UNIVERSITÄT OSNABRÜCK

Institute for Cognitive Science
University of Osnabrück

Bachelor's Thesis

Learning the semantics of Wikipedia hyperlinks

Daniel Bauer

dbauer@uni-osnabrueck.de

December 2007

First supervisor: Prof. Dr. Stefan Evert
Second supervisor: Prof. Dr. Kai-Uwe Kühnberger

Abstract

I claim that hyperlinks in Wikipedia entries often correspond to semantic relationships between concepts, described by the entries. This bachelor's thesis discusses supervised methods to automatically identify new links that correspond to a given relation (hyper-/or hyponymy). Training data is collected by mapping Wikipedia articles to WordNet synsets and then marking links where a relation between the synsets is recorded in WordNet. Also, as a source of data, the assembly of an XML annotated corpus from raw Wikipedia database dumps is described.

Contents

1 Introduction **1**

2 Background and basic concepts **4**
2.1 Ontologies and ontology learning 4
2.2 Types of semantic relations 5
2.3 WordNet 9
2.4 Wikipedia 10

3 Creating an annotated corpus from Wikipedia **13**
3.1 Other corpora 13
3.2 Wikipedia markup 14
3.3 Creating the corpus 15
3.4 Conclusion 18

4 Mapping Wikipedia articles to WordNet concepts **20**
4.1 Manual gold standard for the mapping task 21
4.2 Lesk based disambiguation 22
4.3 Using the vector space model for disambiguation 27
4.4 Comparison and discussion 30
4.5 Annotating relations 31
4.6 Visualizing link structure 32

5 Learning the semantics of Wikipedia hyperlinks **34**
5.1 Pattern based identification of relations 34
5.1.1 Results 40
5.2 Supervised machine learning approach 41

6 Epilogue **46**
6.1 Conclusion 46
6.2 Outlook 47

A Corpus DTD **52**

B Word lists **53**
B.1 Tag set 53
B.2 Stopwords 54

C Results for disambiguation algorithms **55**
C.1 Results for the simple Lesk algorithm 56
C.2 Results for the 'extended' Lesk algorithm 57
C.3 Results for the VSM approach - cosine measure 58
C.4 Results for the VSM approach - dot product measure 59

D Mutual information values for feature selection **60**
D.1 hypernym vs. hyponym ∪ none 60
D.2 hyponym vs. hypernym ∪ none 62

E Results for machine learning experiments **63**
E.1 Classification performance with lexical features 63
E.2 Classification performance with category system features 66

1 Introduction

The "Semantic Web" is an extension of the world wide web which has the goal to augment web content with formal, machine readable semantic information in the style of an ontology [Berners-Lee et al., 2001]. This facilitates the possibility to automatically integrate data from different web sources. The vision behind the semantic web is to provide an environment for intelligent agents. Such agents can autonomously carry out tasks like researching information, buying and selling items on the web and making reservations and appointments.

However, this revolution has yet to happen. At the moment the main drawback for the semantic web is the lack of semantic data. Manual creation of such data, known as ontology engineering, is a tedious task.

For conventional web-content, wikis provide a cheap way to acquire data by letting users generate content collaboratively. Recently, efforts have been made to integrate wikis with the semantic web. Such semantic wikis provide an intuitive web-interface, which allows users to augment wiki content with semantic information. A well known semantic wiki implementation is Platypus Wiki[1] [Campanini et al., 2004].

[Völkel et al., 2005] adapt this idea to Wikipedia. A simple extension of the MediaWiki markup language allows users to create typed links between Wikipedia entries. The type of a link indicates the semantic relation that holds between the concept of an entry and the concept described by the link target. In [Völkel et al., 2006] this idea is implemented in 'Semantic Media Wiki'[2], a modified version of the MediaWiki software on which Wikipedia is based. The software can export the semantic wiki content in OWL [Bechhofer et al., 2004], such that it can be processed by standard semantic web software. The extension is therefore useful not only to facilitate better navigation and search functionality within Wikipedia, but the semantic content can be used as a knowledge base for the semantic web.

The outlook to acquire a huge knowledge base from Wikipedia is especially intriguing because of Wikipedia's size and variety of topics. Currently ontology engineers focus mainly on creating ontologies for a specific domain (e.g medical information). But many language technology applications could profit from a domain independent knowledge base as a source of world knowledge (e.g dialog systems and question answering systems, but also text generation, summarization and machine translation).

Still, with 'Semantic Media Wiki', semantic content has to be created manually, though collaboratively. Once integrated to the working Wikipedia, the new feature has to be accepted by the users, first and even then it will take some time before an ontology of useful size emerges. We might therefore be interested in techniques to automatically generate semantic content from

[1] http://platypuswiki.sourceforge.net, 2007-09-12

[2] http://ontoworld.org/wiki/Semantic_MediaWiki

existing Wikipedia data.

I suggest that some semantic relation can be assigned to a lot of conventional Wikipedia links. For example, consider the Wikipedia entry in figure 1. Links in the article have been marked with different colors, indicating their suggested relational meaning.
Green expresses, that the link target is a hypernym of the concept described in the article, red indicates a hyponym and blue stands for a meronym.

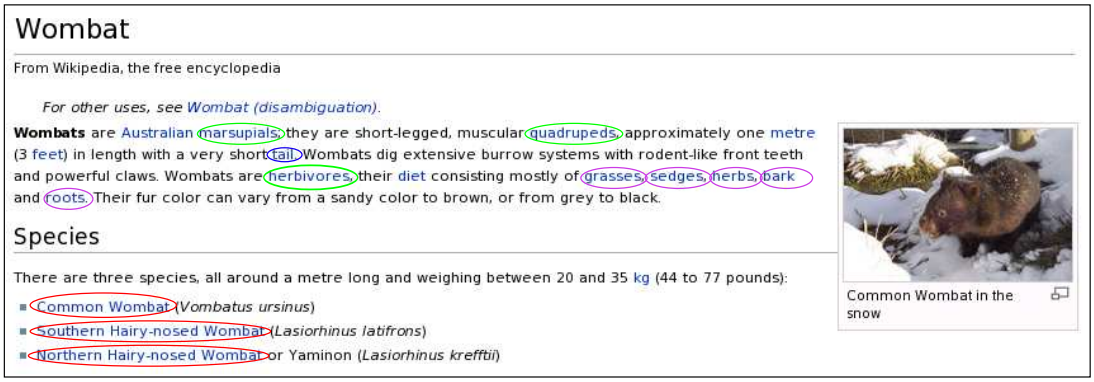


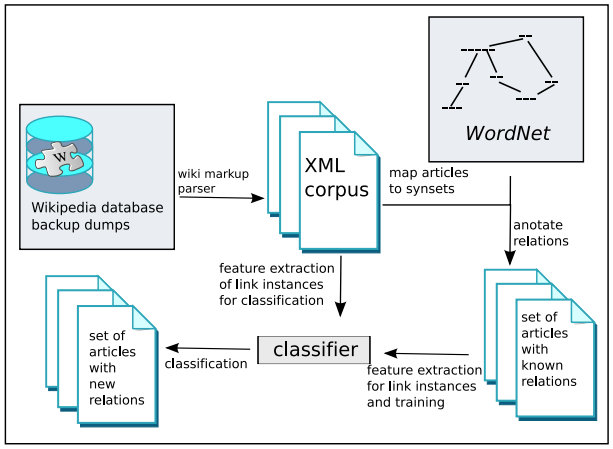
Figure 1: An Wikipedia article. Encircled links are supposed to express a specific semantic relation, indicated by the circle color. green: hypernym, red: hyponym, blue: meronym magenta: 'feeds on'.

The links which are encircled in magenta are intended to show that, in general, arbitrary semantic relations can be identified for links. In this case all the links express a relation, which could be called 'feeds on'. Notice that most of the relevant links appear in the first section of the article. Usually the first section contains a short definition of the concept being discussed in the Wikipedia entry, often similar to a dictionary gloss. I suggest that a lot of semantic data can already be extracted from this section only.

The subject-matter of this thesis is to apply supervised machine learning techniques to the task of identifying semantic relations for Wikipedia links. First a set of features is extracted for each link instance. Then a model is trained to discriminate links, which hold a specific relation, against all others. I restrict myself to the identification of hyponymic relations (hypernymy/hyponymy), as these constitute the taxonomic backbone of ontologies.
As a basis for my work, I need to convert Wikipedia data into a form which is easy to process. In chapter 3, therefore, the assembly of XML annotated corpora from raw Wikipedia dumps is described.
Training a classifier requires training data. The user generated typed links from 'Semantic MediaWiki' would be a valuable source of such data, but since the extension is not yet included

in Wikipedia, typed links were not available. Manual annotation of relation (by the author) was rejected as too laborious, in the first place.
Instead the electronic dictionary WordNet is used to acquire training data. In a first step Wikipedia entries are mapped to WordNet concepts. Then, if a relation is known between WordNet synsets that have been identified to match a Wikipedia article, any hyperlink linking these two articles can be labeled with this relation. Unfortunately, it is not always clear which WordNet concept corresponds to a given entry. Chapter 4 discusses and approaches this problem. Finally the identification of relations itself is described in chapter 5.
Figure 2 gives an overview on the work flow of my venture and shows how the individual components could be assembled to an integrated system.

Figure 2: Work flow of a system to create a corpus with relational annotated links from a raw database dump.



The next Chapter discusses some aspects of techniques and tools we make use of in the remainder of this work in more depth, and presents related works.

2 Background and basic concepts

The first section of this chapter shows how the content of this thesis is related to ontology learning. Afterward an overview on common types of semantic relations is given. Section 2.3 introduces the electronic dictionary WordNet. The last section in this chapter provides information on Wikipedia.

2.1 Ontologies and ontology learning

The subject matter of this thesis can be interpreted as a subtask of ontology learning, the automatic induction of ontologies from unstructured text. We exploit the fact that some structural information is already available from Wikipedia links.

An ontology is a structured, machine readable model of knowledge, in which concepts from a given domain are stored along with their interconnecting relation. The concepts are hierarchically ordered into a taxonomy by a subsumption relation (see below). In addition to a taxonomy an ontology usually also records individuals, viz. instantiations of concepts.

Any object in the ontology comes with a set of attributes, for example the concept *fish* might have the attribute **can swim**. Attributes are thought to propagate down the concept hierarchy. Furthermore in an ontology concepts can be linked by additional relations. For example *lion* and *antelope* could be linked by the relation 'predator of'.

[Buitelaar et al., 2005] subdivide ontology learning into several subtasks. The first is to identify terms in the source text. The second task is to group terms into sets of synonyms, such that each set corresponds to the set of linguistic forms for one concept. Afterwards, in the third subtask, semantic concepts can be formalized. The fourth challenge is to extract the concept hierarchy. In the last step attributes and additional relations between concepts should be learned. This last challenge, however, will not be of our concern and is left for future work.

Using Wikipedia as a data source makes the first subtasks a lot easier. The step of detecting terms and identifying synonyms is rendered unnecessary, because we are only concerned with links and specifically want to identify the relationship between Wikipedia entries. We assume that entries correspond to concepts [3] which are uniquely identified by their URL. Each link description can be seen as a linguistic form of the concept expressed by the link target, i.e. as a term for this concept. Put another way the set of synonyms for a concept expressed by an entry consists of the descriptions of all links pointing to that entry.

The remaining task, on which we will focus, is to extract hyponymic relations between concepts. Both supervised and unsupervised approaches to this task have been made.

[3] But see section 4.1 on difficulties in mapping Wikipedia entries to WordNet concepts.

The latter type of approach (e.g. [Faure and Nedellec, 1998]) is usually based on Harris' distributional hypothesis that words, which appear in a similar context, share a similar meaning. Terms are expressed by vectors of their occurrence frequency within different contexts (usually within different subcategorization frames of verbs). Unsupervised clustering methods can then be used to identify (near-)synonyms, such that a synonym class corresponds to a concept. By hierarchical clustering one can then induce a hierarchy of concepts.
In our case, since we want to extract information from links only, and the term-identification step is skipped, there is not enough data available to create a non-sparse vector representation of terms in context, which can be used for clustering. Furthermore, to identify categorization frames for different words at all, a lot of linguistic preprocessing has to be done. The application of clustering techniques to links does not appear very intuitive anyway, because it ignores the additional structural information they provide. The link based approach is based on the idea that links are used to mark certain relations in the first place. It does not assume that they can be grouped to sets of synonyms depending on their distribution.
Unsupervised approaches are specified to the recognition of synonyms and, when hierarchical clustering is used, hyponyms. In contrast, the following supervised approaches can be generalized to identify other types of relations.
Supervised approaches (e.g [Hearst, 1992]) usually employ lexico-syntactic patterns, either handcrafted or automatically induced from training data, to identify relations. An adaption of this idea to the task of classifying Wikipedia links is presented in [Ruiz-Casado et al., 2005b]. Their attempt will be discussed in section 5.1.
Another possibility, which to my knowledge has not yet been investigated, is to represent each instance as a vector of features and use supervised machine learning classifiers to identify relation. Such an approach is implemented and evaluated in section 5.2.

2.2 Types of semantic relations

In the following I will describe some of the most common semantic relations that have been distinguished traditionally. On the one hand, this overview will help the reader to understand what semantic relations are and how they relate to ontologies. On the other hand, although I will only concentrate on identifying hyponyms and hypernyms, the question of whether our approach can in principle be applied to the identification of other relations will be addressed briefly.

- **Hyponymy and hypernymy:** The hyponymy relation expresses, that one concept is a subordinate of another. For example the concepts *wombat*[4] and *kangaroo* are both a

[4]I will continue to make use of italics to indicate concepts and use quotes for literal term, e.g 'wombat'

kind of *marsupial*. Contrary the hypernymy relation relates a superordinate concept to its sub-concepts (E.g. *marsupial* is a hypernym of *wombat*). The hyponymy relation is also called subordination or subsumption relation. From a set theoretic point of view where we consider the extension of each concept, hyponymy corresponds to the $\subseteq$ relation.

The hyponymy relation is reflexive, viz. each concept is a hyponym of itself. ('a rose is a rose'). Also hyponymy is transitive, which is illustrated by the following example syllogism:

1. *wombat* is a (hyponym of) *marsupial*.
2. *marsupial* is a (hyponym of) *mammal*.
3. therefore *wombat* is a (hyponym of) *mammal*.

Because the hyponymy relation is also anti-symmetric, it defines a partial order on the domain of concepts. The relation can therefore be used to structure concepts into a hierarchy. If a common super concept exists, this hierarchy forms a tree. In computer science one often speaks of such hierarchies as inheritance systems, because the properties of concepts which are higher in the tree are thought to propagate to the leaves[5]

When we talk about ontologies, we are not only confronted with abstract concepts but also with individuals. In knowledge representation one often speaks of the more general IS-A relation (short for 'is a kind of'), which comprises both, concept subsumption and class membership of individuals. In fact hyponymy itself is often understood in this extended sense. However, it can be doubted that such relations can be used to build a complete taxonomy (viz. the result is not a 'taxonomy' in the strict sense), if the distinction between concepts and individuals is blurred [Cruse, 1986]. This problem is illustrated nicely in [Sowa, 2000]:

1. Clyde is an elephant.
2. elephant is a species.
3. therefore Clyde is a species.

Furthermore, according to most definitions (again [Cruse, 1986]), taxonomies structure knowledge into exhaustive decompositions. For example in a taxonomy the concept *animal* could be decomposed into the concepts *male*, *female* and *hermaphrodite*. But *kangaroo* would be a hyponym of *animal*, too.

[5]Obviously this is not true for non-default cases. Certainly penguins are birds, but they do not inherit the ability to fly.

It becomes clear that, strictly speaking, only a subset of the hyponymic relations can be used to structure taxonomies.

Though interesting, in this thesis I will forbear to discuss such philosophical aspects and will rather focus pragmatically on the applications we have in mind. For example when mapping Wikipedia entries to WordNet synsets, I simply assume that both correspond to entities of the same ontological kind (viz. concepts), though many Wikipedia articles describe e.g individual persons.

Due to the importance of the hyper- and hyponymy relation in knowledge representation I focus on learning this relations only.

- **Meronymy and holonymy:** Meronymy is the relation that persists between a whole and its parts. For example the concept *bicycle* can have the meronyms
  - *bicycle seat*
  - *bicycle frame*
  - *bicycle wheel*
  - *chain*

  In contrast holonymy is the relation that relates a part and the whole. E.g *bicycle* is a holonym of *bicycle seat*. Notice that this schema does not correspond to a natural decomposition, viz. there is only one meronym concept *bicycle wheel*, not two.

  Also, several subtypes of meronymy can be distinguished. If a concept can be decomposed into its meronyms (as with the *bicycle* example above), one speaks of a PART-meronymy. A different type is the MEMBER-meronymy. Consider for example the concept *family* with its meronyms *child* and *parent*. Here the concept is not merely made up by the whole of its parts, but by the organization of concepts and how they are related to one another.

  Sometimes other types of meronymy are distinguished, for example:

  - MATERIAL-meronymy: Describes that the object, described by one concept is made of the material, described by the other concept. E.g.: *wood* is a MATERIAL-meronym of *plank*.
  - MASS-meronymy/mereology : Holds if there is no qualitative, but only a quantitative difference between a meronym and it's holonym. E.g.: *centimeter* is a MASS-meronym of *meter*.

  Like hypo-/hypernymy, meronymy and holonymy impose a hierarchical structure on sets of concepts. In knowledge representation one often speaks of a HAS-A relation, or sometimes more specifically about HAS-PART and HAS-MEMBER relations.

In principle, identifying links expressing meronyms and holonyms should be possible with the supervised approach, although links expressing meronymy are expected to appear less frequently than hyponym links.

- **Synonymy:** The term 'synonym' has two different meanings. The first doesn't understand synonymy as a semantic relation between concepts but as a relation between two terms. From this perspective, two terms are synonymous if they refer to the same concept. For example 'car' and 'automobile' are synonyms of each other. According to most definitions two synonymous terms can be used interchangeably. This interchangeability is usually only given for some specific context. [Cruse, 1986] therefore claims, that synonymy in this sense is not a binary relation between two terms, but a ternary relation between the terms and a situation context.

  According to the second meaning of 'synonymy' two concepts are synonymous if they have different intensions, but a common extension. A prominent example for synonymy in this sense is used by Gottlob Frege in [Frege, 1892]: The concepts *morning star* and *evening star* have different intensional meanings, but share a common extension, viz. the planet Venus.

  Synonymy information is already explicitly encoded in Wikipedia by the use of redirect pages.

- **Antonymy:** Intuitively, if two concepts are antonyms of each other, they express each others opposite.

  In fact it is difficult to give a concise definition of semantic antonymy, as the term subsumes several phenomena and differs between lexical categories.

  A certain case of antonymy is on hand, if two adjectives are complementary. For example, the pair 'alive' and 'dead' are antonyms, because for every entity X it holds that X is either alive or X is dead.

  For gradable adjectives, this becomes less clear. For example 'hot' is considered to be an antonym of 'cold'. 'Warm' is in between on the temperature scale and also understood as an antonym of 'cold', but not of 'hot'. Antonyms of this type are called scalar or gradable antonyms.

  If a process leading from a state S_1 to a state S_2 is described by a verb or a noun W_1, then any word W_2 describing the process from S_2 leading to S_1 is an antonym of W_1.

  Due to the vast difference between types of antonymy and the vagueness of the concept, I suggest that it is hard to come up with a generalization of sentence contexts in which

antonyms appear. Therefore supervised approaches, as presented in this thesis, are not suitable for the identification of antonyms.

- **Generic relations:** There are infinitely many generic relations, which correspond to binary predicates. For example the relation **lives in** holds between *wombat* and *Australia*. Under an ontological perspective, these relations correspond to roles of a source concept, that can be filled by role fillers. (viz. the role **lives in** of the concept *wombat* is filled with *Australia*.)

  It should be possible to apply our supervised approach for relation identification to generic relations equally well. However, for most relations training data isn't available from WordNet and training instances must be annotated manually. Possibly the Wikipedia extension proposed by [Völkel et al., 2006], where users can provide such relations when editing the Wiki entry, could provide a cheap source of such data.

  Supposedly for most relations of interest only a small number of instances can be found in Wikipedia. This lack of data poses an interesting challenge to most machine learning classifiers.

2.3 WordNet

WordNet is an electronic dictionary in which all synonymous terms for a common concept are grouped into 'synsets' [6]. Each synset is therefore fully represented by this set of terms and comes with a dictionary gloss. A variety of semantic relations between this synsets is available. WordNet contains nouns, adjectives, verbs and adverbs. For my purposes the noun database is of primary interest, because nouns describe the kind of concept that is usually also described in an encyclopedia. For example, there is no Wikipedia article for the verb 'to learn' but for the action noun 'learning', and likewise there is neither an article for the adjective nor the adverb 'fast'. Furthermore WordNet lists hypernym and hyponym relations only for nouns and verbs, anyway, and for the latter there is only a flat hyponym hierarchy.
WordNet 3.0, the version used for all experiments in this thesis, contained 117797 English nouns. All noun concepts in WordNet are classified in a hyponymy hierarchy. Since an early version there has existed a single root concept 'entity' for all nouns. Meronymy is only available for some of the concepts in WordNet (which is another reason, why I concentrate on learning hyponymy). WordNet does not form an ontology in the strict sense. It is explicitly intended to be a dictionary and therefore records words and their use. In particular it does not form a map of existing things.

[6] In [Fellbaum et al., 1998] Chapter 2, George Miller points out that synonymy of terms in WordNet must be understood as 'interchangeability' in *some* context only.

WordNet synsets can be uniquely identified by their part of speech (noun, adjective, verb, adverb) and their offset in the WordNet source file [Fellbaum et al., 1998].

2.4 Wikipedia

Wikipedia is a collaborative online encyclopedia, based on wiki technology.
A Wiki Wiki Web (short: 'wiki') is a web content management system that allow users to edit content pages directly in their web browser. When editing a page they can make use of a special Wiki markup language (often called wikitext[7]). Most importantly, this language allows the creation of hyperlinks to other pages within the Wiki, but it also usually provides means of formatting text. Wikipedia's wikitext language is described in detail in section 3.2. Wikis usually provide a history function, wherewith changes can be undone and any previous version of the page can be restored (e.g for the case where a malevolent user messes up a page). For an exhaustive discussion of Wiki technology see [Leuf and Cunningham, 2001]. [Ebersbach and Glaser, 2005] provides a short German language overview on this topic.
In Wikipedia these mechanisms are used to enable users to create encyclopedia articles and make arbitrary changes to them. Wikipedia uses the MediaWiki software[8], which was originally developed for Wikipedia, but is now used for several other projects.
Wikipedia is 'free' in the sense intended by the GNU free document license[9]. Under this license all content text can be copied and redistributed freely for both non-commercial and commercial purposes. Usage in other works and changes are allowed under the restriction that the derived work must be 'free' in the same sense.
Wikipedia was started in 2001 as a spin-off of the Nupedia project. Nupedia, also intended to be a free online encyclopedia under the GNU free document license, depended on an editorial staff similar to conventional printed encyclopedias. Users were required to register themselves and new articles were subject to a peer review process. Therefore Nupedia's development process was much slower and soon the project was outranged in size by Wikipedia.
Because every internet user is free to edit articles, the quality of Wikipedia articles can be expected to be lower than the quality of edited encyclopedia articles in both factual information and language (style, spelling, grammar). However, in [Giles, 2005], a team of experts compared the factual correctness of 42 scientific articles from Wikipedia to online articles from Encyclopedia Britannica. On average four errors or important omissions were found in Wikipedia, while Encyclopedia Britannica articles contained three errors and omissions on average. This suggests that the quality of Wikipedia articles is in fact surprisingly good.

[7] Not to be confused with Wikitex, an extension of the Wikipedia Markup language which allows to include LaTeX style setting of mathematical formulas.
[8] http://www.mediawiki.org/wiki/MediaWiki, 2007-09-13
[9] Currently in v.1.2, November 2002. http://www.gnu.org/copyleft/fdl.html (09-02-2007)

Its easy availability, its size, and its structure make Wikipedia a valuable data source for information retrieval and knowledge induction tasks.

Different language versions

Wikipedia is available in a variety of languages, even for rarely spoken languages, dead languages (e.g Latin[10]), dialects and artificial languages. However, many of these side projects contain a considerably lower number of article than 'common' language versions and often have to be considered pure fun projects. In this thesis I utilize two different language versions of Wikipedia. First the English Wikipedia[11] is used. It is the oldest and biggest version of Wikipedia and currently (as of September 2007) contains 1,985,109 articles. The choice is motivated by our need for a lot of data. In order to achieve good results in natural language applications of machine learning techniques, a serious amount of data is usually needed.
Second, I make use of the simple English language version of Wikipedia. This version is intended for learners of the English language, children, and in general people who have difficulties understanding complex concepts as described in the full English language version. It distinguishes itself by the use of easier grammar and a smaller vocabulary. An average article longer than 1000 words in the simple Wikipedia contains only 345 different lemmas. For comparison in the full Wikipedia articles with more than 1000 words contain a vocabulary of 684 different lemmas, on average. The simple English version is much smaller than the original English Wikipedia, as of September 2007 it contained only 19,169 articles. Resulting from the smaller size, the nature of articles is less specific. While the original English Wikipedia contains a lot of articles on more specific concepts (e.g. different varieties of apples like 'Granny Smith' or 'Golden Delicious'), the simple version mostly focuses on general concepts (e.g only the article 'Apple' exists, and none of the varieties are described in separate articles). This criteria make the simple English Wikipedia a good 'toy' example for experiments. Usually I will provide results for both versions. Unfortunately for the final experiments in chapter 5, we have to restrict ourselves to the simple Wikipedia because of time restrictions. Because of the vast amount of training and classification instances from the English Wikipedia, systematic experiments were too time-consuming.

Categories

Most Wikipedia articles are organized into one or more categories. For example the article 'Tornado' belongs to the categories 'Weather hazards', 'Winds', and surprisingly also 'Semi-protected against vandalism'.

[10] http://la.wikipedia.org
[11] http://en.wikipedia.org

The purpose of the category system is primarily to assist users in navigating, rather than to merely classify articles according to a hyponymic relation. For example the category 'Semi-protected against vandalism' serves only administrative purposes, and the category 'Weather' is merely a thematic category.

Categories can be subcategories of other categories and thereby form a network. E.g the category 'Winds' belongs to the categories 'Weather' and 'Sailing'.

Since categories can have both multiple super- and sub-categories, the category system does not form a tree, but a directed graph. The system also contains some cycles. Consider e.g the following example for a 'good-natured' circular classification, taken from Wikipedia's category info page[12]: Education → Social sciences → Academic disciplines → Academia → Education. Though *education* here seems to describe two different concepts, viz. the concrete academic subfield and the general area of knowledge, it represents only a single category. This is again justified by the fact that the category system mirrors rather broad thematic relations than a concept hierarchy. Large cycles can pose a problem for automatic processing of categories.

Still I suppose that the category system can contribute hints to the task of hyperlink classification and we will exploit it as an additional source of features for machine learning in section 5.2.

[12] http://en.wikipedia.org/wiki/Wikipedia:Categories, 2007-09-25

3 Creating an annotated corpus from Wikipedia

While other research based on Wikipedia (e.g. [Ruiz-Casado et al., 2005b]) often retrieves the HTML version of articles and then removes unnecessary elements. My approach is to create an annotated XML based corpus from raw Wikipedia database dumps. This has several advantages. First, extracting only the 'relevant' information from an HTML document is an underestimated task [Bauer et al., 2007]. Second, by storing the data locally in an easily accessible corpus we can access it much faster. Most important a corpus is static, while the online version of Wikipedia changes rapidly. Therefore, as the corpus is available to the public, my results are reproducible. Also the corpus might be valuable for other research based on Wikipedia and, if used as a common reference, allows comparison of results.
In the next section I discuss other corpora built from Wikipedia. The bulk of work in compiling the corpus lies in parsing the markup language, used by Wikipedia, which is described in section 3.2. Section 3.3 describes the process of assembling the corpus and focuses on the description of my markup parser.

3.1 Other corpora

Several corpora built from Wikipedia already exist.
Most prominently known is the Wikipedia XML collection by [Denoyer and Gallinari, 2006]. This corpus is intended to be a source of data for a variety of information retrieval and machine learning tasks with an interest in document structure. It consists of subcorpora created from eight different language versions of Wikipedia. For each subcorpus a hierarchy of Wikipedia categories and an assignment of articles into this categories is available. Additional data collections are focused on more specific tasks, like automatic classification of articles into categories, multi-media information retrieval or entity recognition.
On first glance this corpus appeared very useful. However, on closer inspection many articles in the corpus could not be validated as XML. Others contained a lot of artifacts, mostly wikitext structures, which were not translated correctly. Furthermore only an extract of Wikipedia entries was used, so that many internal link targets were missing in the collection. It would have taken a serious amount of preprocessing work to fit this corpus to my purposes.
Another interesting XML based corpus is described in [Schenkel et al., 2007]. The authors incorporate Wikipedia's category system and article lists to extract semantic information about articles and annotate them with a corresponding WordNet concept. This would have anticipated the work described in section 4.
Unfortunately the latter collection was not yet publicly available at the time of writing. I therefore decided to accept the challenge of assembling my own corpus from original Wikipedia

data as described in section 3.3.

3.2 Wikipedia markup

Wikitext languages are designed to simplify text formatting and structuring and including hyperlinks, tables, images etc. for users without HTML experience. Nevertheless in Wikipedia markup a subset of HTML is available for more advanced users and can be mixed almost freely with other language constructs.
Figure 3 shows an example Wikipedia article in its Wikitext representation and as rendered by a web browser.

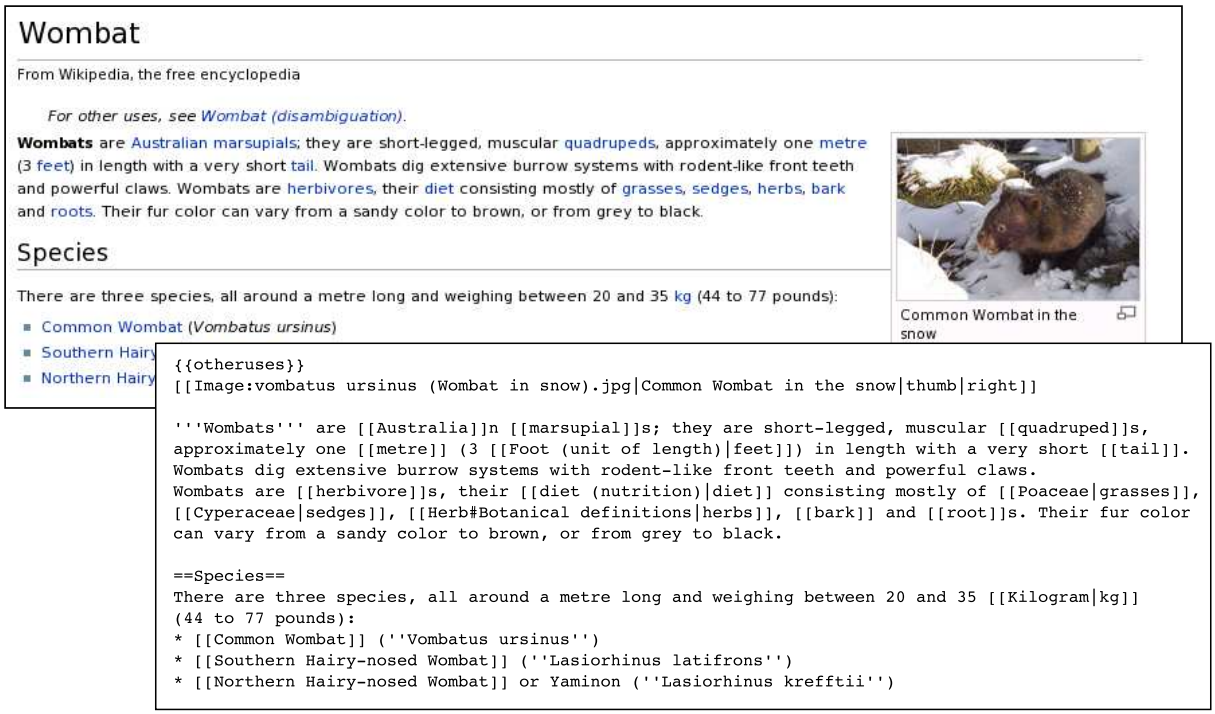


Figure 3: An Wikipedia article as wikitext and rendered by a web browser.

Several difficulties arise from the aim to process Wikipedia markup. First, unfortunately there is no exhaustive and precise specification of the language elements and its semantics. Some features, supported by the original markup to HTML parser, are entirely undocumented but still used in some articles.
Second the validity of markup is not checked when an article is edited and saved. If a user submits erroneous code, it can only be fixed when the page is reviewed by other users. Any parser therefore has to be very robust.

The Wikipedia community has tried to create a grammar specification for the markup language in BNF [13]. Unfortunately up to now their attempts are rather rudimentary. Also it can be claimed, that the language, as licensed by the current MediaWiki parser, is not context free at all, as it contains examples of cross-serial dependencies. For example surrounding a text with `''` can be used to render it in italics and `'''` can be used to render it in boldface. It is then valid to write the code fragment

```
'' italics ''' bold '' font ''',
```

which results in '*italics* **bold font**'.

3.3 Creating the corpus

The process of creating the corpus starts from a raw database backup dump [14] of the full English Wikipedia (Version from May 27th, 2007). The dump contains all Wikipedia articles and other pages in their markup source embedded in a single XML file. Each article comes with additional meta data: its title, a unique numeric id, the last editor etc.

First a SAX based XML parser written in Python is used to consecutively retrieve each article with its content and meta data from the dump. SAX [Brownell, 2002] is the *simple API for XML* to create event driven parsers and is especially well suited for parsing large XML documents without storing them in memory as a whole. Each Wikitext content was parsed and translated into XML (see below), tested for XML validity and then written to the output file.

The resulting corpus is a single, large, valid XML document. The document type is loosely based on the *TEI Lite* standard for linguistic text encoding [Burnard and Sperberg-McQueen, 1995], but strongly simplified.

The high-level structure of the output XML document is designed as follows. The main element is `<wikiCorpus>`, which encloses at least one `<article>` element. Each `<article>` section optionally starts with one or more empty `<category/>` elements, followed by a the `<text>` element. This element contains the main output of the MediaWiki markup parser as described in the next subsection. Appendix A contains the complete XML document type definition (DTD) for the corpus.

[13] http://www.mediawiki.org/wiki/Markup_spec/BNF - 2007-11-28

[14] Wikipedia database dumps can be downloaded from http://download.wikimedia.org/backup-index.html - 2007-08-08

ht

Figure 4: Outline of the Wikipedia corpus XML high-level structure.

```
<wikiCorpus>
   <article title="" id="" wordnet="">
      <category cat=""/>
      ...
      <text>
        [XML annotated parser output]
      </text>
   </article>
   ...
</wikiCorpus>
```

Parsing MediaWiki markup

The implementation of my wikitext parser uses multiple runs of regular expression substitutions. Parsing is sometimes supported by a stack to capture nested structures. Figure 5 shows the wikitext parser output for the example from figure 3. In the following I describe the stepwise operation of the parser.

- The parser first stores passages enclosed by a `<nowiki>`, `<tt>` or `<pre>` tag, because these are intended to appear as-is in the parser output, i.e any markup and HTML code they contain is simply to be ignored by the parser. The passages are only replaced and enclosed in a CDATA section, after the remaining text is processed.

- Next HTML comments and empty XML style tags are removed (e.g. `<references/>`). Common single HTML tags (which do not require an / in the end) are deleted (e.g `<br>` and `<hr>`). Then remaining pairs of matching open and close HTML tags are parsed. Those tags fall mainly into four groups. Some of the tags are removed together with their contents (e.g. `<math>`, `<ref>`,`<del>` and also `<table>` as I considered table data to complex to exploit for my purposes). Others, e.g `<div>` and `<span>` were translated to paragraphs marked with `<p>`. Still others (e.g `<b>`, `<u>`, `<i>`) were interpreted as highlighting their content and translated into `<hi>`. All remaining start and end tags are simply removed remaining the text they enclose. The parser tries to identify misplaced HTML tags and removes them. However, in rare cases misplaced tags cannot be identified and remain in the output, leading to invalid XML. Robust web browsers can still render such invalid HTML code, but transforming it into valid code is not an easy task. Our approach to this problem is to simply remove articles with invalid XML after the conversion process.

- Afterward the parser removes tables and images and special magic words (marked with

```
<text>
  <p>
    <hi> Wombats </hi> are <ref target='Australia' type='article'> Australian </ref>
    <ref target='marsupial' type='article'> marsupials </ref>; they are short-legged, muscular
    <ref target='quadruped' type='article'> quadrupeds </ref>, approximately one
    <ref target='metre' type='article'> metre </ref> (3 <ref target='Foot (unit of length)' type='article'> feet </ref>)
    in length with a very short <ref target='tail' type='article'> tail </ref>. Wombats dig extensive burrow systems
    with rodent-like front teeth and powerful claws. Wombats are
    <ref target='herbivore' type='article'> herbivores </ref>, their
    <ref target='diet (nutrition)' type='article'> diet </ref> consisting mostly of
    <ref target='Poaceae' type='article'> grasses </ref>, <ref target='Cyperaceae' type='article'> sedges </ref>,
    <ref target='Herb#Botanical definitions' type='article'> herbs </ref>,
    <ref target='bark' type='article'> bark </ref> and <ref target='root' type='article'> roots </ref>. Their fur color
    can vary from a sandy color to brown, or from grey to black.
  </p>
  <p>
    <head> Species </head>
    There are three species, all around a metre long and weighing between 20 and 35
    <ref target='Kilogram' type='article'> kg </ref> (44 to 77 pounds):
    <list type='bulleted'>
      <item>
         <ref target='Common Wombat' type='article'> Common Wombat </ref> (<hi> Vombatus ursinus </hi>)
     </item>
      <item>
         <ref target='Southern Hairy-nosed Wombat' type='article'> Southern Hairy-nosed Wombat </ref>
         (<hi> Lasiorhinus latifrons </hi>)
      </item>
      <item>
        <ref target='Northern Hairy-nosed Wombat' type='article'> Northern Hairy-nosed Wombat </ref>
        or Yaminon (<hi> Lasiorhinus krefftii </hi>)</item>
    </list>
  </p>
</text>
```

Figure 5: XML output of the parser for the example article from figure 3.

__UPPERCASE__ in Wikipedia markup)

- Transclusions like info boxes and templates(marked with {{}}), template parameters (marked with {{{ }}}) are also removed. If the disambiguation template was included to mark the page as disambiguation page, the whole page was left out of the corpus. Info boxes provide data on a subject in a very structured way and therefore for the general task to build a knowledge base from Wikipedia they may be of great interest. However they are not directly useful for our purpose of classifying semantic relations.

- Next the parser detects headings and marks them with the `<head>` tag.

- Then internal hyperlinks are processed. Some internal link targets are preceded by a namespace specification, which indicates that they do not refer to an article page within this language version of Wikipedia, but to some other MediaWiki content.
  E.g. `[[Wikipedia:About|about Wikipedia]]` creates a link to the 'About Wikipedia' information page. Such hyperlinks are known as *interwiki links*. All internal links are marked with a `<ref>` tag around the link description, where the attribute `target` specifies the link target and the attribute `type` is either set to `type='article'` for article refer-

ences or to `type='interwiki'`. The example above would therefore be translated as `<ref type='interwiki' target='Wikipedia:About'> about Wikipedia </ref>`. In contrast to the WikipediaXML corpus, where link targets are expressed in terms of article IDs, I decided to retain the plain text article titles, as this is better readable for humans. Within a single namespace this titles are unambiguous.

- Internal links in the `Category` namespace are used to mark an article's category membership. These links are removed and and an empty `category` element is inserted before the article text (see high-level structure, above), where the `cat` attribute was set to one category, the article belongs to, each. In rare cases this feature is also used to mark disambiguation pages (viz. the entry belongs to the category "disambiguation"). In this case, the article is left out of the corpus.

- External links are annotated with a `<xref>` tag and again the `target` attribute specifies the target URL. External links without description are marked with empty `<xptr/>` tags.

- Then highlighted text is enclosed by the `<hi>` tag.

- Afterwards lists are detected and annotated. Top level blocks of lists are detected and processed recursively. Lists are marked with with the `<list>` tag, where the attribute `type` is either set to 'bulleted' for unordered lists, 'numbered' for ordered lists or 'glossed' for definition lists. Each list item is enclosed by an `<item>` element.

- Finally multiple line breaks are interpreted as paragraph breaks. The paragraphs are annotated with the `<p>` tag. Paragraphs starting with the headings 'see also', 'reference', 'external links' and 'notes' and the last paragraph, which usually contains links to other language versions of the given article are removed.

Out of 1,832,334 entries in the English Wikipedia, 12,153 had to be left out of the corpus. Most of them could not be parsed at all, usually due to encoding problems. A few of these pages failed the test for valid XML after parsing. 90,992 other entries were marked disambiguation pages and therefore also left out. The final corpus therefore includes 1,729,189 articles.
For the simple Wikipedia from 18,677 entries, 59 could not be processed. Also 42 disambiguation pages were left out. The final simple Wikipedia corpus includes 18576 articles.

3.4 Conclusion

In this chapter I have described the assembly of an XML annotated corpus from raw Wikipedia database dumps. The resulting corpus is much larger, than the WikipediaXML collection by [Denoyer and Gallinari, 2006], which contains only about 600.000 documents in single XML files.

The corpus is ‘closed’, in the sense that it contains only few links whose target is not included itself.
Another advantage of my corpus is, that it consists only of valid XML.
On the other hand a lot of information gets lost in the conversion process. For example the corpus does neither include images or their positions nor information from info boxes or tables, which is done in the WikipediaXML collection. Also the text structure in my corpus is very flat and captures only a single level of paragraph nesting.
The main difference between my corpus and the WikpediaXML collection is, that the latter is intended as a general purpose collection for the evaluation of various information retrieval algorithms. In contrast my corpus is mainly designed as a data source for the experiments in this thesis.
Then again, as the parser uses exclusively regular expression matching (supported by a stack), the parsing process is fast enough that one can easily modify the scripts and extract a new corpus if additional or other information is required. Creating a corpus from the full English Wikipedia took about 12h on an AMD Opteron Dual Core, 2600MHz, 1024KB Cache machine with 16GB RAM (memory was not a critical factor).

4 Mapping Wikipedia articles to WordNet concepts

For the learning algorithms I want to apply an amount of training and evaluation data is needed. As described in Section 2.3 the kind of semantic relation I intend to classify are annotated for the concepts in WordNet. This chapter therefore discusses algorithms to map Wikipedia articles to WordNet synsets.

The basic idea to solve this task is to measure the similarity between the article and each WordNet synset containing its title, thereby ranking the synsets. The most similar synset is then selected as a match.

The result of this step is a 'mapped' Wikipedia corpus, where each article (`<article>` section) is annotated with a `wordnet` attribute, which is either empty (in case no mapping was found) or has an integer offset in the WordNet noun database as a value.

A given article title is either member of no, a single or many possible WordNet synsets. In the latter case the title is probably polysemous, but we cannot be sure, that the actual concept described by the article is included in WordNet. If the title occurs only once, it is probably monosemous but could possibly refer to other concepts, which are not included in WordNet. It is therefore necessary to include a measure of confidence to the disambiguation process. If this confidence is too low, no mapping is performed and the article instance is left out of the training data set.

The mapping task can therefore be divided into two subtasks. The first one, which I call the **sifting task**, is to decide whether there is a corresponding WordNet concept for a Wikipedia article. The second task is the plain **disambiguation task** in which the algorithm has to choose the best fitting concept for an article out of a set of possible WordNet concepts.

In the next section I will describe, how a manual gold standard for evaluating the task was created.

I then discuss two approaches to the disambiguation task. The first one is a modified version of the well known dictionary based Lesk algorithm, based on vocabulary overlap. The second approach is based on the vector space model of document similarity, which is often used in information retrieval and is mainly a reevaluation of the approach presented in [Ruiz-Casado et al., 2005a].

Finally I show how the resulting mapping is used to annotate relations between articles. Also I show how this relations can be visualized.

4.1 Manual gold standard for the mapping task

To evaluate disambiguation algorithms a manual gold standard was created. A small Python script randomly selected articles from the Wikipedia corpus. Titles were stripped of optional disambiguation hints (e.g. for 'Bank (sea-floor)' only 'Bank' was used as search term). However, these hints were used for the disambiguation itself (see below). If a search term was found in the WordNet noun database, the first three paragraphs of the article were presented to a human judge (the author) along with all potential corresponding WordNet synsets and their glosses. The judge chose the best fitting synset according to his intuition or 'none' if no counterpart could be identified. This process was repeated until a counterpart synset was found for 100 articles for which the search term appeared in multiple synsets (polysemous terms).

Difficulties with article disambiguation

This manual disambiguation yielded some insights into the difficulties, the disambiguation algorithm has to cope with.

Most obvious the English Wikipedia contains a good deal more entries than WordNet and a broader range of subjects. Even if an article name was found in WordNet, in more than half of all cases none of the synsets captured the appropriate meaning. Most of these articles described some work of fiction such as books, movies or TV-series episodes or fictional entities (e.g. the article 'Parturition (Star Trek: Voyager)' is about a TV-series episode and obviously does not match the unique WordNet synset {*parturition, birth, giving birth, birthing*}). Identifying those cases, where none of the WordNet synsets fits, is therefore an important part of the mapping task. [Ruiz-Casado et al., 2005a] seem to ignore this problem.

Another serious problem resides in articles, that do not only describe a single entity, but rather give a 'list' of possible senses of the article name, either with full descriptions, or with hyperlinks to single articles. For the latter case Wikipedia provides a device called disambiguation pages. Their use is recommended by Wikipedia guidelines, but not strictly enforced. Only if this pages were marked appropriately by the author, they were left out of the corpus (See section 3.3).

A variant of this type of article simply gives an etymological overview for a word. The concept described by such articles can be understood as the word itself, not any of its denotations and this is not the type of entity encoded in WordNet (and also not what one usually expects to find in an encyclopedia, but in a dictionary).

Sometimes WordNet senses are very close in meaning, e.g for the term 'apple', two senses are found. The first {*apple*} refers to the fruit, while the second one {*apple, orchard apple tree, Malus pumila*} refers to the tree on which the fruit grows. Algorithms only based on the vocabulary of a single gloss will have a hard time, distinguishing this concepts, especially since WordNet

glosses are rather short.
Finally the granularity of distinction between close meanings varies between WordNet and Wikipedia and also from domain to domain. For example in the simple Wikipedia a single article 'Apple' describes both, the fruit and the tree. In such cases it is difficult to decide which WordNet concept fits best if all possibilities seem to be either to specific or too broad. For the apple example one could well consider either choice appropriate.

Results of the manual mapping

Out of 26519 random article titles in the full English Wikipedia, 25619 [sic] were not found in WordNet at all. From the remaining 900 articles, those included in the gold standard, the judge could not identify an appropriate WordNet synset for 453 articles. This leaves 447 articles, from which 347 had a monosemous and 100 had a polysemous title. These counts result from the mapping procedure which was repeated until 100 Wikipedia entries with polysemous title were found.
For comparison, I also mapped articles from the simple English Wikipedia to WordNet concepts. The same strategy as above was used and worked significantly better, due to the less fine grained nature of concepts in the simple Wikipedia. Out of 955 articles only 675 were not found in WordNet. From the remaining 280 articles, 23 had no appropriate sense in WordNet, 157 were monosemous, 100 polysemous.

4.2 Lesk based disambiguation

The Lesk algorithm

Word sense disambiguation (WSD) is the task of determining the correct sense for a given ambiguous word in context. Our task of mapping Wikipedia articles to WordNet synsets is quite similar, but instead of a small context window around the given word, it comes with a whole article, elaborating on the word's meaning.
Probably the most prominent dictionary based WSD algorithm is the Lesk algorithm ([Lesk, 1986]. It is based on the intuition, that texts about a similar topic usually use a common vocabulary of specific words. To identify the correct sense of a word in its context, the Lesk algorithm tries to assign a correct dictionary entry to the word. For each ambiguous word in a phrase the Lesk algorithm collects a set of indicator words. For each word in context of the ambiguous word (usually four to eight words), indicator words are collected from their dictionary glosses. The algorithm calculates the overlap between the set of all indicator words and the set of words from each possible gloss for the ambiguous word. Finally it chooses the sense whose

gloss achieves the highest overlap. [Lesk, 1986] reports disambiguation accuracies of 50-70% for short samples from 'Pride and Prejudice' and from an Associated Press news story.
An adaption of this algorithm to WordNet is described in [Banerjee and Pedersen, 2002]. The authors compare every possible pair of words from the context window surrounding the ambiguous target word and assign each pair the senses that achieve the highest score. To calculate this score for a pair of words so-called *relation pairs* are defined in the following way. In WordNet there is a number of directly related senses under each relation (antonymy, hypernymy etc.) for each sense. A relation pair indicates that the senses under relation r of one sense have to be compared to the senses under relation s for the other one. E.g in a relation pair one chooses the antonym for one sense and the hypernym for the other sense. For each possible sense combination of of a pair of words, the gloss overlaps for all possible *relation pairs* are counted and then summed up. Since WordNet includes seven relations (over all POS), there is a total of 49 of such relation pairs to be considered for each pair of senses.
However, this approach is not well suited for our purpose of mapping Wikipedia articles to WordNet synsets, because it is computationally expensive, especially since articles are usually longer than the context window used by [Lesk, 1986] and [Banerjee and Pedersen, 2002]. Instead in this chapter we will relay on the broader context we get from each article. Furthermore I suppose that this context provides more reliable information for disambiguation, than additional indicator words extracted from WordNet.
The results of the Lesk based algorithms will afterwards be compared to the reevaluation of the vector space model based approach.

Simplified Lesk algorithm

For first experiments I implemented a simplified version of the Lesk algorithm. The algorithm directly calculates the vocabulary overlap between the gloss for each possible sense of the title and the first paragraph of an article. In almost every Wikipedia article the first paragraph contains a brief definition of the concept to be described, usually similar to a dictionary gloss[15]. I therefore expect the disambiguation algorithm to perform better, when it is applied to this paragraph only. For comparison I also tried using the overlap between the whole article and the gloss, though this is expected to yield noisy data.
Both, glosses and article paragraphs, are POS tagged and lemmatized by the TreeTagger ([Schmid, 1994]) using the default parameter file for English, that uses the Penn Treebank tag set [16]. The TreeTagger is a POS tagger, that uses decision trees to predict transition proba-

[15] In some articles the first paragraph contains a reference to another article or a disambiguation page. Such paragraphs, if not already removed during corpus generation, have been identified by regular expression matching and were skipped.
[16] See appendix B.1 for the tag set.

bilities. For English [Schmid, 1994] reports a tagging accuracy of 96,36% on data from the Penn Treebank.

Common stop words are removed from paragraphs and glosses, as they are expected to yield overlaps without any significance for the semantic relatedness of words [17]. The article's title and the names of all categories to which the article belongs are added to the paragraph's word list. The overlap is calculated by counting the number of occurrences in the target paragraph for each word from the gloss.

Figure 6: WordNet glosses for possible senses of the article 'Water (molecule)'. Overlapping vocabulary is marked in bold face. The simple Lesk algorithm selects the correct sense (1)

1.* **water**, **H2O** --	**binary compound** that occurs at **room temperature** as a clear **colorless odorless tasteless liquid**; freezes into ice below 0 degrees centigrade and boils above 100 degrees centigrade; widely used as a **solvent**
2. body of **water**, **water** --	the part of the **earth's surface** covered with **water** such as a river or lake or ocean
3. **water** --	once thought to be one of four elements composing the universe (Empedocles)
4. **water** system, **water** supply, **water** --	a facility that provides a source of **water**
5. urine, piss, pee, piddle, weewee, **water** --	**liquid** excretory product
6. **water** --	a **liquid** necessary for the life of most animals and plants

title: Water (molecule)

text:
Water (H2O, HOH) is the most abundant molecule on Earth's surface, composing of about 70% of the Earth's surface as liquid and solid state in addition to being found in the atmosphere as a vapor. It is in dynamic equilibrium between the liquid and vapor states at standard temperature and pressure. At room temperature, it is a nearly colorless, tasteless, and odorless liquid. Many substances dissolve in water and it is commonly referred to as the universal solvent. Because of this, water in nature and in use is rarely clean, and may have some properties different than those in the laboratory. However, there are many compounds that are essentially, if not completely, insoluble in water. Water is the only common, pure substance found naturally in all three states of matter—for other substances, see Chemical properties.

categories: Water

WordNet glosses vary in length and notably glosses for word senses that are more common are often longer. Therefore simply choosing the gloss, that yields the highest overlap might give the wrong result in some cases. A better approach is to use an association measure, which weights the overlap by the article and gloss length and therefore yields comparable scores.

Here I use the Dice coefficient, which is based on the harmonic mean. In our case, it is given by

$$Dice(P, G) = \frac{2 \cdot O(G, P)}{|G| + |P|},$$

where P is the set of word tokens from the article paragraph, G is the set of possible indicator

[17] See appendix B.2 for the list of stop words.

words (word types) from the WordNet gloss and the function O yields the overlap, i.e how many tokens of any word type in G appear in P. [18].
This measure does not only allow us to compare the association score for different glosses, but also for several article instances. We can therefore address the sifting task by specifying a threshold, such that the algorithm decides that none of the senses in WordNet fits the concept described by the Wikipedia article, if the association score for all glosses falls below this threshold.

'Extended' Lesk algorithm

The second algorithm I implemented is actually closer to the original Lesk algorithm. The Wikipedia article paragraph is not directly compared to the WordNet gloss, but if possible indicator words are extracted for each word in the paragraph. If one of the context words is polysemous itself, the gloss which maximizes the overall score is used. This technique enriches the article by additional indicator words, thereby leading to higher overlap scores for 'good' senses and thus hopefully resulting in better accuracies.

Results and discussion

When evaluating the mapping process one has to separate strictly between the two subtasks. For the disambiguation task the appropriate WordNet concept for a Wikipedia entry has to be selected, where one of the possible concepts fits definitely. The sifting task consists of the rejection of all possible concepts, that were found in WordNet for a given Wikipedia article, if the confidence for mapping them is too low.
For the disambiguation task random guessing of the correct synset yielded an average accuracy of 35.9% on the simple Wikipedia corpus (36.7% on the English Wikipedia). Since WordNet synsets are ordered according to their frequency of use, another baseline can be drawn by simply choosing the most common synset for each entry, which yielded 56% accuracy for the simple Wikipedia and 49% accuracy for the English Wikipedia. The worse result for the English Wikipedia is not surprising, because this version includes more articles for rarely used concepts, as we have observed before.
Table 1 compares the disambiguation results for both simple Lesk and extended Lesk algorithm. On the plain disambiguation task the simple Lesk algorithm in general shows better results than the full version. The best accuracy achieved with the simple Lesk algorithm was 79% on the English Wikipedia (67% on the simple Wikipedia), while the extended algorithm only achieved a top accuracy of 76% (65% on the the simple Wikipdia). This suggests, that augmenting the article or paragraph with additional indicators from WordNet yields noisy data.

[18] Note how this is different from the cardinality of the intersection $|P \cap G|$, because multiple occurrences of a word type from G in P are counted individually. P contains word tokens, while G is a set of word types.

Table 1: Performance of Lesk based algorithms on the disambiguation task (only select the single correct match from a set of WordNet concepts.)

.

	Simple Wikipedia		English Wikipedia	
Random choice (avg.)	35.9%		36.7%	
Most common	56%		49%	
Paragraphs	first	all	first	all
Simple Lesk	63%	67%	75%	79%
Extended Lesk	65%	59%	69%	76%

On the other hand, at least for the simple Lesk algorithm, using the whole article works better than using the first paragraph only. As already observed in [Lesk, 1986], the number of indicator words has a strong influence on the disambiguation performance of the Lesk algorithm. My results show that, for the special case of disambiguating encyclopedia articles, indicator words extracted from a larger portion of semantically related context are more valuable than those extracted from WordNet glosses for only a view context words.

The lower results for the simple Wikipedia can be attributed to the smaller vocabulary, used in this version. Often general words like 'make' are used, where the English Wikipedia contains more specific words like 'produce', 'cook', 'assemble' etc. A given word therefore can appear in broader contexts and becomes less informative for disambiguation.

Appendix C.1-C.2 shows evaluation results for the sifting task.

This task was only evaluated on the English Wikipedia gold standard because in the simple Wikipedia case only for very few entries none of the possible synsets fits. The decision whether any of the possible synsets corresponds to an entry after all, could have been evaluated on 32 articles in the evaluation set, only, which certainly is not representative.

As for the interpretation of these results, we might be interested in two aspects. First the best disambiguation accuracy achieved with an optimal threshold value is of interest, second we want to know whether a stable threshold can be found. Such a stable threshold discriminates precisely all map-able from non-map-able articles.

A low association threshold yields high precision and small recall because only few articles, for which we can be confident that there is no possible mapping, fall below the threshold. In contrast, a high threshold causes the rejection of too many articles and leads to a high recall with a low precision. For a stable threshold both high precision and recall are needed. To estimate

the general quality of a threshold value, the F-score can be calculated. It is given by

$$F = \frac{2prec \cdot rec}{prec + rec},$$

where *prec* is the precision and *rec* is the recall. Low F-scores indicate that the association measure is not well suited to the identification of non-map-able articles.
For the simple Lesk algorithm an optimal Dice threshold was found at 0.2, with an F-score of 85.35 for all paragraphs and 89.85 when only the first paragraph was used. The total disambiguation accuracy at this threshold was 68.53% (75.61% only on the articles with multiple possible synsets) when all paragraphs were used, and 74.13% (71.65% only multiple synset cases) with only the first paragraph. Suprisingly the identification works better, if we restrict ourselves to indicator words from the first paragraph of each entry, though we have seen that the disambiguation itself profits from additional data.
The threshold with the best F-score for the extended Lesk algorithm in experiments with complete articles was found at 0.35 (F-score 86.21). The accuracy with this threshold was 58.45% (73.17% only multiple synset cases). For the first paragraph the optimal threshold was 0.3 (F-score 86.40), with an accuracy of 66.31% (75% only multiple synset cases).
In general the simple Lesk algorithm also performs slightly better on the second task. In addition it is much faster than the 'extended' Lesk algorithm.

4.3 Using the vector space model for disambiguation

[Ruiz-Casado et al., 2005a] assign articles from the simple English Wikipedia to WordNet concepts, using an approach based on the vector space model of semantic document similarity. They represent articles and WordNet synsets by vectors of word occurrence frequencies in the article and the synset gloss respectively. The dot product or cosine measure is calculated for each gloss and the article. Then the synset with the gloss, that yields the highest score is selected as match. The results are evaluated manually. The authors report a disambiguation accuracy of 91.11% (83.89% only on polysemous words). This approach is described in detail and reevaluated in this section.
In contrast to our discussion the authors assume that a mapping is always possible, if the article title was found in WordNet. This is, especially for the full English Wikipedia, not the case.
Furthermore the vector space based approach seems a little unintuitive at first glance, as it follows rather in the footsteps of corpus based word sense disambiguation than dictionary based WSD. Using dictionary based approaches for this task is more manifest, because we do not only try to disambiguate senses of words in context by means of a dictionary, but we want to disambiguate whole encyclopedia entries which are (at least partly) similar in style to dictionary

glosses. However, in principle the approach is very similar to the simple Lesk algorithm, because both are based on a similarity of word distributions.

Term weighting

As with the Lesk based approaches, WordNet glosses of different length pose a problem. A long gloss is more likely to contain words from the article and therefore these are often favored by the algorithm. One simple solution would be to normalize the weights by document length.
If, for example, a gloss contains a lot of repetitions of 'the', this certainly doesn't yield a lot of information on how much it is semantically related to the Wikipedia entry.
As a term weighting method, *tf_idf* [Salton and Buckley, 1988] is used. It is based on two values, the term frequency and the document frequency. The term frequency tf_{td} indicates how often a term t appears in a given document d. In this case a 'document' refers either the text of an Wikipedia article or the gloss for a WordNet concept. The document frequency df_t is the number of documents in which t occurs. The inverse document frequency is given by $idf_t = log\frac{N}{df_t}$ where N is the total number of documents. *idf* therefore can be interpreted as a measure of how informative or topic specific a word is. If a word occurs within all documents, $df_t = N$, and since $log\frac{N}{N} = log1 = 0$, idf_t becomes 0. If a word appears only in a single document the value becomes maximal: $idf_t = logN$.
The tf_idf weighting for a term t in a document d is:

$$tf_idf_{td} = tf_{td} \cdot idf_t = tf_{td} \cdot log\frac{N}{df_t}$$

In fact tf_idf is the name of a complete scheme for term weighting, which occurs in several versions [Manning and Schütze, 1999]. However the equation above (using plain tf_{td} as term occurrence measure, logarithmized document frequency and no normalization) is the most common one.

Measuring similarity

The similarity between each gloss vector and the article vector is then calculated to rank the gloss vectors. Two different measurements are used in [Ruiz-Casado et al., 2005a], viz. the simple dot product and the cosine measure.
The dot product of two vectors $\mathbf{v} = (v_1, \cdots, v_n)$ and $\mathbf{w} = (w_1, \cdots, w_n)$ is given by

$$\mathbf{v} \cdot \mathbf{w} = \sum_{i=1}^{n} v_i \cdot w_i.$$

Table 2: Performance of the VSM based algorithms on the entry disambiguation task (only select the single correct match from a set of WordNet concepts) with different measures

.

	Simple Wikipedia		English Wikipedia	
Random choice (avg.)	35.9%		36.7%	
Most common	56%		49%	
	first	all	first	all
cos	65%	54%	72%	61%
eucl.	32%	41%	47%	62%
dot	70%	66%	76%	65%

The cosine measure can be interpreted as a normalized variant of the dot product. For two vectors $\mathbf{v}$ and $\mathbf{w}$ it is given by

$$cos(\mathbf{v}, \mathbf{w}) = \frac{\mathbf{v} \cdot \mathbf{w}}{|\mathbf{v}||\mathbf{w}|} = \frac{\sum_{i=1}^{n} v_i \cdot w_i}{\sqrt{\sum_{i=1}^{n} v_i^2}\sqrt{\sum_{i=1}^{n} w_i^2}}.$$

The cosine measure is often used as a similarity measurement for word frequency vectors in NLP tasks.

Another often used (distance) measurement is the euclidean distance

$$eucl(\mathbf{v}, \mathbf{w}) = \sqrt{\sum_{i=1}^{n} (v_i - w_i)^2}.$$

Notice that the euclidean distance is a metric and becomes smaller, the more similar a gloss is to the article vector. Therefore it ranks glosses in reverse order.

As with the Lesk algorithm a threshold value needs to be specified for the dot product, cosine measure and euclidean distance to address the sifting task.

Results and Discussion

Table 2 shows results on the plain disambiguation task for cosine, euclidean and dot product measure.

The best accuracy for both language versions was achieved with the dot product measurement. This result accords to those given in [Ruiz-Casado et al., 2005a]. Euclidean distance showed bad performance. In the simple Wikipedia case, when only the first paragraph was considered,

the accuracy of 32% was even below the random baseline of 35.9%. Because of the bad results I didn't pursue any further experiments with this measure.
Appendix C.3-C.4 provides results for the task in which articles without a WordNet counterpart had to be identified. Again results are only given for the English Wikipedia corpus.
With cosine measure a stable threshold value for experiments with the whole article was found at 0.4 (F-Score 84.34). With this threshold an overall accuracy of 77.87% (71.65% for articles with multiple synsets) was achieved. Evaluated only on the first paragraph the best cosine threshold was at 0.35 (F-Score 87.39) and yielded an accuracy of 76.71% (75.91% on articles with multiple synsets).
As for the dot product measure a good threshold with all paragraphs was found at 32 (F-Score 89.89%) and achieved an accuracy of 67.41% (72.87%). With only the first paragraph the best threshold was at 6 (F-Score 87.92), with an accuracy of 66.97% (57.62%).

4.4 Comparison and discussion

The results for both mapping approaches are lower than those reported by [Ruiz-Casado et al., 2005a]. This is mostly due to the different evaluation method.
In my experiments the accuracies are a measure of resemblance of the manually mapped corpus, as the results were evaluated on a gold standard (which could be called 'in vitro' evaluation), while [Ruiz-Casado et al., 2005a] evaluate their results by manually investigating the quality of disambiguation afterwards (which one could call 'in vivo' evaluation). In fact a manual evaluation of 200 randomly selected mappings from the resulting mapped English Wikipedia corpus (simple Lesk algorithm, Dice threshold 0.25) showed an accuracy of 90.05%.
I suppose that manual subsequent inspection gives better results for the following reason. Since WordNet senses for a given word often have a very similar meaning. Therefore one would more frequently accept a mapping as correct, if the other senses are out of sight, even if one would have chosen another mapping manually. To further investigate this effect, I would be interested in the annotator agreement between several human judges on this task.
My approach to evaluation, on the other hand, exhibits the advantage that one does not have to manually inspect the classification on a single instance level over and over again for different experiments.
In contrast to the results by [Ruiz-Casado et al., 2005a], the simple Lesk algorithm yielded the best results for plain disambiguation. The simple Lesk algorithm is also especially intriguing because of its simplicity. On the other hand, when the sifting task was added it was difficult to find a stable Dice threshold. The cosine measure was more suitable for this task.
A simple possibility to make the sifting task easier is to use simple heuristics. For example one could immediately remove articles whose name contains certain disambiguation hints in brackets

(E.g. “TV-Series” or “Video Game”).
Of course, the quality of any supervised classifier depends on the quality of the training data. I suppose that the data, we have attained in this chapter, is sufficient in size and quality for this purpose. However, we have to keep in mind, that the training data is not entirely free of errors. Especially, if part of it is used for evaluation of classifier performance, the results have to be interpreted in relation to the data quality.

4.5 Annotating relations

Having mapped articles from a Wikipedia corpus to WordNet synsets when possible, we now can annotate links with relations, if a relation is known in WordNet between a link's origin concept and target concept. First a dictionary, mapping article names to WordNet offsets, is created from a ‘mapped’ Wikipedia corpus. Then the following algorithm is applied:

- For each article
  - Look up an article's WordNet Synset S. If none was found proceed to next article.
    - Get the transitive closures $C_{hyper}(\{S\})$ under the hypernym operation.
    - Get the set of hyponyms $hypon(\{S\})$ of S. This restriction was made because retrieving the complete closure under hyponymy, i.e. the complete hyponymy subtree, turned out to be very slow in practice.
    - For each link in the article
      - Look up the WordNet Synset T of the link target. If none was found proceed to the next link.
      - If $T \in C_{hyper}(\{S\})$, add the attribute `rel='hypernym'` to the `<ref>` element of the link.
      - If $T \in hypon(\{S\})$, add the attribute `rel='hyponym'` to the `<ref>` element of the link.
      - Else add the attribute `rel='none'` to the `<ref>` element of the link.

By this technique in total 15575 hypernyms and 5192 hyponyms could be annotated in the full English Wikipedia corpus. For the simple Wikipedia corpus the algorithm found 1974 hypernyms and 417 hyponyms.
To get an impression of the quality of the relation annotation, 100 hypernyms and 100 hyponyms were selected randomly and evaluated by the author. Nine links were incorrectly marked as hypernyms, corresponding to a precision of 91%. For hyponyms the precision was 93%, as seven links were incorrectly marked as hyponyms. These final results of the annotation are

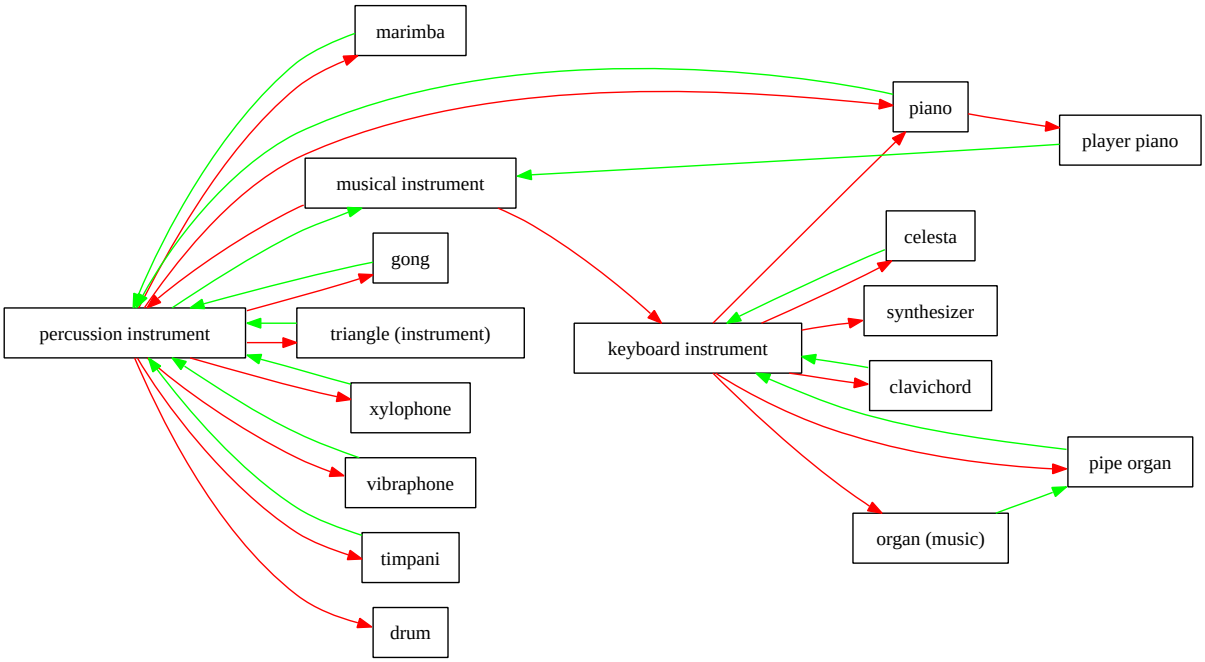
surprisingly good, considering the difficulties and results of the mapping task. On the other hand, if a false concept is assigned to an article in the mapping step, this doesn't necessarily influence the annotation of links. A link is only annotated with a relation, if the relation is present in Wordnet. If one of the articles is mapped to the wrong concept usually such a relation cannot be found. A bad mapping therefore rather decreases the number of training instances than the quality of the annotation.

4.6 Visualizing link structure

A small tool (`graphviz-tool`) was written to visualize the link structure in a relation annotated corpus. The process starts from an initial article S, and then recursively follows all links, marked with given relations, up to a level of N. If a link was found, 'edge statement' in the DOT language is created. DOT is a graph layout description language, used by the programs in the Graphviz graph drawing collection [Ellson et al., 2003]. The Graphviz tools can then be used to render the graph description file, created by `graphviz-tool`, in a variety of image formats. Figure 7 shows an example plot of the links structure in the relation annotated Wikipedia corpus (simple Lesk mapping algorithm).

This type of visualization, though not of immediate use for our work, gives a good impression of the Wikipedia link structure. Included to an online system these visualizations could serve as an additional navigation aid for Wikipedia readers. Visualizations could also be interesting for other research based on Wikipedia. For example they could be used to investigate where and for which purpose Wikipedia users include links.

Figure 7: Extract from the hyponymy link structure in the English Wikipedia after mapping articles to WordNet synsets. Green arrows indicate existence of a hypernym link between the articles, red arrows indicate existence of a hyponym link. The seed term 'piano' was used.



5 Learning the semantics of Wikipedia hyperlinks

As discussed before, we assume that certain hyperlinks in Wikipedia articles can be assigned some semantic relationship, between the concept described by the source article and the concept described by the link target. In this chapter I discuss supervised approaches to identify links that express a certain relation.
We will consider two general types of approaches to this problem. The first approach is based on pattern matching with the phrase, in which two named entities between which we want to classify a relation, appear. The second approach is based on supervised machine learning techniques.

5.1 Pattern based identification of relations

When inspecting the links that have been tagged as hypernyms in section 4.5, they mostly appear in the first paragraph of an article and noticeably they are often embedded in phrases of similar shape. As observed before, the first paragraph seem to correspond to a definition of the concept described in the article, similar in style to a dictionary gloss. Consider for example the following hypernym links from the relation annotated simple Wikipedia corpus. The title of an article is marked in boldface and the link is written in italics.

> **Air** means earth's *atmosphere*.
>
> It is a *sculpture*.
>
> An **electron microscope** is a *microscope* that uses electrons .
>
> The **bubonic plague** is a very deadly *disease*.
>
> **Richard Strauss** was a German *composer* .
>
> Sir **Isaac Newton** was an English *physicist* and *mathematician*.
>
> A **vertebrate** is an *animal* with a spinal cord or spine inside its body.
>
> The **aardvark** is a *mammal* from Africa .

These patterns share a similar structure, where the article title is in subject position, followed by a form of 'to be' in predicate position and finally the link as head of the noun phrase in object position. There it is sometimes modified by a preceding adjective or a subsequent prepositional phrase (example 7) or relative clause (e.g example 3). In example 2, the subject is replaced by a pronoun.

The idea behind pattern based approaches is to collect phrases which, as in the example, express some type of semantic relation from text and, either manually or automatically, abstract from them in some way. The abstracted pattern is then used to identify this relation by matching it with potential phrases.
One of the first approaches to learn hyponymy from text using patterns is described in [Hearst, 1992]. The authors use handcrafted pattern to identify hypernyms, but do not discuss how patterns can be automatically induced. [Ruiz-Casado et al., 2005b] use generalized patterns to identify links expressing hyponymy and meronymy in Wikipedia. They collect context phrases for links, for which a relation is known from WordNet and automatically abstract from them. I will try to reevaluate and discuss this approach in the next section.

Pattern extraction

For each article and each link which was annotated as a hypernym in the previous step, the link was extracted in context of it's surrounding sentence. The sentence was POS tagged, using, again, the tree tagger [Schmid, 1994]. A first step of simplification was done by cutting the sentence after the link, where certain markers, indicating a prepositional phrase or subordinate (usually relative-) clause was found. Those where the punctuation marks `,`, `;`, prepositions (with,in,on,by,from), and relative pronouns (who, which, that...). Parenthesized parts were removed. If the article name appeared in the sentence, it was replaced by the keyword `TITLE` and the link itself was replaced by the keyword `TARGET`. For the examples from above, this yields the following patterns:

TITLE means earth's TARGET.

It is a TARGET.

An TITLE is a TARGET

The TITLE is a very deadly TARGET

TITLE was a German TARGET

Sir TITLE was an english TARGET and mathematician

A TITLE was an english physicist and TARGET

A TITLE is an TARGET

The TITLE is a TARGET

The patterns are so similar because of the the lexicographic style, in which most Wikipedia articles are written. This makes the task of classifying relations (esp. hypernyms) somewhat easier than classifying relations between arbitrary named entities in other raw text.
However, obviously the patterns are still very specific. In order to identify a whole class of links that share a common relation one needs to find abstractions from them.

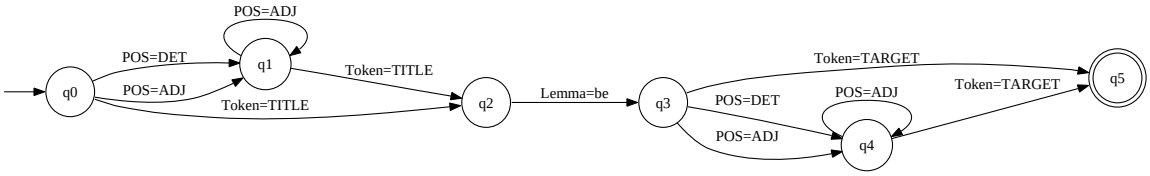
Preliminary experiment

A handcrafted prototypical abstraction of the phrases above, using a regular expression like notation, could look like this:

DET? ADJ* 'TITLE' be DET? ADJ* 'TARGET'

where for each token either the word (in quotes), a lemma or a POS tag is specified. `?` marks the previous token as optional, `*` allows zero or more repetitions of the previous token.
In a preliminary experiment, I tried to identify hypernyms by matching sentences, containing a target link, with a finite state automaton created from the handcrafted pattern (see figure 8).

Figure 8: An FSA for a simple pattern, to identify hypernyms



In the simple Wikipedia corpus, the pattern matched 5379 times. From these matches 417 hypernyms were already annotated from WordNet, and 4962 were new ones. Since there were 1974 hypernyms annotated in section 4.5, this amounts to a recall of 21.12% [19]. The low recall indicates that the pattern is far to specific to match all hypernyms.
The first 200 matches of were manually evaluated by the author. 164 links were correctly identified as hypernyms (from which 22 hypernyms were already known from section 4.5), yielding a precision of 82%. A common cause for false positives were phrases like "**Wombats** are *Australian marsupials*", where *Australian* links to the article "Australia" and *marsupials* links to the article "Marsupial". Here only the link *marsupial* expresses a hypernym relation. Unfortunately both links represent a single instances for classification and produce the patterns "TITLE are TARGET marsupials" and "TITLE are australian TARGET" resp. and the automaton matches both of them.

[19] Of course this value is only relative to the quality of the Wikipedia article to WordNet mapping.

The results suggest that a set of carefully handcrafted patterns might work well for identifying relations. On the other hand I doubt that patterns this simple can be found to identify hyponymy and other relations. In any case building patterns by hand is very costly in terms of labour. Therefore we would like an approach to induce abstract patterns automatically.

Pattern abstraction

In this section I describe a way to automatically abstract from patterns. The method is taken from [Ruiz-Casado et al., 2005b] and based on a modified version of the minimal edit distance algorithm. The algorithm does not generate patterns in the style of regular expressions (as above), but uses it's own syntax. The abstract patterns allow for disjunction of two tokens (using | as a separator), and includes wildcards (using * to match any sequence of tokens).
The idea is to derive a generalized pattern from two pattern each, replace this two pattern by the result and continue the process until only a small number of pattern remains.
Minimal edit distance or Levenshtein distance [Levenshtein, 1966], is a distance measure between strings. It is defined as the minimal number of insertions, deletions and substitutions needed to derive one string from another. To calculate this distance an efficient dynamic programming algorithm exists.
For two strings $A = A[1] \cdots A[m]$ and $B = B[1] \cdots B[m]$ an $(m+1) \times (n+1)$ matrix $\mathcal{M}$ is created and filled in the following way. The first row is filled with the values $0 \cdots m$, and the first column with the values $0 \cdots n$. The remaining places are filled iteratively from left to right and top to bottom in the following way.

$$\forall i \in \{1, \cdots, m\}, \forall j \in \{1, \cdots, n\} : \mathcal{M}[i,j] = \begin{cases} \mathcal{M}[i-1,j] & \text{(deletion)} \\ \mathcal{M}[i,j-1] & \text{(insertion)} \\ \mathcal{M}[i-1,j-1] + cost(i,j) & \text{(substitution)} \end{cases}$$

where

$$cost(i,j) = \begin{cases} 1 \text{ } if \ A[i] = B[j] \\ 0 \text{ } else. \end{cases}$$

Each place $\mathcal{M}[i,j]$ contains the minimal edit distance between the string $A[1] \cdots A[i]$ and $B[1] \cdots B[j]$. $\mathcal{M}[0,0] = 0$ expresses the fact that the edit distance between two empty strings is 0. The values in the first row give the costs for the deletion operations needed to derive the empty string from each prefix of a, while the values in the first column contain the cost for the insertion operations needed to derive each prefix of b from the empty string. The value in $\mathcal{M}[m,n]$ is the minimal edit distance between A and B.

Parallel to the matrix $\mathcal{M}$, a matrix $\mathcal{D}$ can be filled [20], where each place $\mathcal{D}[i,j]$ records the operation that was selected for the calculation of $\mathcal{M}[i,j]$: Either the literal `D` for deletion, `I` for insertion, `E` if $A[i]$ and $B[j]$ were equal (no operation, $cost(i,j) = 0$) or U for a real substitution (if $A[i] \neq B[j]$ and therefore $cost(i,j) = 1$). If more than one operation yields the minimum cost, the precedence is `D`, `I`, `E`/`U` (this is not mentioned explicitly in [Ruiz-Casado et al., 2005b], but follows from their examples).

Table 3: Example matrices $\mathcal{M}$ and $\mathcal{D}$ for the phrases 'the TITLE is a very deadly TARGET' and 'a TITLE is a kind of TARGET'.

$\mathcal{M} =$

0	1	2	3	4	5	6	7
1	1	2	3	3	4	5	6
2	2	1	2	3	4	5	6
3	3	2	1	2	3	4	5
4	4	3	2	1	2	3	4
5	5	4	3	2	2	3	4
6	6	5	4	3	3	3	4
7	7	6	5	4	4	4	3

$\mathcal{D} =$

E	I	I	I	I	I	I	I
D	U	I	I	E	I	I	I
D	D	E	I	I	I	I	I
D	D	D	E	I	I	I	I
D	D	D	D	E	I	I	I
D	D	D	D	D	U	I	I
D	D	D	D	D	D	U	I
D	D	D	D	D	D	D	E

[20] For this purpose the algorithm was introduced as using a full matrix with space requirement $O(m \cdot n)$. Notice that for calculating the minimal edit distance it is sufficient to consider the previous column and row in each step only. This requires a smaller space of only O(max(m,n)).

Afterward the matrix $\mathcal{D}$ for two pattern $A = A[1] \cdots A[m]$ and $B = B[1] \cdots B[n]$ is used to derive a generalized pattern G in the following way.

- initialize $i = m, j = n$ and G with the empty string
- while $i \neq 0$ and $j \neq 0$:
  - if $\mathcal{D}[i,j] =$ `E` : set$G = A[i]G$. Set $i = i - 1$ and $j = j - 1$
  - if $\mathcal{D}[i,j] =$ `U` : set$G = A[i]\,|\,B[i]G$. Set $i = i - 1$ and $j = j - 1$
  - if $\mathcal{D}[i,j] =$ `I` : set $G = *G$. Set $i = i - 1$
  - if $\mathcal{D}[i,j] =$ `D` : set $G = *G$. Set $j = j - 1$
- return G

The symbol | allows either the single word before or after it, while * allows any sequence of words. Notice how this differs from the usage of this symbols in regular expressions.
For the examples from table 5.1, the algorithm yields the generalized pattern 'the|a TITLE is a very|kind deadly|of TARGET'.

Constraints on abstraction by POS tags

This generalized example pattern is not exactly what one would expect intuitively. One reason for this is that the algorithm allows arbitrary disjunction between words, independent of their POS. To solve this problem [Ruiz-Casado et al., 2005b] allow a substitution operation for two words A[i] and B[j] only if both are of the same POS. Otherwise the cheapest insertion or deletion operation is chosen. On the other hand the additional costs for substitutions were dropped. This way the substitution operation is preferred for words with the same POS. Phrases with the same POS sequence have a minimal edit distance of 0.
For illustration the phrases from table 5.1 were POS tagged as 'the/DT TITLE/NN is/VBZ a/DT very/RB deadly/JJ TARGET/NN' and 'a/DT TITLE/NN is/VBZ a/DT kind/NN of/IN TARGET/NN'. Table 9 shows the matrices $\mathcal{M}$ and $\mathcal{D}$. Using the same generalization algorithm from the matrix $\mathcal{D}$ as above, the result is 'a/DT|the/DT TITLE/NN is/VBZ a/DT * TARGET/NN'.

Figure 9: Example matrices $\mathcal{M}$ and $\mathcal{D}$ for the POS constrained algorithm with the tagged phrases 'the/DT TITLE/NN is/VBZ a/DT very/RB deadly/JJ TARGET/NN' and 'a/DT TITLE/NN is/VBZ a/DT kind/NN of/IN TARGET/NN'.

$\mathcal{M} =$

0	1	2	3	4	5	6	7
1	0	1	2	3	4	5	6
2	1	0	1	2	3	4	5
3	2	1	0	1	2	3	4
4	3	2	1	0	1	2	3
5	4	3	2	1	1	2	2
6	5	4	3	2	2	2	3
7	6	5	4	3	3	3	2

$\mathcal{D} =$

E	I	I	I	I	I	I	I
D	U	I	I	I	I	I	I
D	D	E	I	I	I	I	I
D	D	D	E	I	I	I	I
D	D	D	D	E	I	I	I
D	D	D	D	D	I	I	U
D	D	D	D	D	D	I	D
D	D	D	D	D	D	D	E

Deriving a set of abstracted pattern

The following algorithm is used to derive a small set of generalized patterns from all the patterns extracted from hyperlink contexts.

- Start from a set of phrases P.
- Calculate the minimal edit distance for all possible pairs pattern
- While there are at least two phrases and all distances $\leq \Phi$:
  - Generalize the two most similar pattern p_1 and p_2 to p_{new}
  - Remove p_1 and p_2 from P and add p_{new} to P
  - Recalculate minimal edit distance for all $p \in (P \setminus p_1)$ and p_1

In their experiments [Ruiz-Casado et al., 2005b] set the threshold Φ to 5.

5.1.1 Results

My first implementation of this algorithms reproduced the simple examples given in the paper, where always two concrete patterns are generalized. Unfortunately, the authors do not describe in detail, how their algorithm generalizes already abstracted patterns.
The problem with further generalizing abstract patterns is how to handle the $*$ wildcard. One manifest possibility would be to allow the disjunction of $*$ and any other token, because the wildcard can stand for any sequence of tokens, already. The disjunction would then again yield $*$. However, this alternative yields patterns with a lot of wildcards and few lexical tokens (and

usually even generalizes to the most general pattern which consists of a single *, after a few steps.
The other possibility would be to disallow the disjunction of * with other tokens and instead force an insertion or deletion operation, which would possibly result in another *.
Finally I have do draw the conclusion that the algorithm, as presented in [Ruiz-Casado et al., 2005b], does not converge to a useful abstracted set of pattern, though the general approach to pattern generalization is interesting and works at least for simple examples.

5.2 Supervised machine learning approach

The second type of approach is based on supervised machine learning techniques. A target phrase of a link and it's context is represented as a vector of features. Values for a number of training instances are collected and used to train a machine learning classifier. Then the classifier decides whether a given relation holds between a article and link target.
To use supervised machine learning techniques, we first need to represent link instances as vectors of features. For my approach I used a selection of handpicked features from two different domains.
First I make use of a set of 'lexical features', based on the standard bag of words and bag of POS tag approach. This features can be generalized to bags of word- or POS-n-grams which can also capture part of the structure of a surrounding phrase for each link. Notice that, since we do only extract lexical features from the concrete pattern without additional information any classifier learned from this features can be seen as an abstraction of patterns, too.
Second I want to make use Wikipedia's category system.
In the following I will describe how such features can be extracted for link instances. Afterward experiments will investigate which combinations of the proposed features work best with which classifiers.

Lexical features

A first intuition is that certain words in context may give information about the type of relation of a link. For example words like 'is' 'kind' 'type' 'variant' may be a hint for hypernymy. While 'part', 'belongs' may be an indicator for meronymy.
One obvious disadvantage of the basic bag of words approach is that it does not account for the occurrence of indicator words in a specific position. For example the word 'is' might be a good indicator for hypernymy if it appears between 'TITLE' and 'TARGET', but if it appears before 'TITLE' or behind 'TARGET' it is of no use. Therefore we consider words which appear before the link and, if 'TITLE' is in the sentence, behind 'TITLE' only.

Feature selection

Since there are 52892 different lemmas (without proper names) even in the simple English Wikipedia, it would be inconvenient to store occurrence frequencies for all of them. In general it's a good idea to keep dimensionality low, to avoid what is known as the 'curse of dimensionality', viz. the exponential growth of space and time requirements with higher dimensional spaces. We therefore need to restrict ourselves to consider frequencies for words that are most meaningful for detecting links with a certain semantic relation.
The type of feature selection I have adopted is a simple maximum relevance approach. The mutual information between each feature and the target classification was calculated. The classification was either the binary classification

$$\text{hypernymy (class 1) vs. hyponymy} \cup \text{none (class 0)}$$

or

$$\text{hyponymy (class 1) vs. hypernymy} \cup \text{none (class 0).}$$

Were 'none' were those links, for which link source and target had a WordNet mapping, but no relation was found. Then the n most informative features were used to train the classifier.
Appendix D shows the 20 most informative lexical features (both lemma n-grams and POS-n-grams for different n) with respect to each class. POS unigrams were considered too uninformative in the first place and left out.

Experiments with lexical features

The main experiments were done with support vector machines ([Cortes and Vapnik, 1995]) using the *libSVM* implementation by [Chang and Lin, 2001].
The classfier reacted extremely sensible to skewed class distributions (vastly different class sizes). For the hypernymy relation only about 10% of the instances were positive. Just classifying everything as negative therefore already yields an accuracy of $> 90\%$. During training the support vector machine tries to minimize the classification error. Since almost any attempt to improve upon the 90% baseline fails, the resulting model always produces a single class classification. The recall, in this case becomes 0. To solve this problem, only about as many negative instances as positive instances were used for training and evaluation.
As an additional feature the length of each phrase was always included for each instance.
The features were scaled to the interval $[-1 : 1]$, to avoid too strong influence of features with large variance on the training process.
Experimental results were collected in 5-fold crossvalidation with random sampling. This means that the available link instances were randomly divided into 5 partition. In each fold the classifier

was trained on data from 4 of the partitions and evaluatated on the remaining one. The results from each fold were averaged.
For comparision the same experiments were repeated with a C4.5 decision tree learner [Quinlan, 1993] as implemented in the *WEKA* machine learning workbench ([Holmes et al., 1994]) and with the same preprocessing as above.

Results

Appendix E.1 shows results for experiments on the simple Wikipedia training data for different sets of lexical features.
For experiments with SVMs a radial basis function kernel was used. The kernel parameters were empirically set to $\gamma = 0.8$ and $C = 1$. Changing the parameters had only non-significant influence on the results.
The results for hypernym classification are encouraging. The highest F-score (75.97, Prec. 76%, Rec. 75.15%) was achieved with lemma uni-grams. With longer n-grams precision increased slightly but precision decreased. The reason is that bi- or tri-grams are on the one hand more specific but on the other hand they appear less frequently. Therefore the training algorithm suffers from data sparseness problems, because some informative n-grams aren't seen in the training data at all.
POS tri-grams achieved the best precision (80.49%), but the recall was as low as 67.35%, yielding an F-Score of 73.29. POS bi-grams (F-score 72.82, Rec. 69.48%, Prec. 76.50%) performed about as good as lemma tri-grams.
As for the combination of word and POS n-grams the combination of lemma uni-grams and POS bi- or tri-grams worked well. Compared to the performace for lemma uni-grams alone, the precision was better, but the recall decreased.
In comparison to SVMs the C4.5 decision tree classifier performed slightly worse in terms of F-score, because of a low recall, but achieved considerably higher precisions. As is the case with SVMs, the top precision with decision trees was achieved with POS tri-grams (87.54%, Rec. 59.73%, F-Score 73.64). In general with decision trees the same effects are visible as with SVMs.

The accuracy achieved for hyponym classification with SVMs is significantly lower. Phrases in the context of hyponym and hypernym links often contain the same words, but in a different order, therefore many hypernyms are wrongly classified as hyponyms. Furthermore phrases in the context of hyponyms tend to be very short, as they often appear in enumerations and lists. Because of this, with lemma tri-grams the precision is very high, but the recall is low. Surprisingly with POS tri-grams, the effect is the other way around.

Hyponym classification using decision trees achieved significantly worse recall than using SVMs and increase the precision only slightly with some feature combinations.

Features extracted from Wikipedia's category system

Another interesting source of information is Wikipedia's category hierarchy. As we have discussed in section 2.4, the category system does not reflect ontological relations, but is primarily intended to be a navigation aid. Nevertheless the category structure might be useful as a heuristic for classification.
The following method is applied to extract category based features form a link instance.

- Retrieve all transitive superordinate categories (to a level of N) for an article a and link target t from the database. Ignore common 'administrative' categories[21].
- Check if a least common superordinate category S exists. Notice that S is a category, while t and a are articles.
  - Let $dist(x, Y)$, be the smallest distance in the superordinate graph from an article x to a category Y. Use $d = dist(a, S) + dist(t, S)$ as a feature.
  - If $dist(t, S) > dist(a, S)$, set a binary feature $higher = 1$, else if $dist(t, S) < dist(a, S)$ set $higher = -1$, else set $higher = 0$
  - If $dist(t, S) = 1$, set $direct_1 = 1$ else set $direct_1 = -1$
  - If $dist(a, S) = 1$, set $direct_2 = 1$ else set $direct_2 = -1$
- Otherwise set $d = N$, $direct_1 = 0$, $direct_2 = 0$, $higher = 0$.

By this a feature vector $f = (d, higher, direct1, direct2)$ is created.
The first element *distance* shows how far two articles in the category network are away from each other. This can be interpreted as a heuristic for relatedness of the concepts, described by this articles. However, some categories serve only administrative purpose and e.g the distance between two articles might be small, because they are both too short and therefore belong to the category *stubs*.
The *higher* attribute indicates which of the articles is further away from the common super concept.
The elements $direct_1$ and $direct_2$ indicate wether the common superconcept S is a direct superconcept of a or t resp. In connection with the *higher* attribute, this can serve as an indicator

[21] Simple Wikipdia: 'Basic English 850 words','Stubs','Wikipedia','Project','Articles_that_need_to_be_wikified','Basic_English'

on whether the concept described by one article dominates the concept described by the other one.
Like the lexical feature, category features were scaled to the unit interval $[-1, 1]$

Results

Results for category features are given in Appendix E.2. Obviously for hypernym classification the plain category features don't even work as well as the worst lexical features. Furthermore using the category features in addition to lexical features decreases rather decreases the results than improving them.
As with lexical features, compared to SVMs decision trees achieved better precision, but lower recall.
For hyponym classification SVMs worked again much better than decision trees.

6 Epilogue

6.1 Conclusion

This thesis has shown that supervised machine learning techniques can be used to classify semantic relations of Wikipedia hyperlinks. Simple lexical features work well for the detection of hypernyms, but performs significantly worse for hyponyms. This can be contributed to the fact that they are often used in lists or other positions, where not enough context can be extracted. Features generated from the category system are shown to work in principle but always perform worse than lexical features. This underlines the fact that the category system is not intended to mirror a taxonomic hierarchy.
From a machine learning perspective the kind of vector representation that is used poses a challenge, as the feature representation is very sparse. My experiments have shown that both decision trees and support vector machines can cope with such data, with only a slight improvement in the preformance of support vector machines. Depending on the application, however, one might prefer decision trees which achieve higher precision, but traded it for a lower recall.

The pattern learning based approach to relation identification for Wikipedia links, as presented by [Ruiz-Casado et al., 2005b], could not be reevaluated, because important details are missing from the description of their algorithm.

The mapping from Wikipedia articles to WordNet synsets works well. In contrast to previous literature, I have argued that the task has to be divided into the plain disambiguation task, where one has to decide which synset out of a number of possible synsets corresponds best to the article, and a sifting task, where one has to decide whether any possible WordNet synset corresponds to an artice at all. Two types approaches to this problem have been discussed, viz. approches based on the Lesk algorithm for word sense disambiguation and approaches based on the vector space model (VSM). Remarkably a very simple and fast variant of the Lesk algorithm performs best on the plain disambiguation task. When the sifting task is added, VSM based approachs perform slightly better than Lesk based algorithms.
The mapping could be usable by itself. Many techniques that rely on WordNet as a source of semantic information (e.g identification of lexical chains) could profit from additional text, associated with each gloss, which can now be taken from Wikipedia entries.

A further valuable outcome of this thesis are the textcorpora, that habe been extracted from Wikipedia dumps. Their main advantages are the size of the English Wikipedia corpus and the fact that they contain only valid XML according to a relatively simple document type definition.

Because of this it should be easy to adapt the corpora to other applications. Especially the corpus extracted from the simple English Wikipedia proved beneficial as a small but complete data set for fast development and testing of algorithms.
All software that was written in context of this thesis will be made freely available.

6.2 Outlook

These results can possibly be improved by varying the kind of data and machine learning classifiers. For example I did not try other SVM kernel functions but RBF kernels. It would be interesting to further investigate how other kernel types or in general other classifiers can cope with the special kind of data that is extracted.
Extracting features based on a syntactic alanysis could also achieve some improvements, since a major disadvantage of lexical features is that they cannot represent the structure of phrases that embed a link
Also simple heuristics from the article structure could be incorporated, e.g whether a link appears in a list, or in which paragraph it was found.
If the pattern learning approach by [Ruiz-Casado et al., 2005b] could be reemplemented, another interesting possibility would be to use their lexico-syntactic patterns as features in my machine learning setting.
The techniques developed in this thesis could be applied to other relations. A set of well-trained classifiers could in fact be used to create a complete ontology from Wikipedia links.
Wikipedia articles often provide rich numerical data on a topic, e.g the number of inhabitants of a city. [Völkel et al., 2006] suggest an extension of the MediaWiki syntax to allow users to explicitly specify concept attributes for an article in a structured way. A more advanced version of the system, described in this thesis, may automatically extract numerical data from articles and incorporate it in the ontology.
The techniques that were developed in this thesis are independent of a specific language. Since Wikipedia is available in a variety of different language versions one could employ our application to the construction of multi-lingual thesauri that would be useful for machine translation. Fortunately many articles explicitly link to their correspondent in other languages. On the other hand the available data from different language versions could be useful to gather even more training data, itself.
The results could also be interesting, when combined with the semantic Wikipedia approach by [Völkel et al., 2006], where users can manually specify link semantics in the markup language, in a semi automatic way. The classifier, we have trained could be used to make suggestions for correct link semantics to an editor. The editors decision represents an additional training instance for an improved classifier in an adaptive learning setting.

Acknowledgements

Very special thanks to my friends and family who tolerated my moods, while working on this thesis and provided great support. Thanks to my supervisor Stefan Evert for helpful suggestions during our admittedly rare meetings. I'm indebted to IKW's system administrators for setting up a local SQL database for Wikipedia meta data.

References

[Banerjee and Pedersen, 2002] Banerjee, S. and Pedersen, T. (2002). An adapted lesk algorithm for word sense disambiguation using wordnet. *Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics*, pages 136–145.

[Bauer et al., 2007] Bauer, D., Degen, J., Deng, X., Evert, S., Herger, P., Gasthaus, J., Giesbrecht, E., Jansen, L., Kalina, C., Krueger, T., Maertin, R., Schmidt, M., Scholler, S., Steger, J., and Stemle, E. (2007). Filtering the internet by automatic subtree classification, osnabrueck. In *Proceedings of the Third Web as Corpus Workshop (WAC3), CLEANEVAL Session, Louvain-la-Neuve, Belgium*.

[Bechhofer et al., 2004] Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D. L., Pate-Schneider, P. F., and Stein, L. A. (2004). Owl web ontology language reference. W3C Recommendation, http://www.w3.org/TR/owl-ref/, retrieved 2007-10-16.

[Berners-Lee et al., 2001] Berners-Lee, T., Hendler, J., and Lassila, O. (2001). The semantic Web. *Scientific American*, 284(5):28–37.

[Brownell, 2002] Brownell, D. (2002). *SAX2*. O'Reilly.

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

[Burnard and Sperberg-McQueen, 1995] Burnard, L. and Sperberg-McQueen, C. (1995). Tei lite: An introduction to text encoding for interchange. http://www.tei-c.org/Lite/teiu5_en.pdf, retrieved 2007-10-1.

[Campanini et al., 2004] Campanini, S., Castagna, P., and Tazzoli, R. (2004). Platypus Wiki: a Semantic Wiki Wiki Web. *Proceedings of the 1st Italian Semantic Web Workshop Semantic Web Applications and Perspectives (SWAP)*, pages 1–6.

[Chang and Lin, 2001] Chang, C.-C. and Lin, C.-J. (2001). *LIBSVM: a library for support vector machines*. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

[Cruse, 1986] Cruse, D. (1986). *Lexical semantics*. Cambridge University Press.

[Denoyer and Gallinari, 2006] Denoyer, L. and Gallinari, P. (2006). The Wikipedia XML Corpus. *SIGIR Forum*.

[Ebersbach and Glaser, 2005] Ebersbach, A. and Glaser, M. (2005). Wiki. *Informatik-Spektrum*, 28(2):131–135.

[Ellson et al., 2003] Ellson, J., Gansner, E., Koutsofios, E., North, S., and Woodhull, G. (2003). Graphviz and Dynagraph - Static and dynamic graph drawing tools. *Graph Drawing Software*, pages 127–148.

[Faure and Nedellec, 1998] Faure, D. and Nedellec, C. (1998). A corpus-based conceptual clustering method for verb frames and ontology acquisition. *LREC workshop on adapting lexical and corpus resources to sublanguages and applications*, pages 707–728.

[Fellbaum et al., 1998] Fellbaum, C., Miller, G., Miller, K., and Tengi, R. (1998). *WordNet: an electronic lexical database*. MIT Press.

[Frege, 1892] Frege, G. (1892). Uber Sinn und Bedeutung. *Zeitschrift fur Philosophie und philosophische Kritik*, 100:25–50.

[Giles, 2005] Giles, J. (2005). Internet encyclopaedias go head to head. *Nature*, (438):900–901.

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

[Holmes et al., 1994] Holmes, G., Donkin, A., and Witten, I. (1994). WEKA: A Machine Learning Workbench. *Proc Second Australia and New Zealand Conference on Intelligent Information Systems*.

[Lesk, 1986] Lesk, M. (1986). Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *SIGDOC '86: Proceedings of the 5th annual international conference on Systems documentation*, pages 24–26, New York, NY, USA. ACM Press.

[Leuf and Cunningham, 2001] Leuf, B. and Cunningham, W. (2001). *The Wiki way: quick collaboration on the Web*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA.

[Levenshtein, 1966] Levenshtein, V. (1966). Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10:707.

[Manning and Schütze, 1999] Manning, C. and Schütze, H. (1999). *Foundations of statistical natural language processing*. MIT Press.

[Quinlan, 1993] Quinlan, J. (1993). *C4. 5: Programs for Machine Learning*. Morgan Kaufmann.

[Ruiz-Casado et al., 2005a] Ruiz-Casado, M., Alfonseca, E., and Castells, P. (2005a). Automatic assignment of wikipedia encyclopedic entries to wordnet synsets. *Advances in Web Intelligence*, 3528:380–386.

[Ruiz-Casado et al., 2005b] Ruiz-Casado, M., Alfonseca, E., and Castells, P. (2005b). Automatic extraction of semantic relationships for WordNet by means of pattern learning from wikipedia. *Natural Language Processing and Information Systems*, 3513/2005:67–79.

[Salton and Buckley, 1988] Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing and Management: an International Journal*, 24(5):513–523.

[Schenkel et al., 2007] Schenkel, R., Suchanek, F. M., and Kasneci, G. (2007). YAWN: A semantically annotated Wikipedia XML corpus. In Kemper, A., Schöning, H., Rose, T., Jarke, M., Seidl, T., Quix, C., and Brochhaus, C., editors, *12. GI-Fachtagung für Datenbanksysteme in Business, Technologie und Web (BTW 2007)*, volume 103 of *Lecture Notes in Informatics*, pages 277–291, Aachen, Germany. Gesellschaft fÃ$\frac{1}{4}$r Informatik.

[Schmid, 1994] Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. *Proceedings of the International Conference on New Methods in Language Processing*, 12.

[Sowa, 2000] Sowa, J. (2000). *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. MIT Press.

[Völkel et al., 2005] Völkel, M., Krötzsch, M., and Vrandecic, D. (2005). Wikipedia and the Semantic Web-The Missing Links. *Proceedings of Wikimania*.

[Völkel et al., 2006] Völkel, M., Krötzsch, M., Vrandecic, D., Haller, H., and Studer, R. (2006). Semantic Wikipedia. *Proceedings of the 15th international conference on World Wide Web*, pages 585–594.

A Corpus DTD

```
<!ELEMENT wikiCorpus (article+)>

<!ELEMENT article (category*, text)>
<!ATTLIST article
            id CDATA #REQUIRED
            title CDATA #REQUIRED
            wordnet CDATA #IMPLIED>

<!ELEMENT text (p*)>

<!ELEMENT category EMPTY>
<!ATTLIST category cat CDATA #REQUIRED>

<!ELEMENT p (#PCDATA|head|hi|ref|xref|ptr|xptr|list)*>
<!ELEMENT head (#PCDATA|ref|hi|xref|ptr|xptr)*>
<!ELEMENT hi (#PCDATA|hi|ref|xref|ptr|xptr)*>

<!ELEMENT ref (#PCDATA|hi)*>
<!ATTLIST ref
           target CDATA #REQUIRED
           type (article|interwiki|external) "article"
           rel CDATA #IMPLIED>

<!ELEMENT xref (#PCDATA|hi)*>
<!ATTLIST xref
           target CDATA #REQUIRED
           doc CDATA #IMPLIED
           type (interwiki|external) "interwiki">
<!ELEMENT xptr EMPTY>
<!ATTLIST xptr
           target CDATA #REQUIRED
           doc CDATA #IMPLIED
           type (interwiki|external) "external">

<!ELEMENT list (item*)>
<!ATTLIST list
           type (ordered|bulleted|gloss) "bulleted">

<!ELEMENT item (#PCDATA|ref|hi|xref|ptr|xptr|list)*>
```

B Word lists

B.1 Tag set

The modified Penn Treebank tag set used by the TreeTagger.

CC	Coordinating conjunction	CD	Cardinal number
DT	Determiner	EX	Existential there
FW	Foreign word	IN	Preposition or subordinating conjunction
JJ	Adjective	JJR	Adjective, comparative
JJS	Adjective, superlative	LS	List item marker
MD	Modal	NN	Noun, singular or mass
NNS	Noun, plural	NP	Proper noun, singular
NPS	Proper noun, plural	PDT	Predeterminer
POS	Possessive ending	PP	Personal pronoun
PP$	Possessive pronoun	RB	Adverb
RBR	Adverb, comparative	RBS	Adverb, superlative
RP	Particle	SYM	Symbol
TO	to	UH	Interjection
VB	to be, base form, ‘be’	VBD	to be, past tense, ‘was’
VBG	to be, gerund or present participle, ‘being’	VBN	past participle
VBP	non-3rd person singular present	VBZ	3rd person singular present
VH	to have, base form, ’have’	VHD	to have, past tense, ’had’
VHG	to have, gerund or present participle	VHN	to have, past participle
VHP	to have, non-3rd person singular present	VHZ	to have, 3rd person singular present
VV	Verb, base form	VVD	Verb, past tense
VVG	Verb, gerund or present participle	VVN	Verb, past participle
VVP	Verb, non-3rd person singular present	VVZ	Verb, 3rd person singular present
WDT	Wh-determiner	WP	Wh-pronoun
WP$	Possessive wh-pronoun	WRB	Wh-adverb

B.2 Stopwords

i	me	my	myself	we	us	our	ours
ourselves	you	your	yours	yourself	yourselves	he	him
his	himself	she	her	hers	herself	it	its
itself	they	them	their	theirs	themselves	what	which
who	whom	this	that	these	those	am	is
are	was	were	be	been	being	have	has
had	having	do	does	did	doing	would	shall
should	could	must	ought	i'm	you're	he's	she's
it's	we're	they're	i've	you've	we've	they've	i'd
you'd	he'd	she'd	we'd	they'd	i'll	you'll	he'll
she'll	we'll	they'll	isn't	aren't	wasn't	weren't	hasn't
haven't	hadn't	doesn't	don't	didn't	won't	wouldn't	shan't
shouldn't	can't	cannot	couldn't	mustn't	let's	that's	who's
what's	here's	there's	when's	where's	why's	how's	daren't
needn't	oughtn't	mightn't	a	an	the	and	but
if	or	either	because	as	until	while	of
at	by	for	with	about	against	between	into
through	during	before	after	above	below	to	from
up	down	in	out	on	off	over	under
again	further	then	once	here	there	when	where
why	how	all	any	both	each	few	more
most	other	some	such	no	nor	not	only
own	same	so	than	too	very	one	every
least	less	many	now	ever	never	say	says
said	also	get	go	goes	just	made	make
put	see	seen	whether	like	well	back	even
still	way	take	since	another	however	two	three
four	five	first	second	new	old	high	long
name	use	hold	@card@	near	close	set	ask
become	begin	break	bring	come	do	find	forget
get	give	go	have	hold	keep	know	leave
lose	make	may	mean	meet	must	put	should
spend	start	stop	live	happen	suggest	want	will
worry	side	there	's	@ord@	far	1	2
3	4	5	6	7	8	9	10
u							

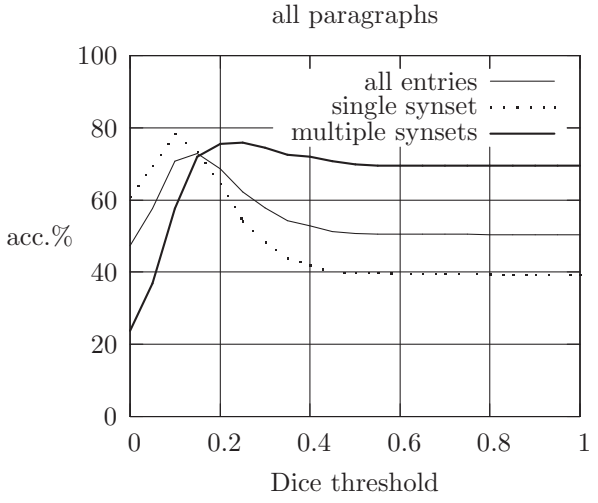
C Results for disambiguation algorithms

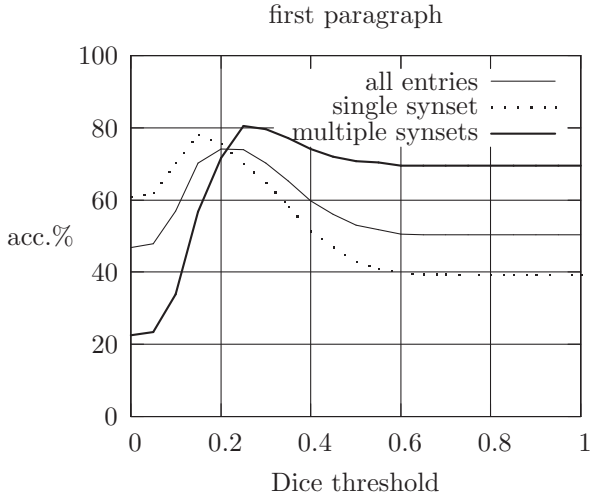
The following tables and diagrams show the influence of theshold parameters for the Wikipedia entry disambiguation algorithms from chapter 4. The threshold enables the algorithm to identify Wikipedia articles that cannot be mapped to any of the possible WordNet glosses.
The evaluated was done on the manual gold standard from section 4.1.
Accuracies are givenfor all articles, for articles which's title appeared only in a single WordNet synset and articles which's title appeared in more than one synset. Experiments were done with both the first paragraph only and the whole Wikipedia entry.

C.1 Results for the simple Lesk algorithm

all paragraphs					
Dice	all	multi	prec	rec	F
0.0	47.26	23.78	???	00.00	???
0.05	57.56	36.89	100.00	18.86	31.73
0.1	70.66	57.62	97.44	50.00	66.08
0.15	**72.79**	71.95	92.22	72.81	81.37
0.2	68.53	75.61	78.67	93.86	**85.35**
0.25	62.15	**75.91**	75.43	96.93	84.84
0.3	57.78	74.39	72.67	99.12	83.84
0.35	54.31	72.56	71.25	100.00	82.60
0.4	52.86	71.95	70.37	100.00	82.16
0.45	51.18	70.73	69.51	100.00	82.01
0.5	50.73	69.82	69.51	100.00	82.01
0.55	50.62	69.51	69.51	100.00	82.01
0.6	50.50	69.51	69.51	100.00	82.01

first paragraph					
Dice	all	multi	prec	rec	F
0.0	46.81	22.56	???	00.00	???
0.05	47.82	23.48	100.00	01.32	02.59
0.1	56.89	33.84	97.37	16.23	27.82
0.15	70.21	56.70	95.04	50.44	82.55
0.2	**74.13**	71.65	93.85	73.68	**89.85**
0.25	73.91	**80.49**	88.51	91.23	88.75
0.3	70.21	79.57	81.85	96.93	86.99
0.35	65.17	77.13	78.05	98.25	84.80
0.4	59.69	74.09	74.10	99.12	83.53
0.45	55.99	71.95	72.20	99.12	82.76
0.5	52.97	70.73	70.58	100.00	82.26
0.55	51.74	70.43	70.37	100.00	82.01
0.6	50.62	69.51	69.51	100.00	82.01

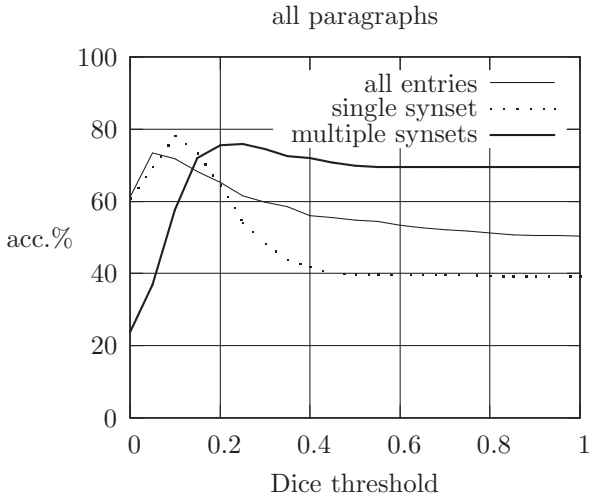


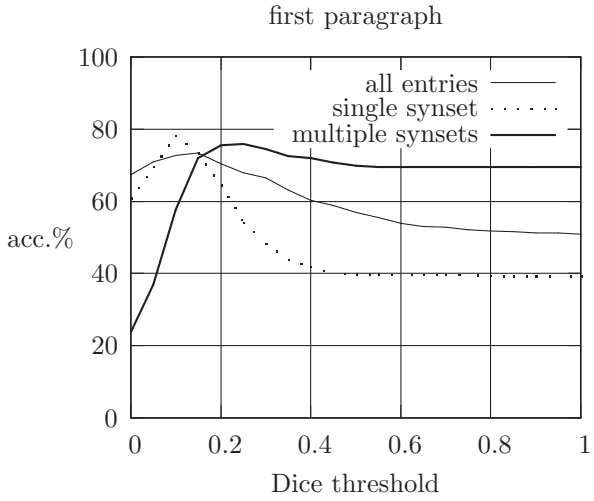


C.2 Results for the 'extended' Lesk algorithm

all paragraphs					
Dice	all	multi	prec	rec	F
0.0	61.25	34.76	???	00.00	???
0.05	**73.35**	58.23	96.24	56.14	70.91
0.1	71.78	65.55	87.56	74.12	80.29
0.15	68.31	70.12	82.70	85.96	84.30
0.2	65.29	71.04	79.92	90.79	85.01
0.25	61.48	71.34	78.23	92.98	84.97
0.3	59.69	72.26	77.30	95.61	85.49
0.35	58.45	**73.17**	77.35	97.37	**86.21**
0.4	56.10	72.87	75.59	97.81	85.27
0.45	55.43	71.65	73.93	98.25	84.37
0.5	54.87	71.95	74.01	98.68	84.59
0.55	54.42	72.26	72.99	99.56	84.23
0.6	53.30	71.34	71.84	99.56	83.46
0.65	52.74	71.04	71.38	99.56	83.15

first paragraph					
Dice	all	multi	prec	rec	F
0.0	67.41	46.95	???	00.00	???
0.05	71.00	53.69	94.12	49.12	64.55
0.1	72.68	62.50	89.76	64.35	75.63
0.15	**73.46**	71.04	86.38	80.70	83.45
0.2	70.44	72.87	83.90	86.84	85.34
0.25	67.97	73.46	80.62	91.23	85.60
0.3	66.41	**75.00**	79.41	94.74	**86.40**
0.35	63.16	73.78	76.84	96.05	85.38
0.4	60.25	72.56	74.75	97.37	84.57
0.45	58.90	71.95	73.93	98.25	84.37
0.5	57.00	71.04	72.67	99.12	83.86
0.55	55.43	70.73	71.84	99.56	83.46
0.6	53.86	70.12	70.81	100.00	82.91
0.65	52.97	69.82	70.81	100.00	82.91



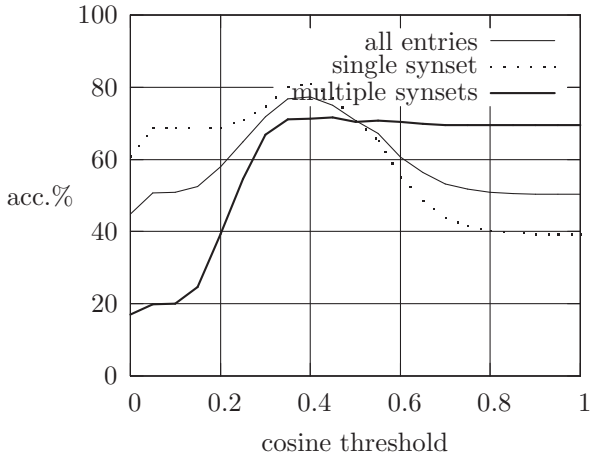


C.3 Results for the VSM approach - cosine measure

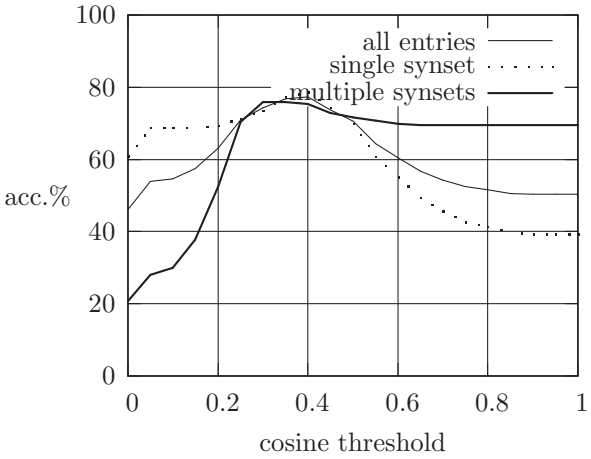
all paragraphs					
cosine	all	multi	prec	rec	F
0.0	44.79	17.07	???	00.00	???
0.05	50.73	19.82	100.00	03.95	07.59
0.1	50.84	20.12	100.00	04.39	08.40
0.15	52.52	24.70	100.00	10.96	19.76
0.2	58.01	39.33	94.87	32.46	48.37
0.25	64.95	54.57	90.58	54.83	68.31
0.3	71.78	66.77	87.69	75.00	80.85
0.35	76.82	71.04	81.15	86.84	83.90
0.4	**77.38**	71.34	77.78	92.11	**84.34**
0.45	75.03	**71.65**	74.15	95.61	83.52
0.5	70.66	70.43	72.55	97.37	83.15
0.55	67.19	70.73	71.38	99.57	83.15
0.6	60.69	70.43	70.59	100.00	82.76
0.65	56.33	69.82	69.94	100.00	82.31
0.7	53.19	69.51	69.72	100.00	82.16
0.75	51.74	69.51	69.51	100.00	82.01
0.8	50.95	69.51	69.51	100.00	82.01
0.85	50.62	69.51	69.51	100.00	82.01

first paragraph					
cosine	all	multi	prec	rec	F
0.0	46.14	20.73	???	00.00	???
0.05	53.86	28.05	100.00	10.53	19.05
0.1	54.54	29.88	96.77	13.16	23.17
0.15	57.45	37.80	96.00	25.00	39.58
0.2	63.16	52.44	95.54	46.93	62.94
0.25	71.00	70.43	92.35	74.12	82.24
0.3	74.36	**75.91**	87.95	86.40	87.17
0.35	76.71	**75.91**	83.87	91.23	**87.39**
0.4	**77.38**	73.22	78.14	95.61	86.00
0.45	73.80	72.87	73.84	97.81	84.15
0.5	70.55	71.65	72.35	98.68	83.49
0.55	64.39	70.73	71.16	99.56	83.00
0.6	60.47	69.82	70.15	100.00	82.46
0.65	56.66	69.51	69.51	100.00	82.01
0.7	54.31	69.51	69.51	100.00	82.01
0.75	52.41	69.51	69.51	100.00	82.01
0.8	51.62	69.51	69.51	100.00	82.01
0.85	50.62	69.51	69.51	100.00	82.01

all paragraphs



first paragraph

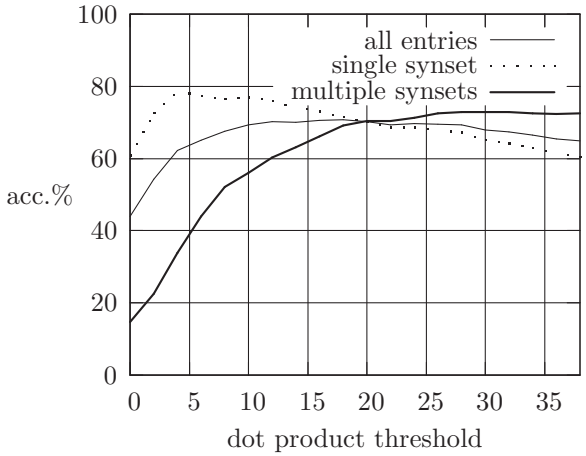


C.4 Results for the VSM approach - dot product measure

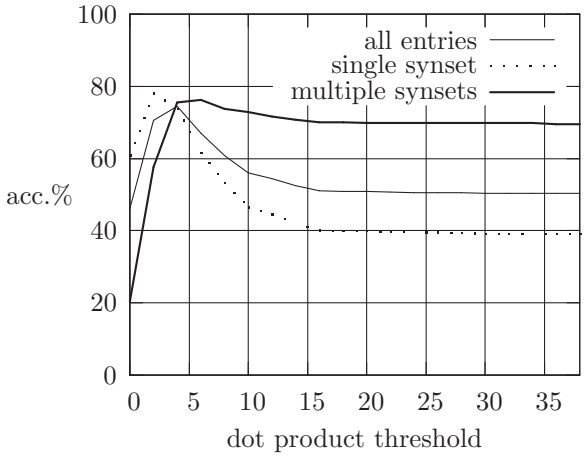
all paragraphs					
dot p.	all	multi	prec	rec	F
0	43.90	14.63	???	00.00	???
2	54.31	22.56	100.00	11.40	20.47
4	62.26	33.84	98.46	28.07	43.69
6	65.06	43.90	98.28	42.54	59.51
8	67.53	52.13	99.28	60.09	70.25
10	69.32	56.10	98.06	66.67	74.86
12	70.21	60.37	95.86	71.05	79.37
14	70.10	63.11	96.09	75.44	81.61
16	70.55	66.16	95.81	80.26	84.52
18	**70.66**	69.21	94.00	82.46	87.35
20	70.21	70.43	92.72	83.77	87.85
22	69.32	70.43	91.12	85.53	88.02
24	69.65	71.34	90.54	88.16	88.24
26	69.54	72.56	89.86	89.47	89.33
28	69.32	**72.87**	88.84	90.79	89.67
30	67.97	**72.87**	88.19	91.67	89.80
32	67.41	**72.87**	87.14	92.11	**89.89**
34	66.52	**72.56**	86.77	92.11	89.55

first paragraph					
dot p.	all	multi	prec	rec	F
0	46.02	20.43	???	00.00	???
2	70.55	57.62	96.09	53.95	69.10
4	**74.47**	**75.61**	90.28	85.53	87.84
6	66.97	76.22	83.73	92.54	**87.92**
8	60.81	73.78	77.66	96.05	85.88
10	56.10	72.87	75.16	98.25	85.17
12	54.42	71.65	72.15	100.00	83.82
14	52.41	70.73	71.25	100.00	83.21
16	51.06	70.12	70.58	100.00	82.76
18	50.95	70.12	70.15	100.00	82.46
20	50.84	69.82	69.72	100.00	82.16
22	50.73	69.82	69.72	100.00	82.16
24	50.62	69.82	69.72	100.00	82.16
26	50.62	69.82	69.72	100.00	82.16
28	50.50	69.82	69.72	100.00	82.16
30	50.39	69.82	69.72	100.00	82.16
32	50.39	69.82	69.72	100.00	82.16
34	50.39	69.82	69.72	100.00	82.16

all paragraphs



first paragraph



D Mutual information values for feature selection

This tables present mutual information (MI) values, that were calculated between lexical features (lemma and POS uni-, bi- and trigrams) and the link classification (hypernym/hyponym/none) for all training instances in the simple Wikipedia. The 10 features with highest MI are listed.

D.1 hypernym vs. hyponym ∪ none

lemma	MI (bit)
TITLE	0.028704544629
be	0.0273100280885
in	0.02028646014
a	0.0129416831271
have	0.00779753127626
it	0.00725439918101
type	0.00475132937144
kind	0.00468072441208
,	0.00400391541752
on	0.00367830660826

lemma bigram	MI (bit)
TITLE be	0.0295031813174
be a	0.0220733859884
in TARGET	0.010041677285
in the	0.0060372849807
a type	0.00581190091008
a TARGET	0.00539641180925
a kind	0.00531974224652
type of	0.00520674314711
kind of	0.00485973398144
be the	0.00457928616104

lemma trigram	MI (bit)
TITLE be a	0.0175152822486
be a TARGET	0.0150187175847
be the TARGET	0.0110098185852
TITLE be the	0.00678126607668
type of TARGET	0.00605866967617
a type of	0.00581190091008
be a type	0.00571368742897
be a kind	0.00555476049894
a kind of	0.00531974224652
kind of TARGET	0.00506724285582

pos bigram	MI (bit)
VBZ DT	0.0367616434423
NN VBZ	0.0141831229656
IN DT	0.0112112984019
VVN IN	0.00936073637103
DT NP	0.00673199241123
NP VBZ	0.00658656581614
IN NP	0.00357139616206
VBZ VVN	0.00293995554242
JJ NP	0.0028546210026
PP VHZ	0.00249682366508

pos trigram	**MI (bit)**
VBZ DT NP	0.0361423314343
NN VBZ DT	0.01918437676
NP VBZ DT	0.00992631463726
IN DT NP	0.00852478370992
DT NP VBZ	0.00774508507171
VVN IN NP	0.00408294671074
NN IN DT	0.00398358163142
DT JJ NP	0.0038868872167
NN NNS VBP	0.00359204186431
VVN IN DT	0.0035146434297

D.2 hyponym vs. hypernym ∪ none

lemma	MI (bit)
s	0.000837902413425
the	0.000755774509829
lans	0.000744599504398
hardball	0.000744599504398
distinguish	0.000744599504398
denote	0.000744599504398
another	0.000571022978174
call	0.000557641937711
thought	0.000554744687313
stand	0.000554744687313

pos trigram	MI (bit)
example of	0.00103079024121
of TITLE	0.000931272156794
TITLE s	0.000886146977981
be another	0.000877656756326
be TARGET	0.000856222142642
when use	0.000744599504398
what distinguish	0.000744599504398
to denote	0.000744599504398
TITLE such	0.000744599504398
TITLE hardball	0.000744599504398

pos trigram	MI (bit)
example of TITLE	0.00167314353256
word to describe	0.000744599504398
when use in	0.000744599504398
what distinguish TITLE	0.000744599504398
use to denote	0.000744599504398
type of it	0.000744599504398
to denote a	0.000744599504398
TITLE such as	0.000744599504398
TITLE mean do	0.000744599504398
TITLE hardball ,	0.000744599504398

pos trigram	MI (bit)
NN VBZ	0.00171669730289
DT NP	0.00111898706875
NP NNS	0.00106320655731
CD RBS	0.000744599504398
CD :	0.000744599504398
: #	0.000744599504398
# CD	0.000744599504398
VBP NP	0.000705116031415
NN :	0.000595060096287
NNS IN	0.000541962426169

pos trigram	MI (bit)
NP NNS VBP	0.00141934928365
NN VBZ DT	0.00123727872893
VVP NP NN	0.000744599504398
RB NP MD	0.000744599504398
NP JJ IN	0.000744599504398
NNS NNS VBP	0.000744599504398
NN : #	0.000744599504398
JJ NNS NNS	0.000744599504398
IN NP JJ	0.000744599504398
DT CD RBS	0.000744599504398

E Results for machine learning experiments

E.1 Classification performance with lexical features

For the following experiments the twenty most relevant features of each type were used (see section 5.2). Experiments with feature combinations, shown in the lower part of the tables, used the ten most relevant features of each type.

Hypernym classification

Hypernym classification performance with RBF kernel (γ=0.8, C=1) with different sets of lexical features on simple Wikipedia data.

Feature set	Accuracy	Recall	Precision	F-Score
lemma uni-grams	76.20%	75.15%	76.81%	75.97
lemma bi-grams	75.85%	73.24%	77.14%	75.14
lemma tri-grams	74.17%	68.06%	77.52%	72.48
POS bi-grams	74.06%	69.48%	76.50%	72.82
POS tri-grams	75.34%	67.35%	80.49%	73.29
lemma uni-grams + pos bi-grams	76.90%	72.10%	79.73%	75.72
lemma uni-grams + pos tri-grams	76.32%	71.95%	78.89%	75.26
lemma bi-grams + pos bi-grams	75.34%	69.95%	78.50%	73.98
lemma bi-grams + pos tri-grams	75.69%	69.78%	78.98%	74.09
lemma tri-grams + pos bi-grams	75.50%	69.52%	78.92%	73.92
lemma tri-grams + pos tri-grams	74.72%	65.87%	79.83%	72.18

Hypernym classification performance with C4.5 decision tree with different sets of lexical features on simple Wikipedia data.

Feature set	Accuracy	Recall	Precision	F-Score
lemma uni-grams	75.05%	69.19%	78.34%	73.48
lemma bi-grams	74.94%	63.64%	82.21%	72.37
lemma tri-grams	72.79%	56.14%	84.15%	67.35
POS bi-grams	73.41%	61.68%	80.56%	69.87
POS tri-grams	73.64%	59.73%	87.54%	70.01
lemma uni-grams + pos bi-grams	76.57%	73.37%	78.38%	75.79
lemma uni-grams + pos tri-grams	74.81%	71.28%	76.69%	73.89
lemma bi-grams + pos bi-grams	73.77%	63.12%	80.18%	70.63
lemma bi-grams + pos tri-grams	74.55%	64.16%	80.97%	71.59
lemma tri-grams + pos bi-grams	73.12%	61.55%	84.67%	71.28
lemma tri-grams + pos tri-grams	73.08%	58.09%	82.95%	68.33

Hyponym classification

Hyponymy classification performance with RBF kernel (γ=0.8, C=1) with different sets of lexical features on simple Wikipedia data.

Feature set	Accuracy	Recall	Precision	F-Score
lemma uni-grams	55.73%	80.78%	54.16%	64.84
lemma bi-grams	56.76%	89.93%	54.03%	67.51
lemma tri-grams	49.74%	23.30%	82.39%	36.32
POS bi-grams	57.78%	84.78%	55.20%	66.87
POS tri-grams	55.21%	92.14%	53.01%	67.30
lemma uni-grams + pos bi-grams	58.29%	83.05%	55.35%	66.43
lemma uni-grams + pos tri-grams	54.87%	79.11%	53.43%	64.78
lemma bi-grams + pos bi-grams	57.26%	84.85%	54.62%	66.46
lemma bi-grams + pos tri-grams	55.21%	91.06%	53.20%	67.16
lemma tri-grams + pos bi-grams	59.49%	88.44%	55.92%	68.52
lemma tri-grams + pos tri-grams	45.81%	22.76%	83.18%	35.74

Hyponymy classification performance with C4.5 decision tree and different sets of lexical features with the full English Wikipedia.

Feature set	Accuracy	Recall	Precision	F-Score
lemma uni-grams	55.21%	34.47%	59.06%	43.53
lemma bi-grams	49.57%	27.30%	49.38%	35.15
lemma tri-grams	57.44%	32.42%	65.07%	43.28
POS bi-grams	56.58%	24.57%	68.57%	36.18
POS tri-grams	58.97 %	35.84%	66.88%	46.67
lemma uni-grams + pos bi-grams	60.17%	32.08%	73.44%	44.65
lemma uni-grams + pos tri-grams	55.90%	34.47%	60.48%	43.91
lemma bi-grams + pos bi-grams	55.73%	23.89%	66.04%	35.09
lemma bi-grams + pos tri-grams	57.44 %	34.81%	63.75%	45.03
lemma tri-grams + pos bi-grams	57.78%	23.33%	69.17%	34.89
lemma tri-grams + pos tri-grams	57.44%	32.42%	64.07%	43.05

E.2 Classification performance with category system features

Hypernym classification

Results for hypernym classification with category features alone and in combination with lexical features on the simple Wikipedia. An SVM with RBF kernel, $C = 1$ and $\gamma = 0.8$ was used.

Feature set	Accuracy	Recall	Precision	F-Score
Category only	65.91	64.22	66.40	65.39
Category + 20 best lemmas	76.04	76.03	76.10	76.07
Category + 20 best lemma bigrams	74.29	71.76	75.70	73.68
Category + 20 best lemma trigrams	74.21	66.28	78.67	71.94
Category + 20 best POS bigrams	75.85	76.49	75.49	75.98
Category + 20 best POS trigrams	75.34	74.13	75.96	75.03

Results for hypernym classification with category features alone and in combination with lexical features on the simple Wikipedia. The C4.5 decision tree learner was used.

Feature set	Accuracy	Recall	Precision	F-Score
Category only	59.61	57.11	60.10	58.57
Category + 20 best lemmas	74.75	70.17	77.23	73.53
Category + 20 best lemma bigrams	75.33	65.86	81.24	72.75
Category + 20 best lemma trigrams	72.69	56.40	88.98	69.04
Category + 20 best POS bigrams	73.80	62.60	80.66	70.49
Category + 20 best POS trigrams	73.56	59.53	82.76	69.25

Hyponym classification

Results for hyponym classification with category features alone and in combination with lexical features on the simple Wikipedia. An SVM with RBF kernel, $C = 1$ and $\gamma = 0.8$ was used.

Feature set	Accuracy	Recall	Precision	F-Score
Category only	56.34%	78.11%	54.71%	64.35
Category + 20 best lemmas	55.38%	75.76%	53.61%	62.79
Category + 20 best lemma bigrams	58.63%	87.33%	55.41%	67.81
Category + 20 best lemma trigrams	58.12%	75.97%	55.73%	64.29
Category + 20 best POS bigrams	57.61%	80.51%	55.28%	65.56
Category + 20 best POS trigrams	57.44%	82.47%	54.85%	65.88

Results for hyponym classification with category features alone and in combination with lexical features on the simple Wikipedia. The C4.5 decision tree learner was used.

Feature set	Accuracy	Recall	Precision	F-Score
Category only	57.44	32.42	65.07	43.28
Category + 20 best lemmas	56.92	37.54	61.54	47.40
Category + 20 best lemma bigrams	57.44	33.11	64.67	43.79
Category + 20 best lemma trigrams	57.44	32.42	65.07	43.28
Category + 20 best POS bigrams	54.70	25.60	61.48	36.15
Category + 20 best POS trigrams	58.97	35.84	66.88	46.67

Declaration of Academic Honesty

I hereby confirm that I have written the bachelor's thesis entitled 'Learning the semantics of Wikipedia hyperlinks' on my own and did not make use of any other means or resources than those mentioned.

Daniel Bauer Saarbrücken, December 2007