotation is one fundamental pillar of the Semantic Web [4] making it possible for Web-based tools to understand and satisfy the requests of people and machines to exploit Web content.

In this section we refer to both semantic annotation and ontology-based IE indistinctly.

In the last years, several attempts have been made to address the annotation of textual Web content. From the manual point-of-view, several tools have been developed to assist the user in the annotation process such as Annotea [33], CREAM [29], NOMOS [42] or Vannotea [44]. Those systems rely on the skills and will of a community of users to detect and tag entities within Web content. Considering that there are 1 trillion of unique Web pages on the Web (see The Official Google Blog, http://googleblog.blogspot.com/2008/07/we-knew-web-was-big.html, last access on March 30th, 2010) , it is easy to envisage the unfeasibility of manual annotation of Web resources.

Recently, some authors have focused on addressing the annotation problem by automating some of its stages. As a result, some tools such as Melita [13] have been developed. It is based on user-defined rules and previous annotations to suggest new annotations in text. Manually constructed rules are used also in other basic approaches to extract known patterns for annotations [3]. Another preliminary work proposing semi-automating the annotation of Web resources is the work described in [32]. The authors propose the combination of patterns (*e.g.,* addressed to extract objects such as email addresses, phone numbers, dates and prices) to tag the candidates to annotate, and then, this set is annotated by means of a domain conceptual model. That model represents the information of a particular domain through concepts, relationships and attributes (in an entity-relation based syntax). Supervised systems also use extraction rules obtained from a set of pre-tagged data [8, 43]. WebKB [7] and Armadillo [1] use supervised techniques to extract information from computer science websites. Likewise, S-CREAM [14] uses machine learning techniques to annotate a particular document with respect to its ontology, given a set of annotated examples.

Supervised attempts are certainly difficult to apply due to the bottleneck introduced by the interaction of a domain expert and the great effort required for compiling a large and representative training set.

SmartWeb [6] resolves the issue of not having pre-existing mark-up to learn from by using class and subclass names from a previously defined ontology. Those are used as examples to learn contexts. In this way, instances can be identified, as they present similar contexts.

Complete automatic and unsupervised systems are rare. SemTag [15] performs automated semantic tagging from large corpora based on the Seeker platform for text analysis and tagging large number of pages with the terms included in a domain ontology named TAP. This ontology contains lexical and taxonomic information about music, movies, sports, health, and other issues, and SemTag detects the occurrence of these entities in Web pages. It disambiguates using neighbour tokens and corpus statistics, picking the best label for a token. KIM [31] is another example of unsupervised domain-independent system. It scans documents looking for entities corresponding to instances in its input ontology.

Another interesting annotation application is presented in [40]. In this case, authors use a reference set of elements (*e.g.*, online collections containing structured data about cars, comics or general facts) to annotate ungrammatical sources like texts contained in posts. First of all, the elements of those posts are evaluated using the TF-IDF metric. Then, the most promising tokens are matched with the reference set. In both cases, limitations may be introduced by the availability and coverage of the background knowledge (*i.e.,* ontology or reference sets). From the applicability point-of-view, Pankow [11] is the most promising system. It uses a range of well-studied syntactic patterns to mark-up candidate phrases in Web pages without having to manually produce an initial set of marked-up Web pages, and without depending on previous knowledge. The context driven version, C-Pankow [12], improves the first by reducing the number of queries to the search engine. However, the final association between text entities and a possible domain ontology is not addressed.

There exist other systems which present a more ad-hoc design and are focused on a specific domain of knowledge, exploiting predefined and expected corpus structures, rules and domain knowledge. In [36] an IE system focused on the Tourism domain is proposed. They combine lexical knowledge, extraction rules and ontologies in order to extract information in the form of instantiated concepts and attributes that are stored in an ontology-like fashion (*e.g.* hotel names, number of rooms, prices, etc.). The most interesting feature is the fact that the pre-defined knowledge structures are extended as a result of the IE extraction process allowing to improve and complete them. They use several ontology learning techniques already developed for the

OntoEdit system [47]. The process starts with a shallow IE model given as baseline. Then, a domain specific corpus is selected. The corpus is processed with the core IE system. Based on this data, one is able to use different learning approaches in a semi-supervised fashion embedded into the Ontology learning framework. As a result, the process is extended. The human expert has to validate each extension before continuing.

Feilmayr *et. al.* [21] propose an ontology-based IE system. They analyse the heterogeneities of individually maintained accommodation websites and discuss the IE techniques in the Tourism domain. As a result, they present a rule/ontology-based IE approach able to cope with the given heterogeneities. A domain-dependent crawler collects Web pages corresponding to accommodation websites. This corpus is passed to an extraction component based on the GATE framework [14] which provides a number of text engineering components. It performs an annotation of Web pages in the corpus, supported by a domain-dependent ontology and rules. Extracted tokens are ranked as a function of their frequency and relevancy for the domain.

Another domain-dependent system is SOBA [6], a sub-component of the SmartWeb (a multi-modal dialog system that derives answers from unstructured resources such as the Web), which automatically populates a knowledge base with information extracted from soccer match reports found on the Web. The extracted information is defined with respect to an underlying ontology. The SOBA system consists of a Web crawler, linguistic annotation components and a module for the transformation of linguistic annotation into an ontology-based representation. The first component enables the automatic creation of a soccer corpus, which is kept up-to-date on a daily basis. Text, images and semi-structured data are compiled. Linguistic annotation is based in finite-state techniques and unification-based algorithms. It implements basic grammars for the annotation of persons, locations, numerals and date and time expressions. On the top, rules for extraction of soccer-specific entities, such as actors in soccer, teams and tournaments are implemented. Finally, data is transformed into ontological facts, by means of tabular processing (wrapper-like techniques are applied) and text matching (by means of F-logic structures specified in a declarative form).

[35] proposes to use shallow natural language processing and domain-specific ontologies (applied to the manufacturing and vehicle domains) to automatically construct a structured representation from a set of unstructured documents. Concepts and relations are identified in the text by means of linguistic patterns. The result is stored in an ontology-like fashion. Apart from the basic linguistic analysis of text (tokenization, POS tagging and chunking), which results in the extraction of noun and verb phrases, the system maps them to the input ontology by simple word matching. Breadth first search is used to search for concepts in the domain ontology which match the extracted entities. Extracted noun phrases are compared against all the concepts in the domain ontology, whereas verb phrases are matches against a manufacturing taxonomy. In the case of multiple matchings, the one with the highest amount of matchings in the same sentence is selected.

The basic idea of the approach by Yildiz and Miksch [55] is to use the information on the input ontology to construct automatically a set of extraction rules to be used by the information extraction system. They look on the text for the words that appear in the name of the concepts, the name of the properties and the comment section of the concepts and attributes. For each appearance of one of these words, they apply rules (regular expressions related to the datatype of each property, as specified in the ontology) to the word’s neighbourhood to find appropriate values. For instance, if there is an ontology on digital cameras in which the Digital Camera class has an Optical Zoom property (of the float type), the system looks for the string “optical zoom” in the text and searches for a float numerical value near it.

3.5 Ontology-driven Information Extraction

The methods described in the previous section may be qualified as *document-driven*, since they analyze sequentially a given set of documents available in a corpus, trying to annotate the information of those documents with respect to the input ontology. A complementary approach, which can be qualified as *ontology-driven*, is commented in this section. The basic idea of the techniques in this category is to focus the processing on the ontology basic elements (classes, relations), leveraging this knowledge to find resources that can be analysed to obtain useful information (in most cases, instances of the ontology classes). As commented in [39], this kind of methods presents some benefits:

- Focusing on the ontology components seems a natural way to exploit all kinds of ontological data (*e.g.* using synonyms to broaden the search for documents to be analysed).
- These systems can consider a huge amount of different resources (*e.g.* the Web), and are not constrained by a limited corpus of documents.
- The systems concentrates all their resources on searching directly for information related to the ontology components, rather than having to analyse a potentially large number of documents that do not contain interesting information.

One of the most well-known examples of ontology-driven information extraction systems in OntoSyphon [39], a domain-independent and unsupervised system which focuses on finding instances of the classes of the input ontology. For each class of the ontology, the following steps are taken:

- Use a basic set of Hearst patterns [30] to generate lexico-syntact phrases that permit to obtain candidates to instances of the class. For example, for the Bird class, the patterns used would be "birds such as …", ·birds including …", "birds especially …", "… and other birds", "… or other birds".
- Use those phrases in a Web search engine (or in a simplified setting such as the Binding Engine [7]) to extract the candidate instances.
- Evaluate those candidates to assess which of them have a good chance of being instances of the class. The evaluation measures proposed in [39] depend basically on the number of patterns from which a given candidate has been obtained and the number of hits of each candidate (redundancy is taken as a signal that the candidate is probably good), although more complex evaluations based on the urn model and on variations of PMI [50] are also proposed.

The work on information extraction by Vicient [52] is also guided by the classes of an input ontology, although the set of Web pages to be analysed is fixed and no Web searches are performed. His methodology is domain-independent, but the work cantered the analysis in a Tourism ontology, which was manually constructed. The aim of this work, very much related to the objectives of the DAMASK project, was to generate a matrix in which each row corresponded to a destination city, each column was related to a class of the ontology, and each cell of the matrix showed the subclasses of the class on the column which denote elements that are present in the city on the row. For instance, if the row is London and the column is Religious-Building, the related cell would contain a list such as "Cathedral, Mosque, Synagogue, Abbey, Church", which are subclasses of Religious Building that are represented by real buildings in London. For each class of the ones considered in the matrix columns (selected by the user from the input ontology), the systems analyzes the Wikipedia pages related to the touristic destinations in the following way:

- All the subclasses of the class are recursively searched in the basic text of the page (*e.g.* "St.Paul's Cathedral" identifies an item of the Cathedral class, and "London Central Mosque" an instance of the Mosque class).
- The subclasses are also searched in the list of categories associated to the Wikipedia page.
- The text associated to each of the images of the page is also compared with the subclasses of the class.

The numerical attributes related to the CityClass are instantiated by analyzing the infoBox that appears at the beginning of the Wikipedia page. Although the work may be considered as a first step in the direction of the DAMASK objectives, it has to be noticed that the identification of the subclasses of each class within the Web pages is purely syntactical.

Van Hague *et al.* [51] present an ontology-driven domain-independent method that, although it is not focused precisely on Information Extraction but rather on Ontology Mapping, uses similar ideas. Their aim is to find a mapping between pairs of concepts belonging to two input ontologies. For each pair (C1, C2), where C1 is a class of the first ontology and C2 is a class of the second ontology, they perform the following tasks:

- Use a basic set of hyponymy-detector Hearst patterns (C1 such as C2, such C1 as C2, C1 including C2, etc).
- Send the patterns to a Web search engine, and collect the hit counts obtained in each case.
- Accept all hyponymy relations supported by a number of hits above a certain threshold.

Another approach for ontology-driven information extraction is given in [27]. In this work the aim is to find instances of the classes of the input ontology. The procedure follows these steps:

- Select one of the binary relations of the ontology and one instance corresponding to the domain or the range of the relation (for example, the relation "acts in" –between Actors and Movies- and an instance of Actor, "Sean Connery").
- The system contains a set of manually-constructed text patterns associated to the relation (in the same example, the relation "acts in" is associated to the pattern "[Movie] starring [Actor], [Actor] and [Actor]"). Take each pattern and apply it to the instance (*e.g.* "[Movie] starring Sean Connery, [Actor] and [Actor]").
- Send each of these instantiated patterns to a Web search engine, and collect candidates to instances of the classes appearing in the pattern (in the example, with the previous pattern we would obtain candidates to instances of the classes Movie and Actor).
- Check the correctness of each candidate, by sending to the Web search engine phrases expressing the instance-class relation (which are constructed semi-automatically) and accepting the instance candidate when the number of hits obtained exceeds a certain threshold.

A similar approach to ontology-driven population is reported by Matuszek *et al.* [37]. This work is framed in the Cyc project, the ambitious effort that has been going on for some decades to formalize all the world's commonsense knowledge. In particular, the authors have developed techniques for automatically finding instances of the components (domain, range) of the relations on the ontology. Their approach follows these steps:

- Choose a query that represents information that wants to be found out (e.g. the Prime Minister of a certain country). The authors have limited the search to 134 binary predicates.
- Translate the query into a search string. The system contains 233 manually created generation templates for the 134 chosen predicates.
- Send the query to a Web search engine, and detect the class instance candidates.
- A candidate is deemed as correct if it successfully passes three tests: it does not create any logical inconsistency with the knowledge already present in Cyc, a specifically generated search string containing the candidate and the class provides enough hits, and a human curator finally validates the candidate.

The main drawback of the last two methods is that they contain some steps that cannot be made automatically, and therefore they require a certain amount of manual work before they can be executed for a given domain ontology.

4 Summary

Information Extraction (IE) methods aim to find specific items of information within electronic resources (usually text documents), by applying some kind of extraction rules. These rules may be given by a domain expert, may be learnt from documents tagged by a domain expert, or may be learnt directly from the texts through the use of some generic information patterns. In the DAMASK project we are interested in this last option, as we want to develop an unsupervised IE framework.

The relation between ontologies and IE is twofold: on the one hand, the semantic knowledge given by a domain ontology may guide the IE process (as in the case of the DAMASK project) and, on the other hand, the IE results may help to improve or enrich an initial domain ontology.

In this document we have considered two different kinds of methods involving ontologies and IE. In the ontology-based (or document-driven) methods, each document of the corpus is analyzed sequentially, and the aim is to annotate each document by relating specific pieces of information to the concepts, instances and relations in the ontology. On the contrary, in the ontology-driven techniques the idea is to consider each of the ontological elements and to use them to search for resources (e.g. Web pages) that can provide interesting information related to each component of the ontology. Some initial work developed in our group [52] along the initial steps of the DAMASK project felt into this category.

5 References

[1] Alfonseca E, Manandhar S (2002). Improving an ontology refinement method with hyponymy patterns. In. Proc. of. 3rd International Conference on Language Resources and Evaluation, LREC 2002. Las Palmas, Spain.

[2] Banko M, Etzioni O (2008). The Tradeoffs Between Open and Traditional Relation Extraction. In. Proc. of. 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL 2008. Columbus, Ohio, USA, pp 28-36.

[3] Baumgartner R, Flesca S, Gottlob G (2001). Visual Web Information Extraction with Lixto. In. Proc. of. 27th International Conference on Very Large Data Bases, VLDB 2001. Apers PMG, Atzeni P, Ceri S, Paraboschi S, Ramamohanarao K, Snodgrass RT (eds.), Morgan Kaufmann, Roma, Italy, pp 119-128.

[4] Berners-Lee T, Hendler J, Lassila O (2001) The Semantic Web - A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. Sci Am 284:34-43.

[5] Berners-Lee T, Hendler J, Lassila O (2001) The Semantic Web,. Scientific American Magazine 284:34-43.

[6] Buitelaar P, Cimiano P, Frank A, Hartung M, Racioppa S (2008) Ontology-based information extraction and integration from heterogeneous data sources. International Journal of Human-Computer Studies 66:759 - 788.

[7] Cafarella M, Downey D, Soderland S, Etzioni O (2005). KnowItNow: fast, scalable information extraction from the web. In. Proc. of. Human Language Technology Conference/Conference on Empirical Methods in Natural Language Processing, HLT/EMNLP 2005. Association for Computational Linguistics, Vancouver, Canada, pp 563 - 570.

[8] Califf ME, Mooney RJ (2003) Bottom-up relational learning of pattern matching rules for information extraction. The Journal of Machine Learning Research 4:177-210.

[9] Cimiano P (2006) Ontology Learning and Population from Text. Springer-Verlag.

[10] Cimiano P (2006). Text Analysis and Ontologies. In. Proc. of. Summer School on Multimedia Semantics. Kallithea, Chalkidiki, Greece.

[11] Cimiano P, Handschuh S, Staab S (2004). Towards the self-annotating web. In. Proc. of. 13th international conference on World Wide Web. Feldman S, Uretsky M (eds.), ACM, New York, NY, USA, pp 462 - 471.

[12] Cimiano P, Ladwig G, Staab S (2005). Gimme' The Context: Context-driven Automatic Semantic Annotation with C-PANKOW. In. Proc. of. 14th international conference on World Wide Web. Ellis A, Hagino T (eds.), ACM, Chiba, Japan, pp 462 - 471.

[13] Ciravegna F, Dingli A, Petrelli D, Wilks Y (2002). User-system cooperation in document annotation based on information extraction. In. Proc. of. 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, EKAW 02. Gómez-Pérez A, Benjamins R (eds.), Springer Berlin / Heidelberg, Sigüenza, Spain, pp 122-137.

[14] Cunningham H, Maynard D, Bontcheva K, Tablan V (2002). GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In. Proc. of. 40th Anniversary Meeting of the Association for Computational Linguistics, ACL 2002. Philadelphia, US.

[15] Dill S, Eiron N, Gibson D, Gruhl D, Guha R, Jhingran A, Kanungo T, Mccurley KS, Rajagopalan S, Tomkins A (2003) A case for automated large-scale semantic annotation. Web Semantics: Science, Services and Agents on the World Wide Web 1:115-132.

[16] Ding L, Finin T, Joshi A, Pan R, Cost RS, Peng Y, Reddivari P, Doshi V, Sachs J (2004). Swoogle: A Search and Metadata Engine for the Semantic Web. In. Proc. of. Thirteenth ACM International Conference on Information and Knowledge Management, CIKM 2004. Evans DA, Gravano L, Herzog O, Zhai C, Ronthaler M (eds.), ACM Press, Washington, DC, USA, pp 652-659.

[17] Embley DW, Campbell DM, Smith RD, Liddle SW (1998). Ontology-Based Extraction and Structuring of Information from Data-Rich Unstructured Document. In. Proc. of. Seventh ACM International Conference on Information and Knowledge Management, CIKM 1998. Gardarin G, French JC, Pissinou N, Makki K, Bouganim L (eds.), ACM Press, Bethesda, Maryland, USA, pp 52-59.

[18] Embley DW, Tao C, Liddle SW (2002). Automatically Extracting Ontologically Specified Data from HTML Tables of Unknown Structure In. Proc. of. 21st International Conference on Conceptual Modeling, ER 2002. Spaccapietra S, March ST, Kambayashi Y (eds.), Springer Berlin / Heidelberg, Tampere, Finland, pp 322-337.

[19] Etzioni O, Banko M, Soderland S, Weld DS (2008) Open information extraction from the web. Commun ACM 51:68-74.

[20] Etzioni O, Cafarella M, Downey D, Popescu A-M, Shaked T, Soderland S, S.Weld D, Yates A (2005) Unsupervised Named-Entity Extraction from the Web: An Experimental Study. Artif Intell 165:91–134.

[21] Feilmayr C, Parzer S, Pröll B (2009) Ontology-Based Information Extraction from Tourism Websites. Journal of Information Technology 11:183-196.

[22] (1998) WordNet: An electronic lexical database. MIT Press, Massachusetts, USA.

[23] Flesca S, Manco G, Masciari E, Rende E, Tagarelli A (2004) Web wrapper induction: a brief survey. AI Commun 17:57-61.

[24] Freitag D (1998). Toward General-Purpose Learning for Information Extraction. In. Proc. of. 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, COLING-ACL 1998. ACL / Morgan Kaufmann, Montreal, Quebec, Canada, pp 404-408.

[25] Freitag D, McCallum A (1999). Information extraction with HMMs and shrinkage. In. Proc. of. AAAI-99 Workshop on Machine Learning for Information Extraction. Califf ME (ed.) AAAI, Orlando, Florida, USA, pp 31-36.

[26] Gaizauskas R, Wilks Y (1998) Information Extraction: Beyond Document Retrieval. Computacional Linguistics and Chinese Language Processing 3:17-60.

[27] Geleijnse G, Korst J, Pronk V (2006). Google-based Information Extraction. In. Proc. of. 6th Dutch-Belgian Information Retrieval Workshop, DIR 2006. Delft, The Netherlands, pp 39-46.

[28] Gómez-Pérez A, Fernández-López M, Corcho O (2004) Ontological Engineering with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web. Springer Verlag, Berlin, Germany.

[29] Handschuh S, Staab S, Studer R (2003). Leveraging Metadata Creation for the Semantic Web with CREAM. In. Proc. of. 26th Annual German Conference on AI, KI 2003. Günter A, Kruse R, Neumann B (eds.), Springer Berlin / Heidelberg, Hamburg, Germany, pp 19-33.

[30] Hearst MA (1992). Automatic acquisition of hyponyms from large text corpora. In. Proc. of. 14th conference on Computational linguistics - Volume 2, COLING 92. Kay M (ed.) Morgan Kaufmann Publishers, Nantes, France, pp 539 - 545.

[31] Kiryakov A, Popov B, Terziev I, Manov D, Ognyanoff D (2004) Semantic annotation, indexing, and retrieval. Journal of Web Semantics 2:49-79.

[32] Kiyavitskaya N, Zeni N, Cordy JR, Mich L, Mylopoulos J (2005). Semi-Automatic Semantic Annotations for Web Documents In. Proc. of. 2nd Italian Semantic Web Workshop on Semantic Web Applications and Perspectives, SWAP 2005. Bouquet P, Tummarello G (eds.), CEUR-WS, Trento, Italy, pp 210-225.

[33] Koivunen M-R (2005). Annotea and Semantic Web Supported Collaboration (invited talk). In. Proc. of. Workshop on End User Aspects of the Semantic Web at 2nd Annual European Semantic Web Conference, UserSWeb 05 Dzbor M, Takeda H, Vargas-Vera M (eds.), CEUR Workshop Proceedings, Heraklion, Crete, pp 5-17.

[34] Labsky M, Svatek V, Nekvasil M (2008). Information Extraction Based on Extraction Ontologies: Design, Deployment and Evaluation. In. Proc. of. 1st International and KI-08 Workshop on Ontology-based Information Extraction Systems. Adrian B, Neumann G, Troussov A, Popov B (eds.), CEUR, Kaiserslautern, Germany.

[35] Li Z, Ramani K (2007) Ontology-based design information extraction and retrieval. Journal of Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 21:137-154.

[36] Maedche A, Neumann G, Staab S (2003) Bootstrapping an Ontology-based Information Extraction System. In: Szczepaniak PS, Segovia J, Kacprzyk J, Zadeh LA (eds) Intelligent exploration of the web. Physica-Verlag, pp 345 - 359.

[37] Matuszek C, Witbrock M, Kahlert RC, Cabral J, Schneider D, Shah P, Lenat D (2005). Searching for common sense: populating cyc from the web. In. Proc. of. Twentieth National Conference on Artificial Intelligence (AAAI-05) and the Seventeenth Innovative Applications of Artificial Intelligence Conference (IAAI-05). AAAI Press, Pittsburgh, Pennsylvania, USA.

[38] McCallum A (2003). Efficiently Inducing Features of Conditional Random Fields. In. Proc. of. 19th Conference in Uncertainty in Artificial Intelligence, UAI 2003. Meek C, Kjærulff U (eds.), Morgan Kaufmann, Acapulco, Mexico, pp 403-410.

[39] McDowell LK, Cafarella M (2008) Ontology-driven, unsupervised instance population Web Semantics: Science, Services and Agents on the World Wide Web 6:218-236

[40] Michelson M, Knoblock CA (2007). An Automatic Approach to Semantic Annotation of Unstructured, Ungrammatical Sources: A First Look. In. Proc. of. IJCAI-2007 Workshop on Analytics for Noisy Unstructured Text Data. Knoblock CA, Lopresti D, Roy S, Subramaniam LV (eds.), Hyderabad, India, pp 123-130.

[41] Nedellec C, Nazarenko A (2005) Ontology and Information Extraction: A Necessary Symbiosis. In: Buitelaar P, Cimiano P, Magnini B (eds) Ontology Learning from Text: Methods, Evaluation and Applications. IOS Press, Amsterdam, The Netherlands, pp 3-14.

[42] Niekrasz J, Gruenstein A (2006). NOMOS: A SemanticWeb Software Framework for Annotation of Multimodal Corpora In. Proc. of. 5th International Conference on Language Resources and Evaluation, LREC 06. Genoa, Italy, pp 21-27.

[43] Roberts A, Gaizauskas R, Hepple M, Davis N, Demetriou G, Guo Y, Kola J, Roberts I, Setzer A, Tapuria A, Wheeldin B (2007). The CLEF corpus: semantic annotation of clinical text. In. Proc. of. AMIA 2007 Annual Symposium. American Medical Informatics Association, Chicago, USA, pp 625-9.

[44] Schroeter R, Hunterd J, Kosovic D (2003). Vannotea - A Collaborative Video Indexing, Annotation and Discussion System for Broadband Networks. In. Proc. of. Knowledge Markup and Semantic Annotation Workshop, K-CAP 03. Handschuh S, Koivunen M-R, Dieng-Kuntz R, Staab S (eds.), ACM, Sanibel, Florida, pp 9-26.

[45] Skounakis M, Craven M, Ray S (2003). Hierarchical hidden markov models for information extraction. In. Proc. of. 18th International Joint Conference on Artificial Intelligence, IJCAI 2003. Gottlob G, Walsh T (eds.), Morgan Kaufmann, Acapulco, Mexico, pp 427-433.

[46] Soderland S (1999) Learning information extraction rules for semistructured and free text. Machine Learning 34:233-272.

[47] Staab S, Maedche A (2000). Ontology engineering beyond the modeling of concepts and relations. In. Proc. of. ECAI-2000 Workshop on Ontologies and Problem-Solving Methods. Benjamins R, Gómez-Pérez A, Guarino N (eds.), Berlin, Germany.

[48] Studer R, Benjamins VR, Fensel D (1998) Knowledge Engineering: Principles and Methods. IEEE Trans Know Data Eng 25:161-197.

[49] Stumme G, Ehrig M, Handschuh S, Hotho S, Madche A, Motik B, Oberle D, Schmitz C, Staab S, Stojanovic L, Stojanovic N, Studer R, Sure Y, Volz R, Zacharia V (2003) The Karlsruhe View on Ontologies. Institute AIFB, Universität Karlsruhe, Karlsruhe, Germany.

[50] Turney PD (2001). Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL. In. Proc. of. 12th European Conference on Machine Learning, ECML 01. De Raedt L, Flach P (eds.), Springer-Verlag, Freiburg, Germany, pp 491-502.

[51] van Hage WR, Katrenko S, Schreiber G (2005). A Method to Combine Linguistic Ontology-Mapping Techniques. In. Proc. of. 4th International Semantic Web Conference, ISWC 2005 Gil Y, Motta E, Benjamins VR, Musen MA (eds.), Galway, Ireland, pp 732-744.

[52] Vicient C (2009) Extracció basada en ontologies d'informació de destinacions turístiques a partir de la Wikipedia. Universitat Rovira i Virgili, Tarragona.

[53] Xiao L, Wissmann D, Brown M, Jablonski S (2004) Information Extraction from the Web: System and Techniques Appl Intell 21:195-224.

[54] Yangarber R, Grishman R, Tapanainen P, Huttunen S (2000). Unsupervised Discovery of Scenario-Level Patterns for Information Extraction. In. Proc. of. 18th International Conference on Computational Linguistics, COLING 2000. Morgan Kaufmann, Saarbrücken, Germany, pp 940-946.

[55] Yildiz B, Miksch S (2007). Motivating ontology-driven information extraction. In. Proc. of. International Conference on Semantic Web and Digital Libraries. InPrasad A, Madalli D (eds.), Indian Statistical Institute Platinum Bangalore, India, pp 45–53.