Ontology Learning and Information Extraction for the Semantic Web

Martin Kavalec

University of Economics, Prague, 2006

Prohlášení

Prohlašuji, že doktorskou práci na téma ,,Ontology Learning and Information Extraction for the Semantic Web" jsem vypracoval samostatně. Použitou literaturu a podkladové materiály uvádím v přiloženém seznamu literatury.

V Praze dne 14. 6. 2006

Acknowledgements

I would like to thank my supervisor, Petr Berka, for his help, kindness and patience. Lots of my thanks belong to Vojtěch Svátek for many fruitful ideas and suggestions and for cooperation in evaluation of the experiment results. With a nice reminiscence on my stay at FZI in Karlsruhe, I thank Alexander Maedche and Philipp Cimiano for their inspirating ideas. Last, but not least I have to thank my parents, friends and my spouse Gizela, without their help and support I would hardly finish this work.

Abstract

The three main topics of this work are the semantic web, information extraction and ontology learning. The goals of this work are to give an overview of this three areas and describe their mutual relations (chapters 2 and 3). This presents a theoretical basis for the research part of the work, which focuses on study of relations between ontology concepts and their lexical representations in natural language texts.

The research part of the work consists of two studies. The topic of the first study (chapter 4) is extraction of information. The goal of the work is to find a way of learning, how relevant information may be identified on web pages. An important aspect of information extraction is adaptability to new tasks, therefore the goal is not to find indicators of important information, but to find how such indicators can be obtained for a new task. We propose a method which uses linguistic analysis of the texts of web pages and the categorization of the web pages in a web directory. This method was developed and experimentally tested on web pages of offering products and services. To identify the important information in this general setting, we focus on ways of expressing an offer. A simple web page summarizing tool, which extracts important sentences form the page was created.

The second study (chapter 5) is oriented to ontology learning, namely on relation extraction. It aims to improve the method of relation extraction from text, based on mining of association rules. The original method suggests to an ontology engineer, which pairs of ontology concepts are related to each other. The goal of our work is to extend these methods, to be able to add information, how the concepts are related, or, in other words, to suggest a name for the proposed relation. Ontology learning is a creative task, and the role of automatic methods is to assist to a human ontology designer. From this fact follows an associated goal to this task: to propose a method for evaluation of ontology learning tasks where human interaction is necessary.

A modification of the association rules mining algorithm and its underlying data structures was designed, which enables to keep and process the information how the relation was expressed in the original text. It was experimentally verified on two sets of text: descriptions of countries from the website of the Lonely Planet guide and on a semantically annotated corpus called SemCor.

Keywords ontology learning, information extraction, semantic web, natural language processing, association rules

Abstrakt

Tato práce má tři hlavní témata: sémantický web, extrakci informací a učení ontologií. Cílem práce je podat přehled těchto tří oblastí a popsat souvislosti a mezi nimi, jakožto teoretický základ pro vlastní výzkumnou práci. Ta se zaměruje na vztahy mezi koncepty v ontologii a způsobech vyjádření těchto vztahů v přirozeném jazyce.

Výzkumná část práce sestává ze dvou studií, tématem první z nich (kapitola 4) je extrakce informací. Cílem práce je najít způsob, jak identifikovat relevantní informaci na webových stránkách. Důležitý aspekt extrakce informací je přizpůsobitelnost extrakčních nástrojů na nové úlohy. Cílem proto není najít identifikátory relevantní informace pro konkrétní úlohu, ale navrhnout obecnou metodu, jak tyto identifikátory najít. Byla navržena metoda, která využívá lingvistickou analýzu textu webových stránek a jejich zařazení ve vyhledávacím katalogu stránek. Tato metoda byla vyvinuta a otestována na stránkách, nabízejících zboží nebo služby. K identifikaci relevantních informací v takto široké oblasti stránek jsme se zaměřili na způsoby vyjádření nabídky. V rámci experimentů byl vytvořen jednoduchý sumarizační nástroj pro webové stránky, který z nich extrahuje nejdůležitější věty.

Druhá studie (kapitola 5) je zaměřena na učení ontologií, konkrétně extrakci relací. Jejím cílem je vylepšit metodu extrakce relací z textových dokumenů, vycházející z dolování asociačních pravidel. Původní metoda navrhuje tvůrci ontologie pouze dvojice konceptů, mezi nimiž by mohla existovat relace. Cílem práce je doplnit informaci o charakteru této relace, jinými slovy navrhnout i pojmenování této relace, na základě toho, jak bývá vyjádřena v textu. Učení ontologií je kreativní úloha ve které automatické metody hrají jen podpůrnou roli, proto související cíl je navrhnout metodu evaluace úloh učení ontologií, ve kterých je nutné posouzení a korekce výstupu tvůrcem ontologie.

V rámci této studie byla navržena modifikace algoritmu dolování asociačních pravidel a jeho podpůrných datových struktur, která udržuje a zpracovává informaci, o tom jak byla relace vyjádřena v textu. Tato modifikace byla experimentálně ověřena na dvou sadách textů: na popisech zemí z webu turistického průvodce Lonely Planet a na sémanticky anotovaném korpusu textů SemCor.

Klíčová slova učení ontologií, extrakce informací, sémantický web, zpracování přirozeného jazyka, asociační pravidla

Contents

Chapter 1

Introduction

The World Wide Web today comprises a tremendous source of information, covering really broad scale of human information needs. But nearly all of the information can be used only by a human, reading it in a form of a web page in a www browser; automated processing of web content is rather limited, the most important and visible examples are web crawlers of full-text search engines.

Yet, there is a wide range of scenarios, when we need to use data from various information sources available on the web and frequently we also need the information structured in another way, than it was originally published. Even though a computer program can download the data from the web, their format doesn't simply allow their machine processing (because the formatting is oriented for human consumption, focused on presentation issues, rather than on the logical structure of the problem). Then, the simplest way of integrating various information sources on the web is copy, paste and edit to a suitable structure. Often, this is a tedious work, which could be easily automated, provided that the information was presented in machine readable format.[1]

The *Semantic Web* is an initiative of the W3 Consortium, with the goal to overcome this obstacle, i.e. with the goal to specify standards for publishing machine processable information on the web, where each published information has a well defined meaning. This allows for integration of the data from disparate sources. Besides the scenario mentioned in the example above, this feature of semantic web may also ease the development of communication interfaces in B2B communication. The topic of semantic web is presented in chapter 2.

One of key components of semantic web are ontologies. Ontologies are

[1] It is expected that the human readable version would coexist with the machine readable one.

knowledge models of a particular domain of interest, describing important concepts of the domain, their properties a relations between them. Ontologies are formalized models, which allows their machine processing and are shared and agreed on in the community, for which they are relevant. This way ontologies can provide well defined meaning for each information published on the semantic web.

However, process of creating an ontology demands time of experts. Ontology learning is a discipline, which studies the possibilities of algorithmic support to ontology engineers during construction of ontologies. Principles and methods of ontology learning are described in chapter 3.

Information extraction aims to identify information with a particular meaning in a set of free- or semi-structured texts. For example, in a set of seminar announcement texts we look for the topic of seminar, name of the speaker, place, date and time of the talk. In other words, it studies methods which can fill some predefined structure from unstructured text information. There are two applications of information extraction, relevant to this thesis:

- The predefined structure may be expressed by an RDF schema and the extracted information may be stored in RDF triples – then we use information extraction to 'populate' the semantic web, as described in section 2.3.

- In ontology learning from text (see section 3) we use statistic methods or machine learning algorithms, which are tailored to structured data. Therefore we usually have to find a way to transform free text to some structure and that is the point, where information extraction fits in.

The goals of this work are to give an overview of the semantic web, information extraction and ontology learning and their mutual relations. This presents a theoretical basis for the second part of the work, which focuses on study of relations between concepts and their lexical representations in natural language texts.

The topic of the first study (4) is extraction of information. The goal of the work is to find a way of learning, how relevant information may be identified on web pages, the experiments work with web pages offering products and services. An important aspect of information extraction is adaptability to new tasks [35], therefore the goal is not to find indicators of important information, but to find how such indicators can be obtained for a new task.

The second study (5) is oriented to ontology learning, namely on relation extraction. It aims to improve methods of relation extraction from text, based on association rules. Original methods suggest to an ontology engineer,

which pairs of ontology concepts are related to each other. The goal of our work is to extend these methods, to be able to add information, how the concepts are related, or, in other words, to suggest a name for the proposed relation. Ontology learning is a creative task, and the role of automatic methods is to assist to a human ontology designer. From this fact follows an associated goal to this task: to propose a method for evaluation of ontology learning tasks where human interaction is necessary.

Chapter 2

Information Extraction and the Semantic Web

2.1 Motivation for the Semantic Web

The World Wide Web today represents huge information resource. From its early beginnings it expanded admirably in many aspects – in the number of pages, users, topics covered. During this expansion also technical capabilities of WWW clients and servers evolved and hand in hand with this evolved the ways and purposes of using WWW. From publishing medium it extended to an application platform – many people use WWW to access e-mail services instead of traditional e-mail clients, to search library catalogues, to do shopping, booking of airplane tickets, theatre tickets, WWW also happened to be a standard communication channel for e-banking... Lots of current web pages are more application interfaces than documents.

There are many situations, where it would be very useful and practical to combine these WWW applications. There is a (rather loose) parallel to business information systems – there are strong benefits of integrated information systems compared to set of separated applications.

For example, Joe wants to see some particular movie. His information need is following: I want to know, in which cinemas a movie XYZ is played, the cinemas should be well accessible by public transport from my home. I want to know, how I will get there and when I have to go to be there on time. I can leave home at 18:00 and want to be back at 22:00. So he goes to a movie portal to find, in which cinemas he can watch the movie in near future in his city. Then he finds locations of the cinemas on an internet map, chooses a cinema, where he can get easily by public transport and finds the closest station to the cinema. Then he goes to website of public transport

and finds connections, departure and arrival times of connections from his home to the cinema.

As we can see, the task requires integration of information from 3 sources:

- the movie portal – information, which movies are available
- plan of the city – provides connection of cinemas with public transportation via information, which are the near stations to a selected cinemas
- public transport lines, stations and timetable

If the information from all the sources were machine readable, a software agent could automate this task.

The main obstacle of information integration on current web lays in its orientation to human users with focus on presentation issues of the structure and meaning of information. Current WWW technology provides presentation layer of applications delivered through it.

The *Semantic Web* is an initiative of the W3 Consortium, with the goal to overcome this obstacle, i.e. with the goal to specify standards for publishing machine processable information on the web. The standards have to address two main issues:

1. allow to describe reality in level of detail and in structure suitable for the publisher
2. to enable automatic integration of information from independent sources

It is possible to achieve these two goals simultaneously, but it is not as simple as publishing ordinary web pages. The problem of wider acceptance of semantic web is in the fact that there are not many applications which could exploit its possibilities, so the webmasters are not motivated to publish any information for the semantic web. On the other hand, it is useless to develop an application for semantic web, when there are no data it could work with. Information extraction from ordinary web pages is one of possible ways to extending the volume of semantic web data.

The two aforementioned features of semantic web technologies make it also suitable for integration of information systems of cooperating businesses, we may expect that semantic web technologies will find its place in B2B communication.

Semantic web is not focused only on 'application oriented' web pages and information integration is one of motivations for semantic web. For 'traditional', document oriented web pages it can provide rich metadata which can

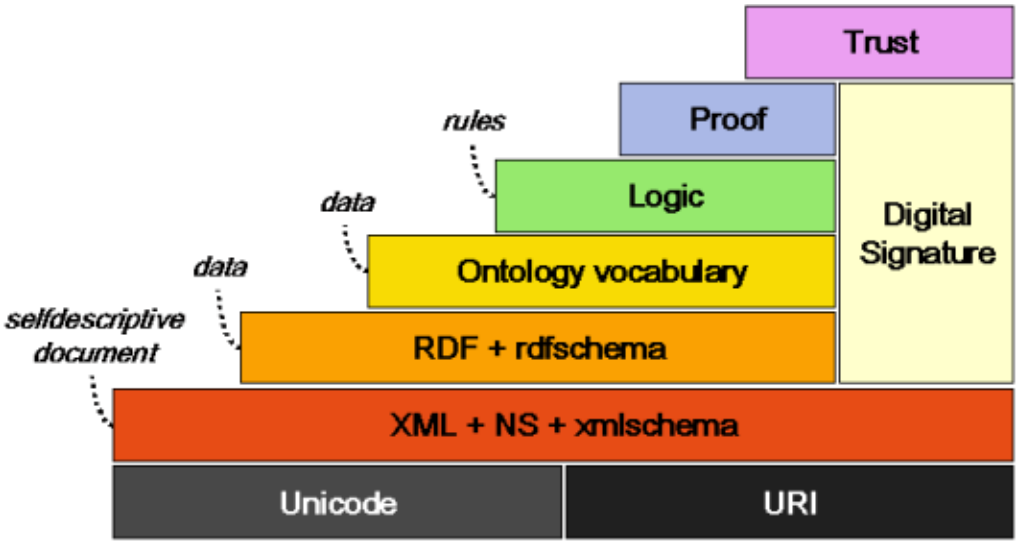

Figure 2.1: Layers of the semantic web

support better management of knowledge contained in them, for example in providing semantic search.

2.2 What is the Semantic Web

"The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation." – Tim Berners-Lee, James Hendler, Ora Lassila, The Semantic Web, Scientific American, May 2001

The basis of semantic web are standards defining how to express information to meet the requirements and to enable benefits of semantic web. The standards build upon currently accepted web standards such as URI or UNICODE, and add layers necessary for the semantic web. The structure is depicted in figure 2.1.

As mentioned above, the base standards are UNICODE for character coding and URI for globally unique identification of both real world objects and abstract entities.

XML, XML namespaces and XML schemas provide integration of semantic web with other web standards based on XML. XML is used as one of possible means of storing base information elements of semantic web, i.e. RDF triplets (see next paragraph). Use of XML simplifies machine processing; another way of storing RDF is format called Notation-3, which is better readable for humans end easily editable without special tools (i.e. in plain

text editor). There are also specialized systems for storing and querying large volumes of RDF triplets such as OpenRDF[1]. In fact, any system dealing with RDF needs to build some internal representation of the RDF data to be able to process it, so the XML encoding or Notation-3 should be considered as data exchange formats between semantic web systems.

Namespaces ensure uniqueness of identifiers when integrating data from independent sources; namespaces are not used only for XML encoded RDF triplets, but for other formats too.

2.2.1 RDF

Standard RDF (Resource Description Framework, citeRDF) defines the basic way of representing facts. It is designed to be able enable simple integration of data from independent sources.

All information is in RDF represented in a form of a triple *subject – predicate – object*, e.g. in a human readable form [*Rembrandt painted painting-Aristotle-contemplating-a-Bust-of-Homer*], or [*painting-Aristotle-contemplating-a-Bust-of-Homer is-exhibited-in Metropolitan-museum-of-New-York*]. For machine processing we need unique identifiers (URIs) instead of human readable strings on the positions of subject and predicate, the object in the triple can be either URI or an literal (e.g. [*painting-Aristotle-contemplating-a-Bust-of-Homer has-height "143.5 cm"*].

The use of URIs allows referencing other RDF documents available on the web, in way similar way to HTML pages linking one to another. Whereas in HTML the purpose of the links is navigation through the web, in RDF documents we add claims about entities mentioned in another documents, or we use predicates defined in another RDF document (and identified by URI) to assert some facts about "our" objects and predicates. It is the use of common predicates that enables machine processing of facts expressed in RDF – the computer program of course still doesn't really understand the meaning of the triplets, but because the author of the program, user of the program and the authors of published information share understanding and meaning of the identifiers for subjects, objects and predicates, the program can do something meaningful for the user.

2.2.2 RDF Schema

As we can see, to achieve the advantages of semantic web, the stakeholders have to agree on use of common set of identifiers. Standard called *RDF*

[1]former Sesame, http://www.openrdf.org

Schema was established for their definition. RDF is tightly coupled with RDF Schema [4], together they are denoted as RDF/S. RDFS defines classes of entities and a hierarchy of these classes, e.g. [*Painting is-subclass-of Artwork*]. This allows the publishers of information to use level of detail appropriate for their information needs. For example, an software agent can handle RDF triples with claims about paintings. It is still is able to processes RDF data containing more detailed concepts, like 'oil on canvas' or 'watercolour', provided the corresponding RDF schema declares these concepts as subconcepts of painting.

It also enriches the possibilities for querying – when we deploy the schema, we can pose one simple query for all artworks, even though we store their specific type (painting, sculpture, ...) for the individual items.

Because predicates specify the features of entities, they are called *properties* in RDFS. RDFS defines the hierarchy not only on classes, but also on the properties, e.g. it is possible to say that [*painted is-subproperty-of created-an-artwork*]. This again extends the possibilities of querying. The hierarchies on classes and their properties allow to use appropriate level of detail and to integrate RDF descriptions from different sources (provided they use the same schema or one schema is an extension of another). Other important feature of RDF/S is that an object can be instance of more than one class (and the classes can be unrelated in the is-subclass-of hierarchy). This allows to describe the object from different points of view and to reflect different information needs of different participants. One object (with one URI) can be described in terms of two independent RDF schemas e.g. the museum can describe a painting using different schema than the insurance company, by which it is insured. It would be naive to expect that everybody on the world would agree on using the same schema, so there is a need for possibility to define equivalence of classes and predicates from different schemas and other ways to map one schema to another. When two schemas evolve independently and describe the same area of interest, the community should use a standardization process to merge the schemas and agree on one common schema.

Furthermore, RDFS specifies *domain* and *range* of predicates. The meaning of these terms is different than in database systems, it is not considered as an integrity constraint, but as a fact, applicable for inferencing. For example, if we encounter a triple [*Georges-de-la-Tour painted painting-The-Fortune-teller*] and the schema specifies that domain of *painted* is *Painter* and range of *painted* is *Painting*, we can infer that [*Georges-de-la-Tour is-a Painter*] and [*painting-The-Fortune-teller is-a Painting*].

2.2.3 Higher ontology languages

But we cannot say that Semantic Web is missing integrity constraints. They are expressed on higher levels in richer ontology languages.

There are more ontology languages used on semantic web. The current proposed standard is called OWL (Web Ontology Language). OWL is based on DAML+OIL language,which has two roots: DAML (DARPA Agent Markup Language), created in DAML Project supported by US Defence Advance Research Projects Agency, and OIL (Ontology Inferencing Layer) developed in Europe.

OWL provides us with a several integrity constraints. At first, it allows to specify value constraints by *owl:allValuesFrom* or *owl:someValuesFrom*. But the more usual point of view on constraints in OWL considers them as part of class definition, i.e. some set of constraint defines a class of individuals, which satisfy the constraint).

Other important features of ontologies are cardinality restrictions on properties, such as *owl:cardinality, owl:minCardinality* and owl:maxCardinality. *owl:cardinality* specifies exact number of values required for given property, *owl:minCardinality* and *owl:maxCardinality* determine the minimal and maximal cardinality of given property.

Other cardinality restrictions are specified by functional characteristics of properties, *owl:FunctionalProperty* and *owl:InverseFunctionalProperty*. Functional property can have only one unique value for each instance x, InverseFunctionalProperty can have only instance x for a given value of the property y.

But important feature of higher ontology languages is inferencing. The hierarchy of classes and properties, specified in RDFS, provides us with a basis for inferencing. OWL extends this basis by other features, especially important are logical characteristics of properties, such as transitivity, reflexivity, symmetry or being inverse of another relation, being mutually exclusive with another relation. For example, we can use transitivity of relation *is-located-in* to infer that [*Charles-bridge is-located-in Czech-Republic*], from facts that [*Charles-bridge is-located-in Prague*] and [*Prague is-located-in Czech-Republic*].

Apart from characteristics of features, characteristics of individuals are important for inferencing, such as *owl:sameAs* or *owl:differentFrom*.

Inferencing can be also used as an mechanism of constraint checking, for example, in OWL we can say that [*Person is-disjoint-with Non-Living-Object*]. In the RDF Schema we can specify that [*Painter is-subclass-of Person*], [*Artwork is-subclass-of Non-Living-Object*] and that [*Artwork is-domain-of is-exhibited*]. We already know that [*Georges-de-la-Tour is-a*

Painter]. So, we also know that [*Georges-de-la-Tour is-a Person*]. Then from [*Georges-de-la-Tour is-exhibited in Metropolitan-Museum-of-New-York*], and from domain specification we can infer that [*Georges-de-la-Tour is-an Artwork*], so [*Georges-de-la-Tour is-a Non-Living-Object*], which is a contradiction.

2.2.4 Semantic Web Query Languages

The semantic web should be the web of information sources accessible to machines, it should be an information environment for software agents. These agents would accomplish tasks, which we have to do manually now, because the tasks require integration of information from independent sources.

Let's go back to the example of Joe, looking for a cinema, where he can watch the movie he is interested in. It would be very inefficient for the agent to fetch whole database of cinemas with schedule of movies, whole plan of the city and public transport and then search for cinemas and times, matching the user's request. Much better approach is to follow the same steps, as Joe would do manually. Joe can deal with different query interfaces of the different sources, but for a computer program, some unified interface is needed. Therefore, a standard query language is important part of the semantic web, as well as a standard for returning results of the queries.

Several query languages appeared in various research projects, the most important are RDQL from HP Labs' project Jena[2], RQL, part of RDF-Suite from ICS Forth[3], Crete and SeRQL,[4] main query language in Sesame from VU Amsterdam and Aduna.

The most widely implemented is probably RDQL, apart from Jena it is available in OpenRDF, PHPxmlclasses and RDFStore projects.

Furthermore, RDF Data Access Working Group of W3C is proposing another RDF query language called SPARQL (Standard Protocol And RDF Query Language), which is based heavily on RDQL.

2.3 Information Extraction for the Semantic Web

Information extraction is in general a automatic process by which we aim to identify information with a particular meaning in a set of free- or semi-structured texts. For example, from a set of seminar announcement texts we

[2]http://www.hpl.hp.com/semweb/jena2.htm
[3]http://139.91.183.30:9090/RDF/RQL/
[4]http://www.openrdf.org/doc/sesame/users/ch06.html

look for the topic (or title) of seminar, name of the speaker, place, date and time of the talk.

The research in this area is strongly tied with the Message Understanding Conferences. In each of this conferences, the participants obtain labeled texts with, where the labels mark the information to be extracted (and the class of the information, i.e. what is name of the speaker, what is title of the talk etc.). Then they build a system which should learn (from the labeled texts), how to find the same information from similar unlabeled texts. The texts are usually short newspaper messages, for example reporting terrorist attacks or acquisitions, mergers or changes on key posts of companies.

The solution typically consists of two sub-tasks:

1. Identification of *named entities*, such as names of persons, organizations or places

2. Extraction of relations among the named entities, e.g. Person X has visited place Y and has met person Z, or Company X raised its share in company Y to Z %.

These sub-tasks are usually solved in two separate steps.

2.3.1 Specific issues of information extraction from web

In comparison to extraction from plain text, we have some additional information, when we extract some information from WWW pages:

- structure of the web page and its formatting (i.e. structure of the document object model and meaning of some HTML elements such us headings or lists.)

- topology of the web, i.e. how are pages interconnected by hyperlinks and which words are used in the hyperlinks

- explicit metadata in the web page

- information from the analysis of URL structure – e.g. occurrence of some key terms or abbreviations, their position, co-occurrence with numbers

- information about images, their dimensions and other properties (number of colors, color histograms, ...)

Possibilities of information extraction from web by analytic modules specialized on aforementioned types of information are explored by project Rainbow.

A big advantage of web is large amount of documents available for experiments. This is important for statistical methods of processing of the web page texts. Furthermore, we can select specific documents using search engines (see [5]) of web catalogues (see section 4). On the other hand, syntactical analysis of the text on web is of limited use, because large part of information is expressed by other means, e.g. by arrangement to a table or to a bulleted list.

Web information extraction techniques

Following groups of approaches are being used for information extraction from web:

- Wrapper induction
- Methods based on logical representation of a document
- Finite state methods, such as hidden Markov models or conditional random fields

Wrapper induction is technique specialized for web information extraction. It exploits the HTML mark-up of page, so each wrapper is is based on formatting of specific website. For each type of information (e.g. name of product or product price) we look for a pair of delimiters which enclose it. It fits well pages with regular structure, containing bulleted or numbered lists or tables (like price lists, or department pages enlisting the staff and so on).

For the induction of a wrapper we need several training examples, in which we mark the information to be extracted. The algorithm then identifies the delimiter pair for each type of information. The extraction algorithm is simple, it looks for the occurrences of the delimiters in the page and text enclosed by them is returned as information of the type corresponding to the delimiters. For example, the wrapper can learn that price is delimited by HTML code `<td><b>` at the beginning and `</b></td>` at the end (i.e. a field in the table in bold face). Then strings enclosed by these tags are returned as a price. (It is possible to further restrict the content, for price it is reasonable to expect numbers, decimal point or comma and a currency symbol).

There are different types of wrappers with different ability to adopt different levels of complexity of the page structure. But, the higher power of the wrapper (i.e. the more complex structures it is able to handle), the higher is

difficulty of their training (the more examples we need and the lower precision and recall we reach).

The specificity of a wrapper to a concrete website (or usually part of a website) is a big disadvantage of wrappers. Wrappers are not suitable for extraction from a larger set of independent sources, because for each source we had to prepare a specific wrapper. On the other hand, the same specificity allows us to reach rather high precision and recall. Another disadvantage of wrappers is their fragility. After a wrapper is trained, it cannot adapt to even slight changes of the pages. For real-world usage it is necessary to detect such situations, to prevent filling of a database by incorrectly extracted data.

Transformation of the document to a logical representation is a basis for other group of approaches. It means that the analyzed document is split to a sequence of tokens and then is represented by assertions about this tokens, e.g. token is emphasized, token X follows immediately token Y. This way it is easy to include linguistic information about tokens, like token is a proper noun in the dative case. The set of possible assertions depends on the type of extraction task. Then machine learning algorithms are used to find important combinations of assertions, inductive logical programming fits particularly well this type of task. These combinations of assertions (in case of ILP they are Horn clauses) are rules for extraction – set of tokens, which satisfies the the conditions in the clause is identified as the information to be extracted. Interesting aspect of this technique is a possibility to incorporate some background knowledge relevant for the extraction task (by defining appropriate possible assertions and also useful combinations of assertions, again formulated as Horn clauses).

Hidden Markov models are *generative statistical models*, which consist of a set of internal states and an alphabet of output symbols. In each state the system emits a symbol from the output alphabet and changes the state to a next one. There is a probability of transition to state s_j probabilistic distribution $P(w|s_i)$ which to each state s_i assigns a probability of emission of a output w from the output alphabet and probabilistic distribution $P(s_j|s_i)$ which to each state s_i assigns the probability of next state s_j, given the current state s_i.

For web information extraction, the output alphabet is set of tokens consisting of HTML tags and words of a natural language. The method is language independent. The internal states are in a simplest form four:

- B – background state – this state emits with high probability all the words, which are not interesting for the extraction task and most of

the HTML markup, too. (The HTML markup may be also detected to be useful as a prefix or suffix state)

- P – prefix state – this state emits with high probabilities tokens preceding the target information
- S – suffix state – this state emits with high probabilities tokens succeeding to the target information
- T – target state – this state corresponds to tokens, which should be extracted

These models are trained from a training data, in which the tokens are labeled by their corresponding state (for practical reasons, only P, S and T states are labeled, unlabeled tokens are considered to correspond to the B state). From such data, the probabilities of transitions between states and the emission probabilities are computed.

For the extraction task, the Viterbi algorithm is used. This algorithm gets as its input a sequence of tokens from the output alphabet and finds the most probable sequence of internal states, which could produce the input sequence. So, in the information extraction scenario tokens with the target state assigned are considered to be the extracted information.

For each type of information different model is trained, so for example, in the seminar announcement task, one model extracts the topic of seminar, another model extracts the name of the speaker and so on. It is possible to build more complex models, frequently the target state is divide to more states to model the structure of extracted information (e.g. for names of speakers we can divide the target state to state corresponding to title before a name, to state for given name, state for middle name, state for surname and state for title after a name. The probabilities of transition can adapt the situation, when some of the name component is missing. It is also possible to restrict the structure of model by requiring that certain probabilities have to be zero, e.g. title before a name cannot be immediately followed by title after a name.

The main shortcoming of hidden Markov models is that the next state of the model depends solely on the previous state. It is possible to lenghten the "history" to k previous states by working with new set of states in which each new state is a combination of k original states: For $k = 2$ the sequence $s_1, s_2, s_3 \ldots$ is transformed to $s'_1, s'_1, s'_2, \ldots$ where $s'_1 = (s_0, s_1), s'_2 = (s_2, s_1), \ldots$.[5] But this is only very limited solution, because it leads to increase

[5] A dummy starting state s_0 is added to the original sequence to ensure that in each original state we have a history of two previous

of number of states, exponential with respect to k and with high number of states it is not possible to estimate the probabilities of transitions among them. So it is not possible to model long-range dependencies in HMM.

Conditional random fields [22] are another finite state model which addresses the shortcoming of the short history of the hidden Markov model. Conditional random fields are not a generative models, so there are no hidden internal states and observed output states. In conditional random field we have an input sequence which is to be labelled by an output sequence. X is a random variable over the input sequences, Y is a random variable over corresponding label sequences, the tokens in Y are taken from a finite alphabet, in information extraction setting it may consist of two labels – *background* and *target* or *background* and labels for the individual types of information to be extracted, e.g. *name, place, topic...*

The conditional random field is defined by

- a graph, in which the vertices correspond to labels of the output sequence Y and the edges define dependencies among the vertices: probability on the vertex v depends only on the observed sequence X and the labels on its neighbouring vertices.

- two fixed sets of *feature functions*, f_k for edges of the graph and g_k for the vertices of the graph. Edge feature functions evaluate domain dependent features of the observed sequence for each edge of the graph G. The arguments of the edge feature function $f_k(e, \mathbf{y}|_e, \mathbf{x})$ are an edge e of the graph G, components of output sequence corresponding to the edge e denoted as $\mathbf{y}|_\mathbf{e}$ and the observed sequence $\mathbf{x}$. Similarly for the vertex feature functions g_k.

The simplest and commonly used modeling graph is a simple chain, connecting previous token with its nearest following token.

An example of a vertex feature function in an information extracting task may be following: g_k is true if the token x_i is a known first name, token x_{i-1} is a string 'prof.' and the tag for y_i is 'speaker name', else g_k is false. The feature functions may use any element of the input sequence $\mathbf{x}$.

The model is based on computation of probability of output sequence $\mathbf{y}$, given the input sequence $\mathbf{x}$. By the definition of the model, this probability $p_\vartheta(\mathbf{y}|\mathbf{x})$ is maximized, when expression

$$\exp(\sum_{e\in E,k} \lambda_k f_k(e, \mathbf{y}|_e \mathbf{x}) + \sum_{v\in V,k} \mu_k g_k(e, \mathbf{y}|_v \mathbf{x}))$$

is maximized. Symbol ϑ denotes the set of parameters of the model $(\lambda_1, \ldots, \lambda_k, \ldots, \mu_1, \ldots, \mu_k)$. The algorithm for estimation of these parameters from training data is described in [22].

2.3.2 Information extraction for the semantic web

The information extraction techniques could be also deployed to transform the information usual web pages to the data for semantic web. It is expected that the semantic web will be based on large number of smaller ontologies, which can evolve in time. Therefore it is important that the algorithms used for the extraction would be adaptable to these changes. Furthermore, because of the variety of human interests and activities, it is necessary that the adaptation of the algorithm to a particular domain of activity wouldn't be too demanding on human and time resources.

When adapting an extraction system to a particular domain we work with two basic types of resources [35]:

- linguistic resources: tokenizers, part of speech taggers, morphological analyzers, chunkers or syntax analyzers
- semantic resources: ontologies and factual bases

For some tasks it is necessary to combine these resources, e.g. for anaphora or metonymy resolution. Other example can be training of probabilistic models – in common situations, we do not observe large number of words to get reliable estimates of their probability of occurrence in a text. By grouping them to a classes based on hierarchy from a domain ontology we can increase the reliability of the estimates.

Another problem of information extraction for semantic web is larger number of classes, compared to classical scenarios, in which case we deal with approximately 5 classes. In simpler ontologies for semantic web, we can expect tens by magnitude. We can expect that some methods successful in classical information extraction may not work for the semantic web task. On the other hand, we can take the advantage of the large volume of documents available and of some regularities in expressing a particular type of information, such as book and article citations, for building extraction patterns and extracting bibliographic data from web.

In [35] Stevenson and Ciravegna formulated the main requirements on IE techniques to be applicable on extraction for semantic web. The methods should be

1. adaptable on a limited sample of training data

5. Consequences – Pragmatics, logic
4. Semantics
3. Syntax
2. Morphology
1. Tokenization

Figure 2.2: Levels of language analysis

2. capable to identify relations without the necessity of deep syntactic analysis; on the other hand, the system should be able to use linguistic information, in cases when it is available and reliable, in other cases it should back off to simpler methods

3. able to use ontological resources, when available

Our own experiment for information extraction, trained on a web directory and pages addressed from it, is described in chapter 4

2.3.3 Natural language processing

Information extraction and ontology learning from texts have a common denominator: the work with free text in natural language. Natural language processing methods and tools represent an important basis for both text mining and information extraction and as a consequence, for ontology learning, too.

In natural language processing, the analysis of text is separated to several levels which follow the linguistic view of a language, which are depicted on fig. 2.3.3.[6]

Tokenization This lowest level is rather technical task of splitting the input text onto list of tokens. In some Asian languages this is not so trivial task (as well as in speech recognition). This step also includes other normalizations of text, like lowercasing ordinary words.

[6]This view of levels expects analysis of text in digital form. There is notable part of NLP research – automatic speech recognition, for which the basic levels are phonetics and phonology, but this part of NLP is not related to ontology learning.

Morphology determines the stem of word ant its morphological categories (part of speech, gender, declination, singular or plural...). The result is not single-valued – frequently there are more possible morphological interpretations for a word. E.g. 'fly' may be a verb (move in air) or a noun (insect).

An important step in NLP builds on morphology: Part-of-speech tagging. This task reduces the ambiguity of morphological analysis taking sequentiality of tokens into account. Hidden Markov models can be successfully applied in this task. (e.g. the probability of noun after a definite article is very high, opposed to probability of a verb).

Syntax In this level sentence structure is examined.

With shallow parsing, the main components of sentence are identified (noun phrases, verb phrases, ...). The components build a component structure of a sentence, which reflects a derivation tree of grammar rules like:

$$\begin{aligned} \text{sentence} &:= \text{noun phrase} + \text{verb phrase} \\ \text{verb phrase} &:= \text{verb} + \text{object} + \text{adverbial phrase} \end{aligned}$$

In deep parsing, each word is related to another, building a dependency structure of sentence (e.g. an adjective depends on subject or object to which it adds some information)

Semantics adds semantic functions to sentence, like Agens (who acts), Paciens (on whom the agens acts), Goal, Instrument, Effect,...

Consequences, pragmatics, logic – the nodes of syntactic structure of sentence are coupled with real world objects (or their corresponding instances in an ontology). Information in sentence is represented in form of logic formulas, which can be evaluated.

On one hand, this separation of levels of the language analysis is helpful in that it allows us to focus on smaller and simpler tasks, on the other hand, natural language processing is a complex task which cannot be performed in a unidirectional chain of steps. Correct results of a lower level of analysis frequently depend on a results of higher level. Assignment of morphological categories of individual words in a sentence depends also on syntactic structure of the sentence. Similarly, the correct syntactic analysis may require knowledge of semantic of the words. In such cases, the results of a lower level of analysis are ambiguous and this ambiguity is resolved at the higher level.

An important task in NLP is part of speech tagging, which lays between morphological analysis and syntax analysis. It determines part of speech for each word in text. In flexive languages, it also specifies other morphological categories such as gender, case, singular/plural and so on. It stands lower than syntax analysis because the word context is sufficient for resolving the ambiguities of morphological analysis and in fact, POS tagging is a prerequisite for syntax analysis. POS-tagging can be performed efficiently, its computational complexity is linear to number of words, and the precision of tagging is around 97–98%.

Two phases of syntax analysis are distinguished, *shallow* syntax analysis and *deep* syntax analysis. Shallow syntax analysis (or chunking) splits the sentences to noun phrases, verb phrases, prepositional phrases or adverbial phrases. It is still relatively fast and robust. Deep syntax analysis (or parsing) determines function of each word in the sentence and it is computationally demanding and also less reliable.

2.3.4 RDF/S syntax

The in previous text we used a 'pseudo' syntax in form [Leonardo-da-Vinci is-a Painter], for sake of better readability. There are W3C recommendations which specify syntax for RDF [2] and RDFS [4], based on XML.

To define schema displayed by picture 2.3, following RDFS document should be availabel at (fictious) URL `http://art.ex`

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

<rdfs:Class rdf:about="http://art.ex/artist">
  <rdfs:label xml:lang="en">Artist</rdfs:label>
</rdfs:Class>

<rdfs:Class rdf:about="http://art.ex/painter">
  <rdfs:label xml:lang="en">Painter</rdfs:label>
  <rdfs:subClassOf rdf:resource="http://art.ex/artist"/>
</rdfs:Class>

<rdfs:Class rdf:about="http://art.ex/artwork">
  <rdfs:label xml:lang="en">Artwork</rdfs:label>
</rdfs:Class>
```

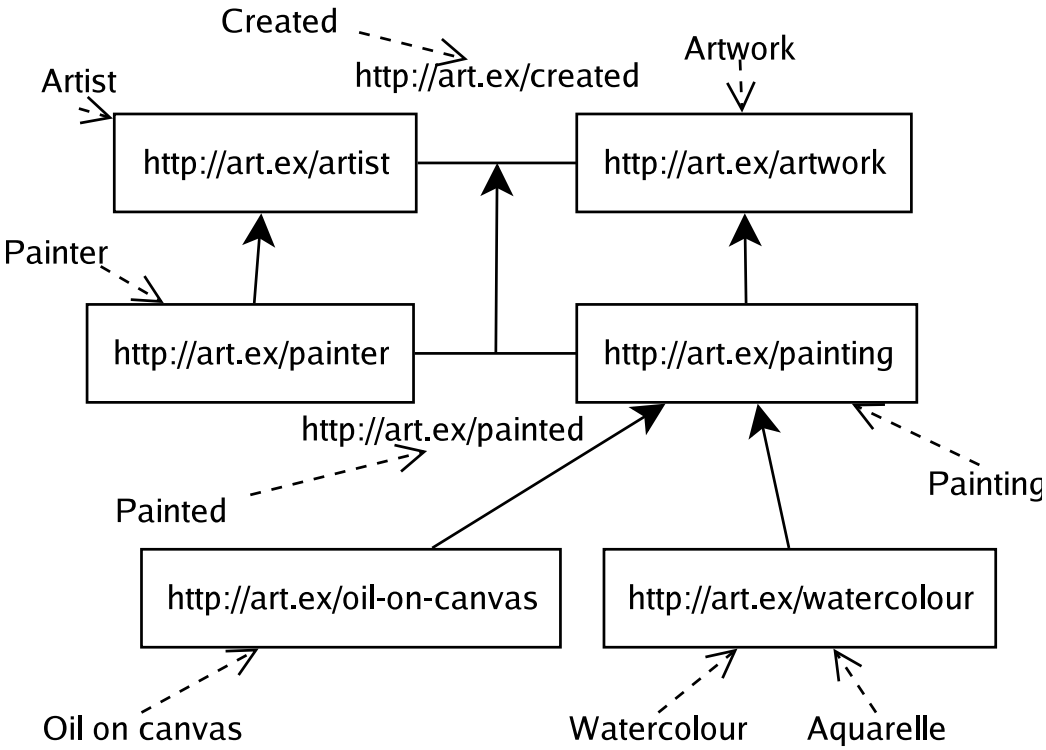

Figure 2.3: RDF Schema

```
<rdfs:Class rdf:about="http://art.ex/painting">
  <rdfs:label xml:lang="en">Painting</rdfs:label>
  <rdfs:subClassOf rdf:resource="http://art.ex/artwork"/>
</rdfs:Class>

<rdfs:Property rdf:about="http://art.ex/created">
  <rdfs:label xml:lang="en">created</rdfs:label>
  <rdfs:domain rdf:resource="http://art.ex/artist"/>
  <rdfs:range rdf:resource="http://art.ex/artwork"/>
</rdfs:Property>

<rdfs:Property rdf:about="http://art.ex/painted">
  <rdfs:label xml:lang="en">painted</rdfs:label>
  <rdfs:subPropertyOf rdf:resource="http://art.ex/created"/>
  <rdfs:domain rdf:resource="http://art.ex/painter"/>
  <rdfs:range rdf:resource="http://art.ex/painting"/>
</rdfs:Property>
</rdf:RDF>
```

If we want to express information that Leonardo da Vinci painted a painting of Mona Lisa, it can be done by following RDF/S code:

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:art="http://art.ex/">

<art:artist rdf:about="#leonardo-da-vinci">
  <rdfs:label>Leonardo da Vinci</rdfs:label>
  <art:painted>
    <art:artwork rdf:about="#mona-lisa">
      <rdfs:label>Mona Lisa</rdfs:label>
    </art:artwork>
  </art:painted>
</art:artist>
<rdf:RDF>
```

This example also demostrates, how XML namespaces are used to connect the RDF data with its schema. From use of relation `art:painted` one may infer that Leonardo da Vinci is a `art:painter` too.

Chapter 3

Ontology learning overview

In section 2.2 we described ontologies as a key component of the prospective semantic web. Ontologies are important part of a growing number of knowledge management systems, as well.

The difficulty of their manual development is however a significant drawback. Therefore, methods of computer support of this process are intensively examined. The development of ontologies is also intellectually demanding task, so it is not expected that fully automatic methods of their construction could give satisfactory results. The research in ontology learning is oriented to development of methods, which would support a knowledge engineer in the process of creation of an ontology.

3.1 Ontologies

The term "ontology" has been introduced by Aristotle in Metaphysics, IV, 1, and denotes a philosophical discipline which studies the nature and the organization of being. It tries to answer questions "what being is?" and "what are the features common to all beings?". [24]

In computer science and throughout this thesis the term *ontology* can be defined as a formal, explicit specification of a shared conceptualization of some domain ([15]). Formal means that the ontology is machine readable and shared means that it is accepted in a community or by a group. In other words, an ontology is an engineering artifact, constituted by a specific vocabulary used to describe a certain reality, plus set of explicit assumptions regarding the intended meaning of the vocabulary. [24]

Both the definitions relate ontologies to communication, in Gruber's definition it lays in words "shared conceptualization", which imply "understanding", in Maedche's definition it is coined by the word "vocabulary". Natural

language is too ambiguous for human-machine or machine-machine communication. In an ontology, each concept has definite and unique meaning. This way an ontology defines a formal language and commitment to it enables sharing and reusing of knowledge across systems. Ontologies, as formalized knowledge models, are studied and developed also outside the context of the semantic web. But for the semantic web with its goal to facilitate better co-operation between humans and computers in processing information available on the web, sharing and reusing knowledge across system is a necessity and therefore ontologies are one of its key components.

There are different types of ontologies, according to their level of generality [16], 3.1.

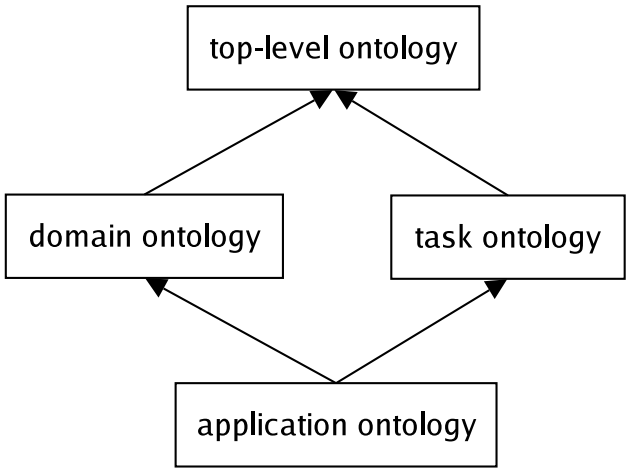

Figure 3.1: Different kinds of ontologies

Top level ontologies consist of very general concepts like space event or time. They serve as a basis for more specific ontologies, therefore also called *foundational ontologies* or *upper-level ontologies*. Upper-level ontologies contain more detailed concepts and reach to some more general domain dependent concepts.

Domain ontologies contain concepts related to a generic domain. These concepts are specializing the concepts from the top-level ontology.

Task ontologies describe the concepts and vocabulary relevant to a generic task or activity by specializing the top-level ontologies.

Application ontologies are the most specific. Generally, ontologies are designed to be shared and reused, but to support concrete tasks within

a context of a concrete application, these ontologies contain application specific concepts, for which a wider acceptance and consensus (i.e. outside the scope of the application) is not necessary. It is possible that during the time some application specific concepts naturally become widely accepted (for example as a result of industry-wide standardization of a business process) and then such concepts become part of domain or task ontologies.

3.1.1 Structure of an ontology

An ontology a knowledge model with a rather complex structure. The key role is played by *concepts* or in RDFS and OWL *classes*. They represent a type of an entity in particular domain. For a university, some of the concepts would be 'Student', 'Lecturer', 'Subject', 'Course' or an 'Exam'. A concept represents the whole set of entities of a particular type. A single entity is called an *instance* or an *individual*. Each individual has some *properties* and this way also each concept has some properties, which are relevant for the instances of the concept. There are two types of properties: the *datatype properties*, whose value is a *literal* – a general string value, which is not further interpreted in scope of the ontology. E.g. a Student or a Lecturer has a name or an Exam takes place in a particular date and time. The dataype properties are also called attributes.

The second type of property are object properties a these properties hold a reference to another object: e.g. a Student may pass an Exam and an Exam is related to a Subject. Object properties are also called *relations.* A hierarchical arrangement of concepts is an important feature of ontologies. There is a specific relation defining this concept hierarchy.

Following the semiotic point of view on ontologies as a mean of communication, an ontology consist from an *ontology structure* and a *lexicon* for the ontology structure. The ontology structure defines the set of concepts and relations used in the modeled domain and is sufficient for logical inferencing and machine-to-machine communication.

The ontology structure itself is independent of a human language. The concepts and relations are identified by artificial identifiers. URIs are the standard mean of identifications in the semantic web environment.

However, for human-machine communication, as well as for applications 'connecting' the ontology with documents in a human language, the lexicon is needed. In multilingual environments there may be multiple lexicons for the same ontology structure.

The lexicon defines two mappings: mapping of lexical items (words or terms of a language) to the concepts of the ontology structure and mapping

of lexical items to the relations of the ontology structure. Both the mappings are of type M:N, one term may reffer for multiple concepts and one concept may be referred to by multiple terms, similarly for names of relations.

The structure of the ontology is illustrated in diagram 3.2. The concepts are displayed as boxes, arrows represent the hierarchy, simple lines the non-taxonomical relations. Note that there exists a hierarchy on the relations, too; the relation 'painted' is a specialization of relation 'created'. Lexical items start with a capital and dotted arrows show the mapping of the lexicon to the concepts.[1]

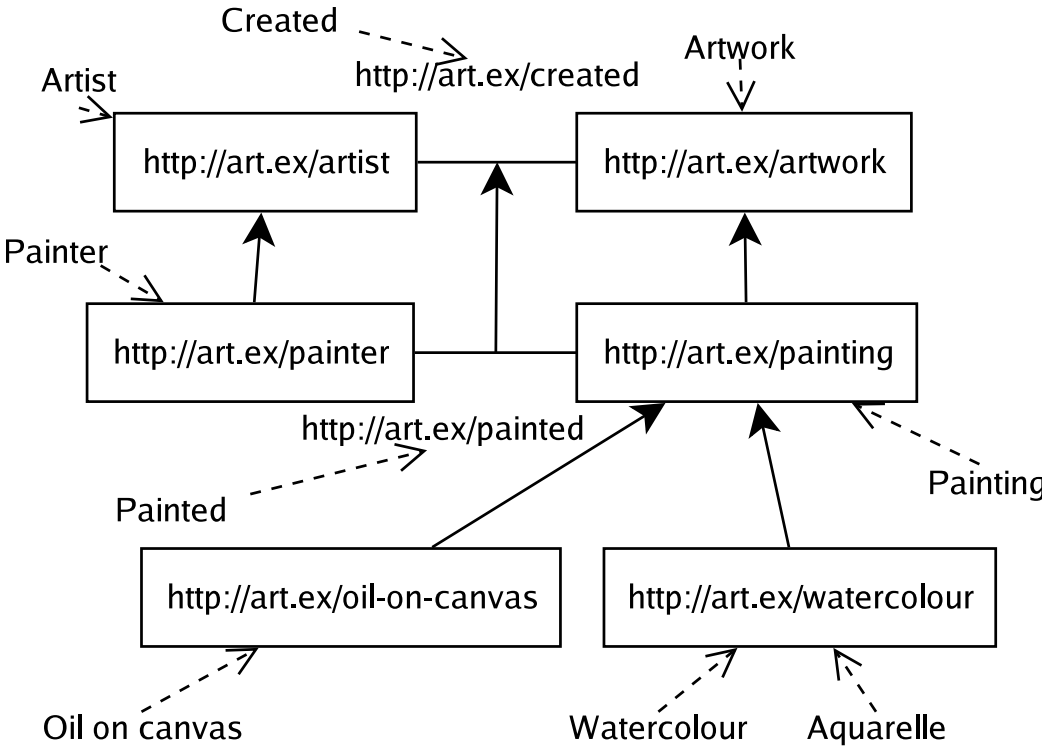

Figure 3.2: Structure of an ontology

The instances of the ontology concepts are contained in the knowledge base. Similarly to the ontology, the knowledge base consists of *knowledge base structure* and *knowledge base lexicon*. The *knowledge base structure* The ontology structure defines the structure of the knowledge base. Knowledge base maps the set of instances to the set of ontology concepts, for each concept

[1]The diagram is used to ilustrate the relation between lexicon and the ontology structure, in praxis the lexical items are not displayed separately of concepts.

C and instance I is determined, whether the instance I is the instance of the concept C or not. One instance may be an instance of several concepts, for example, in a knowledge base of movies a particular person may be at the same time instance of 'Actor' and instance of 'Director'.

Similarly, there exists mapping of pairs of instances and ontology relation. For each ontology relation R and pair of instances (I_1, I_2) is determined, whether the pair instantiates the relation R or not.

The knowledge base lexicon defines mapping between the lexical items and the instances in the knowledge base structure. Again, the mapping is M:N, one lexical item may refer to more than one instance (e.g. several persons are named 'John Smith') and one instance may be referred to by multiple lexical items (e.g. an artist may have a real name and a pseudonym).

3.2 Approaches to ontology learning

Ontology learning is a transdisciplinary field, connecting researchers from a variety of disciplines: natural language processing, information extraction, knowledge representation, logic, philosophy, machine learning, databases, etc. It is relatively new field and its methods and techniques are still shaping. Ontology learning is then a rather broad term grouping together various approaches and heuristics.

These approaches may be distinguished from several points of view: by character of the source data used for learning, by their focus on a specific task in ontology learning or

3.2.1 Sources for ontology learning

Two types of resources may be used in ontology learning:

- generic resources
- domain specific resources.

Generic resources are independent on the domain, for which the ontology is build. Unlike domain dependent resources, generic resources may be tightly coupled with an ontology learning algorithm. Upper-level ontologies or lexical databases, such as WordNet, are used in this role.

In fact, WordNet is such a popular resource in ontology learn that it deserves a more detailed description here. WordNet is freely available lexical base, which contains nouns, verbs, adjectives and adverbs organized into synonym sets (called synsets) and various relations between the synsets. For

each synset a short description is provided, for example descriptions of synsets containing noun 'fork' are following:

- fork – (cutlery used for serving and eating food)
- branching, ramification, fork, forking – (the act of branching out or dividing into branches)
- branch, fork, leg, ramification – (a part of a forked or branching shape; "he broke off one of the branches"; "they took the south fork")
- fork – (an agricultural tool used for lifting or digging; has a handle and metal prongs)
- crotch, fork – (the angle formed by the inner sides of the legs where they join the human trunk)

For nouns, following relations are available:

synonyms words with the same meaning

hypernyms synsets with more general meaning

hyponyms synsets with more specific meaning

coordinate terms words with similar meaning or used in the same context

holonyms words, which denote the whole, of which the selected synset is a part

meronyms words, which denote parts of the whole

derivations derived verbs, adjectives or adverbs

domain domain, in which the synset is used

domain terms other terms, characteristic for the domain

For verbs, similar relations are available. There are no hyponyms, holonyms and meronyms, but there are another relations, suitable for verbs. Coordinate terms, derivations or assignment to a domain are common for all parts of speech.

troponyms particular ways of doing something

antonyms opposite activities

sentence frames ways of use of the verbs in a sentence

For adjectives, synonyms, antonyms (if it makes sense), domain and 'value of' (e.g. *young* is value of *age*) are available. Adverbs are similar to adjectives, in addition they contain their adjective stem.

WordNet also contains an estimated frequency of usage of a word. The lexical base is not domain focused and covers wide range of topics, but the coverage is not balanced, for example it contains many detailed terms specific for biology and some other scientific domains are not covered in such detail. This wide coverage causes high degree of ambiguity because in general, domain-unfocused dictionary a single term can have many different meanings. But anyway, it contains valuable information for ontology learning To its popularity contributes also the fact that it is distributed with API for many programing languages, so researchers can connect it with their algorithms.

Various approaches to ontology learning can be also classified by the different knowledge sources, which can be used as input:

- free text
- terminology dictionaries or glossaries in machine readable form
- knowledge bases in other forms (i.e. from rule based systems)
- semi-structured schemata, such as XML schemas or other sources with predefined structure
- relational schemata – finding relevant concepts and relations from relational databases

Detailed overview of all these directions of research is given in [32], together with tools supporting them.

Ontology learning from texts and dictionaries

The widest range of approaches described is oriented to ontology learning from texts. This follows from two facts: The first one is that majority of human knowledge is expressed in general text, then textual resources are the most available ones, and therefore many researchers concentrate on them. The second one is connected complexity and variety of natural language processing possibilities – because there are many ways to process free text, many different approaches for learning ontologies from them are possible.

Ontology learning from machine readable dictionaries is very close to ontology learning from texts, because some techniques for learning are applicable for both the dictionaries and general texts. Such an example may be Hearst patterns, which are used to identify hyponymy or meronymy relations using domain independent lexico-syntactic patterns in form 'X is a Y' or 'X such as Y'. These patterns may be used to find the relations in general texts, but are best suited to dictionaries or glossaries, where these typical patterns are most frequent.

Ontology learning from texts relies on combination of various levels of text analysis, data mining and knowledge modelling. When dealing with free text, ontology learning takes strong inspiration and intertwines with *text mining* and *information extraction*. Their mutual relations can be characterised as follows:

- Both text mining and ontology learning seek, in a corpus of texts, *frequent terms* as well as tuples of terms with *frequent co-occurrence*. Ontology learning however aims at higher level of abstraction, via aggregating terms to more general classes. In terms of ontology engineering, aggregation of terms maps on concept taxonomies, while co-occurrence of terms maps on non-taxonomic relations.

- In contrast, information extraction retrieves *concrete occurrences* of tuples of terms from individual texts, and typically feeds them into a database. In terms of ontology engineering, tuples correspond to relation instances, i.e. facts, associated with certain concepts and relations from an ontology.

Ontology learning from non-textual information

Techniques for ontology learning from semi-structured data use for example the tree structure of XML data described by an XML Schema. Another technique finds common sets of RDF properties in resource descriptions to suggest new concepts for the corresponding ontology (if we, for example, encounter many book descriptions which refer to historical persons and their deeds or artworks, we may propose to add a concept 'biographic book'.

Learning from relational schemata includes reverse engineering of existing relational schema and mapping between database tables and ontology concepts or ontology relations. It is also used to populate ontology with instances from the relational database.

3.2.2 Tasks in ontology learning

Typically each ontology learning method addresses learning of one of the ontology components, because for each of the component different algorithms are appropriate. Another reason for the separation of the learning tasks is that some steps depend on the previous steps – for example, to be able to learn relations between concepts, we have to know the concepts we work with. From this point of view we distinguish these ontology learning tasks [7]:

1. Terminology extraction (acquisition of lexical items)
2. Handling of synonyms and multilingual variants
3. Concept formation
4. Extraction of *taxonomic relations*
5. Extraction of *non-taxonomic relations* and their lexical representations
6. Extraction of rules

Term extraction

Term extraction is a prerequisite for any other ontology learning from texts, since it connects the ontology concepts with their representations in an natural language. Term extraction was studied earlier in other contexts than ontology learning, for example in information retrieval for keyword based indexing. Term extraction usually uses simple linguistic processing, i.e. part-of-speech tagger together with a set of patterns expressing possible forms of terms[2]. in combination with a statistical processing step, which filters out rare random co-occurrences or which provides comparison of term distribution between generic and domain specific corpora. There are also methods, which build on deeper linguistic analysis.

Synonyms and multilingual variants

These are frequently identified by WordNet[3], or, in case of translations EuroWordNet[4].

[2] A simplified illustrating example of such pattern may be (ADJ)*(NN)+. It means that a sequence of nouns, preceded by optional sequence of adjectives may be a term

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

[4] http://www.elda.fr

There are also algorithms, which identify synonyms by clustering, building on hypothesis that terms which occur in similar contexts are similar in meaning.

Concept formation

In most of research this topic is addressed from linguistic or lexical point of view and is covered by previous two tasks. However, concepts in ontologies are more than sets of synonyms and their prospective translations to other languages. (Therefore, WordNet is not a real ontology; even though WordNet glosses may be considered as intensional definition, they are not formalized and thus not directly usable machine reasoning). Besides this *lexical information*, full concept specification should include an *intensional definition* of the concept (to allow reasoning) and a *set of concept instances* (its extension) [7].

Approaches to concept formation can be distinguished by focus on one of these three components of a concept. Extensionally oriented methods may build on lists or hierarchies of named entities or use information extraction to cover the extension of the concept. Ontology population is tightly related to ontology learning in this area, but it is only a part of the task – before population, it is necessary to identify the concept first.

Intensional concept learning may extract formal or informal definitions of concept. Extraction of formal definitions may build on formal concept analysis methods or it may use machine learning algorithms which produce decision rules or decision which can then serve as formal concept description. This approach borders with relation learning, since relations of concept to other concepts in ontology are important part of its formal definition.

Extraction of informal concept descriptions is quite rare approach represented for example in [40]. Since it is usual, that results of ontology learning methods are reviewed and further processed by an ontology engineer, informal concept descriptions may be helpful, even though the final ontology should provide formal and machine readable definitions. Furthermore, informal concept descriptions may be well appreciated by human users of the ontology, especially in non-technical domains.

Taxonomic relations

There are two main approaches to induction of taxonomies from textual data. The first one is linguistically oriented and searches the texts for lexico-syntactic patterns, which detect hyponymy relations (called Hearst patterns, introduced in [17]). There are also approaches, which analyze internal struc-

ture of noun phrases. The head of noun phrase corresponds to a parent concepts, noun phrases with various modifiers correspond to its subconcepts.

The second approach clusters concepts by contexts in which they appear. It builds on hypothesis that terms with similar meaning have similar lexico-syntactic contexts. The definitions of context may vary here, it may range from several neighbouring words through paragraph or chapter level to whole documents. (E.g. we may have documents describing statistical regression in general, and then documents about linear regression or about logistic regression. The general document will mention all three concepts frequently, whereas, in the specific documents, their main topic will gain the highest frequency).

Statistical methods for hierarchical clustering may be used process occurrences of concepts within observed contexts. Since ontology structure is a formal system where taxonomic relations are used for reasoning in logic, formal concept analysis is also a suitable method here. It builds concept lattices based on subsumption relation from a matrix of instances (i.e. ontology concepts in our case) and their attributes (occurrence of ontology concept in a textual context).

Non-taxonomic relations

Apart from above mentioned Hearst patterns, which can be used to find non-taxonomic relations such as holonymy or meronymy (part-of relation).

In other approaches, text mining methods are used, which combine statistical analysis with various levels of linguistic analysis. Frequent focus of work are verbs and the research may build on methods for acquisition of selection restrictions for verb arguments, studied in NLP.

Relation extraction for ontology learning may also build on association rules. This approach was introduced in [24] and is a basis for chapter 5 of this thesis.

Both taxonomic and non-taxonomic relations may be acquired also from non-textual sources, e.g. by reverse engineering relational database schema or from semi-structured data, such as XML schemas.

Rules

The extraction of rules is probably least researched area in ontology learning. There is a PASCAL lexical entailment challenge, which increased awareness of this topic and attracted researchers to address this problem, so we can expect increased research activity on this field in the near future. Main focus is to learn lexical entailments for application in question answering systems.

There are other ontology building tasks, which are not based on their target, but rather on their source, because they reuse existing ontologies to build domain specific ontologies: *ontology pruning, mapping and merging.* Again, WordNet is commonly used for this task, although it is not a true ontology.

3.2.3 Process of ontology learning

The tasks of extraction of taxonomic and non-taxonomic relations are not strictly separated, since some methods may be used for both of them (e.g. association rules or Hearst patters).

The learning steps may be (and usually are) performed repeatedly, because for each step there may be heuristics using information from a previous step. For example, we may find new concepts for ontology, based on knowledge of current concepts and relations between them, to be more concrete, in domain of artworks we can find words occurring within some context together with words 'Painting', 'Oil on canvas', 'Watercolour' to discover new possible terms for artistic techniques and suggest their inclusion to the ontology. Experience with ontology learning shows that no single approach is best and combinaiton of different methods seems promising.

Typically, there are four steps in ontology learning [24]:

1. Import and reuse existing available resources
2. Extract new items for the ontology
3. Prune the ontology
4. Refine the ontology

Import and reuse

When building a new ontology, it is usually possible to reuse some existing ontologies or supporting resources. There are top- or upper-level ontologies available, such as CyC[5] or SUMO[6]. Lists of relevant named entities may be also useful for the following extraction step. For example, we used TAP knowledge base in our experiments describet in section 5. If more ontologies are imported, ontology merging and alignment methods are applied.

[5] `http://www.cyc.com`

[6] `http://ontology.teknowledge.com/`

Extraction

In this step, various methods and algorithms are applied to find new relevant concepts and relations from available data (possible source data are described above in section 3.2.1). These algorithms may depend on current state of the prepared ontology, therefore iterative methods are used. For example, in the task of discovery of non-taxonomic relations, hierarchy of concepts may be used to find more general relations. These generalized relations have then higher support in data and also may be more comprehensive.

Pruning

For practical applications, it is necessary to balance completenes of the ontology with the needs of the apliaction. An ontology with too wide coverage may become inmanagable and computionally intractable. With too wide coverage also the problem of ambiguity of the lexicon arises, because one term may correspond to different concepts in different contexts. Especially import and reuse of existing ontologies may result in superfluos items in ontology, which are not relevant for the new one. The extraction of new items also may add 'too much'. The pruning must keep the ontology in a consitent state, removing a concept or a relation usually leads to further necessary changes.

Pruning is typically data driven and there may be several strategies for it. In the simple case, pruning may be based on lexical item frequencies. Concepts with lexical items, which do not appear, or appear very seldom in the set of documents, relevant for the domain are may be removed from the ontology. Their lexical items may be deleted (for unused concepts) or remapped to more general concepts (for concepts on too detailed level). In the more complex case, statistical distributions of lexical items of the domain specific document collection is compared to a generic reference collection.

Refinement

The role of this step is similar to the extraction step: to enrich the current ontology with new concepts and relations. In contrast to extraction, the goal is not to build the ontology, but fine tune it, to add only a few missing pieces. In the extraction step the ontology is in the stage of its creation, major changes of it are expected and the extraction algorithms may work independendently of it, during refinement, the algorthms have to consider the ontology in detail. The refinement step is also used to update an existing ontology with new concepts arising in the domain.

Chapter 4

Information Extraction and Ontology Learning Guided by Web Directory

This chapter presents our effort to create an information extraction tool for collecting general information on products and services from the free text of commercial web pages. A promising approach is that of combining information extraction with ontologies. Ontologies can improve the quality of information extraction and, on the other hand, the extracted information can be used to improve and extend the ontology.

We describe the way we use Open Directory as training data, analyse this resource from the ontological point of view and present some results related to information extraction.

4.1 Motivation for this work

Lack of explicit semantics and, consequently, poor machine understandability are commonly known problems of the current World Wide Web and motivations for creation of *semantic web*, as described in 2 In order to excavate *implicit* semantics from the full text of web pages, we can take advantage of both:

- Collections of operational *extraction patterns* (most often, in the form of rules) that specify at which points in the stream of (marked-up) text valuable information should be taken over. The nature of the patterns can be linguistic or surface-form-based (e.g. regular expressions).

- *Ontologies of problem domains* consisting of both the conceptual and lexical part. The identification of lexical items in the text leads to the abstraction of generic concepts, which can, in turn, be used as classes for extracted textual metadata characterising the web pages.

The dividing line between the extraction patterns and lexical ontologies is not always clear; we can roughly distinguish the patterns as being (to some extent) structural and having a lower degree of domain dependency.

A promising approach is that of combining information extraction with ontologies. Ontologies can improve the quality of information extraction and, on the other hand, the extracted information can be used to improve and extend the ontology, see [25] A common strategy for this process is *bootstrapping*: a certain amount of manually labelled training data is initially provided, which serves for iterative labelling of unseen data associated via some properties with the original data. We however assume that the amount of manual labelling can be further restricted via the *reuse of public resources* with similar content and structure as the target knowledge.

The goal of our effort described here is to extract information about (mostly generic) *products, services and areas of competence of companies*, from the free text chunks embedded in web presentations.[1] For this sort of information, an abundant reusable resource are web directories such as Yahoo! or Open Directory. We have based our experiments on the 'Business' branch of *Open Directory* (`http://dmoz.org`). Both the hierarchy of the *directory headings* and the categorization of *links* listed in each node are valuable sources of information. From the categorization of web links we can obtain *labelled* training data for information extraction, while the hierarchy could be used as source for building a (lightweight) ontology of the domain corresponding to the given branch.

Manual construction of proper extraction rules is practically impossible, so we follow common approach and build them (semi-)automatically. Of course, this requires some training data and task involving natural language require large amounts of training data. Solutions to this problem include unsupervised learning methods or acquisition of existing training data. On the web exists lots of data which can be acquired cheaply, but they are not created as training data for some specific task. i

[1]Currently, we do not consider other company information such as cooperation with other companies or financial results, which is much sparsely present in common web pages. We also ignore the possibility to extract company information (as a specific sort of web page metadata) from the micro-level structures of *HTML mark-up*, which is the subject of a project running in parallel.

4.2 Mining indicator terms through directory headings

The general description of the company profile, area of competence, products and services is usually not too extensive but stylistically well-formed. This favourises the use of deeper *linguistic* techniques, in contrast to surface techniques (such as regular-expression-based), which are often used for information extraction from idiosyncratic, abridged documents (e.g. advertisements or medical records).

Our assumption is that the *directory headings* (such as `.../Manufacturing/Materials/Metals/Steel/...`) coincide with the generic names of products and services—let us denote them *informative terms*—offered by the owners of the pages referenced by the respective directory page. By matching the headings with the page full texts, we obtain sentences that contain the informative terms. The terms situated near the informative terms in the syntactical structure of the sentence are candidates for *indicator terms*, provided they occur frequently on pages from various domains. The resulting collection of indicator terms can be, conversely, the basis of extraction patterns for discovering informative terms in previously unseen pages.

The knowledge asset embedded in web directories is the judgement of human indexers who have assigned the pages under the particular heading(s). Naturally, informative terms on the page need not always correspond to the existing directory headings, e.g. due to synonymy. As consequence, our method will extract (without the help of a thesaurus) only a fraction of the sentences with informative terms. This however does not disqualify the method, since, in this training phase, we aim at discovering indicator terms rather than at identifying the informative terms themselves. The small degree of completeness of the method is actually compensated by the hugeness of the material available[2] in the directories. Namely, the 'Business' subhierarchy of Open Directory that we have exploited in our experiments points to approx. 150,000 pages overall, each of these containing the 'heading' terms (from the referencing node or one of its ancestors) in two sentences, on the average.

We have tested the training phase of our method on a sample of 14,500 sentences[3] containing the 'heading' terms. The syntactical analysis has been carried out using the free *Link Grammar Parser*[4] [34]. Our working hy-

[2]As we dispense with manual labelling, processing a larger sample of data is merely the matter of computer time/storage.

[3]I.e. about 5% of the total of such sentences.

[4]The choice was motivated partly by the immediate availability of the parser, partly

pothesis was that the aforementioned indicative function is, in most cases, conveyed by *verbs* (and verb phrases). Therefore, in the initial experiments, the verbs that occurred the closest (in the parse tree) to informative terms have been counted, arranged into a frequency table, and ordered by ratio of their relative frequency of occurrence near some informative term to their relative frequency in general. Eightmost promising verbs have been chosen for the experimental collection. Most of these are likely to be associated with informative terms, e.g. 'our assortment *includes*...', 'we *manufacture*...', 'in our shop you can *buy*...'. Results of testing the indicators are available in Tab. 4.1.

For the test, 130 sentences containing some indicators were randomly selected and each of them was *manually labelled*. The labelling amounted to the subjective estimation whether the sentence contains the target informative terms or not. This is sometimes difficult—e.g. due to missing context, special terminology and domain specific product names; see for example the sentence:

> We are equipped to run any grade of corrugated from E-flute to Triplewall, including all government grades.

Therefore, some unclear sentences were labelled with '?' and then counted once as negative and once as positive test cases. Some sentences contained the company name but no usable information on the products, e.g.

> Industrial Metals Inc. is committed to provide you with exceptional service.

Although named entities are often valued in the information extraction field, we considered these sentences as negative test cases, too, since we focus on *generic* names of products/services or of their providers. The testing results (including ad-hoc inspections not covered by the presented table) suggest that some general[5] verbs–such as 'use' or 'include'–need to be extended to more complex *phrases*, possibly again via selecting the neighbouring terms with frequent occurrence. Also, clearly, certain nouns and noun phrases could play the role of indicators, too.

Due to the tedium of the aforementioned manual labelling, we are not able to measure directly the *coverage* of a collection of indicators: this would amount to considering the full set of sentences in the selected sample of web

by the hypothesis that a linked-based parser could support the presumed 'navigation' over the dependency structures better than parsers based on constituent grammars.

[5]Even the verb 'to be', which has no significance of its own, could presumably be the starting point for finding useful indicator phrases.

Table 4.1: Test of the indicative verbs

indicator	−	?	+	precision
include	8	4	18	60–73%
provide	9	3	28	70–78%
offer	6	1	21	75–79%
specialize	0	1	18	95–100%
(other)	3	5	5	38–77%
total	26	14	90	77–80%

pages. An indirect measure of coverage, which can be obtained automatically, is the number of pages in the sample that contain one or more indicators from the collection. On the pages directly referenced by directory nodes, this measure was rather low, between 10-20%; however, if we manually prefiltered out pages with no or minimal free-text content (such as intro or menu pages), the proportion increased to 70-80%: the fact that this result was obtained for a collection of *eight* indicators suggests that the cross-domain variability of these terms might be relatively limited. Note that, even if a set of indicators could not directly be used, due to low coverage, for systematic filling of information extraction templates, it could still be acceptable for the discovery of new terms for the *ontology of products and services*, see section 4.5.

4.3 Integration of indicator-based analysis into a modular architecture

Indicator-based linguistic analysis, as described here, has only limited capabilities with respect to the heterogeneous content of commercial web pages. In order to bring useful results, it is thus being integrated into a modular architecture currently under development. The central idea of the architecture, named *Rainbow*[6] [38] (Reusable Architecture for INtelligent Brokering Of Web information access) is the separation of different web analysis tasks according to the *syntactical type of data* involved. Communication within *Rainbow* is based on the simple *SOAP* [3] communication protocol. Services provided by the individual modules – acquisition of data from the web, conversion to well-formed XML, different forms of semantic analysis of data

[6]Beyond the acronym, the name is motivated by the idea that the individual modules for analysis of web data should synergistically 'shed light' on the web content, in a similar way as the different colours of the rainbow join together to form the visible light.

and, finally, visualisation of results – are described by means of *WSDL*, the Web Service Description Language [10]. Indicator-based linguistic analysis, as described in this section, has been implemented as one of the web services within the first prototype of *Rainbow*, currently in the form of sentence extraction. The 'interesting sentences' are part of the output of the visualisation component, which can be installed as a plug-in panel of the Mozilla browser. In addition to linguistic analysis, *explicit metadata* (in META tags) are currently processed; moreover, *similar pages* are displayed thanks to the respective web service provided by Google.

For the next version of the architecture, an earlier-developed URL analyser [37] is being adapted; separate modules for the analysis of HTML structures, inline images, and link topology structures are also under design. Shared domain *ontologies* will serve for verification of semantic consistency of web services provided within the distributed system. Clearly, an advanced version of the architecture should be able to overcome the mentioned problem of directory links pointing to the 'barren' pages of the particular website: analysis of keywords and HTML structures on the start-up pages, as well as of the URLs of embedded links, will navigate the proper metadata extractor towards the most promising pages or page sections. Such parts of company websites, named e.g. *about-us*, *profile* etc., are quite common and usually contain larger segments of syntactically correct text.

4.4 Ontological analysis of web directories

Web directory hierarchies are sometimes mistaken for ontologies; however, as already observed by Uschold [39], they are rarely valid taxonomies. It is easy to see that subheadings are often not specializations of headings; some of them are even not *concepts* (names of entities) but *properties* that implicitly restrict the extension of a preceding concept in the hierarchy. Consider for example `.../Industries/Construction_and_-Maintenance/Materials_and_Supplies/Masonry_and Stone/Natural_Stone/International_Sources/Mexico`.

Semantic interpretation of a representative sample of directory paths has revealed that

- terms and phrases in individual headings belong to quite a small set of *classes*, and
- surface 'parent-child' arrangement of headings belonging to particular classes corresponds (with a certain degree of ambiguity) to 'deep' ontological *relations*.

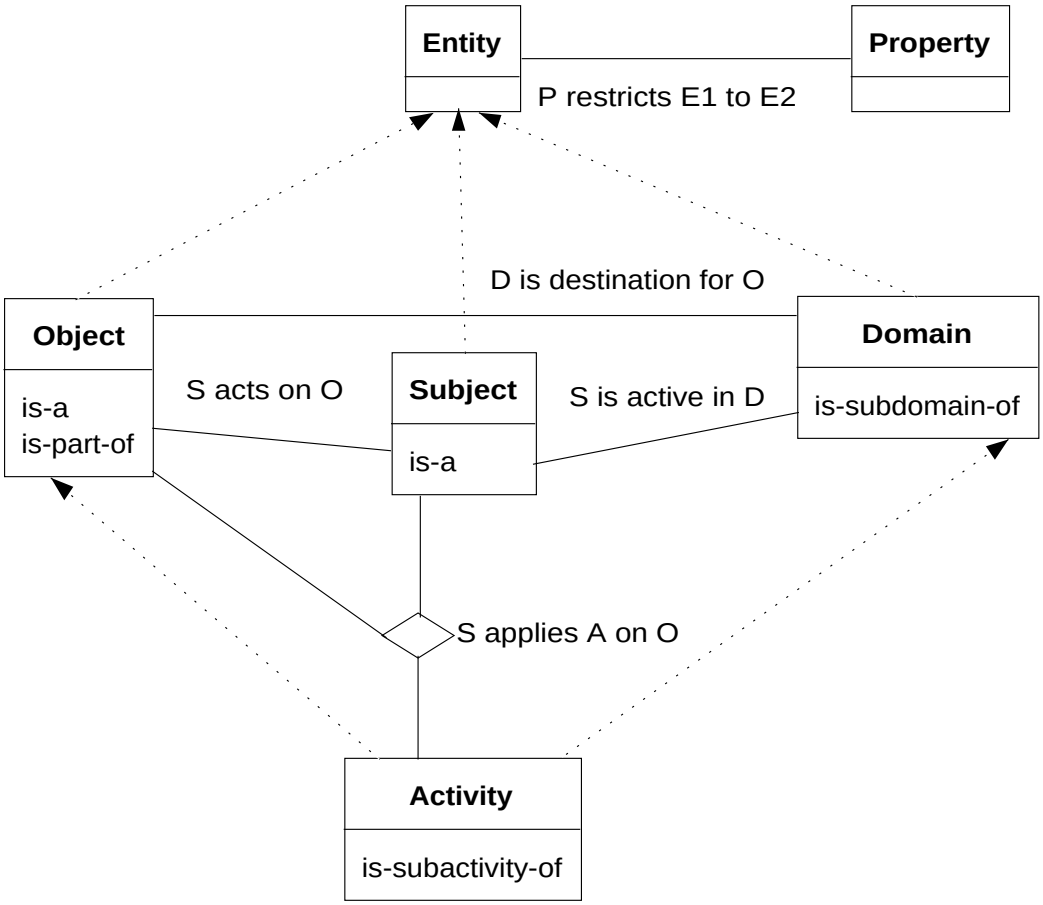

Figure 4.1: The ontology of web directory headings

The result of this effort was a *meta-ontology of directory headings* plus a collection of *interpretation rules*. The diagram at Fig. 4.1 depicts the essence[7] of the *meta-ontology*. Boxes correspond to classes, full edges to named relations, and dashed edges to the class-subclass relationship. Reflexive binary relations are listed inside the respective boxes. Examples of informally expressed *interpretation rules* are in Tab. 4.2.

[7]For better readability, we have e.g. omitted the notion of 'Location', which may also be important to extract but is not directly related to the commercial profile of the company.

Table 4.2: Examples of interpretation rules

Rule no.	Path pattern	Ontology relation
1	Subj/Prop	'Prop_Subj' *is-a* Subj (or, Prop *restricts* Subj *to* 'Prop_Subj')
2	Dom1/Dom2	Dom2 *is-part-of* Dom1
3	Obj1/Obj2	Obj2 *is-a* Obj1
4	Dom/Prop	'Prop_Dom' *is-part-of* Dom

Rule no.	Example
1	`Publishers/Academic_and_Technical`
2	`Security/National_Security`
3	`Electric_Motors/AC_Motors`
4	`Manufacturing/Electrical`

4.5 Information extraction and ontology learning

Plain indicator terms, gathered by means of the fully automated technique described in section 4.2, are by themselves powerful enough to extract *sentences* that are likely to contain *some kind of* interesting information about the company. We can even, in many cases, access this information thanks to simple heuristics over the parse-tree, such as:

> If the immediate object of the *indicator verb* is a generic *set-semantic expression* such as 'range of', 'family of', 'assortment of' etc. then output the *indirect attribute* of the object; otherwise output the *object* itself.

Universal extraction patterns however impose strong assumptions on the whole collection of indicators. A more sensitive method should take account of the *classes* of indicators/headings revealed by ontological analysis. If we learn the indicators for each class of information (such as 'subjects', 'objects' or 'domains') separately, we could be able to perform true *information extraction* in the sense of filling database templates. Conversely, if the informative terms thus discovered coincide with the headings of directory nodes referencing the particular page, we can automatically 'restore the identity' of these headings. With the help of generic interpretation rules such as those shown in Tab. 4.2, fragments of true taxonomies (possibly several interconnected ones, for 'subjects', 'objects'..., as specified by the meta-ontology) could be

built. We can understand this as a two-step *ontology learning* process using two resources: text and the hierarchies of headings. Obviously, the result of this process will still be rather incomplete, and should be enhanced using other ontology-learning techniques, taking into account co-occurrences (and linguistic dependencies) of terms in the text beyond the headings.

These two tasks represent a *closed loop*: as soon as we have classified the headings, we can learn class-specific indicators[8]. From the other side: as soon as we have class specific indicators, we can use them for the classification of headings. Since the first step in this loop has to be done by a human, a more viable approach seems to be that one starting by *classifying the directory headings*. For this task we could use the WordNet lexical database. One reason for this are some regularities and similarities in the structure of Open Directory: some of the headings could thus be even classified semi-automatically with the help of heuristic rules. Another interesting possibility is to classify the headings by matching them to a generic lexical ontology such as *WordNet*.

4.6 Related work

The combination of information extraction and ontology learning has previously been described by Maedche [25]. The main novelty of our approach is the use of a public *web directory*.

Li, Zhang and Yu also use the Link parser and describe in [23] how to learn mapping from the link grammar to RDF statements. Their work shows advantages of link grammar over constituent grammar for this task and demonstrates feasibility of this task.

While directories have already been used for learning to classify *whole documents*, by Mladenic, [31], their use for *information extraction* seems to be innovative.

There is also some similarity to Brin [5], which targets on automated discovery of extraction patterns using *search engines*. The patterns can be used to find relations, such as books, i.e. pairs (author, title). However, the patterns are simply based on characters surrounding the occurrence of the investigated relation. In comparison, we aim at finding less structured information, for which such simple patterns wouldn't be sufficient.

Finally, the use of bootstrapping and other statistical methods for information extraction has also been presented e.g. in [28] and [33].

[8]The class specific indicators will apparently be more complex than the current ones.

Chapter 5

Discovery of Lexical Entries for Non-Taxonomic Relations in Ontology Learning

In 3 we described techniques of *ontology learning* (OL), which has been suggested as promising technology for building lightweight ontologies with limited effort.

In [24], three core subtasks of OL have systematically been examined: lexical item extraction (also used for concept extraction), taxonomy extraction, and *non-taxonomic relation*[1] *extraction*, considered as most difficult.

Two variants of the task are distinguished (e.g. in [14]):

- The name, to say, (semantic) *type* of the relation can be specified beforehand. We could, for example, search for pairs of companies such that one *acquired* another. This is a well known information extraction (IE) task addressed at MUC conferences since more than a decade, with template-based methods being quite successful.

- Alternatively, we may 'mine' for undefined relations. In this scenario, pairs of frequently co-occurring terms have first to be identified by *text mining*. Each term is then assigned *semantic type* (e.g. *Company* or *Product*). The type of relation is then assessed by the semantic type of terms involved (assuming e.g. that companies *produce* products [14]) and/or by characteristic terms (verbs, adjectives) occurring in the neighbourhood of the terms.

[1] Although it might be useful to distinguish the terms 'relation' and 'relationship' (set of tuples vs. high-level association between concepts), we mostly speak about 'relations' since this term is systematically used in the ontology engineering community.

The non-taxonomic relation extraction technique [26] embedded in the *Text-to-Onto* tool [27] of the KAON system[2] produces, based on a corpus of documents, an ordered set of binary relations between concepts. The relations are reviewed and *labelled* by a human designer and become part of an ontology. Empirical studies [24] however suggest that designers may not always appropriately label a relation between two general concepts (e.g. 'Company' and 'Product'), even if they know that *some* relation between them has evidence in data. First, various relations among instances of the same general concepts are possible; for example, a company may not only *produce* but also *sell*, *consume* or *propagate* a product. Second, it is often hard to guess which among synonymous labels (e.g. 'produce', 'manufacture', 'make'...) is preferred by the community. *Lexical items* picked up from domain-specific texts thus may give an important cue.

This chapter is organised as follows. Section 5.1 describes how texts are processed to enable further quantitative analysis, i.e. how concept and verb co-occurrences are defined and identified in text and how data about these co-occurrences are structured, so that we can apply association rules mining algorithm on them.

Section 5.2 describes the principles of our method and quantitative criteria for choosing lexical items (namely, verbs) to be suggested as relation labels. Section 5.4 presents and discusses the results of an experiment in the tour-planning domain. Section 5.6 compares our approach with related research. Finally, section 6 summarises the experiments and outlines directions for possible future work.

5.1 Text modelling

A standard approach to *relation discovery* in text corpus is derived from *association rule learning* [1], originally applied on relational data.

In the *text-mining setting,* which is a basis for ontology learning from texts, the free flow of text must be transformed to structure, suitable for further analysis, in our case association rule learning. Association rules are induced from sets of items observed together, which are called *transactions*. For example, typical application of association rules mining, market basket analysis, focuses on sets of items bought frequently together. This is also the reason, why the sets are called transactions – in this scenario they really represent a transaction between buyer and seller.

So, in the text-mining setting, two (or more) lexical items are understood as belonging to a *transaction* if they occur together in a document

[2]Karlsruhe Ontology infrastructure, `http://kaon.semanticweb.org`.

or other predefined unit of text; frequent transactions are output as *associations* among these items. Furthermore, ontology learning tools discover binary relations not only for lexical items but also for ontological concepts [26]. This presumes existence of a semantic *lexicon* (mapping lexical items to underlying concepts) and preferably a *concept taxonomy*, which enable aggregation of relation instances along the 'is-kind-of' and 'is-a' axes.

Modification of the method, which is the subject of this work, relies on an extended notion of transaction. Following up with our prior work on lexical item extraction from business websites presented in chapter 4, we hypothesised that the 'predicate' of a non-taxonomic relation can be characterised by *verbs* frequently occurring in the neighbourhood of pairs of lexical items corresponding to associated concepts. Information about the verbs is present in the texts, but it gets lost when the texts are transformed to a set of concept co-occurrences.

Text pre-processing

First, we need to preprocess the texts, to be able to identify the co-coccurences of lexical items (and corresponding concepts).

The pre-processing of texts in Text-to-Onto follows an usual approach in NLP: in the first phase the text is tokenized onto tokens. The tokens are individual words, numbers, acronyns or punctuation marks such as '.', ',', '!', '...' and so on. The list of tokens is processed by part-of-speech tagger, which assigns a POS-tag to each of the tokens. Furthermore, stem is assigned to each word token by a stemmer.

Further, *ordinal numbers* are assigned to each token, these numbers are used later to determine *distance between concepts* and *distance between a concept and a verb*. These distances determine, which concept pairs are considered as possibly related and which verbs may be candidates for the relation label. Within a document, the ordinal numbers must be increasing, but the step may vary. This approach allows us to enforce some conditions on considered pairs without modification of the pair discovery algorithm: e.g. by using prohibitively high number for a step crossing a paragraph boundary it is possible to limit the search of the pairs within one paragraph of text only. Similarly, using higher step when crossing sentence boundary favors concept pairs occurring within one sentence.

In the next steps, we identify the information, which we need to be able to discover the relations between concepts: *concept occurences* and *verb phrases*.

These steps are domain independent and so they do not depend on the ontology, for which we seek the relation labels.

5.1.1 Concept occurrences

For the relation extraction task we need the occurrences of concepts in the text. In the ontology model, each concept may have lexical items. Apart from 'standard' term corresponding to the concept, there may be synonyms and also a stemmed version of the concept. These lexical items are used to find occurrences of the concepts in the texts. This way the ontology provides us with a dictionary of the domain, terms covered by this dictionary are recognized, the others are ignored.

5.1.2 Verbs and verb phrases

For the relation labels we need to identify verbs and verb phrases. This step is based on POS tagging. There are several POS-tags for verbs:

- VB for usual verbs in present tense and VBD for verbs in passive or perfect tense
- MD for modal verbs
- HV (stands for 'have' in past perfect), BE, and DO for auxiliary verbs,

Modal verbs are ignored, since the meaning is expressed by following normal verb. Of course, modality expreses important meaning too, but it usually expresess a property of a single class (i.e. 'birds can fly') and here we are concentrated on finding possible relations between two classes.

When an auxiliary verb is encountered, the meaning has to be specified more precisely. If it is followed by another verb, then we expect that the meaning is expressed by it and the auxiliary verb is ignored. If the auxliliary verb is followed by an noun phrase (object of the verb), the noun is appended to the auxiliary verb to form a meaningfull verb phrase.

5.1.3 Co-occurrences of verbs and concepts

Having concept occurrences and verb phrases collected, we will study the ways they occur together. This approach builds on a heuristic assupmtion that two concepts are related, if they occur close to each other in the text and similarly for a pair of a concept and a verb.

A good relation label should be typical or characteristic for the pair of concepts and at the same time specific for them, i.e. the suggested verb phrases shoudn't occur frequently with many other concept pairs.

From the occurrences of verbs and occurrences of concepts we focus on following co-occurences:

- pairs of concepts
- pairs concept–verb
- triples concept–concept–verb

The co-occurences are identified by a simple rule: if the items (verbs, concepts) occur in surface distance closer than a chosen threshold, the co-occurence is counted. (The surface distance is determined as a difference between ordinal numbers assigned to tokens during tokenization. As mentioned above, the numbering may take into account punctuation and paragraph boundaries, or even chunk boundaries, if a chunker were used)

We expect that two concepts may be semantically related, even though there is no direct syntactic relation in the analyzed text. They may be mentioned in two separate sentences. On the other hand, the verbs suggested as labels for the relation will typically be also syntactically related to the concept occurence. Therefore, there are two thresholds used: one for concept–verb distance and another for concept–concept distance.

Definition 1 *CC(n_c)–transaction holds among concept c_1 and concept c_2 iff c_1 occurs within n_c words from an occurrence of c_2.*

Definition 2 *VCC(n_c, n_v)–transaction holds among a verb v, concept c_1 and concept c_2 iff c_1 and c_2 both occur within n_v words from an occurrence of v and $CC(n_c)$ holds.*

In the experiments described further we heuristically set the thresholds to 8, which takes into account possible articles, prepositions, adjectives or nested clauses in a sentence. With too small threshold we would lose some important relations between concepts or candidates for labels of such relations. Furthermore, the counts for estimating co-occurrence probabilities would be too low and the probablity estimates would be unreliable. On the other hand, with too high threshold, many unrelated items would be considered as related and this would introduce noise.

Eight words would probably be too large a distance in general language processing. We however do not count noun-verb (or noun-noun) pairs, but concept-verb (or concept-concept) pairs instead, i.e. only occurrences of terms contained in the ontology lexicon are considered. So even if many unrelated nouns appear within the window, they are not counted, because they are not covered by the ontology lexicon.

By ignoring the *order* of concepts and verb and because of *stemming*, passive and active sentences are treated equally in our approach: this avoids us from usage of deep parsing. Let's consider two sentences:

1. ‘Many tourists visit this museum to see. . . ’

2. ‘This museum is visited by many tourists, who come to see. . . ’

Because the order of concepts is not important and all lexical items are stemmed, both sentences yield the same triple (‘museum’,’tourist’, ’visit’), which is desirable.

On the other hand, we cannot capture the differences in meaning of sentences like ‘A company was hired by a person to accomplish some task’ vs. ’A company hired a person to accomplish some task’. It however seems that achieving higher frequencies for concepts an labels is more important here, especially when at the end, a human will judge the triple. In our example, we expect that the ontology engineer knows that a company may hire a person as well as a person may hire a company, and will appropriately model both relationships in the ontology. Although comprehensive text analysis addressing the aspects of tense might still more ease the role of ontology engineer, its costs would probably not outweigh its benefits.

5.1.4 Implementation

The computation of VCC(n) transactions and associated frequency measures has been implemented as a modification of the relation extraction module of the *Text-to-Onto tool* [27]. Resulting concept–concept–verb triples are shown in a separate window popping up from its parent window of ’bare’ relation extractor, upon choosing one or more among the relations. A screenshot of KAON environment is at Fig. 5.1; note the list of verbs potentially associated with relations between ’Country’ and ’City’, in the front window. In addition, complete results are output into a textual protocol.

To compute the VCC transactions and the AE scores, the concept and verb occurrences were stored to an SQL database. Figure 5.2 shows the structure of the database on the conceptual level. On physical level, lookup table ‘concept’ was added, which contains concept ID and concept URI for performance reasons. The table for concept locations contains only integer document ID, concept ID and concept position. Storing full concept URI with each occurence would result in much larger table, moreover it, we need to do grouping based on concepts. Tables for concept locations and verb locations are named `cloc` and `verbloc`. Columns `cp` and `vp` denote the concept and verb position. The numbers 8 in the inequalities correspond to the thresholds for CC and VCC transactions.

From this database structure, we can identify the CC and VCC transactions by following queries:

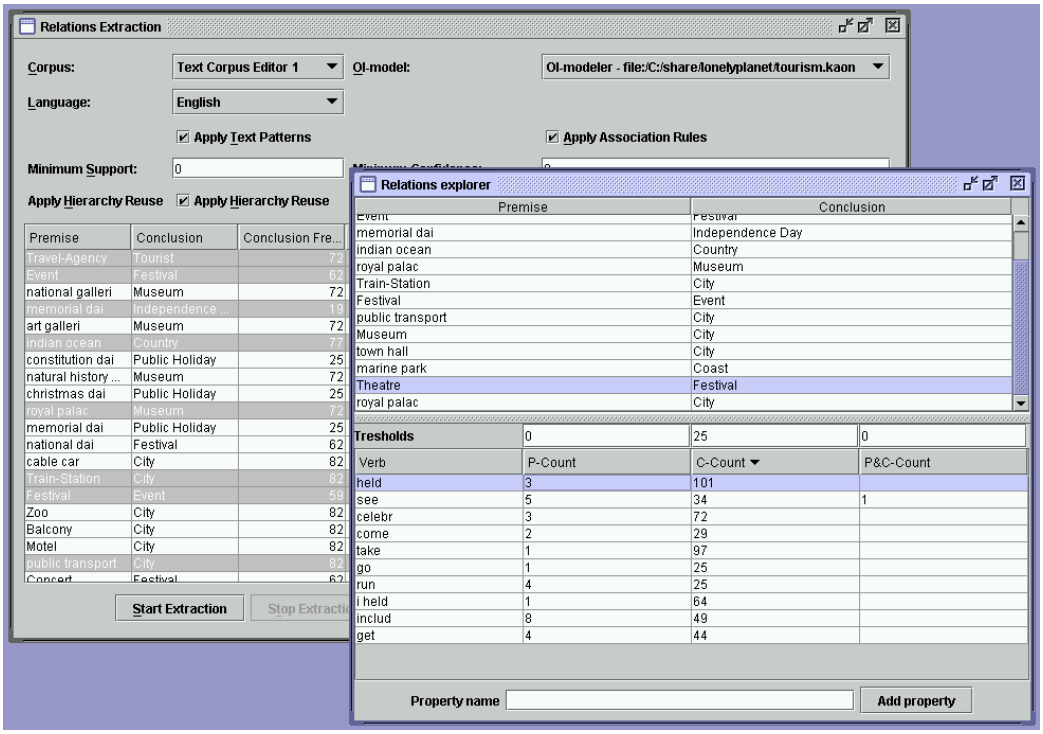

Figure 5.1: KAON environment with interface for non-taxonomic relation discovery

```
insert into cctransactions
select t1.docid,
  t1.concept as c1, t1.cp as p1
  t2.concept as c2, t2.cp as p2
from cloc t1 join cloc t2 on
  (t1.docid=t2.docid and t1.cp<t2.cp and t2.cp-t1.cp<=8);

insert into vcctransactions
select t1.docid,
   t1.concept as c1, t1.cp as p1,
   t2.concept as c2, t2.cp as p2,
   verb, vp
from cloc t1 join cloc t2 on
  (t1.docid=t2.docid and t1.cp<t2.cp and t2.cp-t1.cp<=8)
left join verbloc on
  (verbloc.docid=t1.docid and abs(vp-t1.cp)<=8
   and abs(vp-t2.cp)<=8);
```

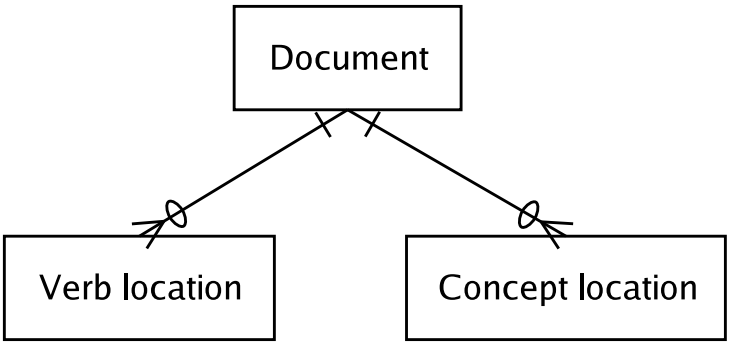

Figure 5.2: Conceptual diagram of the database of concept and verb locations

It may seem, that the queries are computionally very expensive, but the conditions on equality of `docid` together with indices on `docid` reduce the complexity greatly.[3]

5.2 Seeking Labels for Relations in *Text-to-Onto*

5.2.1 Method Description

Good candidates for labelling a non-taxonomic relation between two concepts are the verbs frequently occurring in VCC(n) transactions with these concepts, for some 'reasonable' n. Very simple measure of association between a verb and a concept pair are conditional frequencies (empirical probabilities)

$$P(c_1 \wedge c_2/v) = \frac{|\{t_i|v, c_1, c_2 \in t_i\}|}{|\{t_i|v \in t_i\}|} \tag{5.1}$$

$$P(v/c_1 \wedge c_2) = \frac{|\{t_i|v, c_1, c_2 \in t_i\}|}{|\{t_i|c_1, c_2 \in t_i\}|} \tag{5.2}$$

where $|.|$ denotes set cardinality, and t_i are the VCC(n)-transactions. The first one helps to find concept pairs possibly associated with a given verb; the second one helps to find verbs possibly associated with a given concept pair.

[3]On a workstation with 1.8GHz Athlon CPU and 256MB RAM and PostgreSQL database server the computation of vcctransactions for the data from the LonelyPlanet corpus takes cca 8 seconds

However, conditional frequency of a pair of concepts given a verb is not the same as conditional frequency of a *relation* between concepts given a verb. A verb may occur frequently with each of the concepts, and still have nothing to do with any of their mutual relationships. For example, in our first experimental domain, lexical items corresponding to the concept 'city' often occurred together with the verb 'to reach', and the same held for lexical items corresponding to the concept 'island', since both types of location can typically be reached from different directions. Therefore, conditional frequencies $P(City \wedge Island/'reach')$ and $P('reach'/City \wedge Island)$ will be relatively high, and might even dominate those of verbs expressing a true semantic relation between the concepts, such as 'located' (a city is located on an island).

To tackle this problem, we need a measure expressing the *increase* of conditional frequency, as defined in (5.1) and (5.2), compared to frequency expected under assumption of *independence* of associations of each of the concepts with the verb. Our heuristic 'above expectation' (AE) measure thus is:

$$AE(c_1 \wedge c_2/v) = \frac{P(c_1 \wedge c_2/v)}{P(c_1/v).P(c_2/v)} \tag{5.3}$$

(the meaning of $P(c_1/v)$ and $P(c_2/v)$ being obvious). This measure resembles the 'interest' measure (of implication) suggested by Kodratoff [21] as operator for knowledge discovery in text[4]. The 'interest' however merely compares the relative frequency of a pattern (in data) conditioned with another pattern, with its unconditioned relative frequency. Our AE measure, in turn, compares a conditional frequency with the product of two 'simpler' conditional frequencies.

We could also reorder the triples by an alternative measure, $AE(v/c_1 \wedge c_2)$: this would yield (possibly even more useful) information on which verbs most typically occur with a certain relation.

5.3 Performance Evaluation Techniques

A straightforward analogy of other ontology learning evaluation techniques (see [24] as well as most papers in this volume) is to compare the results of labelling with relation names from a *reference* ('gold standard') ontology created by human evaluators upon reading/browsing a sample of texts. The

[4]There is also some similarity with statistical measures such as χ^2. These however involve applicability conditions that are hard to meet in OL, where a high number of relatively infrequent features have to be examined.

precision and *recall* measures, well-known from information retrieval, can be used to quantify the results. However, as mentioned above, finding suitable names for non-taxonomic relations is more tedious for humans than just listing concepts or even building a concept taxonomy. Moreover, the reliability of 'gold standard' design can be assured, for other tasks, by presenting to the human designer an (almost) exhaustive list of candidate patterns, such as frequent terms (for concept extraction) or concept pairs (for suggestion of taxonomic or anonymous non-taxonomic relations). Names of relations, on the other hand, are linked to lexical items much more loosely than names of concepts[5]: partly because they are not reflected at the lexical level at all, partly because they are dispersed in large synonym sets, and partly because they only pertain to a small subset of occurrences of a term[6]. By consequence, many 'correct' relations would presumably be missing in the reference ontology. An evaluation method exclusively relying on matching relation names from reference ontology with subsequently learnt labels thus might improperly penalise the labelling tool in terms of precision. The solution is to employ two types of precision[7]: *prior* (with respect to reference ontology built prior to learning) and *posterior* (with respect to posterior evaluation of learning results). The latter may be subjectively biased (since the expert may directly control the evaluation result) but makes up for human omissions.

Let us now elaborate on this general idea towards a possible *procedural scenario*. Given a previously extracted collection of concepts C, arranged into a taxonomy, the evaluation of extracted relation labels may look as follows:

1. A domain expert *suggests possible named relations* for all pairs of concepts. We thus obtain a set of reference (concept-concept-label) triples that forms, together with the original taxonomy, a *reference ontology*. Since the number of such pairs might be large, only a subset of concepts, $C^* \in C$, could actually be used. Low-level concepts should be pruned as relations among them are less likely to achieve sufficient frequency counts. On the other hand, a few top-level concepts might be pruned as well in some situations since the interpretation of associations among them would be too uncertain. Obviously, the concept-pruning strategy impacts the evaluation results.

[5]For example, Maedche [24] showed that only 10-15% of human-provided relation labels were found among extracted lexical items, versus 20-25% for concept labels.

[6]While the first two aspects also represent inherent limits for any labelling tool, the third is a specific hindrance to reference ontology design based on a set of extracted frequent terms.

[7]Recall, on the other hand, can only be computed as 'prior'.

2. The labelling tool to be evaluated is run on the document collection and suggests a set of labels for each concept pair from C.

3. The empirical labels are *compared* for equality or synonymy with labels suggested by the expert. The comparison can be carried out either merely by human judgement or using a lexical resource such as WordNet.

   One of the following types of (non-)match with the reference ontology is identified for any learnt concept-concept-label triple $t = (c_1, c_2, lab)$ such that $c_1, c_2 \in C^*$:

   - t directly matches some *reference triple* (concept pair with verb suggested by expert) $t' = (c_1, c_2, lab')$, i.e. lab and lab' are synonyms or (provided human judgement is used) reflect the same relation between c_1 and c_2
   - t could be matched with a reference triple if c_1 and/or c_2 are properly generalised/specialised in the concept taxonomy
   - t could be matched with a reference triple if lab is replaced with a hyper/hyponym (this would only work if a proper lexical resource is used)
   - combination of the previous cases
   - no match can be found even across taxonomies of both types.

   These situations can be used to compute both *prior precision* and *recall* of labelling, with respect to the set of triples in the reference ontology. Prior precision is the proportion of learnt triples that match some reference triple. Prior recall is the proportion of reference triples that match some learnt triple. Partial match via taxonomies (detected in the previous phase) can either be taken into account or not.

4. Learnt triples $t = (c_1, c_2, lab)$ not (or incompletely) matching with reference triples, i.e. such that either $c_1 \notin C^*$, $c_2 \notin C^*$, or lab is not synonym of lab' from any reference triple $t' = (c_1, c_2, lab')$, are *submitted to the expert* for posterior evaluation.

5. The expert may declare some of the non-matching learnt triples as relevant, and *augment* accordingly the set of correct hits. Two different augmentation variants are possible, 'strict' and 'relaxed':

   - *Strict augmentation*: a triple only becomes relevant if it should have been part of the reference ontology, i.e. the non-match was due to omission.

- *Relaxed augmentation*: a triple always becomes relevant if the expert judges it as a meaningful relation; it may thus not necessarily be relevant for the application domain of the ontology.

6. *Posterior precision* is computed as proportion of reference triples that all marked as correct hits.

Note that the distinction of prior and posterior precision can in principle be applied on any ontology learning task; in relation labelling, however, the span between the two is potentially widest due to (often) numerous alternative relations between the same concepts.

In the experiments described in sections 5.4 and 5.5, we only applied fragments of the above scenario, mainly due to small size and specific nature of data.

5.4 Experiment in Tourism Domain: *Lonely Planet* Collection

5.4.1 Problem Setting

For the first experiment we adopted the Lonely Planet text collection[8]: 1800 short documents in English, about 5 MB overall[9]. These are free-text descriptions of various tourist destinations encompassing geography, history and available leisure activities. Our goal was to verify to what extent such a text collection can be used as support for discovering and *labelling* non-taxonomic relations for an ontology of the domain. Such an ontology could be used for diverse purposes, from ad-hoc question answering about world geography to tour recommendation applications.

Non-taxonomic relation extraction is a task typically superimposed over several other tasks, which can be carried out via manual modelling or inductively from text: lexical item extraction, mapping of lexical items to concepts, and taxonomy building:

- In *Text-to-Onto*, *lexical item extraction* has previously been used for discovery of potential *concept* labels, based on the well-known TFIDF (term frequency - inverse document frequency) measure. In contrast, our goal was *relation* labelling, which is also a form of lexical item extraction but requires a more focused approach. Since our hypothesis

[8] `http://www.lonelyplanet.com/destinations/`

[9] The same dataset was later used in other experiments with the *Text-to-Onto* tool [11]

was that 'relational' information is most often conveyed by verbs, we involved a *part-of-speech* (POS) tagger into the process of frequent transaction discovery[10]. About 75000 verb occurrences were identified in the collection.

- Although mapping *lexical items to concepts* can hardly be accomplished automatically. We thus adopted portions of the *TAP knowledge base*[11] recently developed at Stanford. TAP is a large repository of lexical items, such as proper names of places, companies, people, but also names of sports, art styles and other less traditional 'named entities'. It has previously been used for automated annotation of web pages [12] but its use as lexicon for ontology learning is novel.

- TAP includes a simple *taxonomy*, which is however not compatible with standard upper-level ontologies and contains ontologically unsound constructs. We therefore (manually) combined the TAP taxonomy with a small hand-made tourism ontology, and slightly extended via the *Text-to-Onto* term extraction facility. Although *Text-to-Onto* also contains an automatic taxonomy-building tool, we did not use it to prevent error chaining from one ontology learning task to another.

5.4.2 Analysis and Results

The whole analysis consisted of several phases, in which we used different components of *Text-to-Onto*. The output of earlier phases was stored and subsequently used for multiple (incl. debugging) runs of the last phase.

1. First, occurrences of ontology concepts (i.e. lexicon items) were found in text and stored in an index. For all 157 concepts, there were about 9300 such items with about 70000 occurrences.

2. Next, we used the POS tagger to identify the occurrences of verb forms in the text. About 75000 verb occurrences were identified; they were stored in another index.

3. We post-processed the POS tags to couple verbs such as 'to be' or 'to have' with their presumed syntactical objects, to obtain more usable verb constructs (these were subsequently handled in the same way as generic verbs).

[10]The same POS tagger, QTag `http://www.english.bham.ac.uk/staff/omason/software/qtag.html`, was previously used in *Text-to-Onto* for term extraction but not in the context of relation discovery.

[11]`http://tap.stanford.edu`

4. Finally, we compared the indices from step 1 and 2, recorded the VCC(n)-transactions, and aggregated them by triples.

Table 5.4.2 lists the 24 concept-concept-verb triples with $AE(c_1 \wedge c_2/v)$ higher than 100% (ordered by this value); triples with occurrence lower than 3, for which the relative frequencies do not make much sense, have been eliminated. The symbol $C(v, c_1, c_2)$ stands for $|\{t_i|v, c_1, c_2 \in t_i\}|$, i.e. how many times the verb occurred close enough to both concepts.

c_1	c_2	v	$C(v, c_1, c_2)$	$P(c_1 \wedge c_2/v)$	$AE(c_1 \wedge c_2/v)$
island	wg_region	*locate*	3	0.95%	750.00%
country	wg_region	*locate*	10	3.17%	744.68%
continent	country	*is_country*	26	10.12%	431.10%
us_city	wg_region	*locate*	4	1.27%	350.00%
country	island	*made*	5	1.68%	270.42%
country	island	*locate*	5	1.59%	239.36%
country	island	*consist*	10	7.41%	234.78%
museum	us_city	is_home	3	1.74%	234.55%
country	island	*comprise*	6	5.56%	200.62%
country	tourist	enter	6	2.79%	176.95%
country	island	*divide*	5	3.88%	172.46%
island	us_city	*locate*	3	0.95%	168.75%
city	stadium	known	9	1.25%	165.69%
city	country	allow	24	13.71%	152.89%
city	tourist	is_city	9	1.74%	151.61%
country	us_city	*locate*	9	2.86%	150.80%
city	country	*is_settlement*	6	16.22%	148.00%
island	us_city	connect	3	2.86%	140.00%
country	island	populate	5	6.02%	139.73%
city	island	*locate*	8	2.54%	131.39%
city	country	reflect	5	8.06%	117.42%
city	country	grant	4	12.90%	105.98%
city	park	is_city	11	2.13%	104.23%
city	country	stand	8	5.06%	104.03%

Table 5.1: Final results of label extraction

Note that the table suggests which pairs of concepts should certain verbs be assigned to, as lexical items for non-taxonomic relations.

5.4.3 Evaluation

We can see that triples with high $AE(c_1 \wedge c_2/v)$ (even those with low absolute frequencies, 4 or 5) correspond to meaningful semantic relations, mostly topo-mereological ones: an island or a country is located in a world-geographical region (wg_region), a country 'is a country' of a particular continent and may be located on an island or consist of several islands. [12], a city may be home of a famous museum etc. Hence, $P(c_1 \wedge c_2/v)$ is not very important by itself either, as soon as it reaches some minimal (quite small) value. However, with $AE(c_1 \wedge c_2/v)$ dropping to about 150 %, the verbs cease to pertain to a relation. This leads us to the heuristics that triples below this value should probably not be presented to the ontology designer.

On this small result set, we can simulate the evaluation strategy outlined in section 5.3. For simplicity (and to minimise subjective bias), we only chose as *reference ontology* the set of obvious topo-mereological relations among geographical concepts. For the most frequent six concepts of this kind (City, US City, Country, Island, Continent, World Geographic Region), we identified 17 concept-concept-relation triples that are likely to frequently occur in reality: 14 topological ones (i.e. an object is located within another object, under transitive closure) and 3 mereological ones (i.e. an object consists of other objects). The reference ontology is at Fig. 5.3; line arrows stand for 'located in', full arrow for 'is-a' and diamond arrows for 'consists of'. We could then compute (prior) *precision*, *recall*, and, finally, *F-measure* (harmonic mean of precision and recall) with respect to the reference ontology. For simplicity, we did not take concept taxonomy (in this case, a single is-a link) nor verb hypero/hyponymy into account; only verbs that directly reflect the given relation (italicised in Table 5.4.2) were considered. Furthermore, there are relations that are not included in the reference ontology but still make sense, for example the 'entering' relation between the concepts of Tourist and Country. If we choose the relaxed variant of *augmentation*, we keep such cases as correct hits rather than as misses. We can then compute the *posterior precision*. Fig. 5.4 shows the recall and both types of precisions in a single graph, while Fig. 5.5 shows the F-measure (the X-axis always corresponds to increasing number of triples in the descending order of AE measure). The F-measure value sharply increases as long as the values of AE measure are of order of multiple hundreds, then less sharply for values around 130-230%, and finally monotonically decreases when approaching to 100% (i.e. 'equal-to-expectation' value). The sample size was however so small that no general conclusions could be drawn from these figures.

Set aside the solid recall on topo-mereological relation labels, the total

[12] Example of *multiple relations* between the same concepts, cf. end of section 1.

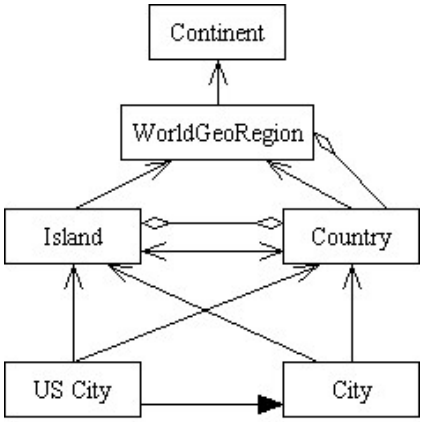

Figure 5.3: Reference ontology for Lonely Planet experiment

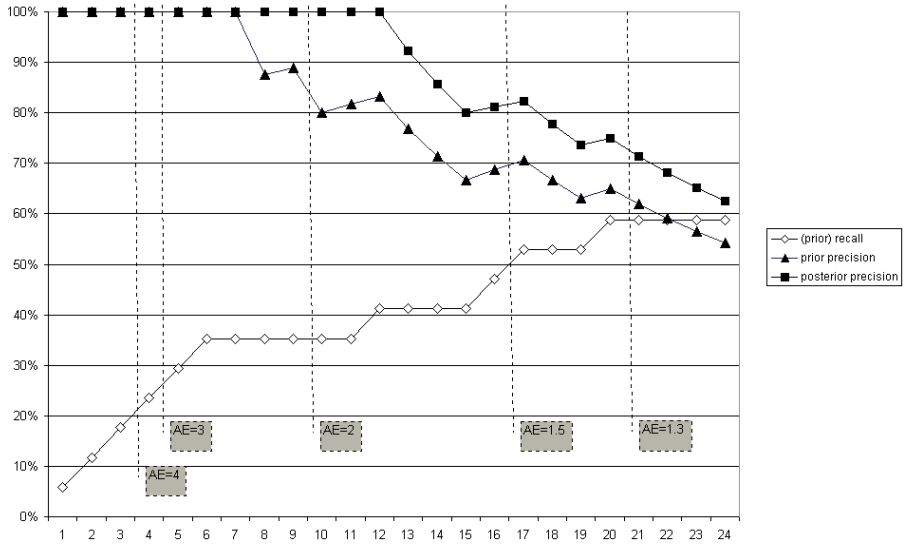

Figure 5.4: Recall and (prior and posterior) precision in Lonely Planet experiment

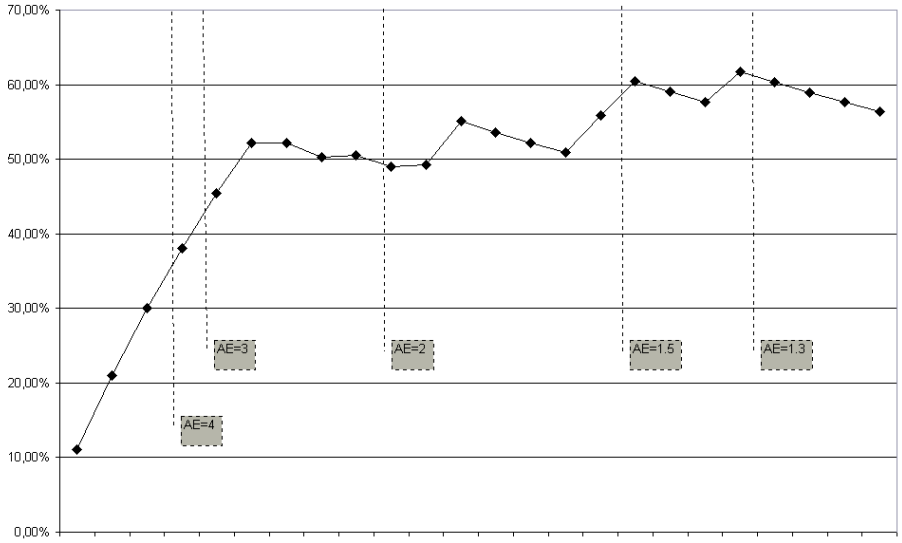

Figure 5.5: (Prior) F-measure in Lonely Planet experiment

number of labels extracted from the 5MB corpus was definitely not impressive. This can be partially attributed to the following:

- *Sparseness of concept taxonomy.* The TAP-based taxonomy was not a true ontology of the domain, and was rather sparse. Construction of a good taxonomy is a demanding task; by complex study in [24], however, it is not as big a challenge as the invention of plausible non-taxonomic relations.

- *Sparseness of lexicon.* The lexicon only covered a part of the relevant lexical space. It listed many names of places (often only appearing in a single document) but few names of activities for tourists or art objects (reusable across many documents). Better coverage would require either comprehensive lexicons (some can also be found on the web) or heavy-weighted linguistic techniques such as anaphora resolution, since the geographical entities initially introduced in the text are often referred to by pronouns.

- *Semantic ambiguity of terms.* Ambiguous words were assigned all possible meanings, which of course added *noise* to the data.

- *Style of underlying text.* The Lonely Planet documents are written in a free style: the same relation is often expressed by different verbs, which decreases the chance of detecting the most characteristic one.

- *Performance of POS tagger.* Sometimes, the tagger does not properly categorise a lexical item. For example, a verb associated with concept *Country* was 'cross'; some of its alleged occurrences however seemed to be adverbs (e.g. 'a tourist going cross the country').

- *Performance of concept extractor.* Since relation extraction was superimposed over (automated) concept extraction, results of the former were negatively influenced by the flaws of the latter.

5.5 Experiments with Semantically Tagged Corpus

5.5.1 Problem Setting

In order to overcome some difficulties arisen in the previous experiment, we adopted *SemCor*[13]: a part of Brown corpus[14], semantically tagged with WordNet[15] senses. All open word classes (nouns, verbs, adjectives and adverbs) are mapped to their WordNet senses. Advantages over an ad hoc document collection such as Lonely Planet immediately follow from reduced ambiguity:

1. We can use the WordNet hierarchy to lift the tagged terms to *concepts* at an arbitrary level of abstraction. There is thus no need for automatic (and error-prone) frequency-based concept extraction.

2. Similarly, we can aggregate the *verbs* along the hierarchy and thus overcome their sparseness of data.

3. We can evaluate our approach without impact of POS tagger, which also exhibited significant error rate in the previous experiment.

Since *SemCor* is a small corpus with very broad scope, we confined ourselves to three very general concepts to avoid data sparseness: *Person*, *Group* and *Location*[16]. We identified each of them with the WordNet synset containing the word sense person#1 (or group#1 or location#1, respectively)

[13] `http://www.cs.unt.edu/~rada/downloads.html`

[14] `http://helmer.aksis.uib.no/icame/brown/bcm.html`

[15] `http://www.cogsci.princeton.edu/~wn`

[16] Admittedly, the combination of a generic corpus and a three-class target 'ontology' does not approximate real-world (say, business) ontology learning settings very well. It was only meant for 'in vitro' evaluation of the method.

[17] and all its hyponyms. Any word tagged with WordNet sense that could be generalised to the synset containing person#1 was thus considered as occurrence of Person (and the like). This way we found 14613 occurrences for Person, 6727 for Group and 4889 for Location. The corpus contains 47701 sense-tagged verb occurrences. In all three experiments below, we set the minimal absolute frequency of triples to 5, to filter out the cases where the relative frequencies were skewed because of sparse data.

5.5.2 Analysis and Results

In the first experiment with *SemCor* we grouped the verbs directly by the *synset* they belong to (i.e. all occurrences of verbs from one synset counted together); this yielded 4894 synsets. Table 5.2 shows the top synsets according to the AE score (only considering those with $AE \geq 2.5$), for the Person-Group pair. In the second experiment we generalised each verb by taking its (first-level) *hypernym synset*; we obtained 1767 synsets. Top ones (again considering those with $AE \geq 2.5$) for the Person-Group pair are in Table 5.3. In the third experiment we attempted to introduce some 'domain bias' through separately processing two *sub-collections* of *SemCor*, news articles and scientific texts, each representing about 15% of the original corpus. We generally observed dissimilar distributions of verb synsets (e.g. news articles concerned 'moving', 'communicating', 'leading', while scientific texts rather dealt with 'observing', 'proposing' or 'transforming') however, only a fraction of verbs suggested as labels for a particular relation was relevant. This was obviously due to data sparseness, even in the hypernym synset setting. In both sub-collections, the relation Person–Group gained highest confidence, the hypernym synsets selected for news articles are presented in table 5.4, hypernym synsets for scientific journals in 5.5

5.5.3 Evaluation

Since building a 'reference ontology' corresponding to the coverage of a generic corpus is inconceivable, we cannot evaluate the labels by means of prior precision and recall. The only remaining measure is then *posterior precision* based on subjective evaluation (i.e. 'relaxed augmentation of empty reference ontology'). We considered as positive hits all cases where *at least one member* of the verb synset corresponded to a meaningful relation among the concepts that would be worth modelling in some domain ontology. To assess the impact of verb abstraction, we separately measured the precision

[17] The sense numbers corresponded to WordNet version 2.0.

Table 5.2: Suggested relations between Person and Group – verb synset version

Verb synset	$C(v, c_1, c_2)$	$AE(c_1 \wedge c_2/v)$
head, lead	10	4.43
act as	13	4.36
leave, depart, pull up stakes	7	4.08
decrease, diminish, lessen, fall	6	3.54
submit, state, put forward, posit	9	3.44
serve	11	3.44
form, organize, organise	10	3.41
stage, present, represent	6	3.22
collaborate, join forces, cooperate, get together	8	2.95
include	25	2.68
meet, ran into, encounter, run across, come across, see	10	2.68
meet, gather, assemble, forgather, foregather	5	2.59

Table 5.3: Suggested relations between Person and Group – verb hypernym version

Verb synset	$C(v, c_1, c_2)$	$AE(c_1 \wedge c_2/v)$
serve, function	13	4.36
attack, assail	6	3.53
meet, ran into, encounter, run across, come across, see	10	2.74
be, follow	11	2.58

Table 5.4: Suggested relations between Person and Group – verb hypernym version, news articles

Verb synset	$C(v, c_1, c_2)$	$AE(c_1 \wedge c_2/v)$
arrive, get, come	7	4.20
direct	10	3.79
order, tell, enjoin, say	5	3.75
include	7	3.57
constitute, represent, make up, comprise, be	4	3.50
note, observe, mention, remark	4	3.33
use, utilize, utilise, apply, employ	4	2.90
be	19	2.40
re-create	8	2.31
travel, go, move, locomote	6	2.24
inform	15	2.21
get the better of, overcome, defeat	7	2.00

Table 5.5: Suggested relations between Person and Group – verb hypernym version, scientific articles

Verb synset	$C(v, c_1, c_2)$	$AE(c_1 \wedge c_2/v)$
serve, function	6	6.38
denounce	5	3.75
propose, suggest, advise	9	3.19
note, observe, mention, remark	4	2.29
find, happen, chance, bump, encounter	5	2.08

for original and abstracted synsets. We only list the graphs for Person-Group pair, in Fig. 5.6, for labels ordered in the decreasing order of AE measure. It seems that the precision is again decreasing more steeply for triples with AE measure under approx. 130%, although some improper labels cause an abrupt decrease near the beginning. Interestingly, most of such highly-scored false hits are related to *communication* (such as 'state', 'write', 'publish', 'announce', 'remark'). We can hypothesise that especially in news articles, (forming a significant part of SemCor), such verbs typically occur near statements involving both persons and groups, yet have nothing to do with the relationship *among* persons and groups. The *hypernym version* had better precision. The most likely reason might be that most 'communication' verbs mentioned above have broad hypernyms such as 'create', which can be considered as proper labels for the Person-Group pair. The labels for other two pairs (Person-Location, Group-Location) had lower precision. While some triples were relevant (such as "Person - born - Location" or "Group - reach - Location"), many other only seemed to reflect the fact that *events* involving persons and/or groups are often said to happen in a particular location. The number of 'correct hits' was hence too low to evaluate the trend of precision curve.

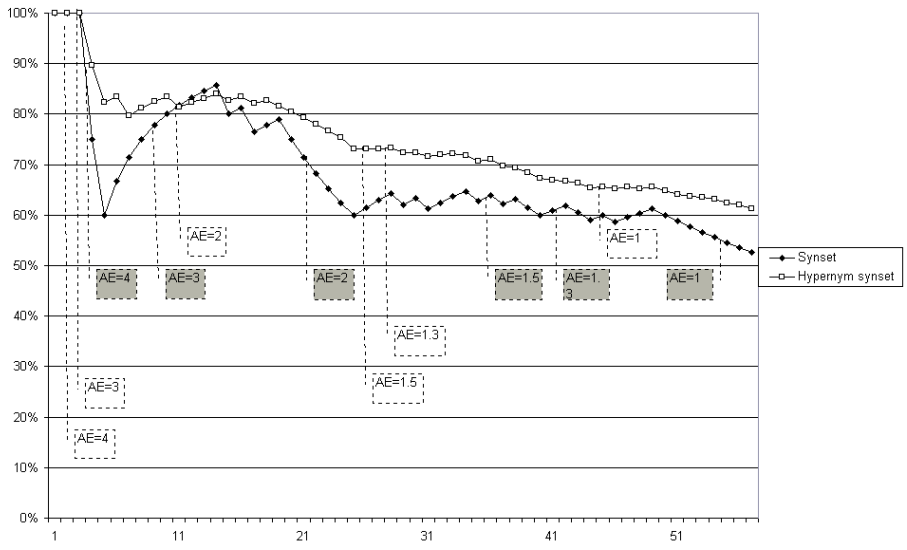

Figure 5.6: Posterior precision in SemCor experiment

Most verbs with high AE measure seem to be potential labels for relations between Person and Group (and similarly for the other two concept pairs not shown here). This supports the hypothesis that our method could provide

useful hints for an ontology designer. Human effort is of course still needed to filter out incidental results or e.g. to handle semantically incomplete expressions such as 'act as'.

In some cases, the impact of *verb generalisation* seems positive. For example, for the Person-to-Location relation, 'leave' (as definitely an important label) only had $AE = 1.67$ in the verb synset version, while it floated up in the verb hypernym version. On the other hand, generalising may sometimes obscure the original meaning, e.g. the 'serve, function' synset (for Person-to-Group relationship) is result of generalisation of 'act as' (the latter being probably more characteristic for this relationship. Sometimes even the one level of WordNet hypernymy may lead to overly general meaning, e.g. 'form, organize' is generalised to 'make, create', which scores much lower and thus does not appear among the top candidates. It seems that a combination of verbs directly found in text and of their careful generalisations might be the best blend to be presented to ontology designer.

5.6 Related Work

Many approaches to relation learning from text make relatively little distinction between *relations* and *relation instances*, in the set-theoretic sense. Lexical labels are often directly assigned to statements about concrete pairs of entities, i.e. relation instances. Instances are however usually not expected to be part of an ontology. This research should be viewed as ontology *population*, rather than learning. It is quite desirable to fully automate such extraction (within a predefined problem setting), and the performance of extractors can often be precisely measured. In contrast, we focus on relations that *possibly* hold among (various instances of) certain ontology concepts. The design of relations is a creative task: it *can* and *should* be accomplished by a human, for whom we only want to offer partial support.

Yet, many partial techniques are similar. Finkelstein&Morin [14] combine 'supervised' and 'unsupervised' extraction of relationships between terms; the latter (with unspecified underlying relations) relies on 'default' labels, under assumption that e.g. the relation between a Company and a Product is always 'produce'. Byrd&Ravin [9] assign the label to a relation (instance) via specially-built finite state automata operating over sentence patterns. Some automata yield a pre-defined relation (e.g. *location* relation for the '-based' construction) while other pick up a promising word directly from the analysed sentence. Labelling of proper relations is however not addressed, and even the 'concepts' are a mixture of proper concepts and instances. The *Adaptiva* system [6] asks the user to choose a relation from the ontology and

then interactively learns its recognition patterns. Although the goal is to *recognise* relation instances in text, the interaction with the user may also give rise to new proper relations. Such intensive interaction however does not pay off if the goal is merely to *find labels for* important domain-specific relations to which the texts refer, as in our case. The *Asium* system [13] synergistically builds two hierarchies: that of concepts and that of verb sub-categorisation frames (an implicit 'relation taxonomy'), based on co-occurrence in text . Verbs co-occurring with concepts in text are used to cluster the concepts, and vice versa. There is however no direct support for conceptual 'leap' from a 'bag of verbs' to a named relation, which we have thanks to integration of our technique into the whole *Text-to-Onto* environment.

Another line of work, more firmly grounded in ontology engineering, systematically seeks new *unnamed* relations in text. Co-occurrence analysis with limited attention to sentence structure is used, and the results filtered via frequency measures as in our approach. As mentioned before, in prior work on *Text-to-Onto* [26], the labelling problem was left upon the ontology designer. The same holds about the non-taxonomic relation component of *DODDLE* [36], which only differs by a more sophisticated way of transaction construction. In the *OntoLearn* project [40], WordNet and FrameNet mapping was used to automatically assign relations from a small predefined set (such as 'similar' or 'instrument').

Interesting is the *OntoLT* plug-in to Protégé [8], which does not distinguish ontology learning tasks such as creation of classes, slots or instances at the architectural level but rather as action parts of user-definable rules. Its input is a corpus that is linguistically annotated by means of another automatic tool (parser): it thus does not rely on surface patterns. The words are filtered for domain specificity (using the χ^2 measure) in the pre-processing phase. Ontology learning corresponds to slot creation; the lexical label for a new slot is directly transferred from (even a single occurrence of) the linguistic predicate for the phrase on which a slot-creation rule is applied.

Chapter 6

Conclusions

In the experiment on discovery of important information indicators, we tested our method on web pages of small companies, offering their products and services. The method selected several indicator terms, which were themselves able to extract sentences which contained some relevant information about the company and its product.

Such result may be used in a web page summarizing tool and this is what we did with the indicator terms within the project Rainbow [38].

To turn the obtained indicators into extraction patterns suitable for example for an ontology population task further work is necessary, because the information indicators themselves do not specify to which concepts of ontology its subject and object belong.

Our experiments on the field of ontology learning suggest that relation extraction from text may be used not only for discovering 'anonymous' relations between pairs of concepts, but also for providing potential lexical *labels* for these relations.

Serious problems however are the sparseness of data (due to multiple reasons) and domain-dependency of the labels. The experiment with semantically-tagged corpus suggested that referring to the *right sense of words* improves the quality of relation labelling, and so might do the increase of the *degree of abstraction of verbs* by their meaning. The quality of results on the SemCor corpus was comparable to the Lonely Planet experiment despite the smaller and broader corpus; we assume that the presence of reliable semantic information (word senses) made up for the smaller size of corpus. Although we usually lack word sense information in real-world settings, it is often possible to restrict the senses of words with respect to a narrow domain, for which we build the ontology. In particular, polysemous verbs typically become monosemous in the context of domain-specific

applications. In addition, there are techniques developed to cope with ambiguity e.g. as proposed by Mihalcea & Moldovan [29]. Furthermore, existing methods for disambiguation of named entities could be applied in some cases.

Simple, user oriented interactive tools may also be helpful in this task. They may be considered complementary to the statistical measures (which are not always reliable, due to data sparseness). For example, the ontology designer might wonder (e.g. assuming a 'borderline' AE measure) whether a verb really pertains to the relation in text, e.g. is really typical (and thus worth modelling) for cities *to be known* for their museums. Display of the underlying text fragments (which are not overwhelmingly numerous in our case) where the VCC transactions in question would be highlighted could easily and quickly help in this decision.

A problematic point of the method is the *direct mapping* from co-occurrences of terms onto 'deep' ontological relations. In particular the SemCor experiment indicated that the method improperly suggests verbs that typically occur in some larger semantic context involving (among other) the two concepts in question but do not correspond to immediate relation between them. Making the method more *linguistic-aware*, i.e., to employ a chunker or parser to determine the (syntactically) most appropriate verb within the transaction would reduce noise and improve quality of the results. Further research could determine whether the overhead of shallow parsing will be outweighed by better precision. From the point of view of the association rules mining (including our extension), linguistic analysis is a preprocessing step. Application of a chunker or another linguistic processing does not require change of the association rules mining, it simply prepares data of (presumably) higher quality.

Verbs, merely identified by POS tagging (i.e. without structural analysis of the sentence) and related to the ontology concepts by their close co-occurrences can be viewed as a first approximation of relation labels, which can provide useful hints to an ontology engineer in the process of creation of an ontology. Our method of relation labeling and approach to its consecutive performance evaluation was published in [19] and this supports our hope, that this work contributes to current research in this area.

Bibliography

[1] Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: Proc. ACM SIGMOD Conference on Management of Data, 207–216, 1993.

[2] Becket, D., McBride, B.: RDF/XML Syntax Specification (Revised), `http://www.w3.org/TR/rdf-syntax-grammar/` [may, 30th 2005]

[3] Box, D., et al.: Simple Object Access Protocol (SOAP) 1.1 – W3C Note, 2000. `http://www.w3.org/TR/SOAP/`

[4] Brickley, D., Guha, R.V.: RDF Vocabulary Description Language 1.0: RDF Schema, W3C, February 2004, `http://www.w3.org/TR/rdf-schema/` [May, 30th. 2006]

[5] Brin, S.,: Extracting Patterns and Relations from the World Wide Web. In: WebDB Workshop at EDBT'98.

[6] Brewster, C., Ciravegna, F., Wilks, Y.: User-Centred Ontology Learning for Knowledge Management In: 7th Int'l Conf. Applications of Natural Language to Information Systems, Stockholm, LNAI, Springer 2002.

[7] Buitelaar, P., Cimiano, P., Magnini, P.: Ontology Learning from Text: An Overview, Ontology Learning from Text: Methods, Evaluation and Applications, IOS Press, 2005

[8] Buitelaar, P., Olejnik, D., Sintek, M.: A Protégé Plug-In for Ontology Extraction from Text Based on Linguistic Analysis. In: Proc. 1st European Semantic Web Symposium (ESWS-04), Heraklion, Greece, 2004.

[9] Byrd, R., Ravin, Y.: Identifying and Extracting Relations in Text. In: Proceedings of NLDB 99, Klagenfurt, Austria, 1999.

[10] Christensen, E., et al: Web Services Description Language (WSDL) 1.1, W3C Note, 2001. `http://www.w3.org/TR/2001/NOTE-wsdl-20010315`

[11] Cimiano, P., Automatic acquisition of taxonomies from text: FCA meets NLP. In: Proceedings of the International Workshop on Adaptive Text Extraction and Mining (ATEM03), Cavtat 2003.

[12] Dill, S. et al.: SemTag and Seeker: Bootstrapping the semantic web via automated semantic annotation. In: Proc. WWW2003, Budapest 2003.

[13] Faure, D., Nédellec, C.: ASIUM: Learning subcategorization frames and restrictions of selection. In: ECML'98, Workshop on Text Mining, 1998.

[14] Finkelstein-Landau, M., Morin, E.: Extracting Semantic Relationships between Terms: Supervised vs. Unsupervised Methods. In: Int'l Workshop on Ontological Engineering on the Global Information Infrastructure, Dagstuhl 1999.

[15] Gruber, T.: Towards principles for design of ontologies used for knowledge sharing. Int. J. of Human and Computer Studies, 43:907-928, 1994.

[16] Guarino, N.: Formal ontology and information systems. In Proceedings of FOIS'98, Formal Ontology in Information Systems, Trento, IOS Press, 1998.

[17] Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora, COLING'92, pp 539-545, 1992.

[18] Kavalec, M., Svátek, V.: Information Extraction and Ontology Learning Guided by Web Directory. In: ECAI Workshop on NLP and ML for ontology engineering. Lyon 2002.

[19] Kavalec, M., Svátek, V.: A Study on Automated Relation Labelling in Ontology Learning. In: Ontology Learning from Text: Methods, Evaluation and Applications, Frontiers in Artificial Intelligence vol. 123, IOS Press, 2005.

[20] Klyne, G., Caroll,J.J.: Resource Description Framework (RDF): Concepts and Abstract Syntax, W3C, February 2004, `http://www.w3.org/TR/rdf-concepts/` [may, 30th 2005]

[21] Kodratoff, Y.: Comparing Machine Learning and Knowledge Discovery in DataBases: An Application to Knowledge Discovery in Texts. In: ECCAI summer course, Crete July 1999, LNAI, Springer 2000.

[22] Lafferty, J., McCallum, A., Pereira, F.: Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data,

[23] Li Y., Zhang, L., Yu, Y.: Learning to Generate Semantic Annotation for Domain Specific Sentences. In: K-CAP 2001 Workshop on Knowledge Markup & Semantic Annotation, October 21, 2001, Victoria B.C., Canada.

[24] Maedche, A.: Ontology Learning for the Semantic Web. Kluwer, 2002.

[25] Maedche, A., Neumann G., Staab, S.: Bootstrapping an Ontology-Based Information Extraction System. Studies in Fuzziness and Soft Computing, editor J. Kacprzyk. INTELLIGENT EXPLORATION OF THE WEB, P.S. Szczepaniak, J. Segovia, J. Kacprzyk, L.A. Zadeh, Springer 2002

[26] Maedche, A., Staab, S.: Mining Ontologies from Text. In: EKAW'2000, Juan-les-Pins, Springer, 2000.

[27] Maedche, A., Volz, R.: The Text-To-Onto Ontology Extraction and Maintenance System. In: ICDM-Workshop on Integrating Data Mining and Knowledge Management, San Jose, California, USA, 2001.

[28] McCallum A., Nigam, K.: Text Classification by Bootstrapping with Keywords, EM and Shrinkage. In ACL'99 Workshop for Unsupervised Learning in NLP, 1999.

[29] Mihalcea, M., Moldovan, D.I.: A Highly Accurate Bootstrapping Algorithm for Word Sense Disambiguation, International Journal on Artificial Intelligence Tools, vol. 10, 1-2, pp. 5-21, 2001

[30] Missikoff, M., Navigli, R., Velardi, P.: Integrated approach for Web ontology learning and engineering. IEEE Computer, November 2002.

[31] Mladenic, D.: Turning Yahoo into an Automatic Web-Page Classifier. In: Proc. 13th European Conference on Aritficial Intelligence, ECAI'98, 473-474.

[32] Gómez-Pérez, A., Manzano-Macho, D.: A survey of ontology learning methods and techniques. OntoWeb deliverable 1.5, Universidad Politécnica de Madrid

[33] Riloff E., Jones R.: Learning Dictionaries of Information Extraction by Multi-Level Bootstrapping. In Proc. 16th Nat. Conf. Artificial Intelligence (AAAI-99).

[34] Sleator, D., Temperley, D.: Parsing English with a Link Grammar. In Third International Workshop on Parsing Technologies, August 1993.

[35] Stevenson, M., Ciravegna, F.: Information Extraction as a Semantic Web Technology: Requirements and Promises Adaptive Text Extraction and Mining Workshop at the ECML03, Cavtat-Dubrovnik, 2003

[36] Sugiura, N., Shigeta, Y., Fukuta, N., Izumi, N., Yamaguchi, T.: Towards On-the-Fly Ontology Construction Focusing on Ontology Quality Improvement. 1st European Semantic Web Symposium (ESWS-04), Heraklion, Greece, 2004.

[37] Svátek, V., Berka, P.: URL as starting point for WWW document categorisation. In: RIAO'2000 – Content-Based Multimedia Information Access, Paris, 2000.

[38] Svátek V., Kosek J., Labský M., Bráza J., Kavalec M., Vacura M., Vávra V., Snášel V.: Rainbow - Multiway Semantic Semantic Analysis of Websites. In: 2^{nd} DEXA Int'l Workshop on Web Semantics, Prague 2003, IEEE Computer Society Press 2003.

[39] Uschold, M., Jasper, R.: A Framework for Understanding and Classifying Ontology Applications. In: Proceedings of the IJCAI-99 Workshop on Ontologies and Problem-Solving Methods: Lessons Learned and Future Trends.

[40] Velardi,P., Navigli,R., Cucchiarelli, A., Neri, F.: Evaluation of OntoLearn, a Methodology for Automatic Learning of Domain Ontologies, In: Ontology Learning from Text: Methods, Evaluation and Applications, Frontiers in Artificial Intelligence vol. 123, IOS Press, 2005.

Dictionary of terms

Concept Identification of a type of entity (or a class of entities of the same type) in a particular domain.

Information extraction, IE Information extraction is in general a automatic process which aims to identify information with a particular meaning in a set of free- or semi-structured texts. For example, from a set of seminar announcement texts we look for the topic (or title) of seminar, name of the speaker, place, date and time of the talk.

Instance (or an individual) a single entity of a domain, for example, in the domain of art, 'Leonardo da Vinci' is a instance of the concept 'Artist'

knowledge base in a broader sense, any explicit representation of a knowledge, relevant for a domain or a particular task. In scope of this work, it is set of instances of concepts and relations of an ontology and the corresponding lexicon

Knowledge base lexicon set of terms (lexical items) corresponding to set of instances of a knowledge base

Lexical item Word or sequence of words, which specifies how an concept (or its instance) is used in a natural language. Concept or its instance may have multiple lexical items (e.g. in case of synonymy or a word stem may be used in addition). Part of an ontology lexicon or a knowledge base lexicon.

Ontology Shared and formalized conceptualization of a domain of human activity. Describes important concepts of the domain, their hierarchy and other relations in formal way, which enables reasoning and logical inferences.

Ontology learning scientific field studying methods of automatic support of an ontology engineer in the process of creating and maintenance of an ontology

Ontology lexicon set of terms (lexical items) relevant for a particular domain and their mapping to the concepts of the ontology.

POS tagger, part-of-speech tagger linguistic tool which determines part of speech for each word in text. It also specifies other morphological categories such as gender, case, singular/plural and other categories for flexive languages.

Relation extraction a step in the process of creating an ontology. Its goal is identification of relations, relevant for the modelled domain, based on available information resources, usually text documents describing the modelled domain. It aims to find possible relations between concepts, not a concrete instances of the relation.

Relation instance (instance of a relation) a pair of instances, related by an ontology relation. The relation instance represents a single fact in the knowledge base of an ontology (e.g. 'Leonardo da Vinci' painted 'Mona Lisa')

Semantic web "The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation." (Tim Berners-Lee, James Hendler, Ora Lassila, The Semantic Web, Scientific American, May 2001) Semantic web comprises of information published on the web in machine readable way, conforming to standards proposed by W3C consortium. Ontologies are the part of semantic web, which provides the well-defined meaning, mentioned in the quotation.

Shallow parser, chunker A linguistic tool, which decomposes sentences on chunks, which are noun, verb or adverbial phrases.

Appendix A

Results of experiment with SemCor

verb set	$C(v, c_1, c_2)$	$AE(c_1 \wedge c_2/v)$
head, lead	10	4.43
act as	13	4.36
leave, depart, pull up stakes	7	4.08
decrease, diminish, lessen, fall	6	3.54
submit, state, put forward, posit	9	3.44
serve	11	3.44
form, organize, organise	10	3.41
stage, present, represent	6	3.22
collaborate, join forces, cooperate, get together	8	2.95
include	25	2.68
meet, ran into, encounter, run across, come across, see	10	2.68
meet, gather, assemble, forgather, foregather	5	2.59
be, follow	11	2.58
command, require, compel	6	2.40
print, publish	5	2.28
conduct, lead, direct	6	2.14

Table A.1: Suggested relations between Person and Group – verb synset version, part 1

verb set	$C(v, c_1, c_2)$	$AE(c_1 \wedge c_2/v)$
write	11	2.10
join, fall in, get together	14	2.08
hire, engage, employ	8	2.07
name, call	14	2.05
marry, get married, wed, conjoin, hook up with, get hitched with, espouse	5	2.00
attend, go to	18	2.00
comment, notice, remark, point out	9	1.94
condemn, reprobate, decry, objurgate, excoriate	6	1.93
hold, throw, have, make, give	9	1.89
typify, symbolize, symbolise, stand for, represent	5	1.87
produce, bring forth	5	1.81
believe, trust	5	1.81
arrive, get, come	18	1.81
lead, take, direct, conduct, guide	13	1.77
announce, denote	5	1.65
enroll, inscribe, enter, enrol, recruit	7	1.60

Table A.2: Suggested relations between Person and Group – verb synset version, part 2

verb set	$C(v, c_1, c_2)$	$AE(c_1 \wedge c_2/v)$
meet	13	7.53
be, follow	14	5.47
vote	6	4.13
write	24	3.97
be born	8	3.78
station, post, base, send, place	6	3.77
rub	6	3.26
elect	5	3.00
shoot, pip	5	2.25
ask	7	2.10
run	7	2.02
meet, ran into, encounter, run across, come across, see	6	2.01
hold, throw, have, make, give	8	2.00
arrive, get, come	23	1.90
head	5	1.88
ask, inquire, enquire	11	1.84
help, assist, aid	5	1.78
become, turn	17	1.78
propose, suggest, advise	5	1.75
reach, make, attain, hit, arrive at, gain	9	1.67
leave, go forth, go away	16	1.67
remove, take, take away, withdraw	5	1.65
watch, observe, follow, watch over, keep an eye on	5	1.64
dwell, shack, reside, live, inhabit, people, populate, domicile, domiciliate	22	1.61
return, go back, get back, come back	13	1.52
make, create	9	1.51
ride, sit	6	1.43
lead, take, direct, conduct, guide	6	1.34
cause, do, make	6	1.31
talk, speak	5	1.26
shout, shout out, cry, call, yell, scream, holler, hollo, squall	6	1.23
see, consider, reckon, view, regard	5	1.19

Table A.3: Suggested relations between Person and Location – verb synset version

verb set	$C(v, c_1, c_2)$	$AE(c_1 \wedge c_2/v)$
reach, get through, get hold of, contact	8	5.90
service, serve	7	3.15
establish, found, plant, constitute, institute	5	2.94
come, come up	9	2.80
desire, want	5	2.55
play	5	2.50
tend, be given, lean, incline, run	5	2.12
hold, throw, have, make, give	6	1.82
become, turn	6	1.65
be	23	1.42
reach, make, attain, hit, arrive at, gain	7	1.39
include	7	1.33
be	28	1.28
supply, provide, render, furnish	10	1.00
exist, be	13	1.00

Table A.4: Suggested relations between Location and Group – verb synset version

verb set	$C(v, c_1, c_2)$	$AE(c_1 \wedge c_2/v)$
serve, function	13	4.36
attack, assail	6	3.53
meet, ran into, encounter, run across, come across, see	10	2.74
be, follow	11	2.58
unite, unify	7	2.38
direct	21	2.14
announce, denote	9	2.06
appoint, charge	5	2.03
denounce	7	2.03
arrive, get, come	23	2.01
note, observe, mention, remark	9	1.94
hire, engage, employ	12	1.93
promote, upgrade, advance, kick upstairs, raise, elevate	5	1.93
re-create	11	1.81
join, fall in, get together	15	1.80
charge, accuse	5	1.79
interact	12	1.77
include	26	1.72
seize, prehend, clutch	7	1.72
work, do work	17	1.69
meet, encounter, play, take on	6	1.69
get the better of, overcome, defeat	10	1.64
lead, take, direct, conduct, guide	13	1.64
find, happen, chance, bump, encounter	9	1.59
label	15	1.56
change magnitude	6	1.47
propose, suggest, advise	9	1.32
depend on, devolve on, depend upon, ride, turn on, hinge on, hinge upon	5	1.29
transform, transmute, metamorphose	17	1.26
utter, emit, let out, let loose	6	1.25
change integrity	7	1.22
work	12	1.12

Table A.5: Suggested relations between Person and Location – verb hypersynset version

verb set	$C(v, c_1, c_2)$	$AE(c_1 \wedge c_2/v)$
meet	13	7.53
be, follow	14	5.47
hit	14	4.99
guide, run, draw, pass	6	3.43
appoint, charge	5	2.84
leave	6	2.77
announce, denote	6	2.75
grow	5	2.68
hire, engage, employ	6	2.35
request, bespeak, call for, quest	15	2.31
choose, take, select, pick out	14	2.23
complain, kick, plain, sound off, quetch, kvetch	5	2.10
arrive, get, come	27	1.99
initiate, pioneer	6	1.85
travel rapidly, speed, hurry, zip	7	1.83
transform, transmute, metamorphose	17	1.78
meet, get together	6	1.77
supply, provide, render, furnish	6	1.73
proceed, go forward, continue	5	1.73
kill	8	1.70
leave, go forth, go away	24	1.70
direct	9	1.69
meet, ran into, encounter, run across, come across, see	6	1.65
check, check up on, look into, check out, suss out, check over, go over, check into	5	1.64
reach, make, attain, hit, arrive at, gain	9	1.64
refer, pertain, relate, concern, come to, bear on, touch, touch on	5	1.55
communicate, intercommunicate	46	1.55

Table A.6: Suggested relations between Person and Location – verb hypersynset version

verb set	$C(v, c_1, c_2)$	$AE(c_1 \wedge c_2/v)$
talk, speak, utter, mouth, verbalize, verbalise	5	6.78
perform, execute, do	5	4.09
connect, link, tie, link up	5	3.67
exchange, change, interchange	6	3.60
attack, assail	6	2.94
communicate, intercommunicate	19	2.91
compete, vie, contend	7	2.52
initiate, pioneer	6	2.27
register	6	2.12
produce, make, create	7	1.83
keep, maintain, hold	5	1.65
transform, transmute, metamorphose	6	1.65
inform	9	1.54
function, work, operate, go, run	7	1.52
direct	7	1.49
reach, make, attain, hit, arrive at, gain	7	1.36
desire, want	5	1.34
be	28	1.28
move, displace	12	1.22
give	18	1.14
judge	8	1.14
travel, go, move, locomote	28	1.11
support, back up	6	1.10
be	35	1.05
get, acquire	13	0.92
change state, turn	6	0.92

Table A.7: Suggested relations between Group and Location – verb hypersynset version