Extracting Lexical Semantic Knowledge from Wikipedia and Wiktionary

Torsten Zesch and Christof Müller and Iryna Gurevych

Ubiquitous Knowledge Processing Lab
Computer Science Department
Technische Universität Darmstadt, Hochschulstraße 10
D-64289 Darmstadt, Germany
{zesch,mueller,gurevych} (at) tk.informatik.tu-darmstadt.de

Abstract

Recently, collaboratively constructed resources such as Wikipedia and Wiktionary have been discovered as valuable lexical semantic knowledge bases with a high potential in diverse Natural Language Processing (NLP) tasks. Collaborative knowledge bases however significantly differ from traditional linguistic knowledge bases in various respects, and this constitutes both an asset and an impediment for research in NLP. This paper addresses one such major impediment, namely the lack of suitable programmatic access mechanisms to the knowledge stored in these large semantic knowledge bases. We present two application programming interfaces for Wikipedia and Wiktionary which are especially designed for mining the rich lexical semantic information dispersed in the knowledge bases, and provide efficient and structured access to the available knowledge. As we believe them to be of general interest to the NLP community, we have made them freely available for research purposes.

1. Introduction

Currently, the world wide web is undergoing a major change as more and more people are actively contributing to the content available in the so called Web 2.0. Some of these rapidly growing web sites, e.g. Wikipedia (Wikimedia Foundation, 2008a) or Wiktionary (Wikimedia Foundation, 2008b), have the potential to be used as a new kind of lexical semantic resource due to their increasing size and significant coverage of past and current developments.

In particular, the potential of Wikipedia as a lexical semantic knowledge base has recently started to get explored. It has been used in NLP tasks like text categorization (Gabrilovich and Markovitch, 2006), information extraction (Ruiz-Casado et al., 2005), information retrieval (Gurevych et al., 2007), question answering (Ahn et al., 2004), computing semantic relatedness (Zesch et al., 2007), or named entity recognition (Bunescu and Pasca, 2006).

Wiktionary has not yet been exploited for research purposes as extensively as Wikipedia. Interest has nonetheless already arisen, as it has recently been employed in areas like subjectivity and polarity classification (Chesley et al., 2006), or diachronic phonology (Bouchard et al., 2007).

All these tasks require reliable lexical semantic information which usually comes from linguistic knowledge bases like WordNet (Fellbaum, 1998) or GermaNet (Kunze, 2004). They are usually shipped with easy-to-use application programming interfaces (APIs), e.g. JWNL[1] or GermaNetAPI[2], that allow for easy integration into applications. However, Wikipedia and Wiktionary have lacked this kind of support so far which constitutes a significant impediment for NLP research. Therefore, we developed general purpose, high performance Java-based APIs for Wikipedia and Wiktionary that we made freely available to the research community.

In this paper, we first describe Wikipedia and Wiktionary from a lexical semantic point of view, and compare them with linguistic knowledge bases in Section 2. We review existing mechanisms of accessing Wikipedia and Wiktionary in Section 3. In Section 4., we introduce the system architecture that is used to provide structured access to the lexical semantic information contained in Wikipedia and Wiktionary. In Section 5., we show how selected NLP tasks can benefit from the improved access capabilities provided by the proposed APIs. We conclude with a summary in Section 6.

2. Collaborative Knowledge Bases

Wikipedia and Wiktionary are instances of knowledge bases that are collaboratively constructed by mainly non-professional volunteers on the web. We call such a knowledge base *Collaborative Knowledge Base* (CKB), as opposed to a *Linguistic Knowledge Base* (LKB) like WordNet (Fellbaum, 1998) or GermaNet (Kunze, 2004). In this section, we briefly analyze the CKBs Wikipedia and Wiktionary as lexical semantic knowledge bases, and compare them with traditionally used LKBs.

2.1. Wikipedia

Wikipedia is a multilingual, web-based, freely available *encyclopedia*, constructed in a collaborative effort of voluntary contributors. It grows rapidly, and with approx 7.5 million articles in more than 250 languages it has arguably become the largest collection of freely available knowledge.[3]

Articles in Wikipedia form a heavily interlinked knowledge base, enriched with a category system emerging from collaborative tagging, which constitutes a thesaurus (Voss, 2006). Wikipedia thus contains a rich body of lexical semantic information, whose aspects are thoroughly described in (Zesch et al., 2007). This includes knowledge about named entities, domain specific terms or domain specific word senses that is rarely available in LKBs. Additionally, the redirect system of Wikipedia articles can be used as a dictionary for synonyms, spelling variations and abbreviations.

[1] http://sourceforge.net/projects/jwordnet

[2] http://projects.villa-bosch.de/nlpsoft/gn_api/index.html

[3] http://en.wikipedia.org/wiki/Wikipedia:Size_comparisons

Language	Rank	Entries
French	1	730,193
English	2	682,982
Vietnamese	3	225,380
Turkish	4	185,603
Russian	5	132,386
Ido	6	128,366
Chinese	7	115,318
Greek	8	102,198
Arabic	9	95,020
Polish	10	85,494
German	12	71,399
Spanish	20	31,652

Table 1: Size of Wiktionary language editions as of February 29, 2008.

2.2. Wiktionary

Wiktionary is a multilingual, web-based, freely available *dictionary*, *thesaurus* and *phrase book*, designed as the lexical companion to Wikipedia. It is also collaboratively constructed by volunteers with no specialized qualifications necessary.

Wiktionary targets common vocabulary and matters of language and wordsmithing. It includes terms from all parts of speech, but excludes in-depth factual and encyclopedic information, as this kind of information is contained in Wikipedia.[4] Thus, Wikipedia and Wiktionary are largely complementary.

Languages and size Wiktionary consists of approx 3.5 million entries in 172 language editions.[5] Unlike most LKBs each Wiktionary edition also contains entries for foreign language terms. Therefore, each language edition comprises a multilingual dictionary with a substantial amount of entries in different languages (cf. Table 2). For instance, the English Wiktionary contains the German entry *Haus*, which is explained in English as meaning *house*.

The size of a particular language edition of Wiktionary largely depends on how active the corresponding community is (see Table 1). Surprisingly, the English edition (682,982 entries), started on December 12, 2002, is, though the oldest, not the largest one. The French Wiktionary (730,193 entries), which was launched over a year later, is the largest. Other major languages like German (71,399 entries) or Spanish (31,652 entries) are not found among the top ten, while Ido, a constructed language, has the 6th largest edition of Wiktionary containing 128,366 entries.

Table 2 shows the number of English and German entries in the corresponding Wiktionary editions. The English Wiktionary edition exceeds the size of WordNet 3.0 with 176,410 English entries as compared to 155,287 unique lexical units in WordNet. In contrast, the 20,557 German entries in the German Wiktionary edition are considerably fewer than the 76,563 lexical units in GermaNet 5.0.

Lexical semantic information Entries in Wiktionary are accompanied with a wide range of lexical and semantic information such as part of speech, word sense, gloss, etymology, pronunciation, declension, examples, sample quotations, translations, collocations, derived terms, and usage notes. Lexically or semantically related terms of several types like synonyms, antonyms, hypernyms and hyponyms are included as well. On top of that, the English Wiktionary edition offers a remarkable amount of information not typically found in LKBs, including compounds, abbreviations, acronyms and initialisms, common misspellings (e.g. *basicly* vs. *basically*), simplified spelling variants (e.g. *thru* vs. *through*), contractions (e.g. *o'* vs. *of*), proverbs (e.g. *no pain, no gain*), disputed usage words (e.g. *irregardless* vs. *irrespective* or *regardless*), protologisms (e.g. *iPodian*), onomatopoeia (e.g. *grr*), or even colloquial, slang and pejorative language forms. Most of these lexical semantic relations are explicitly encoded in the structure of a Wiktionary entry. This stands in clear contrast to Wikipedia, where links between articles usually lack clearly defined semantics.

	English Wiktionary		German Wiktionary	
	English	German	English	German
Entries	176,410	10,487	3,231	20,557
Nouns	99,456	6,759	2,116	13,977
Verbs	31,164	1,257	378	1,872
Adjectives	23,041	1,117	357	2,261
Examples	34,083	465	1,217	20,053
Quotations	8,849	55	0	0
Categories	4,019	992	32	89
Derived terms	43,903	944	2,319	36,259
Collocations	0	0	1,568	28,785
Synonyms	29,703	1,916	2,651	34,488
Hyponyms	94	0	390	17,103
Hypernyms	42	0	336	17,286
Antonyms	4,305	238	283	10,902

Table 2: The number of entries and selected types of lexical semantic information available from the English and German editions of Wiktionary as of September 2007.

Different Wiktionary editions may include different types of information; e.g. the German edition offers mnemonics, while it currently does not contain quotations. The English edition has no collocations and only very few instances of hyponymy or hypernymy (see Table 2). Like in Wikipedia, each entry in Wiktionary is additionally connected to a list of categories. Finally, entries in Wiktionary are massively linked to other entries in different ways: they are intra-linked, pointing to other entries in the same Wiktionary; they are inter-linked, pointing to corresponding entries in other language editions of Wiktionary; they also link to external knowledge bases such as Wikipedia and other web-based dictionaries.

2.3. Comparison with LKBs

Wikipedia and Wiktionary are instances of collaborative knowledge bases (other examples are dmoz[6] or Citizendium[7]). The properties of such CKBs differ from LKBs in several ways – Table 3 gives an overview.

LKBs are typically constructed by linguists following a the-

[4] http://en.wiktionary.org/wiki/Wiktionary:Criteria_for_inclusion

[5] http://meta.wikimedia.org/wiki/Wiktionary

[6] http://www.dmoz.org/

[7] http://en.citizendium.org/wiki/Main_Page

	Linguistic Knowledge Bases (LKBs)	Collaborative Knowledge Bases (CKBs)
Constructors	Linguists	Mainly non-professional volunteers
Construction approach	Following theoretical model or corpus evidence	Following non-binding guidelines
Construction costs	Significant	None
Up-to-dateness	Quickly out-dated	Mostly up-to-date
Size	Limited by construction costs	Huge or quickly growing
Data quality	Editorial control	Social control by the community
Available languages	Major languages	Many interconnected languages

Table 3: Comparison of linguistic and collaborative knowledge bases.

oretical model or guided by corpus evidence, while CKBs are created by non-professional volunteers that follow non-binding guidelines. The less strict construction approach results in certain advantages: (i) CKBs are usually released under a license that grants free usage, while LKBs are usually more restrictively distributed due to their very costly construction and maintenance process (except for e.g. WordNet that is also freely available); (ii) CKBs are mostly up-to-date while the release cycles of LKBs cannot reflect recent events; (iii) popular CKBs like Wikipedia are usually much larger than comparable LKBs; and (iv) CKBs offer availability for a large number of interconnected languages, including minor languages, for which LKBs might not exist.

The possible high benefit resulting from the use of CKBs in Natural Language Processing comes nonetheless with certain challenges: (i) CKBs are generally less well-structured than LKBs – sometimes only semi-structured –, and contain more noisy information; and (ii) CKBs rely on social control for the assurance of accuracy and comprehensiveness, whereas LKBs typically enforce editorial quality control. However, the collaborative construction approach has been argued to yield remarkable factual quality in Wikipedia (Giles, 2005), and the quality of LKBs like WordNet has also been target of criticism (Kaplan and Schubert, 2001).

3. Related Work

To our knowledge, there is no other API for Wiktionary than the one proposed and described in this paper. Thus, we focus in this section on a comparison of freely available Wikipedia APIs.

The simplest way to retrieve a Wikipedia page is to enter a search term on the Wikipedia web site.[8] However, this approach is not suited for automatic access to Wikipedia articles by an application. The Perl module WWW::Wikipedia (Summers, 2006) offers simple means for retrieving Wikipedia pages by programmatically querying the Wikipedia web site. However, this approach poses enormous load on the Wikipedia servers when used in large-scale applications. Therefore, it is discouraged by the Wikimedia Foundation.[9] Other approaches relying on web crawling and thus not being suited for large-scale NLP applications are: (i) the Wikipedia bot framework (available for different programming languages like Python[10] or Java[11]) that can be used to create small programs called *bots* acting on behalf of a normal user and usually employed for maintenance tasks, (ii) the Wiki Gateway tool box, a unified API for interfacing with a variety of remote wiki engines (Shanks, 2005), and (iii) the system developed by Strube and Ponzetto (2006) relying on a modified version of the WWW::Wikipedia module to retrieve articles.

Crawling can be avoided by running an own server using publicly available Wikipedia database dumps.[12] This gives better, but still insufficient performance, due to the overhead related to using a web server for delivering the retrieved pages. In this setting, retrieving a Wikipedia article usually involves a transfer of the request from an application to the web server. The web server then executes a PHP script that accesses the Wikipedia database, and the database returns the article content encoded using Wiki markup[13] to the PHP script which converts the Wiki markup to HTML. Finally, the web server delivers the HTML encoded data back to the application. This poses a substantial overhead that might render large-scale NLP tasks impossible.

This overhead can be avoided by directly accessing the database dumps. For example, the Perl module Parse::MediaWikiDump (Riddle, 2006) parses the Wikipedia XML dump to retrieve articles. As Wikipedia dumps are very large (over 3 GB of compressed data for the snapshot of the English Wikipedia from Feb 2008), the performance of parsing is not sufficient for large-scale NLP tasks (it may take up to several seconds to retrieve a given article). Additionally, the time that is required to retrieve an article is not easily predictable, but depends on the article's position in the XML dump.

WikiPrep (Gabrilovich, 2007) is a preprocessor that transforms a Wikipedia XML dump into an optimized XML format that explicitly encodes information such as the category hierarchy or article redirects. However, as the resulting data is still in XML format, WikiPrep suffers from the same performance problem as Parse::MediaWikiDump.

In the approach presented in this paper, we import the database dumps into a database. Then, we can use the sophisticated indexing offered by the database that guarantees nearly constant retrieval time for each article. This approach is superior to web server based retrieval, as it only involves querying the database and directly delivering the results to the application. Another important benefit is that

[8] http://www.wikipedia.org/

[9] http://en.wikipedia.org/wiki/Wikipedia:Database_download#Please_do_not_use_a_web_crawler

[10] http://pywikipediabot.sourceforge.net/

[11] http://jwbf.sourceforge.net/

[12] http://download.wikipedia.org/

[13] http://en.wikipedia.org/wiki/WP:MARKUP

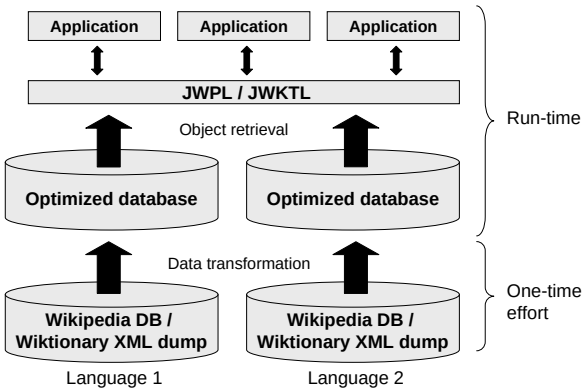

Figure 1: System architecture of JWPL and JWKTL.

the database schema explicitly represents information about an article's links or categories, which is only implicitly encoded in the HTML structure.
The system architecture is described in detail in the following section.

4. Extracting Knowledge from Wikipedia and Wiktionary

If Wikipedia or Wiktionary are to be used for large-scale NLP tasks, efficient programmatic access to the knowledge therein is required. Therefore, we developed general purpose, high performance Java-based APIs abbreviated as JWPL (**J**ava-based **W**iki**P**edia **L**ibrary) and JWKTL (**J**ava-based **W**i**KT**ionary **L**ibrary). JWPL is already freely available for research purposes, JWKTL will be released by the time of the conference at the latest.[14]

4.1. Java-based Wikipedia Library (JWPL)

The original structure of the Wikipedia database is optimized for searching articles by keywords which is performed by millions of users of the online Wikipedia every day. However, an API designed for NLP research has to support a wider range of access paths, including iteration over all articles, a query syntax, as well as efficient access to information like links, categories, and redirects. Thus, JWPL operates on an optimized database (as shown in Figure 1) that is created in a one-time effort from the database dumps available from the Wikimedia Foundation.[15]
The *advantages* of this system architecture are: (i) computational efficiency enabling large-scale NLP tasks, (ii) reproducible research results, and (iii) an easy to use object-oriented programming interface, that cannot be found in this combination by any of the competing approaches outlined in Section 3.
Reproducible experimental results are a direct consequence of using a fixed database dump instead of the online Wikipedia that is very likely to change between two runs of a certain experimental setting.
Computational efficiency is also a consequence of accessing the database using its indexing mechanisms for fast retrieval. The data from the database is directly mapped to Java objects using the Hibernate object-relational mapping framework (Bauer and King, 2004). This also means that JWPL is not restricted to using a certain database, but may run on top of the most common database systems.[16]
The design of the object-oriented programming interface is centered around the objects: WIKIPEDIA, PAGE, and CATEGORY. The WIKIPEDIA object is used to establish the connection with the database (as shown in Listing 1), and to retrieve PAGE and CATEGORY objects. JWPL supports retrieval by keywords or via a query interface that allows for wildcard matches as well as retrieving subsets of articles or categories depending on parameters like the number of tokens in an article or the number of ingoing links. The WIKIPEDIA object also allows to iterate over articles, categories, redirects, and disambiguation pages.
A PAGE object represents either a normal Wikipedia article, a redirect to an article, or a disambiguation page. Each PAGE object provides access to the article text (with markup information or as plain text), the assigned categories, the ingoing and outgoing article links, as well as all redirects that link to this article.
CATEGORY objects represent Wikipedia categories and allow access to the articles within this category. As categories in Wikipedia form a thesaurus, a CATEGORY object also provides means to retrieve parent and child categories, as well as siblings and all recursively collected descendants. JWPL also provides a CATEGORYGRAPH object that e.g. allows to find the shortest path between two given categories (as shown in Listing 2).
Listing 3 presents a more complex example showing how to retrieve a list of 'towns in Germany' from Wikipedia. Executing the given Java code using the English Wikipedia from 9th Feb 2007 yields a list of almost 3,000 towns in Germany.
The next release of JWPL – scheduled for April 2008 – will also contain a parser for the Wikipedia markup language. The parser allows to easily identify and access even more fine-grained information within Wikipedia articles, e.g. sections, paragraphs, templates, links, link texts, link contexts, lists, and tables. Figure 2 visualizes the structure of the Wikipedia article "Natural Language Processing" as analyzed by the parser.

4.2. Java-based Wiktionary Library (JWKTL)

The Wiktionary API (JWKTL) follows a similar system architecture as the Wikipedia API (JWPL), as shown in Figure 1. JWKTL is based on freely available Wiktionary dumps[17] of different language editions in XML format. In order to provide a fast and easy access to the lexical semantic knowledge in Wiktionary, the output of the parser is stored using the Berkeley DB database library.[18] For each Wiktionary entry, the API returns a Java object which contains the extracted information.
The word entries in Wiktionary use the same mark-up language as Wikipedia. As the different language editions of Wiktionary use different structural elements for encoding the lexical semantic information, the Wiktionary parser

[14] http://www.ukp.tu-darmstadt.de/software/
[15] http://download.wikipedia.org/
[16] http://www.hibernate.org/80.html
[17] http://dumps.wikimedia.org/
[18] http://www.oracle.com/technology/products/berkeley-db/index.html

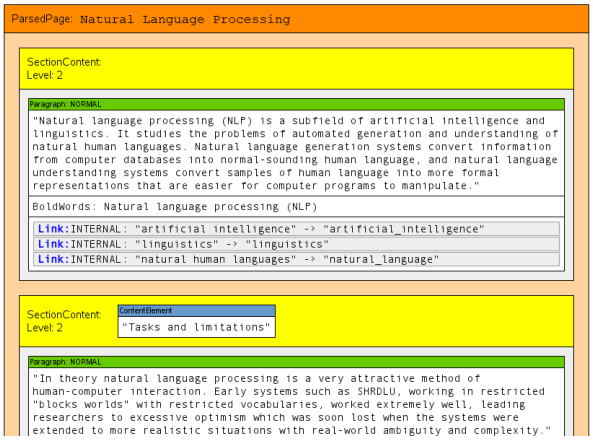

Figure 2: A visualization of the structure of a Wikipedia article as analyzed by the parser.

needs to be adjusted to each language edition. For most language editions, the user community has introduced a layout standard acting as a data schema to enforce a uniform structure of entries. However, as schemas evolve over time, older entries are possibly not updated. Moreover, as no contributor is forced to follow the schema, the structure of entries is fairly inconsistent. Therefore, the parser is designed to be robust against errors of incorrect usage of the markup language.

The API is centered around the Java object WIKTIONARY. It wraps the underlying database and allows to query the database for information about a certain word by using the word's grapheme as query argument (see Listing 4). Additionally, the desired part of speech or word language can also be specified. The API allows to combine several language editions of Wiktionary into one WIKTIONARY object and query the contained entries simultaneously. For each grapheme, Wiktionary contains a page with entries of corresponding words for different languages and parts of speech. In order to allow a structured access to the information available for each word, the API maps each entry to the object WIKTIONARYWORD. Thus, each WIKTIONARYWORD object contains the information for exactly one part of speech and one language. The available information of the entries can be accessed by calling the object's methods, which return the specified information on word or sense level (see Listing 5).

Currently, the proposed API provides robust parsing of the English and the German Wiktionary editions and extracts structured information, including glosses, etymology, examples, quotations, translations, derived terms, characteristic word combinations, lexical relations, as well as links to other language editions of Wiktionary, Wikipedia articles, and external web pages. The parser can be easily adjusted to work with other language editions of Wiktionary.

5. Example Usage in NLP

The APIs for access to Wikipedia and Wiktionary proposed in this paper have already been put into service for large-scale NLP research, such as analyzing and accessing the structure of the Wikipedia category graph (Zesch and Gurevych, 2007), computing semantic relatedness between words (Zesch et al., 2007), and semantic information retrieval (Gurevych et al., 2007).

When analyzing the structure of the Wikipedia category graph, categories assigned to the articles of Wikipedia are viewed as nodes in a directed graph, where the subcategory relation between two categories is cast as a directed edge between the corresponding nodes in the graph. The CATEGORYGRAPH object in JWPL offers means to retrieve graph parameters like diameter, cluster coefficient, or average shortest path length.

The structure of the resulting graph (as defined by the graph parameters) is indicative of the possible performance of graph-based NLP applications, e.g. computing the semantic relatedness between words. This task requires to retrieve the corresponding Wikipedia article for each word, and then to compute the minimum path length between the categories of the two articles (see Listing 2). On this basis, efficient algorithms for computing semantic relatedness using Wikipedia can be easily implemented using JWPL.

Another NLP related task that directly benefits from the capabilities of JWPL and JWKTL is semantic information retrieval. Gurevych et al. (2007) describe work in which electronic career guidance is used to support school leavers in their search for a profession or vocational training. One special challenge in this task is the *vocabulary gap* between the language of the (expert-authored) documents from the database and the language of the school leavers. JWPL has been successfully used to bridge this vocabulary gap by using knowledge extracted from Wikipedia in the retrieval process. Currently, we are working on the integration of knowledge from Wiktionary into information retrieval using JWKTL.

6. Conclusion

Recently, the collaboratively created resource Wikipedia was discovered as a lexical semantic knowledge base that has the potential to trigger major performance increases in such diverse NLP areas as text categorization, information extraction, question answering, computing semantic relatedness, or named entity recognition. Its younger sister project, Wiktionary, has lately emerged as a valuable resource complementing it. We have shown that these collaborative knowledge bases contain lexical semantic knowledge which is not commonly encountered in linguistic knowledge bases. The need of appropriate programmatic access to the knowledge therein is self-evident.

This paper presented Java based APIs that allow for efficient access to Wikipedia and Wiktionary, and demonstrated cases of their usage. As the APIs are freely available for research purposes, we think that they will foster NLP research using the collaborative knowledge bases Wikipedia and Wiktionary.[19]

Acknowledgments

This work was carried out as part of the project "Semantic Information Retrieval from Texts in the Example Domain *Electronic Career Guidance*" (SIR) funded by the

[19] JWPL is already available at http://www.ukp.tu-darmstadt.de/software. JWKTL will be released by the time of the conference at latest on the same website.

German Research Foundation under the grant GU 798/1-2. We thank the students Lizhen Qu and Christian Jacobi for implementing important parts of JWKTL and JWPL, and our colleague Konstantina Garoufi for her valuable contributions to the final version of this paper.

7. References

David Ahn, Valentin Jijkoun, Gilad Mishne, Karin Müller, Maarten de Rijke, and Stefan Schlobach. 2004. Using Wikipedia at the TREC QA Track. In *Proceedings of TREC 2004*.

Christian Bauer and Gavin King. 2004. *Hibernate in Action. Practical Object/Relational Mapping*. Manning Publications Co.

Alexandre Bouchard, Percy Liang, Thomas Griffiths, and Dan Klein. 2007. A probabilistic approach to diachronic phonology. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 887–896.

Razvan Bunescu and Marius Pasca. 2006. Using Encyclopedic Knowledge for Named Entity Disambiguation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 9–16, Trento, Italy.

Paula Chesley, Bruce Vincent, Li Xu, and Rohini Srihari. 2006. Using verbs and adjectives to automatically classify blog sentiment. In *Proceedings of AAAI-CAAW-06, the Spring Symposia on Computational Approaches to Analyzing Weblogs*.

Christiane Fellbaum. 1998. *WordNet An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Evgeniy Gabrilovich and Shaul Markovitch. 2006. Overcoming the Brittleness Bottleneck using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference*, Boston, Massachusetts, USA, July. AAAI Press.

Evgeniy Gabrilovich. 2007. WikiPrep. URL http://www.cs.technion.ac.il/~gabr/resources/code/wikiprep/.

Jim Giles. 2005. Internet encyclopaedias go head to head. *Nature*, 438(7070):900–901, December.

Iryna Gurevych, Christof Müller, and Torsten Zesch. 2007. What to be? - Electronic Career Guidance Based on Semantic Relatedness. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 1032–1039, Prague, Czech Republic, June. Association for Computational Linguistics.

Aaron N. Kaplan and Lenhart K. Schubert. 2001. Measuring and improving the quality of world knowledge extracted from WordNet. Tech. Rep. 751 14627-0226, Dept. of Computer Science, Univ. of Rochester, Rochester, NY, May.

Claudia Kunze, 2004. *Lexikalisch-semantische Wortnetze*, chapter Computerlinguistik und Sprachtechnologie, pages 423–431. Spektrum Akademischer Verlag.

Tyler Riddle. 2006. Parse::MediaWikiDump. URL http://search.cpan.org/~triddle/Parse-MediaWikiDump-0.40/.

Maria Ruiz-Casado, Enrique Alfonseca, and Pablo Castells. 2005. Automatic Assignment of Wikipedia Encyclopedic Entries to WordNet Synsets. In *AWIC*, pages 380–386.

Bayle Shanks. 2005. Wikigateway: a library for interoperability and accelerated wiki development. In *WikiSym '05: Proceedings of the 2005 international symposium on Wikis*, pages 53–66, New York, NY, USA. ACM Press.

Michael Strube and Simone Paolo Ponzetto. 2006. WikiRelate! Computing semantic relatedness using Wikipedia. In *Proceedings of the 21st National Conference on ArtificialIntelligence (AAAI-06)*, pages 1419–1424, Boston, Mass., July.

Ed Summers. 2006. WWW:Wikipedia. URL http://search.cpan.org/~esummers/WWW-Wikipedia-1.9/.

Jakob Voss. 2006. Collaborative thesaurus tagging the Wikipedia way. *ArXiv Computer Science e-prints*, cs/0604036.

Wikimedia Foundation. 2008a. Wikipedia. URL http://www.wikipedia.org.

Wikimedia Foundation. 2008b. Wiktionary. URL http://www.wiktionary.org.

Torsten Zesch and Iryna Gurevych. 2007. Analysis of the Wikipedia Category Graph for NLP Applications. In *Proceedings of the TextGraphs-2 Workshop (NAACL-HLT 2007)*, pages 1–8.

Torsten Zesch, Iryna Gurevych, and Max Mühlhäuser. 2007. Comparing Wikipedia and German Wordnet by Evaluating Semantic Relatedness on Multiple Datasets. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2007)*, pages 205–208.

A Java code examples

Listing 1: Establishing the database connection.

```
// configure the database connection parameters
DatabaseConfiguration dbConfig = new DatabaseConfiguration();
dbConfig.setDatabase("DATABASE");
dbConfig.setHost("SERVER_URL");
dbConfig.setUser("USER");
dbConfig.setPassword("PASSWORD");
dbConfig.setLanguage("LANGUAGE");

// Create a new Wikipedia.
Wikipedia wiki = new Wikipedia(dbConfig);
```

Listing 2: Getting the path length between two categories.

```
// Assuming that a Wikipedia object was already instantiated.
CategoryGraph cg = new CategoryGraph(wiki);
Category c1 = wiki.getCategory("Germany");
Category c2 = wiki.getCategory("France");
int pathLength = cg.getPathLength( c1, c2 );
```

Listing 3: Getting a list of all towns in Germany that are listed in Wikipedia.

```
// Get the category ``Towns in Germany'',
// assuming that a Wikipedia object was already instantiated.
Category topCat = wiki.getCategory("Towns in Germany");

// Add the pages categorized under ``Towns in Germany'' to the list.
Set<String> towns = new TreeSet<String>();
for (Page p : topCat.getPages()) {
   towns.add(p.getTitle().getPlainTitle());
}
// Add the pages categorized under all subcategories of ``Towns in Germany'' to the list.
for (Category townCategory : topCat.getDescendants()) {
   for (Page p : townCategory.getPages()) {
       towns.add(p.getTitle().getPlainTitle());
   }
}
```

Listing 4: Working with a *Wiktionary* object.

```
// create object representing the German edition of Wiktionary
Wiktionary wiktionary = new Wiktionary(DB_PATH, Language.GERMAN);

// add the English edition of Wiktionary
wiktionary.addWiktionary(DB_PATH, Language.English);

// take only entries for German words into account
wiktionary.setWordLanguage(Language.German);

// query Wiktionary for "bank"
List<WiktionaryWord> wordList = wiktionary.getWords("bank");
```

Listing 5: Working with a *WiktionaryWord* object.

```
// get first word from the wordList retrieved in Listing 4
WiktionaryWord word = wordList.get(0);

// get part-of-speech
PartOfSpeech pos = word.getPartOfSpeech();

// get the gloss of the first sense
String gloss = word.getGloss(0);

// get hyponyms for the first sense
List<String> hyponyms = getRelatedTerms(Relation.HYPONYMY, 0);
```