Word Sense Disambiguation with GermaNet

Semi-Automatic Enhancement and Empirical Results

Dissertation

zur Erlangung des akademischen Grades
Doktor der Philosophie
in der Philosophischen Fakultät
der Eberhard Karls Universität Tübingen

vorgelegt von
Verena Henrich
aus Darmstadt

2015

Gedruckt mit Genehmigung der Philosophischen Fakultät
der Eberhard Karls Universität Tübingen

Hauptberichterstatter: Prof. Dr. Erhard Hinrichs
Mitberichterstatter: Prof. Dr. Gerhard Jäger
Dekan: Prof. Dr. Jürgen Leonhardt
Tag der mündlichen Prüfung: 29.4.2015
Verlag: TOBIAS-lib, Tübingen

Abstract

The subject of this dissertation is boosting research on word sense disambiguation (WSD) for German. WSD is a very active area of research in computational linguistics, but most of the work is focused on English. One of the factors that has hampered WSD research for other languages such as German is the lack of appropriate resources, particularly in the form of sense-annotated corpus data. Hence, this work inevitably has to start with the preparation of resources before actual WSD experiments can be performed. The work program is fourfold. Firstly, since sense definitions are necessary to distinguish word senses (both for humans and for automatic WSD algorithms), the German wordnet GermaNet is (semi-)automatically extended with sense descriptions. This is done by automatically mapping GermaNet senses to descriptions in the online dictionary Wiktionary. Secondly, since the availability of sense-annotated corpora is a prerequisite for evaluating and developing word sense disambiguation systems, two GermaNet sense-annotated corpora are constructed. One corpus is automatically constructed and the other corpus is manually sense-annotated. Thirdly, several knowledge-based WSD algorithms are applied and evaluated – using the newly created sense-annotated corpora. These algorithms are based on a suite of semantic relatedness measures, including path-based, information-content-based, and gloss-based methods. Experiments on gloss-based methods also employ the newly harvested definitions from Wiktionary. Fourthly, several supervised machine learning classifiers are applied to the task of German WSD, including rule-based methods, instance-based methods, probabilistic methods, and support vector machines. The classifiers rely on a wide range of machine learning features and their evaluation focuses on several aspects, including a comparison of several algorithms, a detailed analysis of the implemented features, and an investigation of the influence of syntax and semantics on the disambiguation performance for verbs.

Für Jens und meine Eltern

Acknowledgements

I would like to acknowledge everyone who supported me during the work on this thesis. Chronologically the first to thank are those who raised my interest in the topic of natural language processing during my master studies in Darmstadt and Reykjavik: without Prof. Dr. Bettina Harriehausen-Mühlbauer, Dr. Hrafn Loftsson, and Timo Reuter I would not have turned to this research area.

I am very grateful to my supervisor Prof. Dr. Erhard Hinrichs in several – conceptual and institutional – respects: of course, for supervising me working on the thesis and providing me with useful remarks, but also for giving me the opportunity to develop and extend my personal knowledge and strengths in many diverse aspects of computational linguistics and of the university's daily work. He was always patient with me and gave me enough space to tackle those research questions I am interested in, while, at the same time, guided me into the right direction.

I would like to thank my second reviewer Prof. Dr. Gerhard Jäger and the other members of my dissertation committee, Prof. Dr. Fritz Hamm, PD Dr. Helmut Schmid, and Prof. Dr. Andrea Weber, for their valuable feedback and comments.

Due to my computer science background I am thankful to my 'linguistic' colleagues: to Reinhild Barkey mainly for GermaNet-related collaborations and discussions and to Kathrin Beck and Dr. Heike Telljohann for TüBa-D/Z-related linguistic input.

Many thanks to Dr. Yannick Versley for making it possible to reuse his annotation tool for sense annotation in the TüBa-D/Z treebank, to Corina Dima, Christina Hoppermann, and Jianqiang Ma for fruitful discussions during

our ‘PhD meetings’, and to all anonymous reviewers for their comments on papers that were published as part of this thesis. I would like to thank my SfS colleagues Dr. Chris Culy, Marie Hinrichs, Dr. Daniël de Kok, Jochen Saile, Daniil Sorokin, Johannes Wahle, Dr. Holger Wunsch, Dr. Thomas Zastrow, and Ramon Ziai and my external colleagues Prof. Dr. Chris Biemann, Dr. Christian Meyer, Tristan Miller, and Prof. Dr. Torsten Zesch for valuable support and feedback on several aspects of my thesis.

I am thankful to the many student assistants who were part of the GermaNet project at certain times – both on the programmatic as well as on the lexicographic side. In particular, thanks to Tatiana Vodolazova, Anne Brock, Agnia Barsukova, Edo Collins, and Steffen Tacke for their implementation contributions and Johannes Wahle, Sarah Schulz, Valentin Deyringer, and Annabell Grasse for helping with manual annotations.

Since English is not my native language I am grateful to Siân Alsop, Dr. Scott Martens, and Tristan Miller for proof-reading some of my chapters.

This thesis was written using the LaTeX thesis template provided by the Engineering Department of the University of Cambridge.[1]

Mein besonderer Dank gilt meinen Schwiegereltern, die mir jederzeit viel Verständnis und Geduld entgegengebracht haben.

Am allermeisten und von ganzem Herzen möchte ich Jens und meinen Eltern für ihr Verständnis und für ihre bedingungslose Unterstützung danken. Ich widme euch aus Dankbarkeit diese Arbeit, denn ohne euch hätte ich es nicht geschafft.

[1] `http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/ThesisStyle/`

Table of Contents

Citation Conventions

Parenthetical literature citations are consequently and deliberately placed in four different positions throughout this thesis. Firstly, citations at the end of a paragraph (after any end of sentence punctuation mark) generally refer to the paragraph – either to the whole paragraph or to most of it. Secondly, literature citations at the end of a sentence (before the sentence final punctuation) usually refer to the whole sentence. Thirdly, literature citations in the middle of a sentence refer to the phrase or term after which they are placed. Note that this potentially includes the previous case, if such a phrase or term occurs at the end of a sentence. It has to be left to the reader to figure out whether such a citation refers to the whole sentence or to only a part of it – though it mostly refers to the entire sentence or should be clear from the context otherwise. If there are distinct references for several terms or phrases in the same sentence, one sentence contains several citations at different positions. Fourthly, citations that are placed after the punctuation mark of one sentence and before the beginning of the next sentence within the same paragraph refer to both the sentence before and after the citation.

If no particular priority is implied for multiple citations, lists of citations are generally first sorted chronologically and then alphabetically.

The position of footnotes is less complex: footnotes are placed after the end of sentence punctuation to refer to an entire sentence. If this sentence is the last sentence of a paragraph, the footnote might refer to the entire paragraph or only to its last sentence. Footnotes inside a sentence refer to a particular phrase or term (this implies their placement immediately before the sentence final punctuation).

Part I

Introduction

Chapter 1

Introduction

1.1 Goals

The overall objective of this thesis is to overcome the bottleneck of German word sense disambiguation (WSD) resources and to boost research on WSD for German. In order to achieve this goal, the work inevitably has to start with the preparation of the necessary resources before actual word sense disambiguation experiments can be performed.

The work program of this thesis is fourfold:

(i) *Extend GermaNet with sense definitions from Wiktionary:* only about 10% of all entries in the German wordnet GermaNet [Hamp and Feldweg, 1997] have sense definitions. In order for humans to manually annotate word occurrences in a corpus with senses from GermaNet, the distinction of senses has to be clear in the minds of the annotators. Without sense definitions, this distinction is difficult. Further, several WSD algorithms rely on the existence of sense definitions. Therefore, the descriptions from the online dictionary Wiktionary are automatically mapped to senses from GermaNet in order to harvest them as sense definitions.

(ii) *Create sense-annotated corpora annotated with GermaNet senses:* the availability of sense-annotated corpora is a necessary prerequisite for evaluating and developing word sense disambiguation systems. Since the overall objective of this dissertation is to overcome the bottleneck

of German WSD resources, the creation of sense-annotated corpora is indispensable. The manual construction of sense-annotated corpora is expensive; it is therefore timely and appropriate to investigate automatic means of creating such a resource. The presented method for automatically constructing a sense-annotated corpus relies on the mapping between GermaNet and Wiktionary described under (i) above to harvest sense-specific example sentences from Wiktionary itself and additional textual materials from other web-based textual sources such as Wikipedia and online newspaper materials. In order to allow a comparison of the performance of WSD algorithms on these automatically harvested web-based materials with a real *authentic* corpus and to overcome the limitations of such an automatically constructed resource (like the assumably varying quality of web texts and the restriction of not being able to compile most frequent sense information due to the skewed number of annotated target word occurrences), the TüBa-D/Z treebank [Telljohann et al., 2004, 2012] is manually extended by sense annotation for a selected set of lemmas.

(iii) *Evaluate several knowledge-based WSD algorithms on German:* since one of the main purposes of this thesis is to help boost research on WSD for German, it uses the sense-annotated corpora described under (ii) to evaluate and compare a wide range of knowledge-based WSD algorithms. The knowledge-based WSD algorithms are based on a suite of semantic relatedness measures, including path-based, information-content-based, and gloss-based methods. Experiments on gloss-based methods also employ the newly harvested definitions from Wiktionary, which are linked to corresponding GermaNet senses via the automatic mapping between GermaNet and Wiktionary described under (i).

(iv) *Evaluate supervised machine learning classifiers on German WSD:* another set of WSD experiments uses several supervised classification algorithms, including rule-based methods, instance-based methods, probabilistic methods, support vector machines, and combined approaches. The algorithms rely on a wide range of machine learning features such

as morphological information for the target word, structural information from the sentence, or co-occurring words or word classes. The evaluation of the supervised WSD experiments focuses on several aspects, including a comparison of several heterogeneous machine learning algorithms, a detailed analysis of the implemented machine learning features, and an investigation of the influence of syntax and semantics on the disambiguation performance for verbs.

1.2 Word Senses to be Disambiguated

Ambiguity is a pervasive phenomenon in natural languages. The ambiguity of words is due to the fact that words can have more than one meaning (in this context also referred to as a word sense). The resolution of this word sense ambiguity is referred to as word sense disambiguation (WSD). *Word sense disambiguation* is the task of computationally assigning the most appropriate senses to words occurring in a text – where the senses are usually taken from a predefined sense inventory such as a dictionary or a lexicon.[1] [Agirre and Edmonds, 2006a; Navigli, 2009; Kwong, 2012]

The task of disambiguating word senses presupposes the existence of word senses. Word senses are typically defined and listed in dictionaries. Many heterogeneous approaches of how to define word senses exist, including behaviourist, conceptual, cognitive, contextual, definitional, denotational, descriptive, distributional, feature-based, formal, generative, generativist, prototype, relational, structuralist, and truth-conditional approaches [Cruse, 2006; Geeraerts, 2010; Kwong, 2012; Goddard and Wierzbicka, 2014]. Since it would be beyond the scope of this dissertation to account for all existing theories, four prominent yet heterogeneous approaches are described in the following.[2]

In the traditional *componential analysis*, which is also referred to as *feature analysis*, conceptual categories are defined by semantic features that need to

[1] A variety of techniques for trying to solve the task of WSD exists – including knowledge-based approaches and supervised or unsupervised machine learning approaches – which are described in Chapter 2.

[2] See Geeraerts [2010] for an extensive overview of approaches to defining word senses.

be both individually necessary and jointly sufficient. That is, every object that jointly satisfies all features of a definition is classified into the particular category and – the other way around – objects in a category must satisfy each of the described features. For example, the conceptual category of *birds* might be described as *egg-laying vertebrates with wings, feathers, and a beak*, i.e., appropriate objects must satisfy features such as *'being a vertebrate'*, *'egg-laying'*, *'with wings'*, *'with feathers'*, and *'with a beak'*. While this approach is intuitive, it has several deficiencies: for example, features need to represent discrete properties and conceptual categories are supposed to have clear boundaries, although natural objects and their properties often have fuzzy boundaries. Further, the status of all objects in a category is equal, although there usually are main and subordinate representatives. In the example at hand, both *robins* and *penguins* are equally classified into the *birds* category, although *robins* are clearly more characteristic representatives of the *birds* category than *penguins* are. [Goodenough, 1956; Katz and Fodor, 1963; Pottier, 1964, 1965; Greimas, 1966; Lipka, 1987; Murphy, 2004]

Several theories were postulated to overcome these deficiencies. The most prominent among those is the *prototype theory*, where *categories tend to become defined in terms of prototypes or prototypical instances that contain the attributes most representative of items inside and least representative of items outside the category* [Rosch, 1978, page 30]. The features (attributes) in the prototype theory thus have different relevance, which reflect the status of certain objects in a category. For example, *robins* represent prototypical *birds*, while *penguins* are less prototypical *birds*. [Rosch, 1975, 1978]

Another strategy is pursued by the *relational approach*, which defines word senses in terms of their relations – such as synonymy, antonymy, hyponymy, and meronymy – to other words. Prominent resources that encode word senses in this relational manner are wordnets. A wordnet is a lexical semantic resource that groups words into sets of synonyms (*synsets*) and interrelates word senses and synsets in a network. For the example of a *bird*, the English Princeton WordNet[1] [Miller, 1995; Fellbaum, 1998a] encodes a hypernymy relation to the

[1]Note that in this thesis the term *WordNet* with a capitalized *W* and a capitalized *N* consistently refers to the Princeton WordNet, while the term *wordnet* with all small

synset *{vertebrate, craniate}*, meronymy relations to *{wing}*, *{feather, plume, plumage}*, *{beak, bill, neb, nib, pecker}*, a holonymy relation to *{flock}*, etc. It further links both *{robin, redbreast, robin redbreast, Old World robin, Erithacus rubecola}* and *{penguin}* as indirect hyponyms.[1] [Lyons, 1963, 1968; Cruse, 1986; Miller, 1995; Geeraerts, 2010]

The *distributional approach* assumes that *words with semantically similar meanings occur in linguistically similar contexts* (inter alia stated in Rubenstein and Goodenough [1965], Schütze and Pedersen [1995], and Pantel [2005] – each with slightly varying wordings). In this approach the definition of word senses takes corpus occurrences into account. That is, occurrences of a word in question are extracted from large collections of text and all word occurrences with distributionally similar contexts are grouped into clusters. For each cluster that represents a certain amount of attested word occurrences, a separate word sense is defined. This procedure of inducing word senses is also referred to as *word sense discrimination* [Schütze, 1998]. Although it does not prevent word senses of being overlapping, it reflects attested examples of word use. [Harris, 1955, 1956; Schütze, 1992; Kilgarriff, 1997a, 2006; Sahlgren, 2008; Geeraerts, 2010; Kwong, 2012]

Since the present work takes word senses from the German wordnet GermaNet [Hamp and Feldweg, 1997] (see Chapter 3) and since the relational approach underlies the structure of wordnets, this relational approach is the one mainly referred to when speaking about word senses throughout this dissertation.

The task of word sense disambiguation assumes that it is possible to assign predefined word senses to word occurrences in a text, given a particular context. The emerging problems are twofold. Firstly, given the heterogeneity

case letters generally refers a wordnet resource – not necessarily to the English Princeton WordNet, but potentially to wordnets of any language.

[1] All relations in the example are taken from Princeton WordNet 3.1. For reasons of simplicity only a subset of relations is used in the example. The comma-separated lists within curly braces represent word senses linked by the synonymy relation (i.e., synsets). Note that – although not given in the example – synsets in WordNet are mostly accompanied by definitions, for the reason stated by Miller [1995, page 40]: *not enough semantic relations are encoded into WordNet to support such constructions. Following standard lexicographic practice, definitional glosses are included in most synsets.*

of possible sense definitions, the word senses listed in a specific dictionary resource might not necessarily reflect the exact meanings of words in a certain text. Secondly, the interpretation of a word in a text often further depends on implicit information – including pragmatic aspects or background knowledge such as world knowledge or knowledge of a certain situation. For example, if a group of men is having dinner at a restaurant, the noun *tip* in a statement like *he gave a tip to the waitress* seems to refer to the 'gratuity' that one of the men gave to their waitress. However, knowing that the man who payed usually never gives gratuity, but generally likes to communicate and to propagate wisdoms, the occurrence of *tip* in the example rather seems to be used in its 'hint/ advise' sense. Despite these issues, research on automatic word sense disambiguation, which takes word senses as idealizations, has successfully employed existing sense inventories (see Chapter 2).

Word sense disambiguation further implicitly assumes that word senses are discrete [Kwong, 2012], but they are often hard to distinguish: where exactly does one sense start and another one end? Is it at all possible to clearly distinguish word senses, i.e., are they categorical or do they represent overlapping continua? For example, the two words *light* and *dark* describe apparently distinct properties of luminosity. Since luminosity is a continuum, there are certain mediocre states, where it is hard to tell whether it is still light or already dark, i.e., which state exactly represents the transition from light to dark.

Although there is psychologically no doubt that word senses exist and many natural language processing applications assume that they exist, the difficulty in defining and distinguishing word senses raises the fundamental question about their existence. It might sounds strange, but several researchers questioned whether word senses exist at all: two contributions to this debate include *I don't believe in word senses* by Kilgarriff [1997a] and *Do word meanings exist?* by Hanks [2000]. Both Kilgarriff and Hanks conclude that word senses indeed exist, but they propose restrictions: Kilgarriff [1997a, page 91] argued that *word senses exist only relative to a task* and Hanks [2000, page 214] claimed that *traditional descriptions are misleading* and that *words have meaning potentials, rather than just meaning.*

A remaining open question is thus about the right granularity of sense distinctions. Although many natural language processing applications need coarse sense distinctions that correspond to the homograph level, the required sense granularity often depends on the purpose [Kilgarriff, 1997a; Ide and Wilks, 2006]. Regardless of any dictionary or corpus, the noun *tip*, for example, should at least have the three distinct senses describing (i) the peak or end of something pointed, (ii) the money given to a waiter as gratuity, and (iii) the synonymous use as a hint or advice. These three senses are clearly distinct in their meaning. If it is coincidentally the same word string – usually without a common origin – that is used to express such distinct semantic concepts, they are called *homographs* (from Greek *homós* 'same' and gráphō 'write') [Ide and Wilks, 2006].

However, the set of word senses is not identical for all word sense sources. For example, distributional clustering approaches produce different sets of senses depending on the corpora used, and lexicographers building dictionaries do not necessarily agree on the same granularity level of sense distinctions. Already the lexicographers working on the same series of a dictionary have to produce different granularity sets of senses depending on the purpose of a dictionary (main, shorter, concise, pocket) being produced by publishers such as Oxford University Press.[1]

For the example noun *tip*, the Princeton WordNet further differentiates the first of these senses into three more fine-grained senses described as (i-a) *the extreme end of something; especially something pointed*, (i-b) *'point', 'peak', a V shape*, and (i-c) *'crown', 'crest', 'top', 'summit', the top or extreme point of something (usually a mountain or hill)*.[2] These more fine-grained senses share a common origin and describe somewhat similar semantic concepts. In general, all five distinguished senses[3], i.e., including the above-listed homographs plus the three senses below the level of homography, are called *polysemes* (from Greek *poly-* 'many' and *sêma* 'sign'). It has been argued in the literature that WordNet is much too specific in defining too many fine-grained word senses

[1] This example is taken from Ide and Wilks [2006, page 49].

[2] These sense descriptions are taken from WordNet 3.1.

[3] Five senses, because sense (i) is further divided into three senses and thus resulting in the five senses (i-a), (i-b), (i-c), (ii), and (iii).

[Palmer, 2000; Ide and Wilks, 2006].

However, the sense distinction in the German wordnet GermaNet is more coarse-grained and does not distinguish that many different senses than the Princeton WordNet. For example, there is one sense of *Buch* 'book' in GermaNet expressing the artifact *book* in the sense of *several printed pages bound together*. In WordNet, there are two more fine-grained senses expressing this semantic concept described as (i) *a written work or composition that has been published (printed on pages bound together)* and (ii) *'volume', physical objects consisting of a number of pages bound together*. Another example would be *Mund* 'mouth' in the sense of a human anatomy feature. There is one sense in GermaNet and two senses in WordNet representing this semantic concept. The two senses in WordNet are distinguished as (i) *'oral cavity', 'oral fissure', 'rima oris' the opening through which food is taken in and vocalizations emerge* and (ii) *the externally visible part of the oral cavity on the face and the system of organs surrounding the opening.*[1]

The distinction of word senses is clearly much easier on the homograph level for the simple psycholinguistic reason that distinct enough senses are being represented separately in the mental lexicon and are thus easier to distinguish [Ide and Wilks, 2006]. Many researchers have been experimenting with manually or automatically collapsing similar fine-grained senses of existing sense inventories such as WordNet into more coarse-grained ones [Fellbaum et al., 2001; Agirre and Lacalle, 2003; Mihalcea et al., 2004; Navigli et al., 2007; Palmer et al., 2007; Snow et al., 2007]. This has, inter alia, resulted in an immense increase in performance on word sense disambiguation systems [Palmer et al., 2006, 2007].

1.3 Motivation

The ability to disambiguate between word senses is essential to many natural language processing applications. The two most prominent such applications are information retrieval and machine translation. Depending on the applica-

[1] These examples are taken from GermaNet 7.0 and WordNet 3.1.

tion, word sense disambiguation can be an explicit step that has been implemented on purpose or it can be an implicit task of a larger system without representing a separate step. Irrespectively of whether WSD is implemented explicitly or implicitly, it is often a necessary part of a natural language processing system. [Kilgarriff, 1997b; Ide and Véronis, 1998; Resnik, 2006; Navigli, 2009]

Word sense disambiguation has been a very active area of research in computational linguistics [Ide and Véronis, 1998; Agirre and Edmonds, 2006a; Navigli, 2009], with most of the work focusing on English [Kilgarriff and Palmer, 2000]. One of the factors that has hampered WSD research for other languages has been the lack of appropriate resources, particularly in the form of sense-annotated corpus data, which are necessary to evaluate WSD systems. The WSD competitions at SensEval and SemEval (see Subsection 2.2.4) were organized for several languages including English, Italian, Spanish, Basque, Estonian, Catalan, Japanese, Korean, Hindi, Romanian, Swedish, and Turkish [Preiss and Yarowsky, 2001; Mihalcea and Edmonds, 2004; Agirre et al., 2007]. The organization of several competitions and tasks on WSD included the preparation of the necessary resources such as sense-annotated corpora. Thus, SensEval and SemEval have helped a lot to increase research on WSD also for languages other than English. Unfortunately, no one has yet organized any tasks specifically for German WSD. In all, there are very few WSD resources for German and thus very little research on WSD for German (see Section 2.3 for an overview of related work).

With regard to a sense inventory, there are two approaches to WSD: those who use a predefined sense inventory to label word occurrences with and others who create their own sense distinctions by identifying groups of related word occurrences in corpora. Traditionally, most of the research on WSD has been using a sense inventory [Agirre and Edmonds, 2006a]. For English, the Princeton WordNet is clearly the de facto standard sense inventory [Agirre and Edmonds, 2006a; Navigli, 2009], albeit problems such as too fine-grained sense distinctions that arise with its usage are well-known. This work follows the large amount of research using a wordnet – here: the German wordnet GermaNet – as the sense inventory for WSD. The six main reasons why GermaNet

is used as the sense inventory are the following:

(i) The use of a predefined sense inventory makes it possible to evaluate automatic WSD systems against a gold standard, which is otherwise rather difficult (i.e., for unsupervised learning techniques that try to automatically create sense inventories by clustering similar word occurrences in running text).

(ii) Using a wordnet as the sense inventory is fully in line with standard practice for English where the Princeton WordNet is typically taken as the gold standard.

(iii) Existing experience for other languages on creating sense-annotated corpora and on developing systems for automatic WSD can be reused.

(iv) A sense inventory based on a wordnet makes it possible to employ the wordnet's structures, hierarchies, and relations for automatic sense disambiguation systems.

(v) The criticism on the level of granularity in the Princeton WordNet is reduced for GermaNet as it has much fewer distinct senses for a word.

(vi) Experiments by others [Saito et al., 2002] have shown the adequacy of GermaNet's sense coverage for WSD.

1.4 Contributions

The main scientific contributions of this thesis are summarized in the following. The list also includes short descriptions and links to the developed software tools and language resources that are made freely available to the research community.

- The **extension of GermaNet with sense definitions** from the German version of Wiktionary is included in GermaNet (since release 7.0) and is made freely available for download.[1]

[1] `http://www.sfs.uni-tuebingen.de/GermaNet/wiktionary.shtml`

- The **semi-automatic creation of a sense-annotated corpus** for German has resulted in the web-harvested corpus WebCAGe, which is freely available online.[1]

- The **manual sense annotation of a treebank** – in particular, the German TüBa-D/Z treebank – is freely available for academic use.[2]

- The manual **sense annotations in the deWaC corpus are updated**. Initially, Broscheit et al. [2010] manually annotated the deWaC corpus with senses from GermaNet 5.1. In the context of this thesis, these annotations have been updated for GermaNet 9.0.

- A **programming interface for the TüBa-D/Z treebank** has been implemented in order to access the treebank programmatically.

- A wide range of **knowledge-based WSD algorithms** are evaluated for German. These knowledge-based WSD algorithms are based on semantic relatedness measures.[3]

- In the context of these knowledge-based WSD experiments, the **effect of linking GermaNet with other lexical resources** such as Wiktionary is studied on the WSD performance of gloss-based relatedness methods.

- The performance and effectiveness of many different **supervised machine learning algorithms** when applied to the task of German WSD is investigated. The learning algorithms include rule-based methods, instance-based methods, probabilistic methods, support vector machines, and combined approaches.

[1] `http://www.sfs.uni-tuebingen.de/en/ascl/resources/corpora/webcage.html`

[2] The sense annotation have been included in release 9.1 of the TüBa-D/Z: `http://www.sfs.uni-tuebingen.de/en/ascl/resources/corpora/sense-annotated-tueba-dz.html`.

[3] The semantic relatedness algorithms represent a GermaNet reimplementation of Pedersen et al.'s [2005] suite of semantic relatedness algorithms for the Princeton WordNet. The reimplementation was performed by Anne Brock under the supervision of this thesis' author and is made freely available at `http://www.sfs.uni-tuebingen.de/GermaNet/tools.shtml#SemRelAPI`.

- These supervised machine learning algorithms rely on a **variety of distinct machine learning features** – including morphological information for the target word, structural information from the sentence, co-occurring words or word classes, and subcategorization information for verbs. The impact on WSD of these features is analyzed in detail.

- The **influence of syntax and semantics on WSD** is analyzed. This is particularly interesting for verbs where the syntactic structure in which a verb occurs is often highly predictive of different word senses.

- The **combination of WSD algorithms** is studied both for knowledge-based WSD and WSD using supervised machine learning methods.

- Finally, the thesis includes a **comparison of knowledge-based with supervised machine learning approaches** for the task of German word sense disambiguation and it **compares three sense-annotated corpora** for the same task.

1.5 Chapter Guide

The division of this dissertation into chapters closely follows the above outlined work program (Section 1.1). The interaction of all tasks is visualized in Figure 1.1. That is, this thesis starts in **Chapter 2** with an introduction into the task of word sense disambiguation and into the evaluation setup of WSD systems. The chapter further comprises the state of the art of WSD, in particular, it concentrates on related WSD studies that apply knowledge-based approaches and supervised machine learning approaches, which are relevant for WSD experiments in later chapters.

Chapter 3 introduces the GermaNet resource, which serves as a sense inventory for all word sense disambiguation experiments in this thesis. The chapter includes details about GermaNet's representation of word senses and lexical semantic relations, its hierarchical structure, its interlingual links to the Princeton WordNet, its information on nominal compounds, and its encoding of verbal frames.

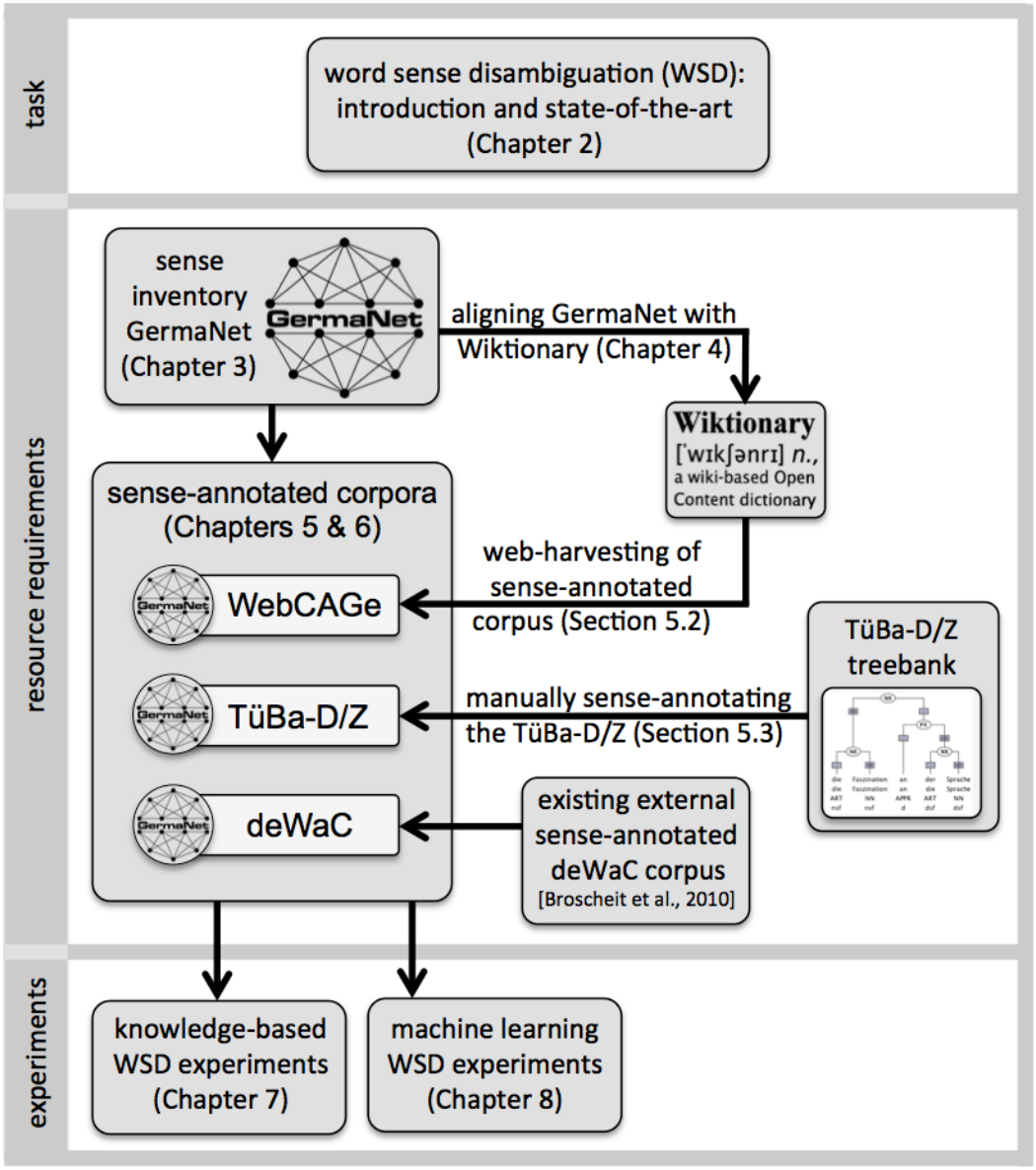

Figure 1.1: Interaction of all tasks in this dissertation.

The semi-automatic extension of GermaNet with sense definitions from Wiktionary is the content of **Chapter 4**. This chapter describes and evaluates the algorithm that maps lexical units in GermaNet to sense definitions in Wiktionary. It further discusses related work on aligning wordnets with other resources.

Since sense-annotated corpora for German were in short supply, there was a

great demand to create such corpora. In **Chapter 5**, the creation of two different sense-annotated corpora for German is described: the automatically web-harvested corpus WebCAGe (short for: *Web-Harvested Corpus Annotated with GermaNet Senses*) and the manually sense-annotated TüBa-D/Z treebank. The chapter also includes a detailed comparison of the two sense-annotated corpora.

The three sense-annotated corpora used as gold standards for the word sense disambiguation experiments in the following two chapters are described in **Chapter 6**. These corpora include the two sense-annotated corpora described in the previous chapter (i.e., WebCAGe and TüBa-D/Z) and the deWaC corpus, which was manually sense-annotated by Broscheit et al. [2010]. To allow a fair and comparable evaluation of WSD systems on different corpora, the sense inventory used for sense annotation must match for all corpora. Thus, the chapter at hand presents updated versions of the three sense-annotated corpora WebCAGe, TüBa-D/Z, and deWaC. The chapter further describes the division of gold standard corpora into training and test sets for evaluating supervised machine learning algorithms.

Chapter 7 explores a wide range of knowledge-based word sense disambiguation algorithms for German. These WSD algorithms are evaluated on the three available sense-annotated corpora WebCAGe, TüBa-D/Z, and deWaC described in the previous chapter. The algorithms are based on a suite of semantic relatedness measures, including path-based, information-content-based, and gloss-based methods, and they are applied and evaluated both separately one algorithm at a time and in combination. Experiments on gloss-based relatedness methods also employ the newly harvested definitions from Wiktionary, which are linked to corresponding GermaNet senses via the automatic mapping between GermaNet and Wiktionary described in Chapter 4.

Chapter 8 applies many supervised machine learning algorithms to the task of German WSD. These algorithms, which are taken from the Weka machine learning tool suite [Hall et al., 2009], include rule-based methods, instance-based methods, probabilistic methods, support vector machines, and combined approaches. The chapter also describes the variety of distinct machine learning features on which the supervised machine learning algorithms

rely – such as morphological information for the target word, structural information from the sentence, co-occurring words or word classes, and subcategorization information for verbs. The evaluation of the supervised WSD experiments focuses on several aspects, including a comparison of several heterogeneous machine learning algorithms, a detailed analysis of the implemented machine learning features, and an investigation of the influence of syntax and semantics on the disambiguation performance for verbs.

Finally, **Chapter 9** summarizes and concludes this dissertation with a comparison of knowledge-based WSD with WSD using supervised machine learning methods and with a comparison of sense-annotated corpora. The chapter further discusses future work for the task of German word sense disambiguation.

Chapter 2

Fundamentals and Related Work on WSD

This chapter outlines the topic of word sense disambiguation (WSD) and presents the state of the art relevant for this dissertation. It starts with an introduction into the WSD task in Section 2.1. Section 2.2 describes the evaluation setup of word sense disambiguation systems – including evaluation measures for WSD systems (Subsection 2.2.1), the WSD evaluation procedure (Subsection 2.2.2), baselines and bounds (Subsection 2.2.3), and WSD evaluation competitions (Subsection 2.2.4). Section 2.3 introduces several approaches to WSD, including knowledge-based approaches and supervised machine learning approaches. It comprises the state of the art and concentrates on related studies relevant for the disambiguation experiments in Chapters 7 and 8, which apply knowledge-based approaches and supervised machine learning approaches to German WSD.

2.1 Introduction to the Task of WSD

Word sense disambiguation (WSD) is the task of computationally assigning the most appropriate senses to words occurring in a text [Navigli, 2009; Kwong, 2012]. In stand-alone WSD systems, these senses are typically taken from a predefined sense inventory. A *sense inventory* is a collection of predefined

senses for the words of a certain language, i.e., a dictionary, a lexicon, or a wordnet. Despite some problems and criticism, e.g., on the granularity level of sense distinctions, the Princeton WordNet [Miller, 1995; Fellbaum, 1998a] is the most commonly used sense inventory for English WSD. What makes WordNet particularly popular as a sense inventory for WSD is its coverage and availability. [Agirre and Edmonds, 2006a; Ide and Wilks, 2006; Navigli, 2009]

There are several techniques for trying to solve the task of word sense disambiguation, which all share a common underlying concept: in order to disambiguate the sense of a *target word*[1] – i.e., the ambiguous word to be disambiguated – they compare linguistic clues from the target word's context with the properties of each sense of that word [Agirre and Edmonds, 2006b]. This disambiguation procedure is illustrated in Figure 2.1.

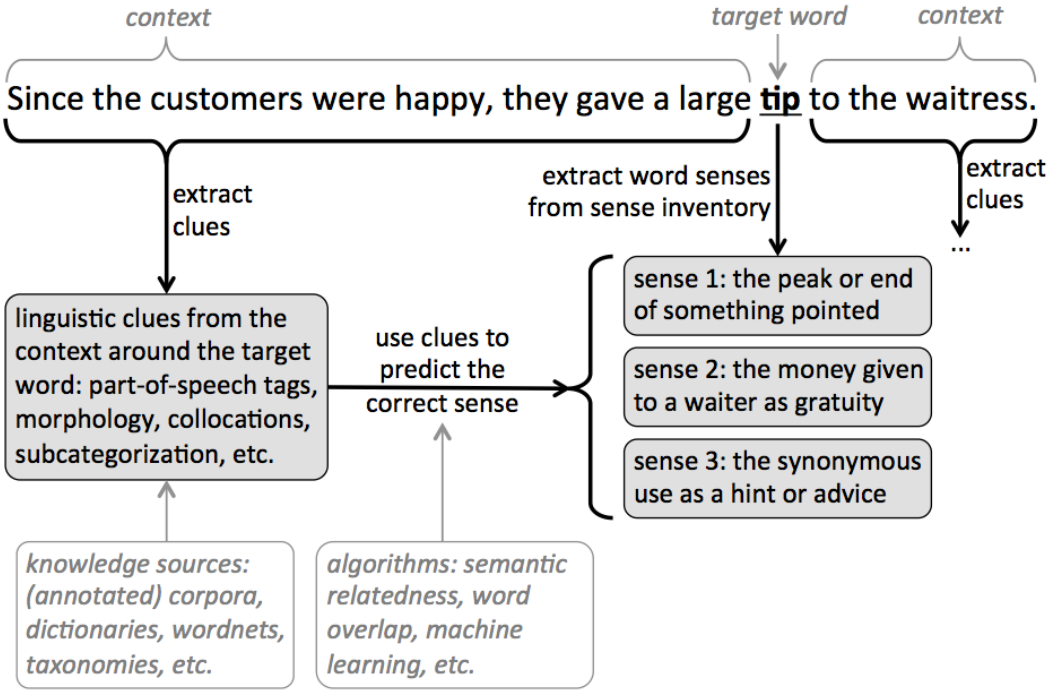

Figure 2.1: The general idea of word sense disambiguation.

The ambiguous target word in Figure 2.1 is the noun *tip*. In the example, this noun has three distinct senses, which are assumed to be taken from a

[1]Sometimes also *head word*.

predefined sense inventory: (i) the peak or end of something pointed, (ii) the money given to a waiter as gratuity, and (iii) the synonymous use as a hint or advice. The task at hand is a classification task, i.e., the goal is to automatically predict the sense that best fits the textual context in which the target word occurs.

The text around the ambiguous target word serves as the context from which linguistic clues are extracted. In the example, sentence (1) serves as the context – with the target word *tip* rendered in boldface.

(1) *Since the customers were happy, they gave a large* ***tip*** *to the waitress.*

This context is important for extracting linguistic clues. Depending on the concrete WSD approach, these clues are further enriched by automatic tools such as lemmatizers, tokenizers, or part-of-speech taggers, as well as by external knowledge sources such as dictionaries, wordnets, or corpora. The variety of information extracted from the context is generally very large, ranging from syntactic information (such as part-of-speech tags, morphology, collocations, and subcategorization) to semantic information (such as frequency distributions, selectional preferences, and semantic roles) to pragmatic information (such as domains and pragmatics) [Agirre and Stevenson, 2006].

For disambiguating the target word, the clues from the target word's context are compared with the properties of each sense of the target word to identify the sense that best fits the context in question. In the example in Figure 2.1, the second sense of the target word *tip* referring to the money given to a waiter as gratuity would be the correct target word sense for the given context. The exact way the clues are extracted from the context and used to predict the correct sense of a target word is determined by the WSD algorithm. Several WSD techniques – including knowledge-based approaches and supervised machine learning approaches – exist, which are described in more detail in Section 2.3 below.

In order to judge the disambiguation quality, WSD systems are mostly evaluated independently of a concrete application, i.e., as stand-alone WSD systems. This type of evaluation is called *in vitro* evaluation. An alternative approach is the *in vivo* evaluation, where a WSD system is evaluated in terms

of its impact as one component in another application such as a translator or a search engine. Although an in vivo setup seems to be more realistic, almost all WSD systems are evaluated in a stand-alone manner, because it is much easier to realize and evaluate. [Ide and Véronis, 1998; Palmer et al., 2006; Navigli, 2009]

The performance of stand-alone WSD systems is evaluated on sense-annotated corpora. A *sense-annotated corpus* is a text in which occurrences of words are annotated with their senses from a given sense inventory. Since sense-annotated corpora serve as *gold standards* (i.e., datasets with correct sense annotations [Kilgarriff, 1998a]) for the development, training, and evaluation of word sense disambiguation systems, their availability is a necessary prerequisite for WSD. However, sense-annotated corpora have typically been constructed manually, making the creation of such resources time-consuming and expensive and the compilation of larger data sets difficult. This problem is also referred to as the *knowledge acquisition bottleneck*.

Two variants of the WSD task are distinguished that require different kinds of sense-annotated corpora [Kilgarriff and Rosenzweig, 2000; Navigli, 2009]:

- **All-words disambiguation:** all or nearly all word occurrences in a limited size of running text need to be disambiguated with the senses of a given sense inventory. Sense-annotated corpora for this task contain sense annotations for all word occurrences, and automatic systems trying to tackle this variant of the WSD task need to be able to disambiguate all words. Due to limitations of how much text can reasonably be annotated in such an all-words, sense-annotated corpus, the resulting numbers of instances for each lemma are limited and may not be of sufficient frequency for supervised machine learning systems which rely on existing training data for each lemma.

- **Lexical sampling:** many occurrences of a selected set of ambiguous word lemmas are disambiguated with the senses of a given sense inventory. Sense-annotated corpora for this task contain sense annotations for many occurrences of the selected set of lemmas (the lexical sample), which is usually chosen in advance. The advantage of annotating

a restricted set of lemmas is that the frequency per lemma can be set appropriately for training machine learning models.

As do many existing sense-annotated corpora (see Section 5.1 for an in-depth presentation of sense-annotated corpora), the sense-annotated corpora constructed and used in this thesis (see Chapters 5 and 6) follow the lexical sample variant. The decision to annotate a lexical sample is primarily motivated by the requirements of machine learning as the intended use of the data. Such data are useful for training automatic machine learning models only if there are enough instances of each item to be classified, which cannot be assured for all words. Consequently, all WSD experiments in this thesis (see Chapters 7 and 8) are evaluated on lexical samples.

2.2 Evaluating WSD Systems

Automatic word sense disambiguation systems can be evaluated by comparing their performance to the performance of other systems. Therefore, the performance of a WSD system is typically measured in terms of a system's coverage, precision, and recall, as well as by a metric called F_1 – as introduced in Subsection 2.2.1. The procedure of how these evaluation scores can be calculated on annotated gold standard corpora is described in Subsection 2.2.2. In order to judge the performance of a WSD system, the performance range is often delimited by lower and upper bounds, which are described in Subsection 2.2.3. Besides these bounds, an even better way of evaluating WSD systems is to compare them to other systems. For this purpose, a common evaluation framework is required, which is provided by the SensEval and SemEval ventures (introduced in Subsection 2.2.4).

2.2.1 Evaluation Measures

The standard measures to evaluate the performance of WSD systems are coverage, precision, recall, and F_1. [Edmonds and Cotton, 2001; Palmer et al., 2006; Navigli, 2009; Kwong, 2012]

The *coverage* of a WSD system is calculated as the number of instances where the WSD system returns a result, compared to the overall number of instances in the test set used for evaluation:

$$coverage = \frac{\#\text{ predictions provided by system}}{\#\text{ total instances in test set}} \tag{2.1}$$

Recall is defined as the percentage of word instances that are correctly disambiguated by the WSD system, out of all annotated word instances in the test set on which the system is evaluated:

$$recall = \frac{\#\text{ correct predictions by system}}{\#\text{ total instances in test set}} \tag{2.2}$$

In WSD terminology, the terms *recall* and *accuracy* are used synonymously [Palmer et al., 2006; Navigli, 2009], i.e., the formula is the same.

Precision is reported as the percentage of word instances that are correctly disambiguated by the WSD system, out of the word instances addressed by the system:

$$precision = \frac{\#\text{ correct predictions by system}}{\#\text{ predictions provided by system}} \tag{2.3}$$

The definitions above imply that for a perfect coverage of 100%, the values of precision and recall are identical, while for a coverage below 100%, recall and precision correlate insofar that it is possible to increase one while decreasing the other. The goal is to find the optimal trade-off between the two measures. For this purpose, the F_1-*measure*, also referred to as the balanced F-score, is introduced, which represents the weighted harmonic mean of recall and precision:

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall} \tag{2.4}$$

For a coverage of 100%, the value of F_1 is equal to the values of recall and precision. Thus, this measure is particularly useful for systems that do not achieve perfect coverage.

2.2.2 Evaluation Procedure

Word sense disambiguation systems such as knowledge-based systems that are designated for disambiguating all words and that do not need any training material are usually evaluated on all sense annotations available in a gold standard (i.e., a sense-annotated corpus). Thus, the knowledge-based WSD experiments in Chapter 7 are evaluated on all available annotations.

By contrast, supervised machine learning systems require a certain amount of annotated instances per lemma for training. The gold standards for evaluating supervised WSD systems need to provide a certain amount of annotations per lemma and ideally – though seldom the case because this is unrealistic – also a certain amount of annotations per word sense. Thus, the supervised WSD experiments in Chapter 8 are evaluated on a subset of annotations that fulfill certain criteria (explained in Section 6.1).

In general, there are two main approaches to evaluate the performance of supervised machine learning algorithms: evaluating on a separate test set or evaluating by cross-validation. For the evaluation on a separate test set, some annotations (referred to as the *training set*) are used for training a supervised system; and another distinct portion of the annotations (the *test set*) are held back while tuning the system on the training data, and later used to obtain final evaluation results. As a drawback of separate, non-overlapping training and test sets – particularly prevalent when the amount of annotated data is limited – the evaluation results strongly depend on the specific split of the data into training and test samples, i.e., there is a large variance [Refaeilzadeh et al., 2009]. However, to meaningfully and accurately estimate an algorithm's ability to generalize, testing on unseen data is crucial. [Palmer et al., 2006; Witten et al., 2011]

The alternative approach to evaluate supervised machine learning systems is by cross-validation. *Cross-validation* partitions all available gold standard annotations into k equally-sized sets. The process of training and evaluation is repeated k times, where each of the k sets serves once as the test set: $k-1$ sets constitute the training portion and the single left-out set is used as the test set to calculate coverage, recall, precision, and F_1. The most common

number k for partitioning all annotations is 10, which is also referred to as 10-fold cross-validation. For the final result, the evaluation metrics obtained for the k repetitions are averaged. [Weiss and Kulikowski, 1991; Refaeilzadeh et al., 2009]

Although evaluation by (repeated) cross-validation on all available data could helps to lower the above-mentioned variance problem occurring for the evaluation with separate training and test sets, there is a major advantage in using a separate, unseen test set: it allows a more realistic estimate of the system's ability to continue to generalize after several experiments with distinct classifiers, parameters, and features on the training set. The supervised machine learning experiments in Chapter 8 are evaluated on a completely unseen test set (Section 6.1 explains the division of gold standard corpora into training and test sets). This evaluation procedure is in line with most related work on supervised disambiguation of a lexical sample, e.g., Hoste et al. [2002a] and Mohammad and Pedersen [2004], and used for many SensEval and SemEval tasks [Kilgarriff and Rosenzweig, 2000; Edmonds and Cotton, 2001; Mihalcea and Edmonds, 2004; Agirre et al., 2007]. It also follows the WSD studies that make use of these SensEval/SemEval datasets, including Lee and Ng [2002], Dinu and Kübler [2007], and de Oliveira et al. [2011] – to name only a few.

There are two approaches for how to calculate overall numbers for coverage, recall, precision, and F_1: by *micro-averaging* over all annotations used for testing or by *macro-averaging* over all lemmas, i.e, calculating according average numbers for each lemma and then averaging the numbers for the lemmas. [Lee and Ng, 2002; Maarouf et al., 2014] While micro-averaging gives equal weight to each annotated instance, which reflects the distribution of occurrences per lemma, macro-averaging gives equal weight to each lemma, which results in a reduced impact of lemmas with many occurrences in favor of lemmas with few occurrences. All results in this dissertation are reported as micro-averaged – as in many other studies, including Patwardhan et al. [2003], Lee et al. [2004], Torres and Gelbukh [2009], and Wiriyathammabhum et al. [2012].

2.2.3 Baselines and Bounds

In order to judge the performance of a WSD system, the performance range is often delimited by lower and upper bounds. The lower bound is typically represented by a baseline, which is a very simple, commonly used disambiguation algorithm, and which is assumed to be outperformed by more elaborate WSD algorithms. Comparing the performance of a WSD system to the performance of a baseline allows estimating the impact and efficiency of a WSD system. The two most commonly employed lower baselines are the *random sense baseline* and the *most frequent sense baseline*. [Navigli, 2009]

The *random sense baseline* randomly selects – for each target word occurrence – exactly one sense from the set of corresponding senses in the sense inventory for the target word lemma. It is generally a good indicator of the difficulty of the task at hand, and is used in both Chapters 7 and 8 on knowledge-based and supervised machine learning WSD experiments.

The *most frequent sense baseline*[1] always (i.e., for each target word occurrence) assigns the sense which has most occurrences in the annotations. [Gale et al., 1992a; Navigli, 2009; Preiss et al., 2009] This baseline is often difficult to beat by WSD systems – especially for skewed sense distributions. Since a separate sense-annotated corpus or training set is required to determine most frequent senses, the most frequent sense baseline is used for comparing WSD using supervised machine learning methods (Chapter 8) rather than knowledge-based methods.[2]

On the other side of the performance scale, the upper bound is often represented by the *inter-annotator agreement* (IAA)[3], which is the percentage of sense annotations where human annotators agree on the annotated sense(s) [Gale et al., 1992a]. IAA is calculated on sense annotations that have been independently performed by two (or more) human annotators.

[1] Sometimes also *most frequent sense heuristic* [Miller et al., 1994; Mihalcea, 2006] or *first sense baseline/heuristic* [McCarthy, 2009; Navigli, 2009].

[2] This procedure follows the approach by Kilgarriff and Rosenzweig [2000] for SensEval, who state that *"baselines which use training data are intended for comparison with supervised systems"* [Kilgarriff and Rosenzweig, 2000, page 28].

[3] Sometimes also *inter-tagger agreement* (ITA).

2.2.4 SensEval and SemEval Competitions

Word sense disambiguation systems were developed since the 1950s [Ide and Véronis, 1998]. However, since the necessary resources such as sense inventories and sense-annotated gold standards for evaluating WSD systems were not publicly available, the WSD systems were all evaluated on different sense inventories and corpora. It was not objectively possible to judge which system performed better with which parameter adjustments and, generally, to compare the results of published works on WSD. Towards the end of the 1990s, a strong interest in the possibility to compare WSD systems in a coherent evaluation setup arose. [Gale et al., 1992a; Resnik and Yarowsky, 1997, 1999; Kilgarriff and Palmer, 2000]

In 1998, the first *open, community-based evaluation exercise for Word Sense Disambiguation programs*, called *SensEval*, was organized [Kilgarriff and Palmer, 2000, page 1]. Further SensEvals followed in 2001 [Preiss and Yarowsky, 2001] and 2004 [Mihalcea and Edmonds, 2004], and – with a broader spectrum of semantic analysis tasks besides WSD and, thus, under the new name of *SemEval* – in 2007 [Agirre et al., 2007], 2010 [Erk and Strapparava, 2010], 2012[1] [Agirre et al., 2012], 2013 [Manandhar and Yuret, 2013], and 2014 [Nakov and Zesch, 2014].[2] The main motivation of these competition workshops was to provide a common setup for evaluating automatic systems: all WSD systems were supposed to disambiguate between senses of the same sense inventory and were evaluated on the same sense-annotated gold standard corpora – including a common scoring system [Melamed and Resnik, 2000]. Such a coherent framework made it possible to compare WSD systems with each other.

The SensEval and SemEval workshops included tasks on all-words disambiguation [Palmer et al., 2001; Navigli et al., 2007], lexical sample disambiguation [Kilgarriff, 2001; Mihalcea et al., 2004], cross-lingual disambiguation [Lefever and Hoste, 2010, 2013], and multilingual disambiguation [Navigli

[1]The SemEval workshop in 2012 did not include a specific task on word sense disambiguation. It is rather listed for the sake of completeness.

[2]For simplicity, the first three competitions are often referred to as SensEval-1, SensEval-2, and SensEval-3 throughout this thesis, while the SemEval workshops are referenced with the corresponding year, i.e., SemEval-2007, SemEval-2010, etc. This notation is common in the literature.

et al., 2013]. Since several WSD competitions were organized for many different languages including English, Italian, Spanish, Basque, Estonian, Catalan, Japanese, Korean, Hindi, Romanian, Swedish, and Turkish [Preiss and Yarowsky, 2001; Mihalcea and Edmonds, 2004; Agirre et al., 2007], the workshops helped a lot to increase research on WSD for many languages. The organization of SensEval and SemEval also included the preparation and distribution of the necessary resources such as sense-annotated corpora. Unfortunately, no one has yet organized any tasks specifically for German WSD.[1]

2.3 WSD Approaches and State of the Art

Word sense disambiguation has been a very active area of research in computational linguistics, from the 1950s up to recent years [Ide and Véronis, 1998; Agirre and Edmonds, 2006a; Navigli, 2009]. Most of the work on WSD has focused on English. [Kilgarriff and Palmer, 2000] One of the factors that has hampered WSD research for other languages has been the lack of appropriate resources, particularly in the form of sense-annotated corpus data. Since such sense-annotated corpora serve as gold standards for the development, training, and evaluation of word sense disambiguation systems, it is not surprising that there has been a steady progress in the development and in the performance of WSD algorithms for languages such as English for which large sense-annotated corpora are available and considerably less on languages with a shortage of such corpora. In recent years, the WSD competitions at SensEval and SemEval boosted research on WSD for many languages by organizing comparative WSD evaluation exercises including the preparation of the necessary resources (see Subsection 2.2.4 above for more details on SensEval and SemEval). [Agirre and Edmonds, 2006a; Navigli, 2009]

This section comprises related work on the topic of WSD, so as to be able to relate the research reported in this dissertation to previous research on this

[1]The more recent tasks on cross-lingual disambiguation [Lefever and Hoste, 2010, 2013] and multilingual disambiguation [Navigli et al., 2013] include German as one of several target languages. However, since these tasks focus on multi- and cross-lingual perspectives rather than explicitly on German, they did not provide monolingual sense-annotated corpora for German.

topic.[1] There are a variety of techniques for trying to solve the task of word sense disambiguation. Following the general WSD terminology (e.g., Agirre and Edmonds [2006a], McCarthy [2009], and Navigli [2009]), the techniques are grouped into (i) knowledge-based approaches and (ii) machine learning approaches, where the machine learning approaches are in turn distinguished into (ii-a) supervised, (ii-b) unsupervised, and (ii-c) semi-supervised.

(i) Knowledge-based approaches mainly use dictionaries or wordnets as knowledge sources to disambiguate between word senses. They calculate word overlaps or semantic relatedness between different words and word senses to predict the correct sense for a given context. Chapter 7 applies a wide range of knowledge-based word sense disambiguation algorithms to German – see Section 2.3.1 below for more details and an overview of related works.

(ii) All machine learning approaches have in common that they use corpora as knowledge sources, but they are different in the exact task they perform. Machine learning approaches are broadly distinguished into supervised and unsupervised, with semi-supervised and minimally supervised approaches having an intermediate status between the two.

(ii-a) Supervised approaches to WSD adapt supervised machine learning methods to solve the task of assigning the correct sense to a word. The task at hand is considered as a classification problem, where the *class* that needs to be predicted is the corresponding word sense (from a given sense inventory). Therefore, supervised machine learning systems use sense-annotated corpora as knowledge sources to train supervised classification algorithms how to predict the correct sense from a given sense inventory. Chapter 8 explores many supervised machine learning algorithms for disambiguating

[1] This dissertation concerns two further topics besides the proper task of WSD, namely the alignment of GermaNet to the web-based dictionary Wiktionary as well as the creation of sense-annotated corpora. Related work on these topics is placed in the corresponding chapters. More specifically, related work on aligning wordnets to Wiktionary is covered in Subsection 4.6 and related work on sense-annotated corpora is discussed in Subsection 5.1.

German word senses – see Section 2.3.2 below for an introduction and an overview of the state of the art.

(ii-b) Unsupervised machine learning approaches to WSD aim to induce word senses by clustering word occurrences with similar contexts in unannotated corpora. This induction of word senses is independent of a predefined sense inventory. It is also known as *word sense discrimination*. Note that the aim of unsupervised machine learning approaches, i.e., to discriminate between word senses independently of a sense inventory, differs from the aim of the proper WSD task (as defined in Section 2.1), which is to disambiguate between senses of a predefined sense inventory. [Schütze, 1998; Pedersen, 2006; Navigli, 2009]

The advantage of unsupervised machine learning approaches is that – since they use unannotated corpora for clustering word senses – they do not depend on sense-annotated corpora and are, thus, not affected by the knowledge acquisition bottleneck (see Section 2.1 above). However, the downside is that the evaluation of the word sense induction task is much more difficult than the WSD classification task. The reason for this difficulty is that there are no clear criteria on how to judge the quality of word sense clusters [Navigli, 2009]. Mainly due to this difficulty, the thesis at hand neglects word sense discrimination methods and focuses on the task of word sense disambiguation.

(ii-c) Semi-supervised or minimally supervised machine learning approaches to WSD aim to disambiguate between word senses with very little sense-annotated data. That is, they tackle a classification task (such as the supervised machine learning methods), but they try to overcome the knowledge acquisition bottleneck by using unannotated corpora – together with small amounts of annotated corpora. Therefore, semi-supervised or minimally supervised machine learning approaches to WSD have an intermediate status between supervised and unsupervised machine learning approaches.

> [Chapelle et al., 2006; Navigli, 2009; Zhu and Goldberg, 2009]
>
> The idea of semi-supervised machine learning approaches is to increase the impact of the small amount of available annotated training data by automatically harvesting new annotations. The two most popular such approaches are bootstrapping methods (e.g., Yarowsky [1995] and Mihalcea [2004]) and methods that rely on 'monosemous relatives' (e.g., Leacock et al. [1998], Mihalcea and Moldovan [1999], and Agirre and Lopez de Lacalle [2004]). Both these methods are explained in Section 5.1.3, where related work on automatically sense-annotated corpora is covered.[1]

Since it would be beyond the scope of this thesis to account for all existing studies and to provide a comprehensive introduction to all possible approaches, this section has a specific focus on related methods and studies on knowledge-based approaches and supervised machine learning approaches to WSD, which are relevant for the WSD experiments in Chapters 7 and 8, as well as on previous studies who worked on WSD for German.[2] All previous work covered in Subsections 2.3.1 and 2.3.2 focus on English – if not otherwise stated – while Subsection 2.3.3 describes related studies on German WSD. The studies described are compared and related to the WSD experiments in this thesis. Further detailed comparisons are provided at the appropriate positions in Chapters 7 and 8.

2.3.1 Knowledge-Based Approaches to WSD

Knowledge-based[3] WSD techniques make use solely of available lexical resources including dictionaries or wordnets. They do not use annotated train-

[1] The automatic web-harvesting of the sense-annotated WebCAGe corpus, which is described in Section 5.2, can be seen as one step of such a semi-supervised system. That is, the automatic web-harvesting can be regarded as one approach to automatically harvest sense annotations. However, the application of a proper bootstrapping method has to be left to future work.

[2] The interested reader is pointed to Ide and Véronis [1998], Agirre and Edmonds [2006a], and Navigli [2009] for a broader range and more in-depth surveys of WSD methods and studies.

[3] Sometimes also *dictionary-based* – depending on the context.

ing data but rather use linguistic clues such as word overlaps with definitions, selectional restrictions, or similarity between two words in a knowledge base in order to tackle the WSD task. Since these linguistic clues are generally not restricted to certain word classes, knowledge-based systems are often applied to disambiguate all words in a running text. This high coverage is the major strength of knowledge-based systems as compared to supervised machine learning systems (see Subsection 2.3.2 below), which are usually applicable to only a restricted set of lemmas for which sense-annotated training material is available. On the other hand, for this restricted set of lemmas, supervised systems usually outperform knowledge-based systems. [Mihalcea, 2006; McCarthy, 2009; Navigli, 2009]

Since the list of knowledge-based methods is long, this subsection concentrates on works and studies relevant for the knowledge-based WSD experiments of this thesis (Chapter 7). Chapter 7 explores a wide range of knowledge-based word sense disambiguation algorithms for German, including word overlap methods and semantic relatedness measures. The most similar related works are those by Patwardhan et al. [2003] and Pedersen et al. [2005] – as described in the following paragraphs.

Gloss-Based Word Overlap

Lesk [1986] introduced a word sense disambiguation algorithm which operates on the assumption that words occurring together in a text tend to share common words in their definitions of a dictionary. The *Lesk algorithm* assigns an ambiguous target word the sense whose definition in a dictionary has most word overlaps with the dictionary definitions of the words in the context of that target word. The two main underlying assumptions to this algorithm (recorded by Banerjee and Pedersen [2003, page 806]) are: (i) words occurring together in a text tend to be used in related senses and (ii) two senses are more related the more words their definitions have in common.

Lesk himself used the Oxford Advanced Learner's Dictionary of Current English as the source of sense definitions, but generally any semantic resource that provides sense definitions can be used. For instance, the studies by Kil-

garriff and Rosenzweig [2000], Banerjee and Pedersen [2002, 2003], Vasilescu et al. [2004], Torres and Gelbukh [2009], Ponzetto and Navigli [2010], and Miller et al. [2012] outlined below all use WordNet as the resource of sense definitions.

The original Lesk algorithm has two main problems: (i) the number of comparisons increases exponentially when more than two words are being compared and (ii) dictionary definitions are often sparse, which results in insufficient word overlaps and low coverage of the WSD algorithm. This is why there are very few studies (among them Vasilescu et al. [2004] and Torres and Gelbukh [2009]) applying the Lesk algorithm in its original version.

Several variants of the Lesk algorithm have been proposed to overcome these problems and to improve the algorithm's disambiguation performance. The most prominent approach to overcome the computational complexity problem for the comparison of more than two words is the *simplified Lesk algorithm* [Kilgarriff and Rosenzweig, 2000]: each word is disambiguated individually by comparing its sense definition directly with the context (rather than with the sense definitions of each word in the context) [Mihalcea, 2006]. Almost all studies using any variant of the Lesk algorithm apply this simplification strategy (with the only known exception of Torres and Gelbukh [2009] – as mentioned above). The study by Vasilescu et al. [2004], which compared several Lesk variants on the all-words dataset of SensEval-2, showed that this simplified variant is much more efficient and precise than the original Lesk algorithm.

Among other Lesk variants, the simplified Lesk algorithm was used as a baseline at the first two SensEval competitions. While hardly any (non-supervised) WSD system was able to beat it at SensEval-1, many WSD systems beat it at SensEval-2. [Kilgarriff and Rosenzweig, 2000; Kilgarriff, 2001]

The most prominent approach to overcome the problem of sparse dictionary definitions is the *adapted Lesk algorithm* [Banerjee and Pedersen, 2002, 2003]. This variant extends the original algorithm in two main respects: (i) the overlap calculation includes definitions of related synsets – based on the underlying idea that two synsets are more related, the more overlaps their definitions and the definitions of corresponding related words have – and (ii) overlaps con-

sisting of sequences of words get higher scores. On the SensEval-2 lexical sample dataset, Banerjee and Pedersen [2002, 2003] achieved second best results compared to the SensEval-2 participating systems, which is about double the accuracy compared to the original Lesk algorithm.

Basile et al. [2007] proposed to use individual disambiguation strategies for each word class and applied the adapted Lesk algorithm to disambiguate adjectives and adverbs. They found that this word class-specific approach achieves higher performance compared to the application of the same WSD algorithm to all word classes.

Other popular approaches to overcome the problem of sparse dictionary definitions are by automatic harvesting of semantic relations [Ponzetto and Navigli, 2010] and of distributionally similar words [Miller et al., 2012]. Ponzetto and Navigli [2010] mapped WordNet to Wikipedia in order to incorporate Wikipedia relations in addition to WordNet relations when applying the adapted Lesk algorithm. On the Semeval-2007 coarse-grained all-words WSD task [Navigli et al., 2007], Ponzetto and Navigli [2010] achieved performance comparable to the best supervised systems on this task. Miller et al. [2012] extended sense definitions and contexts with distributionally similar words taken from an automatically created distributional thesaurus. Their approach overcame the problem of sparse sense definitions and significantly improved the results of the adapted Lesk algorithm.

Another set of proposed variations on the original Lesk algorithm concerns the way two sense definitions (or, for the adapted Lesk algorithm, a sense definition and the context) are compared. Lesk suggested to merely count the words that two sense definitions have in common. Kilgarriff and Rosenzweig [2000] do not simply count the number of words that a sense definition has in common with the context of the target word, but calculate a sum based on the inverse document frequency of each word in common. This inverse document frequency represents the likelihood of a word occurring in an arbitrary sense definition. Ramakrishnan et al. [2004] compute the cosine similarity between the inverse document frequency vector weighted by the term frequency. Yet another approach to computing the overlap was proposed by Basile et al. [2014]. They used a word similarity function defined on a distributional semantic space

to calculate the overlap between a sense definition and the context.

The WSD experiments in Chapter 7 apply a variant of the adapted Lesk algorithm. The adapted Lesk variant is preferred since it overcomes the problem of sparse sense definitions from which the original Lesk algorithm suffers [Banerjee and Pedersen, 2002, 2003]. Due to the low coverage of GermaNet's sense definitions, this sparsity problem is especially prevalent for the German wordnet and – without including related synsets for calculating overlaps – prevents any reasonable Lesk-like disambiguation.

Although all above-cited studies employing the simplified or the adapted Lesk variant, or both variants in combination, are similar to the experiments reported in Chapter 7, the main differences lie in the language (English vs. German), and, consequently, in the lexical resource from which sense definitions are taken (mostly WordNet for English vs. GermaNet for German) and in the gold standard datasets (English vs. German sense-annotated corpora) used for the evaluations.

A further commonality between Chapter 7 and Ponzetto and Navigli [2010] is that both enrich a wordnet with sense definitions in order to overcome the problem of sparse definitions. Therefore, both Chapter 7 and Ponzetto and Navigli [2010] map a wordnet in question to a web-based, collaboratively constructed lexical resource. While Ponzetto and Navigli [2010] used a mapping of the Princeton WordNet and Wikipedia to enrich WordNet with relations from Wikipedia, Chapter 7 uses a mapping of GermaNet to the German Wiktionary to enrich GermaNet with sense descriptions from Wiktionary.

However, even more similar studies compared to the knowledge-based WSD experiments in Chapter 7 are those using a set of semantic relatedness measures rather than only the Lesk algorithm. These algorithms are described in the next paragraph.

Measures of Semantic Relatedness

To take up and continue the approach outlined in the previous paragraph, gloss-based word overlap methods are to be considered as a kind of semantic relatedness measure. In their work on the adapted Lesk algorithm outlined

in the previous paragraph, Banerjee and Pedersen [2003] noted that the more words the definitions of two senses have in common, the more related these senses are. In the continuation of their work, Patwardhan et al. [2003] realized that the measure used for determining semantic relatedness in the disambiguation process can be substituted with any semantic relatedness measure. They extend their WSD experiments to several further semantic relatedness measures [Pedersen et al., 2005], including:[1]

lch: Similarity between two concepts is computed as the negative logarithm of the length of the shortest path between the concepts (limited to hypernymy/hyponymy relations only) over the path length of the overall depth of the wordnet – as introduced by Leacock and Chodorow [1998].

wup: Conceptual relatedness between two concepts is computed as the shortest path length (limited to hypernymy/hyponymy relations) between the two concepts, normalized by the depth of their lowest common subsumer – as introduced by Wu and Palmer [1994].

hso: For computing the semantic relatedness between two concepts, the length of the shortest path between the concepts (not limited to the hypernymy/hyponymy relations) and the change of 'direction' (i.e., the relations in a wordnet can be grouped into upwards, downwards, and horizontal) are considered – as introduced by Hirst and St-Onge [1998].

res: Similarity between two concepts is computed as the information content of their lowest common subsumer in the graph – as introduced by Resnik [1995].

jcn: Similarity between two concepts is computed as the inverse of their distance (measured as the sum of the information contents of the concepts minus double the information content of their lowest common subsumer) – as introduced by Jiang and Conrath [1997].

[1] Since this suite of semantic relatedness measures is used in the experiments of Chapter 7 and more details on the semantic relatedness measures are provided in Subsection 7.1, the descriptions are deliberately kept brief at this point.

lin: Similarity between two concepts is measured as the information content (multiplied by two) of their lowest common subsumer over the sum of the information contents of the concepts – as introduced by Lin [1998].

Since the experiments in Chapter 7 use the same suite of semantic relatedness algorithms (reimplemented for German), the works by Patwardhan et al. [2003] and Pedersen et al. [2005] represent the most similar research studies. Their experiments – and analogously also the experiments in this thesis – are manifold: for instance, experiments are performed with different relatedness measures, different word classes, and different context window sizes. Several of their main findings are generally confirmed by the experiments in Chapter 7 for German: for example, (i) larger context windows improve performance, (ii) word overlap methods based on the adapted Lesk algorithm perform consistently well, and (iii) most semantic relatedness measures other than word overlap measures are ill-suited for disambiguating adjective and verb senses. A more thorough comparison of the knowledge-based WSD experiments in this thesis with those from Patwardhan et al. [2003] and Pedersen et al. [2005] are provided throughout Chapter 7.

Besides the different languages under investigation, the major extension of this thesis over the work of Patwardhan et al. [2003] and Pedersen et al. [2005] is the combination of several semantic relatedness measures in a simple *majority voting* scheme and in a *Borda count* setup. These combinations obtain better overall results compared to the single measures applied. There are no equivalent studies investigating such combined algorithms on a set of semantic relatedness measures.[1]

Several approaches to build a combined knowledge-based WSD system used separate measures depending on the word class. Basile et al. [2007], for example, used separate disambiguation strategies for each word class (including the measures by Lesk [1986] and Resnik [1995]) – based on the assumption that the disambiguation performance strongly depends on the word class. Torres

[1]There are several studies investigating combined WSD systems, including Florian et al. [2002], Florian and Yarowsky [2002], and Klein et al. [2002]. But since (almost) all of these studies combine supervised algorithms rather than semantic relatedness algorithms, they are outlined in the following subsection.

and Gelbukh [2009] experimented with the semantic relatedness measures by Lesk [1986], Jiang and Conrath [1997], and Lin [1998] as well as the combination of these measures that also used different kinds of relatedness measures for different classes of words.

2.3.2 Supervised Machine Learning Approaches to WSD

While WSD systems based on supervised machine learning (ML) methods typically obtain far better results than knowledge-based systems (described in the previous subsection), they presuppose the availability of training data and thus their coverage is restricted to those lemmas for which training data is available [Màrquez et al., 2006; Mihalcea, 2006; Navigli, 2009]. This subsection describes several supervised ML approaches with a special focus on works and studies relevant for the supervised WSD experiments in Chapter 8 so as to be able to relate the research reported in this dissertation to previous research.

Supervised approaches to WSD adapt supervised machine learning methods to solve the task of assigning the correct sense to a word. The task at hand is considered as a classification problem, where the *class* that needs to be predicted is the corresponding word sense (from a given sense inventory). Since the sets of word senses and thus the sets of classes to be predicted differ for each lemma, training and classification of supervised WSD systems are usually (e.g., Ng and Lee [1996], Veenstra et al. [2000], Hoste et al. [2002a], Màrquez et al. [2006], and Dinu and Kübler [2007]) performed separately for each lemma[1] – as it is done in this thesis (Chapter 8). This separate classification of each word lemma is also referred to as *word-experts* [Berleant, 1995].

In order to learn how to predict the corresponding word senses for unseen words, supervised classification algorithms – also referred to as *classifiers* – rely on corpora whose words are already annotated with senses from a given sense inventory. Each such annotated word occurrence is denoted as an *instance*. A certain amount of sense annotations serves for training a supervised method how to predict the correct senses for unseen word occurrences; and

[1]Note that some studies use an alternative approach of training and classifying word senses of all lemmas at once (for example, de Oliveira et al. [2011] or Kawahara and Palmer [2014]).

another set of sense annotations is used to evaluate the performance of the automatic disambiguation prediction. Since these sense-annotated corpora have usually been constructed manually, their availability is restricted and their construction expensive.

The contexts of these annotated words provide linguistic clues specific to particular senses. Supervised WSD systems use these clues – referred to as *features* – in order to disambiguate between word senses. That is, a feature is a distinct bit of information that encodes linguistic clues from the context of a target word [McCarthy, 2009] – such as morphological information for the target word, structural information from the sentence, or co-occurring words or word classes. The values of the set of features are specific to the word instance they represent. Some in the literature frequently used machine learning features include:

Co-occurrence: The occurrence of words in the context of the target word is often used to encode ML features. Probably because these features have proven to perform well, are still easy to obtain and applicable to all words and word classes, they are popular in the literature [Lee and Ng, 2002; Martínez et al., 2002; Mihalcea, 2002b; Kopeć et al., 2012]. This type of feature is also used in the WSD experiments in Chapter 8 (see Subsection 8.1.3).

Parts of speech: Similarly, information on the parts of speech (POS) of those words that occur in the context of the target word are available to all target words. Machine learning features that encode POS information on context words or on the target word itself are also often used features [Lee and Ng, 2002; Martínez et al., 2002; Mihalcea, 2002b; Kübler and Zhekova, 2009; Kopeć et al., 2012]. Subsection 8.1.4 describes the implementation of POS features for the WSD experiments in Chapter 8.

Syntax: Several popular features encode information on predicate-argument structures and syntactic relations of the target word [Fellbaum et al., 2001; Florian et al., 2002; Lee and Ng, 2002]. See Subsections 8.1.8 and 8.1.9 for more details on how this kind of information is employed

as features in the supervised WSD experiments in this thesis.

Several studies investigated the impact of individual features, either manually [Hoste et al., 2002c; Lee and Ng, 2002; Yarowsky and Florian, 2002; Bas et al., 2008] or with the help of automatic feature selection algorithms [Mihalcea, 2002a,b; Le and Shimazu, 2004; Kopeć et al., 2012]. Chapter 8 also implements many different types of features and investigates an algorithm for automatic feature selection to identify the set of features with the highest positive impact on the WSD performance.

Supervised classification algorithms take these features from sense-annotated training data to train a classifier how to predict the correct senses for unseen instances. Several supervised machine learning methods exist, which are different in the exact way the features are used to identify the correct sense of a target word. The most popular ML algorithms in the literature employ different underlying classification theories, which mainly include methods based on decision rules, instance-based approaches, probabilistic approaches, support vector machines, and combined approaches. In previous WSD studies that used supervised ML approaches to disambiguate word senses, the same set of heterogeneous algorithms is prevalent for the task at hand. However, the three predominant and most effective single supervised ML algorithms for the task of WSD seem to be instance-based algorithms, support vector machines, and naive Bayes.

Due to their popularity as supervised machine learning methods – in general and for the WSD task – Chapter 8 explores all of the above-mentioned, popular supervised ML algorithms for solving the task of disambiguating German word senses. That is, Chapter 8 applies rule-based methods, instance-based methods, probabilistic methods, support vector machines, and combined approaches – all of which are described in the following subsections. Unlike for the knowledge-based WSD experiments, where the most similar research project was clearly identified (see Section 2.3.1 above), there are distinct aspects of the supervised machine learning experiments that need to be related to multiple previous studies. The most similar studies are probably those which implement and assess a wide range of features (including Hoste et al. [2002c],

Mihalcea [2002a,b], Le and Shimazu [2004], and Kopeć et al. [2012]) or evaluate and compare a wide range of supervised ML algorithms (including Mooney [1996], Zavrel et al. [2000], Pedersen [2001], Agirre and Martínez [2004], Joshi et al. [2006], and Màrquez et al. [2006]) or both of these (including Lee and Ng [2002], Yarowsky and Florian [2002], Bas et al. [2008], Kopeć et al. [2012], and Wiriyathammabhum et al. [2012]).

Decision Rules

Supervised classification algorithms based on decision rules are among the simplest to understand and implement. The rules use conditions to discriminate between the classes to be predicted. For the task at hand, the rules discriminate between the senses of the ambiguous target words. The conditions encode specific values of the features representing the target words' contexts (usually only a certain subset of the features is included in a classification model [Mitchell, 1997]). The three classification methods in this paragraph differ from each other in the way how they create, represent, and apply these rules.

Decision lists [Rivest, 1987] are ordered lists of rules combining conditions (for the WSD task, conditions correspond to features) with values (i.e., word senses). The order of the rules is relevant for classification: a new instance is sequentially compared to the rules in the list; the assigned word sense is the one from the first rule whose condition is fulfilled by the instance. In this way, decision lists can be though of as extended *if* – *then* – *elseif* – . . . *else* – rules, where highly discriminating rules are at the top of the list and more general rules at the bottom, with the very last rule being the default case. [Rivest, 1987]

There are two main approaches for constructing decision lists: directly from training data [Yarowsky, 1994] or indirectly from a decision tree (see below). In the approach by Yarowsky [1994], feature–sense pairs are collected from training data and sorted by the log-likelihood probabilities of senses having the given feature values.

Besides Yarowsky [1994, 1995, 2000], many other studies – including Mooney

[1996], Paliouras et al. [2000], Martínez et al. [2002], Agirre and Martínez [2004], and Màrquez et al. [2006] – applied decision lists to the task of word sense disambiguation. Decision lists proved successful in the first SensEval workshops [Navigli, 2009].

Decision tables [Kohavi, 1995] represent rules in tabular forms: a set of conditions (for the WSD task, the features – as for decision lists) is contrasted to a set of resulting values (here, again, the word senses). That is, for the WSD task, columns correspond to features and rows represent instances with specific values for these features and assigned word senses. For constructing a decision table, the features and instances are extracted from the training data. Algorithms for constructing decision tables do usually not include every feature and every instance from the training data, but rather a reduced set. During the classification process, a new instance is compared to all feature combinations stored in the decision table, and, if there is at least one matching instance, the assigned word sense is the one which is stored with the majority of these matching instances. [Kohavi, 1995]

Decision tables were previously applied to the task of word sense disambiguation [Paliouras et al., 2000; Pancardo-Rodríguez et al., 2005; Bas et al., 2008; Kopeć et al., 2012], albeit much less often than decision lists (see above) or decision trees (read on).

Decision trees [Quinlan, 1993] encode classification rules as branching structures. Each branching node models a rule condition that partitions the data into subsets; for the WSD task, each such node represents a feature and each of the branches represents a feature value. The terminal nodes in a decision tree depict the classes to be predicted; for the task at hand, the terminals depict the word senses. [Quinlan, 1993; Mitchell, 1997; Pedersen, 2001]

Most decision tree algorithms, including the popular C4.5 algorithm [Quinlan, 1993], induce decision trees from training data recursively and in a top down procedure. They first put the most informative conditions (i.e., features) at the top of the tree and add more general ones at the bottom – with the aim of achieving high classification performance with a minimal set of conditions. In order not to overfit the training data but be able to generalize to new instances, trees are pruned appropriately. [Quinlan, 1993; Mitchell, 1997;

Pedersen, 2001]

The classification of a new instance follows a path from the decision tree's root node to one of its leaf nodes – branching at each inner node according to the value of the corresponding feature. [Quinlan, 1993; Pedersen, 2001] The word sense finally assigned is the one which is represented by the leaf node reached. By following the paths from the root to the leaves and creating one rule for each such leaf node, decision trees can be converted into decision lists [Paliouras et al., 2000; Witten et al., 2011]. (This is the above-indicated indirect way of constructing decision lists.)

Several studies investigated decision trees for WSD, for example, Mooney [1996], Paliouras et al. [2000], Zavrel et al. [2000], Pedersen [2001, 2002], Lee and Ng [2002], Mohammad and Pedersen [2004], Joshi et al. [2006], and Kopeć et al. [2012]. However, although decision trees are popular and widely used supervised algorithms [Pedersen, 2001; Màrquez et al., 2006], their popularity does not hold for the task of word sense disambiguation [Màrquez et al., 2006; Navigli, 2009]. One major obstacle of using decision trees for WSD is their lower performance compared to other machine learning algorithms [Mooney, 1996; Navigli, 2009], which is probably caused by two main problems: (i) for small training sets, leaf nodes include very few instances, which results in unreliable predictions, and (ii) for training data with many features or many feature values, the tree consists of many inner nodes, which causes data sparseness problems [Màrquez et al., 2006; Navigli, 2009].

Decision lists can partly counter these drawbacks mainly due to their simplicity. On the other hand, decision trees allow more complex decisions at each branching node [Rivest, 1987; Màrquez et al., 2006]. However, none of the decision-rule-based approaches reaches the performance and popularity of the approaches described in the following paragraphs.

Instance-Based Learning

In contrast to the decision-rule-based approaches, the instance-based[1] classifiers described in this paragraph achieve high WSD performance and are thus

[1] Also *memory-based*, *exemplar-based*, or *lazy*.

very popular for WSD. The large number of previous research projects investigating instance-based algorithms for WSD includes Mooney [1996], Ng and Lee [1996], Ng [1997], Escudero et al. [2000b], Paliouras et al. [2000], Veenstra et al. [2000], Zavrel et al. [2000], Hoste et al. [2002b], Mihalcea [2002a,b], Decadt et al. [2004], Pancardo-Rodríguez et al. [2005], Màrquez et al. [2006], Bas et al. [2008], Kübler and Zhekova [2009], Kopeć et al. [2012], and Wiriyathammabhum et al. [2012].

During the training phase, instance-based learning algorithms store all training instances to memory, which is why they are also referred to as memory-based. Proper processing is postponed until classification, which is why they are also characterized as lazy. For classification, new instances are compared to the stored instances and a distance function determines the closest training instance(s) for a given test instance. Final predictions depend on the class values of these closest training instances, which are also called nearest neighbors. The majority sense of the most similar stored instances are assigned to a given test instance. [Màrquez et al., 2006]

The *k-nearest neighbor* (kNN) classifier [Cover and Hart, 1967] is the most basic and most popular instance-based learning algorithm. It represents instances as feature vectors and identifies the k nearest neighbors, i.e., the k most similar instances, according to a similarity metric. The algorithm has two relevant parameters. Firstly, the exact similarity metric used for measuring the similarity between instances; popular similarity metrics are the Hamming distance or the Euclidean distance. Secondly, the number k defines how many nearest neighbors, i.e., how many most similar instances, are considered for classification. In the simplest case, where k is 1, only the most similar instance is relevant and the predicted word sense is the one stored for this single nearest neighbor. If k is larger than 1, the assigned word sense is the one associated with the majority of most similar instances. Since this majority voting is be problematic for skewed class distributions, weighted refinement variants, which either give more weight to relevant features or which give more weights to instances more similar to the new instance to be classified, are quite common. [Mitchell, 1997; Daelemans et al., 1999; Daelemans and Hoste, 2002; Màrquez et al., 2006; Navigli, 2009]

The main difference of the instance-based approach compared to other supervised algorithms such as the ones based on decision rules is that there is no reduction, modification or restructuring of the training data; during the training phase, all examples are stored as-is; and the comparison is delayed until the classification phase [Mitchell, 1997; Daelemans and Hoste, 2002]. A well-known problem of instance-based algorithms, which arises from the fact that all available features and instances are stored without any preprocessing or preselection, is that irrelevant features and improperly scaled feature values often lead to incorrect classifications. [Mitchell, 1997] The solution to this problem is to weight features according to their relevance and, in the most extreme case, to disregard irrelevant features altogether. On the other hand, storing all training data can also be advantageous: Daelemans et al. [1999] attribute the high generalization performance of instance-based algorithms to the fact that these algorithms retain outlying instances.

Probabilistic Classification

This paragraph describes two machine learning algorithms based on probabilistic computations: naive Bayes and maximum entropy.

The *naive Bayes*[1] classifier [Duda and Hart, 1973] is among the simplest probabilistic classifiers. During the training process, it estimates for individual features their probabilities of belonging to specific classes. These probabilities are calculated on the basis of the frequencies with which the features occur for certain classes in the training data. For classification, the naive Bayes classifier estimates for a new instance the conditional probabilities with which the instance belongs to a certain class. These probabilities are estimated by multiplying the individual probabilities for each feature to occur for a certain class. The classifier assigns the class (i.e., word sense), which has the highest overall probability (i.e., the product of the probabilities of the individual features belonging to a certain class) for the new instance. [Mitchell, 1997; Paliouras et al., 2000; Navigli, 2009]

The classifier is characterized as *Bayesian* because it uses the Bayes theo-

[1]Spelling variant: *naïve Bayes*.

rem for its probability calculation. The Bayes theorem [Bayes, 1763] specifies the posterior probability for a new instance given the probabilities in the training data. [Mitchell, 1997; Paliouras et al., 2000; Navigli, 2009] The classifier is further referred to as *naive* because it assumes conditional independence of all features given a class. Although this strong assumption is practically seldom fulfilled, naive Bayes performs well for the task of word sense disambiguation compared to other supervised methods. Naive Bayes classification has often been applied to WSD in the literature, for example, by Mooney [1996], Escudero et al. [2000b], Paliouras et al. [2000], Zavrel et al. [2000], Pedersen [2000, 2001], Klein and Manning [2002], Lee and Ng [2002], Yarowsky and Florian [2002], Agirre and Martínez [2004], Lamjiri et al. [2004], Le and Shimazu [2004], Wu et al. [2004], Pancardo-Rodríguez et al. [2005], Joshi et al. [2006], Màrquez et al. [2006], Bas et al. [2008], Kopeć et al. [2012], and Wiriyathammabhum et al. [2012].

A *maximum entropy* classifier aims to maximize the conditional likelihood of classes on the basis of the training data. Therefore, each feature from the set of given features represents a constraint. These constraints are posed by the distribution of the features observed in the training data. The size of the feature set and, thus, also the size of the constraints set is not restricted. In contrast to naive Bayes, the probability model underlying a maximum entropy classifier does not assume feature independence.

Given the feature constraints, a maximum entropy classifier computes probability models that satisfy the constraints. This set of models that satisfy the given constraints can be infinitely large. The learning aim of a maximum entropy classifier is to identify the most uniform model among the set of models. This most uniform model is the model where uncertainty is at its maximum and, since entropy is a measure of uncertainty, it is at the same time the model with the highest entropy. The motivation behind identifying the maximum entropy model is to restrict the probability model to those assumptions that can be extracted from the distribution in the training data rather than to assume any further information that is not observed in the training data. [Berger et al., 1996; Manning and Schütze, 1999]

Maximum entropy models were applied to the task of word sense disam-

biguation by, for example, Zavrel et al. [2000], Dang and Palmer [2002], Suárez and Palomar [2002], Lamjiri et al. [2004], Suárez Cueto [2004], Wu et al. [2004], Chen [2006], Tratz et al. [2007], and Wiriyathammabhum et al. [2012].

Support Vector Machines

Support Vector Machines (SVMs) [Cortes and Vapnik, 1995; Vapnik, 1995] use hyperplanes to separate training instances into two classes – as illustrated in Figure 2.2. Therefore, training instances are mapped into the feature space by a function referred to as a *kernel*. The circles and squares in the left part of Figure 2.2 represent instances of two distinct classes. The idea is to find a hyperplane that separates the instances of these two classes (see the middle part of the figure).

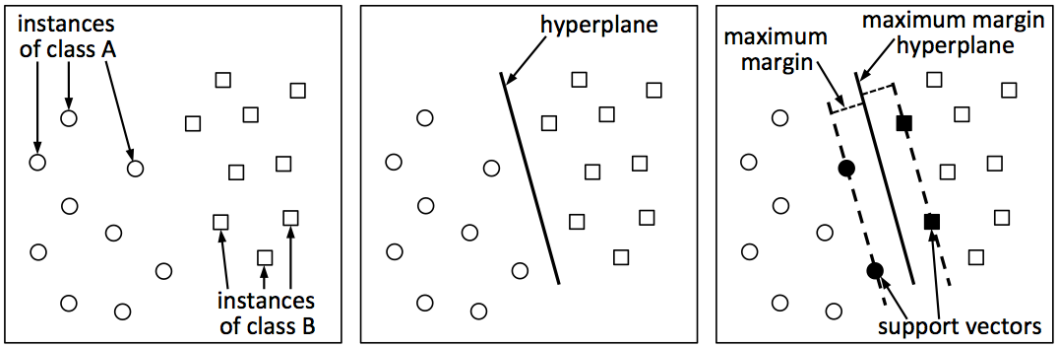

Figure 2.2: The idea of support vector machines.

The optimization goal of SVMs is to identify the hyperplane with the largest margin to the closest instances of both classes. This is illustrated in the right part of Figure 2.2. The instances of both classes that are closest to the hyperplane are also called *support vectors* (the filled circles and squares in the figure). In case the training instances are not separable by a hyperplane, a trade-off between a low training error and a large margin has to be made. This trade-off is controlled by the regularization parameter C. [Cortes and Vapnik, 1995; Witten et al., 2011]

SVMs are, by default, linear classifiers, i.e., they use linear models to map instances into the feature space. However, instead of using a linear mapping of instances into the feature space, SVMs can map instances non-linearly into

high-dimensional features spaces by using non-linear kernels such as polynomial or Gaussian functions [Boser et al., 1992].

The two most relevant parameters for support vector machines are (i) the regularization parameter C to control the trade-off between a low training error and a large margin and (ii) the choice of the kernel function. For support vector machine classification, a test instance is mapped into the same feature space in which the hyperplane is located and classified according to the side of the hyperplane it belongs to.

In the base case, SVMs can discriminate between two classes only. To apply SVMs to multi-class problems, several approaches exist. Among the most popular approaches to solve multi-class problems with SVMs is the one-versus-one method. For this method, an SVM classifier is constructed for every pair of classes, and the individual classification results are combined by a maximum voting strategy to build a final classification result. [Hastie and Tibshirani, 1998; Duan and Keerthi, 2005]

Support vector machines have previously demonstrated high performance among supervised methods and are thus very popular for many kinds of classification tasks. Since their high performance has also been proven for the task of WSD, they are very popular for disambiguating word senses [Zavrel et al., 2000; Cabezas et al., 2001; Lee and Ng, 2002; Agirre and Martínez, 2004; Lee et al., 2004; Wu et al., 2004; Pancardo-Rodríguez et al., 2005; Joshi et al., 2006; Màrquez et al., 2006; Wiriyathammabhum et al., 2012; Maarouf et al., 2014].

Combination

It has been shown for various NLP tasks, including word sense disambiguation [Florian et al., 2002; Florian and Yarowsky, 2002; Klein et al., 2002], that multiple classifier systems outperform single decision systems. Many of the best performing systems at SensEval-3 were combined classifiers [Mihalcea et al., 2004]. This subsection describes two popular combination strategies: voting and boosting.

In a *voting* combination, several distinct machine learning classifiers are

combined as follows: each single classifier votes for a candidate sense of the target word, the votes are summed, and the target word sense(s) with the highest sum are defined to be the overall disambiguation result. Several parameters influence the performance of a voting algorithm, such as the choice of single algorithms to be combined or whether the voting assigns equal weights to each single classifier or individual weights (depending on the single classifier's performance). Related WSD studies that combined single classifiers by voting include Ilhan et al. [2001], Florian et al. [2002], Hoste et al. [2002c], Klein et al. [2002], and Turney [2004].[1]

Boosting pursues a different combination strategy than voting: rather than combining classifiers for different machine learning algorithms, several classifiers are iteratively trained with the same ML algorithm (referred to as the *base* classifier), but on reweighted training instances after each iteration, to finally build an ensemble classifier. The most popular boosting algorithm is AdaBoost [Freund and Schapire, 1996]: the base classifier is initially run on the unweighted training data. After each run, all instances in the training data are reweighted to give higher weights to misclassified instances and the base classifier is rerun on this reweighted data. This process of running and reweighting is repeated for a fixed number of iterations. Boosting was applied to the task of word sense disambigutation by, for example, Escudero et al. [2000a], Lee and Ng [2002], Martínez et al. [2002], Màrquez et al. [2006], and Kopeć et al. [2012].

2.3.3 Related Work on German WSD

Most of the work on word sense disambiguation has focused on English. [Kilgarriff and Palmer, 2000] One of the factors that has hampered WSD research for German has been the lack of appropriate resources, particularly in the form of sense-annotated corpus data. Since these corpora are a prerequisite

[1]Note that the idea of combining individual WSD algorithms by voting to build a joint classifier is not bound to supervised machine learning methods. Indeed, the WSD experiments in Chapter 7 include a method that combines knowledge-based WSD methods. The reason, however, why this subsection on combined methods is placed under the section on supervised machine learning methods is because for the task of WSD it has mostly been applied to supervised ML methods.

for the development, training, and evaluation of word sense disambiguation systems, it is not surprising that there has been little research on languages with a shortage of such corpora. Due to the lack of sense-annotated corpora for German prior to the construction of the sense-annotated corpora described in Chapters 5 and 6, there has been relatively little research on WSD for this language.[1]

As it is the case for the WSD experiments in this thesis, all previous research on automatic word sense disambiguation work for German using GermaNet as a sense inventory[2] [Widdows et al., 2003; Steffen et al., 2004; Broscheit et al., 2010; Henrich and Hinrichs, 2012] focused on the lexical sample task for WSD, i.e., the disambiguation of a fixed set of polysemous target words. The only exception is the study by Saito et al. [2002] who – by manual means – demonstrated the appropriateness of GermaNet's sense coverage for WSD and therefore regard all words in running text.

Saito et al. [2002] is the earliest known German WSD study using GermaNet as a sense inventory. It describes the manual creation of a sense-annotated corpus from German novels for children and young people and from German newspaper articles. All content words (adjectives, nouns, and verbs) were automatically extracted from this corpus and, together with possible GermaNet senses, presented to human annotators. With this manual sense annotation of all words in running text, Saito et al. [2002] demonstrated the adequacy of GermaNet's sense coverage for WSD – an important preliminary step for any automatic WSD system using GermaNet as the sense inventory, but not yet a proper automatic disambiguation system.

The studies by Widdows et al. [2003] and Steffen et al. [2004] applied unsupervised methods for domain-specific WSD on medical texts. Steffen et al.

[1] Note that while this subsection describes related work on German WSD with the focus on the disambiguation systems, Subsection 5.1.2 invokes the same body of research [Saito et al., 2002; Widdows et al., 2003; Steffen et al., 2004; Broscheit et al., 2010] with an emphasis on comparing the employed sense-annotated corpora.

[2] A separate branch of research studies (e.g., Erk [2005], Burchardt et al. [2009], and Rehbein et al. [2009]) investigates the disambiguation of frames (semantic classes) or frame elements (semantic roles) from the German FrameNet (`http://www.laits.utexas.edu/gframenet/`). These works are not outlined in this subsection, because the approaches differ fundamentally: the sense inventories are based on different linguistic theories (frames vs. concepts), which crucially influence the work on WSD.

[2004] applied two separate WSD methods: one method automatically determines domain-specific senses of ambiguous target words on the basis of their relative statistical relevance across several domain-specific corpora. The other method is instance-based and uses k-nearest neighbor classification in an unsupervised manner. Here, they used the Weka machine learning tool suite, which is also used in the WSD experiments in Chapter 8 though in a supervised manner. The best results reported by Steffen et al. [2004] were obtained by a combined approach that applies the two methods in a disjunctive manner: if one method is unable to assign any word sense, the other method is applied. This finding that a combined algorithm improves the results generally corroborates the results in this dissertation. However, the two most significant deviations between the study by Steffen et al. [2004] and the experiments in this thesis are (i) their domain-specificity and (ii) their application of unsupervised methods (vs. knowledge-based and supervised approaches in this thesis).

Widdows et al. [2003] performed unsupervised WSD for English and German and derive their sense inventory for these two languages from the Medical Subject Headings thesaurus (MeSH) contained in the Unified Medical Language System (UMLS). They applied three unsupervised WSD methods: (i) bilingual disambiguation based on a parallel corpus of English–German medical scientific abstracts obtained from the Springer Link web site[1], (ii) collocational disambiguation as introduced by Yarowsky [1995] which automatically extracts multi-word expressions and collocations from UMLS as seed examples for Yarowsky's algorithm, and (iii) disambiguation using related UMLS terms following the idea of the Lesk [1986] algorithm. This third method resembles the knowledge-based approach in Chapter 7 in some respects: both techniques apply a variant of Lesk's [1986] dictionary-based word overlap method and both techniques use related terms and concepts to extend the coverage of the overlaps method. However, the approach by Widdows et al. [2003] achieved best results with manually annotated information on co-occurring concepts, which is not available in the context of this thesis. The availability of this type of information originates from the use of a different sense inventory (i.e.,

[1] `http://link.springer.de/`

UMLS).

The studies of Widdows et al. [2003] and Steffen et al. [2004] both focused on the medical domain and are thus domain-dependent. A more recent study [Broscheit et al., 2010] performed WSD on a domain-independent German corpus – i.e., on the web-harvested *deWaC* corpus (see Section 6.5 for more details). Since the sense annotations in the deWaC corpus have recently been made publicly available, they have been updated to the most recent version of GermaNet in the context of this thesis (see Section 6.5). The revised deWaC sense annotations are employed in the WSD experiments in Chapters 7 and 8.

The disambiguation system by Broscheit et al. [2010] used GermaNet as a knowledge base and the graph-based algorithm *Personalized PageRank* (PPR) of Agirre and Soroa [2009], the unsupervised algorithms of McCarthy et al. [2004] and Lapata and Keller [2007] for determining the most frequent sense, and a simple majority voting algorithm that combines the single algorithms. The best results reported by Broscheit et al. [2010] were not obtained by the voting approach but rather by the PPR algorithm alone.

In general, the performance obtained by the knowledge-based WSD experiments in Chapter 7 are comparable to the results by Broscheit et al. [2010] on the same sense-annotated corpus. The only sizable deviation is in the performance of the combined algorithms: the results reported in Chapter 7 of this thesis improve in combination, which contradicts Broscheit et al.'s finding that there is no improvement. However, it is important to note that a comparison of the results from Broscheit et al. [2010] with the results of this thesis is not entirely fair. There are two main factors that influence the results (see Section 7.3 for more discussion). Firstly, the versions of the underlying sense inventory and, thus, also the sense annotations are different (see Section 6.5 for details). Secondly, the evaluation setups differ both in the employed algorithms as well as in the use of backoff strategies for cases where the WSD algorithms are unable to assign any word sense.

The most recent paper on German word sense disambiguation was published by Henrich and Hinrichs [2012]. It explored several knowledge-based WSD algorithms that are based on a suite of semantic relatedness measures, including path-based, information-content-based, and gloss-based methods.

Since the individual algorithms produced diverse results in terms of precision and thus complemented each other well in terms of coverage, a set of combined algorithms was investigated and compared in performance to the individual algorithms. Among the single algorithms considered, a word overlap method derived from the Lesk [1986] algorithm yielded the best F-score. This result was outperformed by a combined WSD algorithm that uses weighted majority voting. All reported WSD experiments utilized GermaNet as a sense inventory and WebCAGe (which has been constructed as part of this dissertation – see Section 5.2) as a sense-annotated corpus. The experiments [Henrich and Hinrichs, 2012] were performed in the context of a pilot study as part of this thesis, particularly of Chapter 7. The underlying idea to use semantic relatedness measures for WSD is the same. However, the WSD experiments described in the paper used old versions of GermaNet and WebCAGe, whereas Chapter 7 reports on experiments with the most recent versions. Chapter 7 extends Henrich and Hinrichs [2012] by presenting WSD results for two additional sense-annotated corpora, i.e., TüBa-D/Z and deWaC, which were not available in the context of Henrich and Hinrichs [2012]. Moreover, it includes the evaluation of adjectives and verbs while the paper focused solely on nouns.

Chapter 3

Fundamentals of GermaNet

This chapter introduces the German wordnet GermaNet, which serves as a sense inventory for all word sense disambiguation experiments in this thesis.

A wordnet is a lexical semantic resource that interrelates word senses and semantic concepts in a network. It partitions the lexical and conceptual space into a set of semantic concepts that are interlinked by lexical semantic relations. Since wordnets encode principles resembling semantic dictionaries, thesauri, and lightweight ontologies, they can be seen as a combination of all of them. The first resource of this kind and also the most prominent one is the Princeton WordNet for English [Miller, 1995; Fellbaum, 1998a]. Its development began in the 1980s under the auspices of George A. Miller at Princeton University. Following the idea of the Princeton WordNet, several wordnets have been developed for other languages.

The development of the German wordnet GermaNet [Hamp and Feldweg, 1997] started in 1997 at the University of Tübingen. It is modeled after the Princeton WordNet and, thus, the two wordnets share the same basic assumptions. However, since GermaNet is built from scratch, the two resources deviate in several respects – as itemized in Section 3.1.

GermaNet covers the three word classes of adjectives, nouns, and verbs. It encodes word senses and semantic concepts (see Section 3.2) that are interlinked by lexical semantic relations (see Section 3.3). The wordnet is hierarchically structured in terms of the hypernymy relation of synsets (described in Section 3.4). Furthermore, GermaNet includes interlingual links to the

Princeton WordNet (Section 3.5), encodes information on nominal compounds (Section 3.6) and on verbal frames (Section 3.7).

If not otherwise stated, the most recent version of GermaNet available at the time of finishing the writing of this dissertation – i.e., release 9.0, as of April 2014 – is described throughout this chapter. However, since the development of the resource is still in process, several versions of GermaNet are relevant for this dissertation. The overview of the coverage of GermaNet releases given in Section 3.9 reports on the resource's active process of development and illustrates the variety of extensions that entered GermaNet. Together with Appendix A, it provides a detailed summary of the four GermaNet releases most significant throughout this dissertation; including information on which release serves as the basis for which task in this dissertation, and, vice versa, which release contains which newly contributed information.

3.1 Comparing GermaNet with WordNet

GermaNet is based on the Princeton WordNet for English. The two resources share the same basic principles and assumptions. For example, their understanding of lexical units and synsets as well as their encoding of relations between those entries is similar. However, GermaNet is built from scratch and is not simply translated. Mainly due to language specifics, the two wordnets deviate in several respects. In order to better understand the nature of these differences, the following list outlines the most important points of disparity:[1]

Coverage WordNet's coverage is larger than the coverage of GermaNet:[2] 117 659 vs. 93 246 synsets, 206 941 vs. 121 810 lexical units, and 155 287 vs. 110 738 literals. Furthermore, GermaNet does not include adverbs, while WordNet does.

Granularity The sense distinction in WordNet is more fine-grained than in GermaNet. The average number of word senses per literal in WordNet is

[1] The versions compared are WordNet 3.0 and GermaNet 9.0.

[2] This fact is hardly surprising, since the development of WordNet started before the work on GermaNet.

about 1.33 (i.e., 206 941 total word-sense pairs divided by 155 287 unique strings), which is higher than the corresponding average of about 1.10 in GermaNet (i.e., 121 810 lexical units divided by 110 738 literals).

Artificial concepts GermaNet encodes artificial concepts to systematically fill lexical gaps and to avoid unjustified co-hyponymy – as explained in Subsection 3.4. WordNet also uses artificial concepts, but less systematically and without any special label, which makes their automatic identification difficult. [Hamp and Feldweg, 1997]

Adjective hierarchy As for the other word classes, adjectives in GermaNet are ordered hierarchically in terms of the hypernymy/hyponymy relation. By contrast, WordNet encodes antonymous *satellite* synsets (see Miller [1998b] for a documentation of adjectives in WordNet).

Verbal frames Mainly due to language specifics, GermaNet's verbal frames capture more details than those in WordNet: reflexives, grammatical case, expletive subjects, and *to*-infinitives are explicitly encoded in GermaNet. See Subsection 3.7 for more details on GermaNet's verbal frames.

Compounds In GermaNet, nominal compounds are split into their constituent parts and labeled with linguistic information such as foreign words and named entities – see Subsection 3.6 and Henrich and Hinrichs [2011]. This kind of information makes particular sense for German, where compounds are almost always spelled as one word.

Connectedness GermaNet is a completely connected graph hierarchy without any dangling subgraphs, whereas WordNet consists of several distinct hierarchies – one for each semantic field (see Subsection 3.4).

Maintenance format While the development environment of WordNet uses so-called *lexicographer files*, the GermaNet maintenance format has been converted from lexicographer files to a relational database – as described in Appendix B and Henrich and Hinrichs [2010a].

3.2 Lexical Units and Synsets

GermaNet represents meanings of words, i.e., word senses, as *lexical units*. Since two word senses are synonyms if they express the same semantic concept, (near-)synonymous lexical units are grouped together to form *synsets* (synonym sets). That is, GermaNet represents semantic concepts as synsets, which are set-representations of the semantic relation of synonymy.

The German wordnet distinguishes less word senses than the Princeton WordNet (as mentioned in the previous subsection). An explicit guideline among lexicographers was stated in the very first paper on GermaNet:

> *"The amount of polysemy is kept to a minimum in Germanet, an additional sense of a word is only introduced if it conflicts with the coordinates of other senses of the word in the network. When in doubt, GermaNet refers to the degree of polysemy given in standard monolingual print dictionaries."*
>
> [Hamp and Feldweg, 1997, page 10]

Synsets in GermaNet belong to a word class (adjective, noun, or verb) and to a semantic field (see Table 3.1 for the list of 38 semantic fields[1]). Originally, the division into semantic fields was for organizational purposes, i.e., to divide synsets into multiple files. However, since GermaNet's conversion into a relational database (see Appendix B), these semantic fields are not organizationally required anymore, but rather serve as a grouping of synsets into semantically related topics. Since lexical units belong to synsets, corresponding information on word classes and semantic fields can easily be identified for lexical units as well.

Optionally, synsets can be assigned definitions.[2] Prior to the research reported in Chapter 4 on semi-automatically enriching GermaNet with sense descriptions from Wiktionary, only 10% of all synsets were accompanied by

[1] Note that the semantic fields resemble the unique beginners in WordNet (see Section 3.4 below). However, mainly due to language specific differences of the two wordnets, the lists are not exactly identical: for instance, labels *Verhalten* and *privativ* are not available in WordNet, while *act* and *process* are not used in GermaNet.

[2] Note that the terms *definition*, *gloss*, and *sense description* are used interchangeably throughout this thesis.

Table 3.1: Semantic fields in GermaNet.

Semantic field	Semantic field (continued)
Allgemein 'general'	*Motiv* 'motive'
Artefakt 'artifact'	*Nahrung* 'food'
Attribut 'attribute'	*Naturgegenstand* 'natural object'
Besitz 'possession'	*Naturphänomen* 'natural phenomenon'
Bewegung 'motion'	*Ort* 'place'
Form 'shape'	*Pertonym* 'pertainym'
Gefühl 'feeling'/'emotion'	*Perzeption* 'perception'
Geist 'spirit'	*Pflanze* 'plant'
Geschehen 'event'	*privativ* 'privative'
Gesellschaft 'social'	*Relation* 'relation'
Gruppe 'group'	*Schöpfung* 'creation'
Kognition 'cognition'	*Substanz* 'substance'
Kommunikation 'communication'	*Tier* 'animal'
Konkurrenz 'competition'	*Tops* 'tops'
Kontakt 'contact'	*Veränderung* 'change'
Körper 'body'	*Verbrauch* 'consumption'
Körperfunktion 'bodily function'	*Verhalten* 'behavior'
Lokation 'location'	*Zeit* 'time'
Menge 'quantity'	
Mensch 'person'	

definitions. This semi-automatic enrichment has resulted in harvested definitions for about 30% of all GermaNet 7.0 lexical units (i.e., 29 433 out of 99 523, see Table 3.5 in Section 3.9). It should be noted that these harvested sense descriptions are assigned to lexical units rather than synsets; but again, a list of corresponding Wiktionary descriptions can easily be identified for synsets.

Since a lexical unit represents a word sense, it carries information on the word itself, including its orthography. In the context of the recently adopted German spelling reform (*Neue Deutsche Rechtschreibung*, Rat für deutsche Rechtschreibung [2006]), lexical units can have optional orthographic variants. Since GermaNet release 5.2, as well as its obligatory main orthographic form, a lexical unit can also have further optional variants. These variants include spelling variants, which characterize the differences between the old and the new German spellings, and specify whether the old variant is still valid in the new orthography:

Main orthographic form A lexical unit always encodes a main orthographic form, which represents the correct spelling of a word according to the rules of the recently adopted German spelling reform [Rat für deutsche Rechtschreibung, 2006].

Orthographic variant In case of an alternative spelling that is permissible according to the new German spelling, a lexical unit can optionally have an orthographic variant. An example of this kind is the German noun *Delfin* 'dolphin'. Apart from the main form *Delfin*, there is an orthographic variant *Delphin*.

Old orthographic form If the orthography of a word has changed in the context of the spelling reform, the old orthographic form represents the main form from the old German spelling.

Old orthographic variant This encodes an orthographic variant that was permissible prior to the spelling reform. It is encoded only if the variant is no longer valid in the new orthography.

Furthermore, lexical units contain information about whether they represent a named entity or whether they are stylistically marked.

3.3 Lexical Semantic Relations

Lexical units and synsets are interlinked by lexical semantic relations. GermaNet distinguishes two types of relations: *conceptual relations* are established between two semantic concepts, i.e., synsets; *lexical relations* are established between two individual lexical units.

The following list explains all conceptual (semantic) relations. For each relation, it includes an example for each applicable word class.

hypernymy/hyponymy This is the most important conceptual relation in a wordnet. *Hypernymy* connects a more specific item (a *hyponym*) to its more generic concept (a *hypernym*). The inverse direction, i.e., from

the hypernym to the hyponym, is referred to as *hyponymy*. Since hypernymy/hyponymy is a transitive relation, it has a hierarchical structuring function within GermaNet – see Subsection 3.4 below. It is applicable to all three word classes of adjectives, nouns, and verbs, as the following examples illustrate respectively:

↦ *adipös* 'adipose' has hypernym *dick* 'fat'
dick has hyponym *adipös*

↦ *Fußball* 'football' has hypernym *Ball* 'ball'
Ball has hyponym *Fußball*

↦ *boxen* 'to box' has hypernym *schlagen* 'to hit'
schlagen has hyponym *boxen*

component meronymy/holonymy Meronymic relations (also called part-whole relations) make up the second largest portion of conceptual relations in GermaNet. Prototypically, they are annotated between nouns only: the part is referred to as the *meronym*, and the whole object is referred to as the *holonym*. GermaNet's part-whole relation is distinguished into four subclasses – as introduced by Hinrichs et al. [2013] for GermaNet release 6.0 – of which component meronymy is the most prevalent. *Component meronymy* is established between an object and its component parts; and the inverse *component holonymy* relates a component part to its larger object.

↦ *Hand* 'hand' has component meronym *Finger* 'finger'
Finger has component holonym *Hand*

member meronymy/holonymy The relation of a group to one of its members is denoted as *member meronymy*. The inverse direction, i.e., from a member to its group, is denoted as *member holonymy*.

↦ *Flotte* 'fleet' has member meronym *Schiff* 'ship'
Schiff has member holonym *Flotte*

substance meronymy/holonymy This relation connects an object to the substance from which it is made or the substance of which it consists (*substance meronymy*); and the other way around (*substance holonymy*).

↦ *Fahrrad* 'bike' has substance meronym *Stahl* 'steel'
Stahl has substance holonym *Fahrrad*

portion meronymy/holonymy Objects, which can be portioned into smaller units, are involved in the *portion meronymy* relation.
↦ *Tag* 'day' has portion meronym *Stunde* 'hour'
Stunde has portion holonym *Tag*

entailment The most prominent example of entailment is the temporal inclusion of one event by another event – also referred to as a kind of subevent. In GermaNet, events in this sense are realized by verbs or nouns, i.e., the *entailment* relation might be established between two verbs (as in the first example given below) or between two nouns (see second example below). This relation has an inverse relation; the notation says that if concept A *entails* concept B, then concept B *is entailed by* concept A.
↦ *schnarchen* 'to snore' entails *schlafen* 'to sleep'
schlafen is entailed by *schnarchen*
↦ *Anrufer* 'caller' entails *Telefonat* 'phone call'
Telefonat is entailed by *Anrufer*

causation If one action causes another action or if an action results in a certain condition (i.e., a resultative state), a *causation* relation is encoded. This relation has no inverse relation. It predominantly connects verbs with verbs (see first example given) or verbs with adjectives (see second example).
↦ *bremsen* 'to break' causes *verlangsamen* 'to decelerate'
↦ *korrigieren* 'to correct' causes *richtig* 'correct'

association This relation does not concretize the exact nature of the relationship. It rather connects two semantic concepts that are somehow related to each other, but whose type of connection is not one of the previously itemized relations. *Association* relations are defined in both directions, i.e., if concept A is related to concept B, then concept B is equally related to concept A. In general, such associations are possible between all word class combinations, but in practice the relation mostly

involves at least one noun.
↦ *Brauerei* 'brewery' is related to *Bier* 'beer'
Bier is related to *Brauerei*
↦ *rocken* 'to rock' is related to *Rockmusik* 'rock music'
Rockmusik is related to *rocken*

The following list of lexical relations is smaller than the list of conceptual relations. Again, each item explains the relation in question and gives illustrative examples.

synonymy Clearly, the linkage of synonyms is the most prominent lexical relation. Two word senses are *synonyms* if they express the same semantic concept. Synonymy is established indirectly by grouping synonymous lexical units into synsets (i.e., synonym sets). The relation is bidirectional and equally connects two (or even more) lexical units. Synonymy occurs for all word classes, as illustrated by the following examples:
↦ *bunt* 'colorful' has synonym *vielfarbig* 'colorful'
vielfarbig has the synonym *bunt*
↦ *Karotte* 'carrot' has the synonym *Möhre* 'carrot'
Möhre has the synonym *Karotte*
↦ *abspielen* 'to pass a ball' has the synonym *passen* 'to pass a ball'
passen has the synonym *abspielen*

antonymy Pairs of opposites are also called *antonyms*. The antonymy relation links antonymous lexical units in a bidirectional manner. Although antonyms exist for all three word classes – as the three examples show – antonymy is by far most frequently encoded between adjectives.
↦ *kalt* 'cold' has the antonym *warm* 'warm'
warm has the antonym *kalt*
↦ *Frau* 'woman' has the antonym *Mann* 'man'
Mann has the antonym *Frau*
↦ *schließen* 'to close' has the antonym *öffnen* 'to open'
öffnen has the antonym *schließen*

pertainymy This relation encodes derivation with a semantic origin. The base word determines the meaning of the derived word such that the meaning of the derived word cannot be understood without knowing the meaning of the base word. By definition, it connects words of different word classes. In general, all word class combinations are possible, though some are more frequent in GermaNet. Clearly the most prominent example of pertainymy relates a denominal adjective with its nominal base (see first example below). In many other cases it relates a deadjectival nominalization with its adjectival base (see second example), a deverbal adjective with its verbal base (see third example), or a denominal verb with its nominal base (see forth example). Pertainymy is encoded unidirectional, i.e., without an inverse relation.
↦ *schriftstellerisch* 'authorical' has the pertainym *Schriftsteller* 'author'
↦ *Müdigkeit* 'tiredness' has the pertainym *müde* 'tired'
↦ *schläfrig* 'drowsy' has the pertainym *schlafen* 'to sleep'
↦ *bürsten* 'to brush' has the pertainym *Bürste* 'brush'

participle This relation links a derived word to its base in a similar way to pertainymy, but is restricted to relations between deverbal adjectives with participle forms[1] and their verbal bases. As before, this relation does not have an inverse relation.
↦ *isoliert* 'isolated' has the participle *isolieren* 'to isolate'

3.4 The Hierarchy

Wordnets are hierarchically structured in terms of the hypernymy/hyponymy relation of synsets. In the Princeton WordNet, several distinct hierarchies exist. Each hierarchy roughly corresponds to a semantic field (see the description of semantic fields in Section 3.2 above). The top synsets of these hierarchies are called *unique beginners* [Fellbaum, 1998b; Miller, 1998a]. This is different in the German wordnet. Since release 5.2, GermaNet is a completely connected

[1]Note that GermaNet does not generally cover all participles, it includes only those participles whose meanings go beyond the meanings of the base verbs.

graph without any dangling subgraphs. There is a single artificial top node (labeled as *GNROOT*) subsuming all synsets of all word classes. The previous unique beginners are direct or indirect hyponyms of this common top synset.

Artificial concepts such as the top node GNROOT represent non-lexicalized concepts which help to structure the GermaNet hierarchy and to avoid unjustified co-hyponymy[1]. GermaNet contains artificial concepts (i) for purely structural requirements as well as (ii) for lexical gaps in the German language. Artificial nodes of type (i) subsume proper artificial concepts that are unlikely to be lexicalized in any language, whereas artificial concepts of type (ii) are likely to be lexicalized in other languages.

Several semantic concepts express more than one aspect of meaning. To capture multiple equally important hypernyms for a semantic concept, GermaNet allows the encoding of more than one hypernym for a synset. This phenomenon is referred to as *cross-classification* and is systematically employed to capture multiple meaning aspects for a concept rather than encoding only a partial aspect.

3.5 Interlingual Links to Princeton WordNet

The *interlingual index* (ILI) represents an extensive multilingual lexical database which connects wordnets of different languages. The central component of this database is a list of basic vocabulary meanings taken from the Princeton WordNet for English, to which corresponding concepts from wordnets of other languages are linked. That is, the index allows a mapping of concepts of different languages via lexical semantic relations such as hypernymy, hyponymy, synonymy, near-synonymy, holonymy, meronymy, etc. [Vossen, 1998, 2002]

The ILI was initially created in the context of the EuroWordNet project[2].

[1] Two synsets with the same hypernym – see Subsection 3.3 above – are denoted as *co-hyponyms*.

[2] For more information about the EuroWordNet project, please see Vossen [1998] and Vossen [2002], and the project webpages of the GlobalWordNet (`http://www.globalwordnet.org/`) and the EuroWordNet (`http://www.illc.uva.nl/EuroWordNet/#EuroWordnet`) projects.

In the context of this project, 20 000 ILI records were created for GermaNet. These allow the mapping of GermaNet senses to the corresponding entries in the Princeton WordNet (and, as a consequence, also to corresponding entries in other wordnets). For compatibility with recent versions of GermaNet, existing GermaNet–ILI records have been revised[1] and, since release 7.0, integrated into the official GermaNet release. The German part of the ILI is extended to 28 661 records in GermaNet release 9.0 (see Table 3.5 in Section 3.9). By far the most frequent type of interlingual relation is synonymy which occurs in 72.3% of all German ILI records. While near-synonymy is encoded in 16.3% and hypernymy in 6.6%, the other types of relations occur much scarcer.

3.6 Nominal Compounds

Compounding is a highly productive word formation process in German resulting in complex words with two or more constituent parts. In GermaNet, nominal compounds are split into their constituent parts, i.e., *modifier* and *head* [Henrich and Hinrichs, 2011]. The splitting identifies the immediate constituents at each level of analysis and thus reflects the recursive nature of compounds that have more than two constituent parts, such as *Autobahnanschlussstelle* 'motorway junction' – see Figure 3.1. The immediate constituents of this compound are *Autobahn* 'motorway' and *Anschlussstelle* 'junction', with the first constituent further split into *Auto* 'car' and *Bahn* 'way' and the second constituent further split into *Anschluss* 'connection' and *Stelle* 'place'.

What makes compound splitting for German a challenging task is the fact that compounding is not always simple string concatenation, but often involves the presence of intervening linking elements or the elision of word-final characters in the modifier constituent of a compound [Eisenberg, 2006].[2] Compound splitting in GermaNet is supported by an automatic algorithm described in Henrich and Hinrichs [2011]. The basic idea of this algorithm combines several

[1] Note that the revised GermaNet-ILI records link lexical units, although the original ILI designated the mapping of synsets.

[2] Langer [1998] presents a frequency table for German linking morphemes and elisions, according to which approximately half of the compounds he investigated contain some kind of linking morpheme or elision.

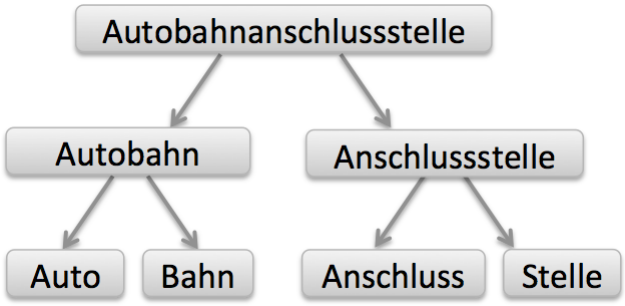

Figure 3.1: Split compound *Autobahnanschlussstelle* 'motorway junction'.

individual compound splitters to identify the immediate constituents of nominal compounds at each level of analysis. All automatically split compounds are manually post-corrected and enriched with information on relevant properties before they are inserted into GermaNet.

The rightmost head constituent of German nominal compounds is, by definition [Eisenberg, 2006], a noun. By contrast, modifiers are manually labeled with appropriate classes – see Table 3.2.

Table 3.2: Modifier classes.

Class	**Example**
adjective	*kurz* 'brief' in *Kurzbeschreibung* 'brief description'
adverb	*nicht* 'non-' in *Nichtraucher* 'non-smoker'
noun	*Tomate* 'tomato' in *Tomatensuppe* 'tomato soup'
particle	*selbst* 'self-' in *Selbstmotivation* 'self-motivation'
preposition	*vor* 'pre-' in *Vorname* 'prename'
pronoun	*niemand* 'nobody' in *Niemandsland* 'no-man's-land'
verb	*klappen* 'to fold' in *Klappstuhl* 'folding chair'

In GermaNet, all modifiers are lemmatized and if a modifier is ambiguous with respect to its word class (due to conversion), both possibilities are specified: for example, the verb *feiern* 'to feast' and the noun *Feier* 'feast' are encoded as modifier alternatives for the compound *Feiertag* 'feast day'; for the compound *Reisekosten* 'playing card', both the verb *reisen* 'to travel' and the noun *Reise* 'travel' are specified as plausible modifiers.

Furthermore, several properties are manually encoded for modifiers and heads, as shown in Table 3.3.

Table 3.3: Properties for compound constituents.

Property	**Example (and explanation, if needed)**
abbreviation*	*IP* 'IP' in *IP-Paket* 'IP packet'
affixoid*	affixoids have a special grammatical status between bound and free morphemes; e.g., *haupt* 'main' in *Hauptbahnhof* 'main station'
foreign word*	*Offset* 'offset' in *Offsetdruck* 'offset printing'
combining form*§ (German: *konfix*)	bound morphemes which are borrowed from a foreign language and whose meaning stems from that particular language, e.g., *bio-* 'organic' in *Biosiegel* 'organic seal'
opaque morpheme*	*Him-* in *Himbeere* 'rasperry'
proper name†	*Valentin* 'Valentine' in *Valentinstag* 'Valentine's Day'
virtual word form‡	*Zieher* nominalization for 'to pull' (word does not exist in isolation) in *Schraubenzieher* 'screwdriver'
word group†	*drei Zimmer* 'three-room' in *Dreizimmerwohnung* 'three-room flat'

*Property occurs both for modifiers and heads.

†Property occurs for heads only.

‡Property occurs for modifiers only.

§The term *combining form* is taken from Bauer [1983, pages 213ff.].

For many applications, it is helpful to have information about the parts of the compound, as usually the semantic interpretation of a compound is based on the meaning of its constituent parts. Therefore, in the most recent version of GermaNet, all nominal compounds have been identified, split into their constituent parts, and labeled with appropriate linguistic information. Altogether, 62.4% of all nouns in GermaNet are compounds (i.e., 54 759 out of 87 811 noun lemmas).[1] In Chapter 7, some WSD experiments use the information on the splitting of compounds in GermaNet.

3.7 Verbal Frames

All verb senses in GermaNet have frames and example sentences illustrating the sense in question. Verbal frames denote syntactic patterns for verbs (also

[1]The numbers are calculated for GermaNet release 9.0.

referred to as subcategorization or valency). They represent sequences of frame labels that refer to different types of phrasal and sentential categories, including nominal, accusative, dative, genitive, or prepositional objects. They purely encode syntactic structures rather than semantic information. The basic idea of GermaNet's verbal frames is based on the CELEX Lexical Database[1] [Baayen et al., 1995]. In the CELEX terminology [Gulikers et al., 1995], these frame label categories are referred to as complements or complementation codes.

Table 3.4 presents all frame labels that occur in GermaNet's verbal frames – ordered alphabetically.[2] The table briefly explains each frame label and gives a sample sentence[3] which exemplifies the frame label in question. In GermaNet, each example sentence is related to a verbal frame; but verb senses might have further frames without sentences.

Table 3.4: Frame labels occuring in GermaNet's verbal frames.

Frame label		**Explanation and example**
Accusative	AI*‡	Verbal infinitive clause; ↦ *Das Kind kann schwimmen*^{AI}. ('The child can swim.')
	AN*	Noun phrase in accusative case (accusative object); ↦ *Er züchtet Dalmatiner*^{AN}. ('He breeds Dalmatians.')
	AR*	Reflexive pronoun in accusative case (accusative object); ↦ *Ich sonne mich*^{AR}. ('I sun myself.')
	AZ*‡	Verbal infinitive clause with *zu* 'to'; ↦ *Sie begann [zu singen]*^{AZ}. ('She began to sing.')

Continued on next page

[1]Although the underlying idea comes from the CELEX database, the concrete encoding is heavily adjusted for GermaNet. For example, GermaNet provides frames for verb senses rather than for lemmas and uses an easily readable compact notation rather than a complex numeric code. Furthermore, GermaNet includes reflexive, sentential, and subject frame labels which are not included in the original CELEX.

[2]The frame labels are abbreviations of two (or three) letters: basically, the first letter denotes the grammatical function and the second letter subdivides the frame labels by syntactic categories.

[3]Most examples are simplified versions of GermaNet's example sentences.

Table 3.4: *Continued from previous page.*

Frame label		Explanation and example
Adverbial	BC*	Causative adverbial, can be realized by an adverb or by an adverbial or prepositional phrase; ↦ *Seine Augen leuchten [vor Freude]^{BC}.* ('His eyes glow with joy.')
	BD*	Directive adverbial, can be realized by an adverb or by an adverbial or prepositional phrase; ↦ *Sie blickt [in die Ferne]^{BD}.* ('She looks into the distance.')
	BL*	Locative adverbial, can be realized by an adverb or by an adverbial or prepositional phrase; ↦ *Ich bin [im Schwimmbad]^{BL}.* ('I am in the swimming pool.')
	BM*	Adverbial of manner, can be realized by an adverb or by an adverbial or prepositional phrase; ↦ *Er ist jung^{BM}.* ('He is young.')
	BO*	Comitative adverbial, realized by a prepositional phrase; ↦ *Sie scherzt [mit ihm]^{BO}.* ('She jokes with him.')
	BR*	Role adverbial, realized by a prepositional phrase; ↦ *Er tritt [als Zeuge]^{BR} auf.* ('He appears as witness.')
	BS*	Instrumental adverbial, can be realized by an adverb or by an adverbial or prepositional phrase; ↦ *Das Baby rasselte [mit seinem Spielzeug]^{BS}.* ('The baby rattles with its toy.')
	BT*	Temporal adverbial, can be realized by an adverb or by an adverbial or prepositional phrase; ↦ *Die Geschäfte öffnen [samstags um 9 Uhr]^{BT}.* ('The shops open Saturdays at 9 o'clock.')
Dative	DN*	Noun phrase in dative case (dative object); ↦ *Sie hört [der Musik]^{DN} zu.* ('She listens to the music.')
	DR*	Reflexive pronoun in dative case (dative object); ↦ *Er merkt sich^{DR} alle Details.* ('He memorizes all details.')
Sentential	DS*	Subordinate clause introduced with *dass* 'that'; ↦ *Dies bedeutet, [dass er gewinnt]^{DS}.* ('This means that he wins.')
	FS*	Interrogative sentence; ↦ *Wir werden klarstellen, [wie es dazu kam]^{FS}.* ('We will clarify how it came about.')

Continued on next page

Table 3.4: *Continued from previous page.*

	Frame label	Explanation and example
Sentential (cont.)	FSo†	Interrogative sentence with question particle *ob* 'whether'/'if'; ↦ *Sie fragt, [ob ich morgen einen Termin habe]^{FSo}.* ('She asks whether I have an appointment tomorrow.')
	FSw†	Interrogative sentence beginning with a 'w'-interrogative pronoun (e.g., *wer* 'who', *was* 'what', *wie* 'how'); ↦ *Der Chef entscheidet, [wer welche Aufgabe übernimmt]^{FSw}.* ('The boss decides who undertakes which task.')
Genitive	GN*	Noun phrase in genitive case (genitive object); ↦ *Er gedachte [seines toten Vaters]^{GN}.* ('He commemorated his dead father.')
Nominative	NE†	Expletive subject *es* 'it'; ↦ *Es^{NE} nieselt.* ('It drizzles.')
	NG†	A second noun phrase or adjectival phrase in nominative case besides the subject (in German *Gleichsetzungsnominativ* or *Prädikatsnominativ*); ↦ *Er ist [ein Künstler]^{NG}.* ('He is an artist.')
	NN†	Noun phrase in nominative case which represents the subject of the sentence; ↦ *[Die Benzinpreise]^{NN} steigen.* ('Petrol prices rise.')
Prep.	PP*	Prepositional phrase; ↦ *Der See glänzt [im Mondschein]^{PP}.* ('The lake glistens in the moonlight.')

*Frame label can be specified either as obligatory or as optional.

†Frame label is always specified as obligatory.

‡Strictly speaking, verbal infinitive clauses do not encode case. However, since infinitive clauses both with and without *zu* 'to' can often be substituted by accusative objects, the corresponding frame labels *AI* and *AZ* are assigned to the *accusative* group in the table.

If a verbal frame requires more than one frame label, the frame labels are concatenated by dots ('.'). For example, the frame *NN.DN.AN.BD* consists of the four frame labels[1] of a nominative subject (*NN*), a dative and an accusative object (*DN* and *AN*), and a directive adverbial (*BD*).

While most of the frame labels are obligatory, some frame labels can be specified as optional. Optional frame labels are indicated by a lower case second letter, e.g., *NN.Dn.AN.Bd* specifies the same frame as above with optional dative and adverbial frame labels. Table 3.4 marks all frame labels that can

[1]Four is the maximum number of frame labels combined in one verbal frame.

be specified either as obligatory or as optional with a superscript * and frame labels that exist only in an obligatory version with a superscript †.

Verbs differ in the degree of correlation between word senses and frames, ranging from total correlation to a complete lack of correlation. That is, while the syntactic structure in which a verb occurs is highly predictive of different word senses for some verbs, syntactic structures are not informative enough to distinguish word senses for other verbs. The manually sense-annotated verbs in the TüBa-D/Z treebank are selected to represent a mixture of correlations among word senses and frames – see Section 5.3. The impact of features encoding verbal frame information on machine learning systems for automatic word sense disambiguation is tackled in Chapter 8.

3.8 Data Formats for GermaNet

Since GermaNet 7.0, the official release package contains two data formats:

XML format The GermaNet XML format was initially developed by Kunze and Lemnitzer [2002], but modifications of the GermaNet data itself led to an adapted XML format. An updated version (for GermaNet 5.3) is presented in Henrich and Hinrichs [2010b]. The most recent version (for GermaNet release 9.0, as of April 2014) subsumes four different kinds of XML files described by four DTDs. Firstly, the XML files which represent all synsets and lexical units of GermaNet are organized around the three word classes included in GermaNet: nouns, adjectives, and verbs. Since the semantic space for each word class is divided into a number of semantic subfields (see Subsection 3.2), there are altogether 54 such synset files. Secondly, one XML file contains all relations, both conceptual and lexical. Thirdly, the interlingual links to the Princeton WordNet (see Subsection 3.5) are stored in a separate file. Fourthly, three files – one per word class – contain the recently introduced GermaNet-Wiktionary alignment, including the Wiktionary sense descriptions that extend several lexical units from GermaNet (see Chapter 4 for more details).

Relational database The working development copy of GermaNet was converted into a relational database – as described in Henrich and Hinrichs [2010a]. The database model follows the internal structure of GermaNet, which means that there are tables to store synsets, lexical units, conceptual and lexical relations, etc. The complete database structure is detailed in Appendix B. Since GermaNet 7.0, a dump of the relational database is included in the official release package.

3.9 Coverage of GermaNet

GermaNet aims for a broad, general, and domain-independent language coverage of the three word classes of adjectives, nouns, and verbs. It covers only the most common technical terms and includes named entities of locations whereas it excludes named entities of persons. The insertion of new words into GermaNet follows a corpus-based approach. That is, lexicographers process frequency-sorted lists of lemmas, extracted from large text corpora. This approach implies a prioritization of base vocabulary. However, at a certain point – when all base vocabulary of the German language has been included – the lemma lists also itemize less frequent words. Concerning the current status of GermaNet, potentially new entries for nouns, for instance, mainly cover compounds whereas potentially new entries for verbs often constitute particle verbs.

The development of GermaNet started in 1997, and is still in progress. Since the resource is publicly released on a yearly basis, six official GermaNet releases were published during the work on this dissertation. Table 3.5 presents coverage numbers for the four most relevant versions (see Appendix A for detailed descriptions).

Comparing the coverage of the first listed release 6.0 from April 2011 with the most recent release 9.0 as of April 2014, the table reports an increase of more than 130% in the number of synsets, lexical units, and literals. This substantial extension requires a consequent treatment and consistent documentation of GermaNet releases throughout the work of this dissertation. It

Table 3.5: Comparison of relevant GermaNet releases.

	Official GermaNet release			
	6.0	**7.0**	**8.0**	**9.0**
Release date	04/2011	05/2012	04/2013	04/2014
Synsets	69 594	74 612	84 584	93 246
Lexical units	93 407	99 523	111 361	121 810
Literals, i.e., lemmas	85 214	89 819	100 750	110 738
Lexical units per synset	1.34	1.33	1.32	1.31
Conceptual relations	81 852	87 115	96 925	105 912
Lexical relations*	3 562	3 544	4 081	4 258
Interlingual index (ILI)	–	19 609	26 307	28 661
Wiktionary sense descriptions	–	29 433	29 544	29 475
Split compounds	–	–	40 474	54 759

*Since the synonymy relation is not explicitly encoded in GermaNet but indirectly determined by the grouping of lexical units into synsets, the specified numbers of lexical relations exclude synonymy.

further presupposes the adjustment and update of all sense-annotated corpora to the same, most-recent version of GermaNet (see Chapter 6), in order to make the experimental WSD results for the three corpora comparable and up-to-date.

Each task and experiment of this thesis uses the most recent version of GermaNet that was available when the work commenced. Thus, several GermaNet releases serve as a basis for different tasks; and, vice versa, several manual and (semi-)automatic extensions entered a new GermaNet release. In all of the following chapters, the corresponding GermaNet version is specified for each task described. Appendix A provides a detailed summary of all significant GermaNet releases referenced in this thesis; including information on which release serves as the basis for which task, and which release contains which newly contributed information.

Part II

Preparation of the Resources

Chapter 4

Aligning GermaNet with Wiktionary

Sense definitions[1] in a dictionary or wordnet serve a number of important functions:

(i) They help to distinguish different senses of a word both for humans and computers. Especially in those cases where a sense does not have synonyms that allow identifying the sense.

(ii) They enhance the usability of dictionaries and wordnets for a wide variety of NLP applications, including word sense disambiguation, machine translation, information retrieval, and semantic similarity measures.

(iii) They facilitate the sense alignment of dictionaries or wordnets with other lexical resources.

Although wordnets pursue the relational approach of defining word senses (see Section 1.2), the Princeton WordNet for English provides sense definitions for most of its synsets. Mainly for the reason stated by Miller [1995, page 40]: *not enough semantic relations are encoded into WordNet to support such constructions. Following standard lexicographic practice, definitional glosses are included in most synsets.* For example, the synset *{pipe, tube}* is explained as *a*

[1] As stated in Footnote 2 on page 58, the terms *definition*, *gloss*, and *sense description* are used interchangeably throughout this thesis.

hollow cylindrical shape. While such definitions or descriptions are useful – not only for the three points (i) to (iii) listed above – many wordnets for languages other than English – including GermaNet – lack comprehensive coverage of such definitions.[1] Rather, these wordnets rely on the implicit mutual disambiguation of word senses by the members of a synset. For example, for the synsets *{pipe, tobacco pipe}* and *{pipe, tube}*, the contrast between the lexical units *tobacco pipe* and *tube* indicates which senses are documented by the two synsets. This type of implicit disambiguation has its limits for those words where the individual senses are synsets with only one member. For example, GermaNet contains two senses for the word *Pfeife*, which can either refer to a whistle or a tobacco pipe, with each sense represented by a single lexical unit as the only member of a synset (see the right part of Figure 4.1).

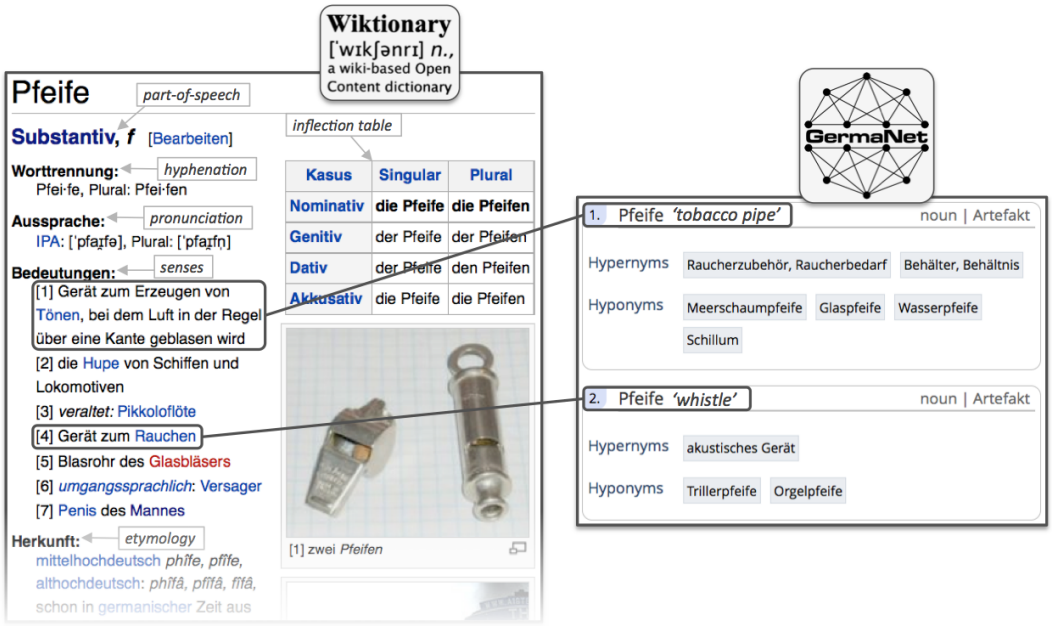

Figure 4.1: Sense mapping example using *Pfeife* 'tobacco pipe; whistle'.

In order for humans to manually annotate word occurrences in a corpus with senses from GermaNet (which is the topic of the next Chapter 5), the distinction of senses has to be clear in the minds of the annotators. Without

[1]The reason for this lack of sense definitions is an entirely pragmatic one: the inclusion of descriptions in a wordnet requires considerable human resources, which are often not available.

sense definitions, this distinction is difficult as the *Pfeife* example has illustrated. Furthermore, several WSD algorithms rely on the existence of sense definitions (see Chapter 7). That is, having sense definitions in GermaNet is obviously beneficial for the purposes of this dissertation, but GermaNet's coverage of definitions is far from complete and does not include, e.g., the synsets for *Pfeife*. Prior to the research reported here[1], only 10% of all synsets in GermaNet were accompanied by definitions. Given the broad coverage of GermaNet, adding descriptions to the missing 62 582 synsets by purely manual, lexicographic work would be an arduous task. Therefore, the possibility of employing automatic or semi-automatic methods for adding sense descriptions would be extremely valuable.

The purpose of this chapter is to explore this possibility on the basis of Wiktionary, a freely available, web-based dictionary containing sense definitions. The two above mentioned senses of *Pfeife* in GermaNet are defined as *Gerät zum Erzeugen von Tönen, bei dem Luft in der Regel über eine Kante geblasen wird* 'instument for producing sounds, where air is usually blown across an edge' and *Gerät zum Rauchen* 'instrument for smoking' in Wiktionary (see the left part of Figure 4.1). The idea is to automatically harvest Wiktionary's definitions by mapping[2] word senses in GermaNet to the corresponding entries in Wiktionary, as illustrated in Figure 4.1. Such a sense mapping relies heavily on word sense disambiguation, i.e., the task of identifying the correct sense of a word in one resource that matches the corresponding sense of the word in a second resource requires the disambiguation of the matching word senses. In the running example this means that, starting with the first sense of *Pfeife* in GermaNet, an automatic disambiguation algorithm has to decide which of the seven sense descriptions from Wiktionary for the word *Pfeife* matches this 'tobacco pipe' sense.

Note that this chapter is an extended version of work that has previously been published by Henrich et al. [2011]. The described algorithm is the same, but this chapter gives more details on the results. Also note that recently this

[1] That is, for GermaNet release 6.0, April 2011.

[2] Note that the terms *mapping* and *aligning / alignment* are used interchangeably throughout this thesis.

chapter has independently been published [Henrich et al., 2014b].

The following Section 4.1 briefly introduces the web-based dictionary Wiktionary and motivates why this resource has been used for the automatic harvesting of sense definitions. A word sense alignment algorithm is developed that performs the automatic harvesting of sense definitions for GermaNet senses – see Sections 4.2 and 4.3. A comparison of different setups of the algorithm yields as the best result an accuracy of 93.8% and an F_1-score of 84.3, which confirms the viability of the proposed method for automatically enriching GermaNet – see Sections 4.4 and 4.5. Related work on aligning wordnets with Wiktionary is presented in Section 4.6.

4.1 Wiktionary

Wiktionary[1] is a web-based dictionary that is available for many languages, including English and German. As is the case for its sister project Wikipedia, it is written collaboratively by volunteers and is freely available[2]. The online dictionary follows the idea of a traditional dictionary by distinguishing several meanings of a word. Distinct word senses are identified by sense descriptions and accompanied with example sentences illustrating the sense in question.

Furthermore, Wiktionary provides information such as parts of speech, hyphenation, possible translations, inflection, etc. for each word. It includes, among others, the same three word classes of adjectives, nouns, and verbs that are also available in GermaNet. Wiktionary provides relations to other words, e.g., in the form of synonyms, antonyms, hypernyms, hyponyms, holonyms, and meronyms. In contrast to GermaNet, the relations are (mostly) not disambiguated.

As an example, a part of a word entry is shown on the left side of Figure 4.1. It shows, inter alia, information on the word's part of speech (*Substantiv, f*), hyphenation (*Pfei-fe, Plural: Pfei-fen)*, inflection (see the table with inflected word forms for all four German cases in singular and plural), and a list of

[1] See `http://www.wiktionary.org`

[2] Wiktionary is available under the Creative Commons Attribution/Share-Alike license `http://creativecommons.org/licenses/by-sa/3.0/deed.en`.

descriptions for the seven distinguished word senses.

The reason why Wiktionary has been chosen to harvest sense definitions for GermaNet is threefold.

(i) Wiktionary is freely available and its license allows the redistribution of harvested materials.

(ii) There is a freely available Java-based library JWKTL[1] [Zesch et al., 2008] that allows accessing all Wiktionary data programmatically.

(iii) The overlap of terms that are in both resources is large enough, as reported by a survey of the overlaps of GermaNet and Wiktionary which shows that, disregarding word sense disambiguation, about 30,488 terms (45.23%) in GermaNet are also present in the German Wiktionary [Zesch, 2010b].

For the present project, a downloaded copy of the German Wiktionary as of February 2, 2011 is utilized, consisting of 46 457 German words comprising 70 339 word senses.

4.2 The Idea of the Alignment Algorithm

In the current scenario of Wiktionary and GermaNet, the aim is to correctly map a GermaNet lexical unit to a Wiktionary sense definition in order to harvest sense definitions from Wikionary. This task includes word sense disambiguation, i.e., given a sense from one resource it has to identify (disambiguate) the correct matching sense from the other resource. For each lemma (also referred to as the *target word*, i.e., the word under consideration) contained in GermaNet, it takes all lexical units representing that lemma and tries to disambiguate which of Wiktionary's senses for that lemma is the correct match.[2]

[1] `http://www.ukp.tu-darmstadt.de/software/jwktl`

[2] Note that this procedure of collecting candidate pairs coincides with the work by Meyer and Gurevych [2011] (see related work in Section 4.6) on aligning WordNet with the English Wiktionary. Eventually – on the level of synsets – both studies (theirs and the one presented in this chapter) have collected as the candidate alignments for a synset all senses from Wiktionary for all synonymous words in that synset.

There can be more than one occurrence of a target word in GermaNet, thus a target word can correspond to a number of lexical units, each belonging to a distinct semantic concept, i.e., synset. The mapping of each sense definition in Wiktionary needs to consider all synsets containing the target word in GermaNet. Further, due to different sense granularities and distinct coverages of Wiktionary and GermaNet, some senses in GermaNet may correspond to more than one, exactly one, or even no senses in Wiktionary. In the other direction, some Wiktionary senses correspond to more than one sense in GermaNet. Even if there is exactly one sense in both resources, this does not necessarily mean that they match. For example, there is exactly one sense for *Angeln* 'fishing' in GermaNet and exactly one sense for *Angeln* in Wiktionary described as *Landschaft im Nordosten Schleswig-Holsteins* 'region in the north-east of Schleswig-Holstein'; but these two senses are clearly distinct.

The other challenge for the mapping is the absence of sense definitions in GermaNet, which prohibits, e.g., simply applying a word overlap disambiguation (such as the Lesk [1986] algorithm, see Subsection 2.3.1) out-of-the-box. Hence, in this chapter a word sense alignment algorithm is developed, which accommodates auxiliary information from GermaNet and Wiktionary to enable a word overlap approach.

Lexical fields: Therefore, the notion of *lexical fields* is introduced. Lexical fields substitute sense descriptions in GermaNet by encapsulating relations and semantic field information.[1] That is, lexical fields are a bag of words that represent the lexical unit in question. Figure 4.2 visualizes the extraction of the lexical field information for one sense of the word *Eisen* 'iron' in GermaNet. All lexical units (the items in the boxes with a white background in Figure 4.2) related to the target word – either directly by a lexical relation or indirectly by a conceptual relation – are extracted.[2] For example, the synonym *Ferrum* 'ferrum', the hypernyms *Schwermetall* 'heavy metal', *Mineralstoff* 'mineral',

[1] A similar kind of technique using all related words for constructing *pseudo glosses* has been used by Gurevych [2005] for the purpose of computing semantic relatedness for any two words in GermaNet.

[2] As indicated by the panel (with the caption *key*) in the top left part of Figure 4.2, conceptual relations connect synsets whereas lexical relations connect lexical units contained in synsets.

etc., the holonyms *Eisenerz* 'iron ore', *Stahl* 'steel', etc., the hyponyms *Magnet* 'magnet', *Gusseisen* 'cast iron', etc. – to name only a few. In addition to the terms obtained via lexical and conceptual relations, the lexical fields are further enriched by the semantic field (see Section 3.2) that the target word belongs to. The word *Eisen*, for example, belongs to the semantic field *Substanz* 'substance'. This is why the term *Substanz* is added to the lexical field. All words that have been added to the lexical field in the example are listed on the right in Figure 4.2.

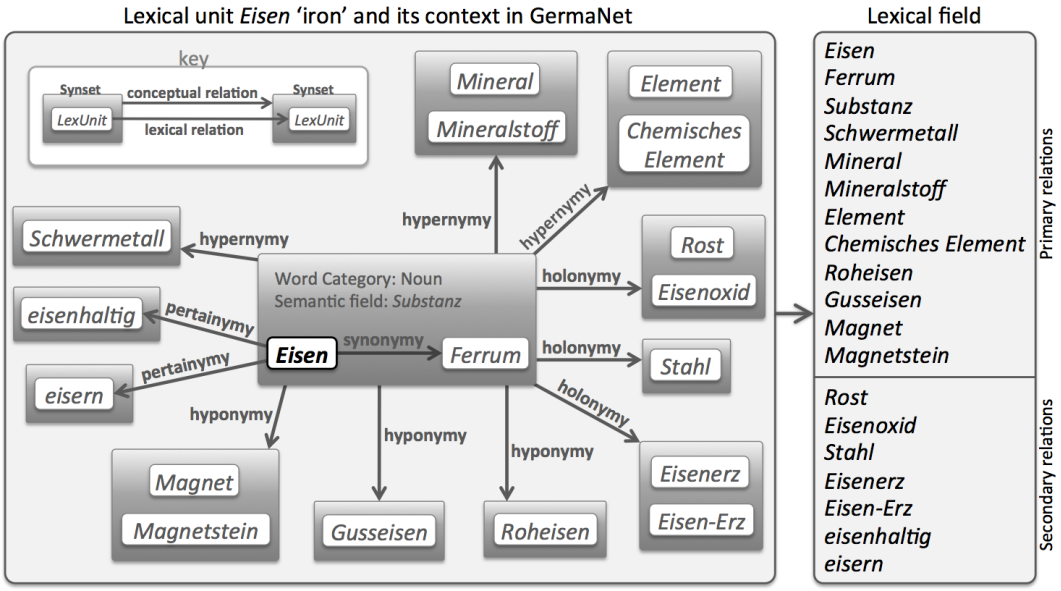

Figure 4.2: Lexical field example using *Eisen* 'iron'.

Given a target word, the alignment algorithm counts the overlaps between each of Wiktionary's sense definitions and each of the lexical fields representing lexical units in GermaNet. For example, the lexical field of *Eisen* 'iron' in GermaNet contains the hypernym *Chemisches Element* (see the box on the right in Figure 4.2), which appears in the first sense definition of *Eisen* in Wiktionary described as *Chemie, ohne Plural: chemisches Element, silberweißes, bei Feuchtigkeit leicht oxidierendes Metall* 'Chemistry, without plural: chemical element, silver-white, on dampness easily rusting metal'.

Coordinated relations: Besides the application of lexical fields for counting word overlaps, the alignment algorithm utilizes the occurrence of the same

relations in GermaNet and Wiktionary – which is referred to as *coordinated relations*. As mentioned in the introductory sections on Wiktionary (Section 4.1) and GermaNet (Section 3.3), the two resources have several relations in common, for example the hypernymy and the hyponymy relations. Thus, if a lexical unit in GermaNet and a sense in Wiktionary both show the same hypernyms, this is a strong indicator for their equality. For example, *Eisen* in the sense of 'iron' in GermaNet and the first sense of *Eisen* in Wiktionary both show, among others, the same hypernym *Schwermetall* and the same two hyponyms *Roheisen* and *Gusseisen*, which are a good indicator that these two entries express the same semantic concept.

Utilizing the overlap information of lexical fields and coordinated relations is the underlying idea of the developed alignment algorithm.

4.3 Implementation of the Alignment Algorithm

Preprocessing: Wiktionary sense descriptions are tokenized and stopwords, such as determiners, are withdrawn. All words are normalized using either stemming (Snowball stemmer [Porter, 1980]) or lemmatization (TreeTagger [Schmid, 1994]).[1] Since compounding is a highly productive word formation process in German [Eisenberg, 2006], splitting compounds that occur in the sense descriptions in Wiktionary and in the lexical fields in GermaNet increases the overlap rate. For instance, after splitting the compound *Wasserwelle* 'waterwave' into its two components *Wasser* 'water' and *Welle* 'wave', an overlap (of *Wasser*) with the first sense definition in Wiktionary, i.e. *Physik: Erhebung von Wasser* 'physics: elevation of water', can be captured. Duplicates, which arise due to compound splitting, are eliminated to avoid multiple overlap counts for the same word. For example, one sense of the word *Welle* has the compound *Wasserwelle* as a synonym and thus the compound appears in the lexical field of *Welle*. Compound splitting therefore results in two occurrences of *Welle* in the same lexical field (of which one occurrence is eliminated to avoid multiple counts).

[1] Several experiments with stemming and lemmatization yielded better results with stemming. Thus, all below described experiments use stemming as a preprocessing step.

Different versions: Basically, two versions of the alignment algorithm are implemented, which can be run separately or in combination.[1] The first variant utilizes the lexical fields in GermaNet as described above. All words are included into the lexical field that are directly connected[2] to the target word. For experimenting with different sets of words, two types of relations are distinguished: *primary relations*, such as synonymy, hypernymy, and hyponymy, constituting the fundamental structure of a wordnet; and *secondary relations*, such as association, causation, entailment, holonymy, meronymy, and pertainymy with a subordinated importance. In the previous example of *Eisen* 'iron', all words that are connected by a primary relation are listed above the line in the box on the right in Figure 4.2, and all words connected by a secondary relation are listed below the line. An overlap of single words is calculated between a tokenized Wiktionary sense description and a lexical field belonging to a target lexical unit in GermaNet.

The second variant of the alignment algorithm counts the overlaps of coordinated relations between GermaNet and Wiktionary (as explained in Section 4.2). Therefore, all relations that occur in both resources, such as synonymy, antonymy, hypernymy, hyponymy, meronymy, and holonymy, are considered.

Overlap count: More precisely, each of the target lexical units in GermaNet is represented by a lexical field (as described in Section 4.2). An overlap is calculated between a lexical field in GermaNet and a sense description in Wiktionary. The overlap is a mere count of the number of words x_i for $x_i \in X$, where X is a set of words representing the lexical field of a target lexical unit in GermaNet found in the set of words of a Wiktionary sense description – optionally augmented by the coordinated relations overlap. Further, in case that there is exactly one sense in both resources for a given word, an initial

[1] Additionally to these options the algorithm can be run in a case-sensitive and in a case-insensitive mode. By distinguishing nouns, which are always capitalized in German, case-sensitivity automatically reduces the number of potentially erroneous combinations, and thus in several runs of the algorithm the case-sensitive mode always outperformed the case-insensitive mode. This is why all reported experiments rely on a case-sensitive setup.

[2] Here, *directly connected* means that the path length between two words is exactly one – disregarding the type of relation (lexical or conceptual).

count of 1 is given to the overall count of overlaps.[1]

The alignment algorithm maps the Wiktionary sense definitions with the highest overlap counts to a given lexical unit in GermaNet and disregards all other overlap counts (even if those are above zero).[2] Notice that the overlap calculation can result in the same overlap score for several senses in Wiktionary. In these cases, more than one Wiktionary sense definition is mapped onto the lexical unit in question[3] and is taken to mean that the lexical unit in GermaNet is jointly described by the Wiktionary sense descriptions in question.

Algorithm setups: As the lexical fields do not consist of continuous word sequences but rather of single words, it is not possible to give more weight to longer sequences of word matches as it is done by Banerjee and Pedersen [2003]. In order to be able to fine-tune the most reliable set of relations, the algorithm includes the possibility of specifying individual weights for different relations.

A set of alignment experiments were conducted that differ from each other in the weight assigned to the terms that make up the lexical field of a given GermaNet lexical unit (see Table 4.1). For setups A – C, only the terms contributed by a single primary relation are considered (given a non-zero weight). Setup D considers only all secondary relations and setup E only all coordinated relations. In setup F, all terms obtained by the primary, secondary, and coordinated relations are given equal weight. In addition, a set of experiments has been conducted where the terms obtained by the different relations were

[1]Needless to say, assigning an arbitrary count of at least 1 to the overlap score between words occurring exactly once in both resources results in a positive mapping of these two senses which, in turn, results in a prediction of false positives for all cases, where those senses do not match (see the example of *Angeln* in Section 4.2 above). However, such cases are rare and therefore the heuristic in question works well in practice (see the evaluation section below).

[2]The use of a threshold that defines the minimum overlap count an alignment candidate should have in order to be classified as correct (which has proven useful in related work, e.g., when used with Personalized PageRank and cosine on word vectors [Meyer and Gurevych, 2011; Niemann and Gurevych, 2011]) is not applicable here, since the absolute overlap counts are skewed and can differ enormously for different target words – mainly depending on the number of words that the lexical fields and the sense descriptions in Wiktionary contain. Further, experiments with more sophisticated calculations, such as introducing a dynamic threshold by determining the average of all overlap counts for a word and defining all counts that are above this average as a predicted mapping, did not show noticeable improvements.

[3]Note that this stands in contrast to other approaches, e.g., Niemann and Gurevych [2011], who have assigned only the one most similar alignment candidate as this has shown best performance in their experiments on mapping WordNet to Wikipedia.

given different weights. Setup G shows the weight assignments that produced optimal results for a precision and recall evaluation (see Section 4.4 below).

Table 4.1: Different algorithm setups (numbers indicate weights).

Setup	Lexical field overlap				Coordinated relations
	Primary relations			Secondary relations	
	Hyper.	Hypo.	Syno.		
A	1	0	0	0	0
B	0	1	0	0	0
C	0	0	1	0	0
D	0	0	0	1	0
E	0	0	0	0	1
F	1	1	1	1	1
G	2	0.5	3	0.5	3

4.4 Evaluation

In order to be able to evaluate the automatic alignment of lexical units in GermaNet with senses in Wiktionary, the mappings produced by the developed disambiguation algorithm were manually checked by two experienced lexicographers. In order to ensure a comprehensive evaluation of lexical items with different degrees of polysemy, the evaluation reports results for five different polysemy classes: words having (i) one sense in GermaNet, (ii) two senses in GermaNet, (iii) three or four senses, (iv) five to ten senses, and (v) more than ten senses in GermaNet. Table 4.2 shows the total number of words in each polysemy class for the three word classes contained in GermaNet that were available for the evaluation. Altogether, 20 997 distinct words[1] with an average of 1.3 senses (i.e., 27 309 lexical units of which 3 241 are adjectives, 19 423 are nouns, and 4 645 are verbs) were manually checked by the lexicographers.

Since the number of senses assigned to a word in GermaNet and Wiktionary may differ and since the lexical coverage of the two resources only partially coincides, a given word sense in GermaNet may have no counterpart

[1] The numbers from Table 4.2 do not exactly add up to 20 997 because some words belong to more than one word class.

Table 4.2: Evaluated words and their sense distributions.

No. of senses	Adjectives	Nouns	Verbs
1	2 328	13 391	1 393
2	319	1 872	510
3 – 4	71	557	320
5 – 10	8	91	135
> 10	0	1	23
Total	2 726	15 912	2 381

in Wiktionary at all, or it may correspond to exactly one or more than one senses in Wiktionary. The same holds true in the inverse direction. Figure 4.3 illustrates the range of possible mappings for the word *Archiv*, for which each resource records three distinct senses. Accordingly, three times three sense combinations need to be considered. This number rises exponentially with more available senses. The solid arrows in Figure 4.3 denote the correct mappings: the first sense in GermaNet ('data repository') corresponds to the first sense in Wiktionary, the second sense in GermaNet ('archive') corresponds to both the second and third senses in Wiktionary, and the third sense in GermaNet ('archived file') does not map onto any sense in Wiktionary.

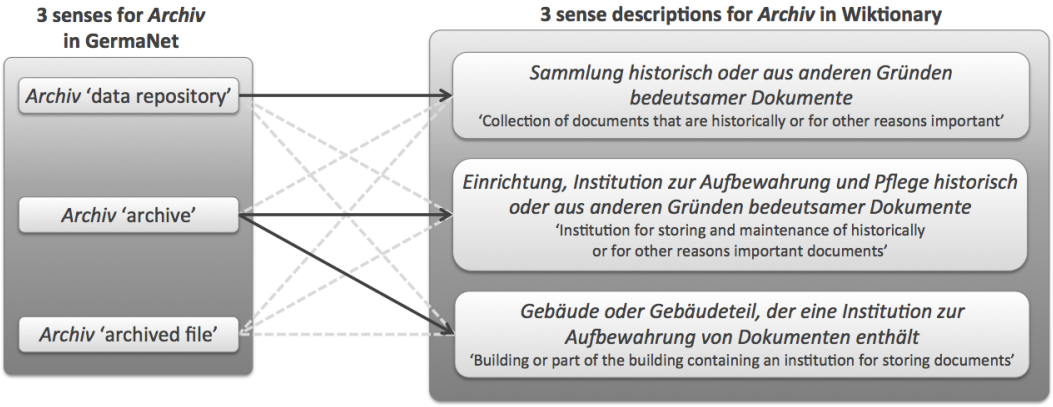

Figure 4.3: Sense mapping example using *Archiv* 'data repository; archive; archived file'.

A truly meaningful evaluation has to reflect the nature of the task at hand: the semi-automatic enrichment of lexical units in GermaNet with appropriate sense descriptions from Wiktionary. A very crude approach would simply map

all word senses recorded in GermaNet for a given word with all corresponding sense descriptions documented in Wiktionary. Such an approach would require a lot of human post-editing to eliminate all inappropriate mappings. Since the unwanted mappings far outnumber the number of correct mappings, this approach would be clearly inappropriate. In fact, the motivation behind the developed alignment algorithm is precisely to map only plausible candidates for correct mappings in terms of word overlap between lexical fields in GermaNet and sense descriptions in Wiktionary. These considerations clearly show that the task at hand requires maximizing accuracy in order to minimize the amount of human post-processing required.

The calculation of accuracy, precision, and recall for the task of sense alignment differs slightly from the calculation for the task of word sense disambiguation as defined in Subsection 2.2.1. In order to calculate the evaluation metrics for the sense alignment task, sets of correct and incorrect mappings are defined:

- *True positives:* all correctly identified mappings for a word.
- *False positives:* all erroneously indicated mappings for a word.
- *True negatives:* all candidate mappings for a word that are correctly not identified as a mapping.
- *False negatives:* all erroneously not identified mappings for a word.

Accuracy, precision, recall, and F_1 are computed per word as follows. *Accuracy* is calculated as the percentage of sense alignments that are correctly identified plus the sense alignment candidates that are correctly not mapped by the alignment algorithm, out of all candidate sense mappings for a word (i.e., all combinations of GermaNet senses with sense descriptions from Wiktionary for a word in question):

$$\textit{accuracy of the alignment per word} = \frac{TP + FP}{TP + TN + FP + FN} \tag{4.1}$$

Recall is defined as the percentage of word senses that are correctly mapped

by the system, out of all word senses that are supposed to be aligned:

$$\textit{recall of the alignment per word} = \frac{TP}{TP + FN} \tag{4.2}$$

Precision is reported as the percentage of word senses that are correctly mapped, out of all word senses that the alignment mapped:

$$\textit{precision of the alignment per word} = \frac{TP}{TP + FP} \tag{4.3}$$

The F_1-measure is calculated per word using Formula (2.4) defined in Subsection 2.2.1 using the above definitions for precision and recall. The overall values for accuracy, precision, recall, and F_1 are then computed as the average of all word values.

4.5 Results

Table 4.3 shows the results for the described task separately for the previously defined polysemy classes (columns). The rightmost column depicts the overall results without classifying words with respect to their number of different senses. The rows show the different algorithm setups A – E (described in Section 4.3) separately for each of the three word classes of adjectives, nouns, and verbs (column *POS*). Rows marked with *All POS* denote results for all word classes.

To begin with, the average scores for all three word classes in all setups A – G are above 90% accuracy. This suggests that human correction is needed for only one out of 10 mappings between GermaNet and Wiktionary suggested by the algorithm.[1] This underscores the overall feasibility of the approach.

As setups A – E each take into account only one relation type (see Table 4.1), a comparison of the results directly reflects the suitability of the different relation types when applied independently of each other. One of the most striking findings among the results of this evaluation is that the use of hy-

[1]To be even more precise, the average accuracies for setups A to E are actually above 93% and thus human correction is needed for only one out of 14 mappings.

Table 4.3: Accuracy of the alignment.

Setup	POS	Number of senses in GermaNet					
		1	**2**	**3 – 4**	**5 – 10**	**> 10**	**All**
A	Adj.	94.7%	83.2%	93.6%	99.0%	N/A	93.3%
	Nouns	94.3%	89.0%	92.9%	90.0%	100%	93.6%
	Verbs	92.3%	87.0%	92.2%	93.6%	85.8%	91.2%
	All POS	94.2%	88.0%	92.7%	92.4%	86.4%	93.3%
B	Adj.	94.4%	79.3%	91.9%	99.1%	N/A	92.6%
	Nouns	94.2%	88.4%	89.9%	87.0%	0%	93.3%
	Verbs	92.3%	88.5%	94.6%	92.9%	85.4%	91.8%
	All POS	94.1%	87.3%	91.7%	90.8%	81.9%	93.1%
C	Adj.	95.0%	81.9%	93.9%	97.9%	N/A	**93.4%**
	Nouns	94.6%	90.2%	93.6%	92.3%	0%	**94.0%**
	Verbs	92.3%	89.7%	96.9%	97.3%	97.8%	92.7%
	All POS	94.5%	89.1%	94.8%	95.4%	93.7%	**93.8%**
D	Adj.	93.1%	81.1%	93.6%	99.4%	N/A	91.7%
	Nouns	94.2%	89.4%	94.5%	91.0%	0%	93.6%
	Verbs	92.4%	87.7%	96.8%	94.7%	92.4%	92.1%
	All POS	93.9%	88.1%	95.2%	93.5%	88.5%	93.2%
E	Adj.	94.1%	85.3%	91.8%	97.3%	N/A%	93.0%
	Nouns	94.4%	87.2%	88.1%	85.6%	100%	93.3%
	Verbs	92.6%	90.5%	97.3%	95.6%	97.1%	**93.0%**
	All POS	94.2%	87.6%	91.5%	91.8%	97.3%	93.2%

pernyms (setup A) and synonyms (setup C) outperforms the use of hyponyms (setup B) and secondary relations (setup D). The fact that hypernyms and synonyms outperform the other relations is not surprising since sense definitions often refer to a hypernym or synonym term which is then described in more detail to fit the specific properties of the entity being described. For example, the English WordNet defines the noun *convertible* as 'a car [hypernym] that has a top that can be folded or removed'.

The single application of coordinated relations (setup E), in turn, is better than all previous setups (for verbs) or among the best (for adjectives). Again, this result is hardly surprising since coordinated relations are present when the same two terms are connected by the same lexical relations in both resources (see Section 4.2 for a more detailed explanation of coordinated relations). Such a scenario is highly predictive for a correct mapping between corresponding

senses in the two resources. By contrast, the results for nouns on setup E do not outperform the other setups. One explanation for this lower performance[1] is the preference of related terms (setups A – D) being often referred to in definitions of nouns with which the occurrence of coordinated relations cannot compete.

A comparison of the results for the three different word classes yields the following tendencies: the results for nouns are slightly higher for all of the different lexical relations (93.3% to 93.6%) than the results for the other two word classes (91.7% to 93.4% for adjectives and 91.2% to 93.0% for verbs). One explanation for this higher performance must be, that related terms are often referred to in order to describe nominal concepts.

For verbs, the hypernymy relation (setup A) seems to perform particularly poorly. The explanation for this low performance is probably, that definitions of verb senses rarely use hypernyms, which supports the assumption that verbs are usually defined differently than adjectives and nouns.

For words with one sense or with five to ten senses adjectives almost always outperform the other two word classes, whereas for words with two senses adjectives show lowest performance compared to nouns and verbs.

Perhaps the three most remarkable observations when comparing the results for the different polysemy classes in Table 4.3 are:

(i) The drop in performance for all three word classes for words with exactly two senses. Compared to the performances for words having one sense there is a drop of between 2.0% to 15.1% for words having two senses for all algorithm setups.

(ii) The increase in performance for all three word classes for words with 3–4 senses compared to words with 2 senses in GermaNet.

(iii) There is no regularity in performance for words with more than 4 senses in GermaNet.

[1]Denoting the performance as *lower* is meant in a relative sense, i.e., compared to the results for the other setups for nouns. Note that setup E for nouns does not perform lower than setup E for adjectives and verbs.

At first glance, one would suspect that the performance of the algorithm would decrease as the number of senses increases. In other words, the difficulty of the task of aligning sense definitions with lexical units should increase with the number of senses available. This expectation holds true for the comparison of words with one and two senses (see (i) above) but is not empirically confirmed for words with more than two senses (see (ii) and (iii) above). The real explanation for what looks like contradictory findings has to do with the ratio of true positives and true negatives. For the task at hand of aligning sense definitions with word senses in GermaNet, the true negatives outnumber the true positives, and the ratio between the two becomes more and more skewed as the number of word senses and the number of corresponding sense descriptions increases. The fact that the alignment algorithm shows no attested degradation in performance for highly polysemous words attests to the suitability of the algorithm for the task at hand.

What still remains to be explained is why words with exactly one sense do not necessarily show best performance. This is due to the heuristic that adds a count of one to the count of overlaps in cases where there is exactly one sense in both resources.

The extreme scores of 0% and 100% for the nouns having more than ten senses in GermaNet also require some explanation. The explanation is simple: there is only one noun with more than ten senses. It is the word *Dollar*, which has a total of 15 different senses denoting national currencies such as *US-Dollar*, *Canadian Dollar*, *Hongkong-Dollar*, etc. with one common hypernym *Währungseinheit* 'currency unit'. In Wiktionary, there is one sense definition for *Dollar* with the wording *Währungseinheit in verschiedenen Staaten, z.B. den USA und Kanada* 'currency unit in different countries, e.g., in the USA and Canada'. Since the hypernym *Währungseinheit* matches the Wiktionary sense description, the alignment algorithm detects an overlap for each sense resulting in a 100% score for setup A which considers only the hypernymy relation. The same holds true for setup E (coordinated relations) since the GermaNet sense of *Dollar* and the Wiktionary sense description of *Dollar* have the same hypernym *Währungseinheit*. For the other relations (setups B, C, and D) the score is 0% because there is simply no lexical overlap for any of

them.

As mentioned above, for the task at hand, the true negatives outnumber the true positives by a wide margin, and this distribution becomes more and more skewed as the number of senses for a word increases. For this very reason, an accuracy-based evaluation is particularly important since it takes false positives into account. However, apart from the accuracy of the mappings proposed by the algorithm, the recall behaviour of the alignment algorithm is also relevant. Poor recall would mean that many empirically correct mappings go undetected by the algorithm and therefore have to be manually added. Therefore, recall is also computed for different setups of the algorithm (see Table 4.4). Recall of the single application of hypernyms (setup A) and coordinated relations (setup E) is better than all other setups of single relations (setups B, C, and D). Again, this result is hardly surprising since coordinated relations are present when the same two terms are connected by the same lexical relations in both resources.

Table 4.4: Accuracy, precision, recall, and F-measure.

Setup	**Accuracy**	**Recall**	**Precision**	**F_1**
A	93.3%	72.1%	71.3%	71.7
B	93.1%	61.2%	60.8%	61.0
C	93.8%	63.8%	63.4%	63.6
D	93.2%	61.3%	60.8%	61.0
E	93.2%	73.6%	73.5%	73.5
F	92.3%	83.8%	82.8%	83.3
G	91.9%	84.6%	84.1%	84.3
Baseline	53.7%	50.7%	44.2%	47.2

The best recall values are obtained by those settings where all relations are taken into account for the construction of the lexical field (setups F and G). Notice also that, compared to accuracy, there is a much wider spread in the results for recall, ranging from 61.2% (setup B) to 84.6% (setup G). This is hardly surprising since recall is bound to improve with the number of terms included in the lexical field as candidate for overlap.

For completeness, Table 4.4 also contains the scores for precision and F_1. The fact that precision does not rise much above 84% means that there still is an error rate of 16%, i.e., 16% of the proposed links are wrong.

The baseline of randomly mapping Wiktionary senses to lexical units in GermaNet (see row *Baseline* in Table 4.4) demonstrates that the mapping task as such is far from trivial. All setups A to G significantly outperform the baseline. This constitutes strong evidence of the feasibility of the approach.

4.6 Related Work on Aligning Wordnets

The alignment of lexical resources has been widely studied in recent years. Most of the work has focused on English resources – investigating the alignment of the Princeton WordNet with other lexical resources. Early studies reported on mapping WordNet to the Longman Dictionary of Contemporary English and with Roget's thesaurus [Kwong, 1998], to the Hector lexicon [Litkowski, 1999], to the Suggested Upper Merged Ontology [Niles and Pease, 2003], or to the Oxford Dictionary of English [Navigli, 2006] – to name only a few. More recently, several studies investigated the alignment of WordNet with Wikipedia (including Ruiz-Casado et al. [2005], Suchanek et al. [2007], Ponzetto and Navigli [2009, 2010], Toral et al. [2009], Niemann and Gurevych [2011], and Fernando and Stevenson [2012]).

Previous work on aligning German resources is limited to the mapping of GermaNet to the German version of Wikipedia [Henrich et al., 2012a] and to the Digital Dictionary of the German Language (*Digitales Wörterbuch der Deutschen Sprache*[1], DWDS) [Henrich et al., 2014a].

The approach by Meyer and Gurevych [2011], which maps WordNet to the English version of Wiktionary, was developed in parallel to the one presented in this chapter (first published in 2011 [Henrich et al., 2011]). Their alignment has very much followed the approach by Niemann and Gurevych [2011] who aligned WordNet with Wikipedia – especially in the conception of their alignment algorithm, i.e., the application of similarity measures and the use of a threshold. As the candidate alignments for a synset in WordNet Meyer and Gurevych [2011] collected for all synonymous words in that synset all senses from Wiktionary. Following Niemann and Gurevych [2011], they

[1] `http://www.dwds.de`

used the Personalized PageRank algorithm [Agirre and Soroa, 2009] and the cosine on word vectors (previously also used by Ruiz-Casado et al. [2005]) to calculate similarity between a Wiktionary definition and a WordNet synset. In a training phase, they determined a threshold learned from a training set that defines the minimum similarity score an alignment candidate should have in order to be classified as correct. In experiments with different configurations of the measures, a higher precision from the PageRank algorithm and a higher recall from the cosine measure resulted in a better F_1-score for a combination of the two measures. Albeit Meyer and Gurevych [2011] did on English data what is described in this chapter on German, i.e., they automatically aligned senses in Wiktionary and WordNet, their intention differs from the one behind this chapter. Their motivation behind the mapping is to extend the coverage of a joint resource whereas this chapter focuses on the systematic enrichment of an existing resource, i.e., GermaNet, by sense definitions.

In a recent study, Matuschek and Gurevych [2013] developed a graph-based algorithm for automatically aligning several datasets available for German and English including GermaNet and Wiktionary. The GermaNet–Wiktionary mapping described in the present chapter was used as a gold standard in their evaluation. Matuschek and Gurevych [2013] proposed a two step approach supposed to be independent of the underlying resources to be mapped: the first step of their algorithm consists of several approaches to constructing a graph for a lexical resource, where they relied not only on existing relations or hyperlinks, but also introduced new relations with the help of monosemous terms occurring in sense descriptions. In a second step, they performed the actual sense alignment by first aligning all terms that are monosemous in both resources, which connects the two resources by an initial set of relations. Then each polysemous sense was mapped to the candidate sense that is connected by the shortest path (computed with the Dijkstra shortest path algorithm [Dijkstra, 1959]) relying on the initial set of relations between the two resources. They experimented with several configurations and combinations evaluated on existing alignments including WordNet-Wiktionary [Meyer and Gurevych, 2011] and GermaNet–Wiktionary (present chapter) and achieved competitive performance compared to the results by the creators of those alignments.

The study that is closest in spirit to the approach presented here is the one by Gonçalo Oliveira and Gomes [2013]. Their procedure of automatically assigning definitions to synsets in the Portuguese wordnet Onto.PT [Gonçalo Oliveira, 2013] follows the approach proposed in Henrich et al. [2011], which is a previously published version of this chapter. Thus, they also follow the word overlap approach to identify matching word senses in order to harvest descriptions. The work by Gonçalo Oliveira and Gomes [2013] differs from the present work especially in the language and therefore also in the resources used. More specifically, they use three Portuguese dictionaries to harvest definitions for the Portuguese wordnet Onto.PT.

What distinguishes the work in the present work from earlier studies is the fact that all automatically aligned data were manually checked and, if necessary, post-corrected.

4.7 Conclusion and Continuing Work

Sense definitions are a crucial component for wordnets. However, as GermaNet rarely contained sense definitions, comprehensive sense definitions were badly needed in order to enhance its usability for a wide variety of NLP applications. The present chapter has described a method for semi-automatically enriching lexical units in GermaNet with appropriate sense descriptions from Wiktionary. It has resulted in harvested definitions for about 30% of all GermaNet 7.0 senses (i.e., 29 433 out of 99 523, see Table 3.5 in Section 3.9). These definitions have already been included into GermaNet (since release 7.0) and have also been made freely available online[1]. An accuracy of more than 90% suggests that human correction was needed on average for only one out of 10 mappings suggested by the algorithm. Moreover, the estimations of recall and precision result in 84.6% and 84.1%, respectively. These numbers underscore the overall feasibility of the approach and verify its usability for the task at hand. This suggests the applicability of the approach to other resources.

The most obvious uses of the outcome of this chapter's work include:

[1] `http://www.sfs.uni-tuebingen.de/GermaNet/wiktionary.shtml`

- An automatic method for creating a sense-annotated corpus harvested from the web that relies on the mapping between GermaNet and Wiktionary is explored in Section 5.2.
- The harvested definitions help human annotators to understand the meaning of a word when manually sense-annotating a corpus, which is the topic of Section 5.3.
- The impact of the harvested descriptions on WSD algorithms relying on the existence of such sense descriptions are explored in Chapter 7.

A natural next step would also be to implement more elaborate alignment algorithms such as Personalized PageRank, cosine similarity on word vectors, or graph-based Dijkstra that have been used in other approaches (see Section 4.6). Though, a comparison with related work shows that the resulting improvement is likely to be modest.[1] Of course, a better automatic alignment would make the manual post-correction easier for human annotators and thus it would definitely be worth experimenting with more elaborated alignment algorithms, if the mapping between Wiktionary and GermaNet had not yet been completely manually post-corrected and the goal of harvesting definitions from Wiktionary not yet been achieved. Apart from that, as the GermaNet–Wiktionary sense alignment dataset is freely available and one of the goals of this thesis is to boost WSD research on German (and aligning resources heavily includes WSD), other researchers are highly encouraged to use it as a gold standard in their experiments with more elaborated alignment algorithms – as it has already been done by Matuschek and Gurevych [2013].

[1]The only comparable work on the same language and resource pair is the one by Matuschek and Gurevych [2013]. They have reported results that are 4.2% (for recall), 9.9% (for precision), and 2.7 (for F_1-score) higher and 8.8% (for accuracy) lower than the ones presented in this chapter. For this comparison, the setups that report highest numbers are taken. The reason why the accuracy in this chapter is higher than theirs whereas the precision in this chapter is lower than theirs lies in the differing focus of parameter adjudication; in this chapter, the aim is to achieve high accuracy.

Chapter 5

Creating Sense-Annotated Corpora

A *sense-annotated corpus* is a text in which occurrences of words are annotated with their senses from a given sense inventory. A sense inventory is a collection of predefined senses for the words of a certain language, i.e., a dictionary or a wordnet. Sense-annotated corpora serve as a gold standard for the development and evaluation of word sense disambiguation (WSD) systems. The task of WSD is to automatically assign senses of a predefined sense inventory to the word occurrences in a text – see Chapter 2. Manually corrected sense-annotated corpora are used to train and to evaluate such automatic WSD systems. The availability of large sense-annotated corpora is a necessary prerequisite for any supervised and many semi-supervised approaches to WSD. It is therefore not surprising that most research on WSD has focused on languages such as English for which large sense-annotated corpora are available and considerably less on languages with a shortage of such corpora. This chapter helps to close this gap by creating two sense-annotated corpora for German, a language for which the availability of sense-annotated corpora is restricted.

Thus far, sense-annotated corpora have typically been constructed manually, making the creation of such resources expensive and the compilation of larger data sets difficult, if not completely infeasible. It is therefore timely and appropriate to explore alternatives to manual annotation and to inves-

tigate automatic means of creating sense-annotated corpora. In this chapter (Section 5.2), an automatic method for creating a domain-independent sense-annotated corpus harvested from the web is described. It relies on a correct sense mapping between GermaNet and Wiktionary (described in Chapter 4) to harvest sense-specific example sentences from Wiktionary itself and additional textual materials from other web-based textual sources such as Wikipedia and online newspaper materials. The data obtained by this method have produced the German WebCAGe (short for: *Web-Harvested Corpus Annotated with GermaNet Senses*) resource.

In order to allow a comparison of the performance of WSD algorithms on these automatically harvested web-based materials with a real *authentic* corpus and to overcome the limitations of such an automatically constructed resource (like the assumably varying quality of web texts and the restriction of not being able to compile most frequent sense information due to the skewed number of annotated target word occurrences), the TüBa-D/Z treebank is manually extended by GermaNet sense annotation for a selected set of lemmas (see Section 5.3). This manual creation of such a corpus is along the same lines as many other sense-annotated corpora (see Section 5.1), although the systematic selection of verb lemmas to be manually sense-annotated sets the presented work apart from related work. This selection is determined by the goal of analyzing the influence of syntax and semantics on automatic WSD which is particularly interesting for verbs where the syntactic structure in which a verb occurs is often highly predictive of different word senses.

There are two kinds of sense-annotated corpus (see Chapter 2 for details):

- **All-words disambiguation:** all or nearly all word occurrences in a limited size of running text are annotated with the senses of a given sense inventory.

- **Lexical sampling:** all occurrences in a text of a selected set of ambiguous word lemmas chosen in advance are annotated with the senses of a given sense inventory.

Both sense-annotated corpora described in this chapter (WebCAGe and

the sense-annotated TüBa-D/Z) follow the lexical sample variant[1] and use GermaNet as the sense inventory. For the WebCAGe corpus, the automatic harvesting method meant that only the lexical sample option was possible. For sense annotation in the TüBa-D/Z, the decision to annotate only a lexical sample is motivated by the requirements of machine learning as the intended use of the data. Such data are useful for training automatic machine learning models only if there are enough instances of each item to be classified. Due to limitations of how much text can reasonably be annotated manually in an all-words, sense-annotated corpus, the resulting numbers of instances for each token are not of sufficient frequency for machine-learning applications.

The purpose of this chapter is to describe and evaluate the construction of two sense-annotated corpora for German: the automatically web-harvested corpus WebCAGe (Section 5.2) and the manually sense-annotated TüBa-D/Z treebank (Section 5.3). The chapter continues with a detailed comparison of the two sense-annotated corpora in Section 5.4 and ends with concluding remarks in Section 5.5.

Parts of this chapter have already been published: Henrich et al. [2012b] describe the automatic harvesting method used to create the WebCAGe sense-annotated corpus, and Henrich and Hinrichs [2013, 2014] introduce the manual sense annotation in the TüBa-D/Z treebank.

5.1 Related Work on Sense-Annotated Corpora

There are many sense-annotated corpora available; primarily for English, much fewer for other languages. For English, there are many sense inventories used in sense-annotated corpora, including Princeton WordNet, HECTOR lexicon [Atkins, 1993], and Longman's Dictionary of Contemporary English (LDOCE) [Procter, 1978]. In German, there are much fewer sense-inventories used. This section discusses both manually and automatically constructed sense-annotated corpora. With relatively few exceptions to be discussed shortly (Section 5.1.3), the construction of sense-annotated corpora has focussed on

[1] As do many existing corpora – see Section 5.1.

manual methods (Sections 5.1.1 and 5.1.2).

5.1.1 Manually Sense-Annotated Corpora for English

The manual annotation of a sense-annotated corpus is usually done either (i) token-by-token, i.e., by sequentially going through all word tokens in the corpus, or (ii) lemma-by-lemma, i.e., one word lemma at a time. The first listed procedure is applicable only if most or all word tokens of a running text are annotated in an all-words manner. The problem with this token-by-token procedure is that annotators have to look up all senses of a word for each annotation they do. By contrast, in the lemma-by-lemma annotation procedure, an annotator first takes a look at all senses of a word in the sense inventory, then goes through all occurrences of that word lemma in the text and – having in mind all possible senses of the word – annotates each occurrence with the corresponding sense from the inventory. The advantage of this lemma-by-lemma procedure (compared to a token-by-token procedure) is a higher quality since the annotator who processes one word at a time has all senses of that particular word in mind and apparently can conduct a more consistent annotation. [Kilgarriff, 1997c; Palmer and Xue, 2010]

The annotation procedure for many manually sense-annotated corpora starts with an initial annotation step in which two or more human annotators sense-annotate the same set of word occurrences. In a second adjudication step, another annotator – the adjudicator – goes through all those occurrences where there was a mismatch between the annotations from the initial step or where there was a comment marking unclear or uncertain cases and tries to solve those conflicts. In some annotation projects – especially in those where the sense-annotated corpus is being created by the same research group which maintains the sense inventory – the sense inventory is updated with senses that are missing during the annotation process in order to improve the sense inventory and make it even more feasible for the annotation process. [Palmer et al., 2001; Palmer and Xue, 2010]

The organization of several shared task competitions on WSD at SensEval and SemEval (see Subsection 2.2.4) included the preparation of the necessary

evaluation resources such as sense-annotated corpora. Several sense-annotated corpora were constructed and used in the context of these competitions. The sense-annotated corpus which was used as a gold standard in the first SensEval shared task competition [Kilgarriff, 1998a,b; Kilgarriff and Rosenzweig, 2000] is a part of the Hector corpus [Atkins, 1993]. It comprises texts taken from the British National Corpus[1] (BNC) that were manually annotated with senses from the Hector lexicon. In a 20M-word pilot for the BNC, 200 000 instances of altogether about 300 lemmas were manually sense-annotated in a lexical sample manner (statistics are taken from Kilgarriff [1998b]). During the sense-annotation process, the sense inventory was subsequently updated with missing word senses.

From SensEval-2 onwards, WordNet was mainly used as the sense inventory for the English tasks. In the second SensEval competition, there was a lexical sample task and an all-words task – both using WordNet as the sense inventory. The lexical sample task exclusively annotated verbs [Fellbaum et al., 2001; Palmer et al., 2001]. In the gold standard for this task, between 75 and 300 occurrences in the Penn TreeBank II Wall Street Journal [Marcus et al., 1993] of about 30 highly polysemous verbs were annotated – adding examples from the BNC if there are too few in the Penn Treebank. For the all-words task, 5 000 word occurrences of running text were annotated in the treebank [Palmer et al., 2001]. The WordNet sense inventory was updated in the process of annotation.

For the SensEval-3 all-words task, again 5 000 word occurrences of running text were annotated with WordNet senses [Snyder and Palmer, 2004]. For the SensEval-3 lexical sample task [Mihalcea et al., 2004], 5 adjectives and 32 nouns were annotated with WordNet senses, and 20 verbs were annotated with senses taken from Wordsmyth[2]. The reason for using a different sense inventory for verbs in the latter corpus is the weak performance of WSD systems in the SensEval-2 English lexical sample task, which the organizers of SensEval-3's lexical sample task claimed is due to the high polysemy of verbs in WordNet [Mihalcea et al., 2004]. The lexical sample sense-annotated corpus

[1] `http://www.natcorp.ox.ac.uk/`
[2] `http://www.wordsmyth.net/`

is constructed by volunteers in the context of the Open Mind Word Expert project (OMWE) [Chklovski and Mihalcea, 2002] – see below.

Furthermore, in the SensEval-3 competition, the Princeton WordNet Gloss Corpus was used [Litkowski, 2004]. This corpus, which was created in the context of the 'eXtended WordNet project', contains WordNet sense annotations of the words occurring in WordNet's glosses [Harabagiu et al., 1999; Mihalcea and Moldovan, 2001].

Although this section does not list all tasks in the subsequent SemEval[1] competitions, it is interesting to note that an ongoing discussion about the granularity of senses in the sense inventory had already started by the second SensEval [Palmer, 2000; Edmonds and Cotton, 2001] (also see Section 1.2). One of the most often reported obstacles when creating a sense-annotated corpus is a too fine-grained distinction of senses in the dictionary – especially when using the Princeton WordNet as a sense inventory. To counter this problem, several approaches cluster WordNet senses to use a more coarse-grained sense inventory for sense annotation (see, e.g., Fellbaum et al. [2001], Mihalcea et al. [2004], or Palmer et al. [2007]). As a result, some tasks in subsequent SemEval competitions used semantically grouped WordNet senses to serve as more coarse-grained sense distinctions, such as, for example, the lexical sample task [Pradhan et al., 2007] and the all-words task [Navigli et al., 2007] at SemEval-2007.

Several sense-annotated corpora were created besides SensEval and SemEval. One of the best known of these corpora is SemCor (*Semantic Concordance*) [Miller et al., 1993; Landes et al., 1998], which was constructed by the research team that maintains WordNet. The textual sources for SemCor were taken from the Brown Corpus [Francis and Kučera, 1982] and the novella *The Red Badge of Courage* written by Stephen Crane in 1895 [Crane, 1895]. SemCor comprises 186 files from the Brown Corpus that are sense-annotated with all open class word tokens plus another 166 files in which verbs are sense-annotated. Altogether, 234 113 occurrences of 23 346 word lemmas were annotated with WordNet senses in an all-words manner. Since SemCor was created

[1] After the third SensEval competition it was renamed SemEval (see Subsection 2.2.4).

in the same research group who maintains WordNet, a subsequent update of the sense inventory was performed during the annotation process.

In the *interest* corpus [Bruce and Wiebe, 1994, 1998], 2 369 occurrences of the noun *interest* were manually annotated in the Penn TreeBank Wall Street Journal with senses taken from Longman's Dictionary of Contemporary English (LDOCE) [Procter, 1978].

The Penn TreeBank Wall Street Journal corpus was annotated by several researchers. Wiebe et al. [1997], for example, annotated the 25 most frequent verbs in a lexical sample manner with their WordNet senses. Palmer et al. [2000] annotated verbs and the corresponding headwords in their noun arguments and adjuncts in 5 000 words of running text with their WordNet senses.

In the DSO corpus [Ng and Lee, 1996], a total of 192 800 word instances taken from the Brown Corpus and the Wall Street Journal were annotated with WordNet senses of 191 frequent words (121 nouns and 70 verbs).

In the *line*, *hard* and *serve* corpora [Leacock et al., 1993, 1998], about 4 000 examples of each of the three words *line* (noun), *hard* (adjective), and *serve* (verb) were manually sense-tagged in texts taken from the Wall Street Journal, the American Printing House for the Blind, and the San José Mercury News.

The work of Mihalcea [2007] followed a two-step approach for creating a sense-annotated corpus. It started with automatically harvesting Wikipedia text passages containing polysemous words that are part of a hyperlink. The result of this step was a set of hyperlink labels for each polysemous word. In a second step, all these labels were manually mapped to their corresponding WordNet sense. This manual mapping is equivalent with annotating multiple word occurrences in Wikipedia with word senses from WordNet. In their experiments, Mihalcea [2007] reported on the annotation of 30 word lemmas with an average of 316 occurrences each.

MASC (*Manually Annotated Sub-Corpus*) [Ide et al., 2010; Passonneau et al., 2012] is a corpus with multiple layers of linguistic annotation – including WordNet sense annotation. It consists of 500 000 tokens from different genres of contemporary American English text from the Open American National Corpus (OANC). For each lemma from a selected set of 114 polysemous words, about 1 000 occurrences were annotated with WordNet senses. The

sense inventory was updated during the sense-annotation process.

Since the quality measured in terms of inter-annotator agreement is reported higher when the manual annotation was performed by experts, i.e., linguists, lexicographers, or trained students in the field, most of the sense annotation was performed by experts. However, there is always a trade-off between quantity and quality. On the one hand, supervised algorithms require a lot of training material, i.e., a lot of sense-annotated occurrences. On the other hand, employing experts who do the manual sense annotation is very costly.

In the Open Mind Word Expert project (OMWE) [Chklovski and Mihalcea, 2002], a different approach was investigated in order to increase the number of sense-annotated examples at low cost. OMWE gathered sense annotations from web users while playing a game in which they are supposed to disambiguate words in context. The underlying corpus data includes texts from the Penn TreeBank and Los Angeles Times, and the sense inventory is taken from WordNet. With this method, 70 000 instances of 230 lemmas were sense-annotated with WordNet senses [Palmer et al., 2006, page 85]. A portion of the OMWE data was used in the SensEval-3 competition.

Note that although this list of sense-annotated corpora is long, it is certainly not complete.

5.1.2 Manually Sense-Annotated Corpora for German

There are only a few sense-annotated corpora for GermaNet. Some sense-annotated corpora for German exist, but have either not been distributed or are not annotated with senses from GermaNet.

Saito et al. [2002] manually developed a German corpus from novels for children and young people and from newspaper articles, which altogether comprised 5 625 word tokens. Following the token-by-token all-words annotation variant, all content words with entries in GermaNet (2 199 word tokens) were annotated with GermaNet senses. Although five annotators were involved, all

word tokens in all parts of the corpus were annotated only once.[1]

In the context of the MuchMore project[2], an English-German parallel medical corpus was obtained from scientific abstracts from the Springer Link website. In this corpus, 2 421 occurrences of 25 nouns relevant to the medical domain were manually annotated with GermaNet senses [Raileanu et al., 2002]. The same medical corpus was manually annotated for 24 ambiguous German UMLS[3] (*Unified Medical Language System*) terms[4] – each of which occurs at least 11 times in the corpus [Widdows et al., 2003]. UMLS consists of several components. For the sense annotation, the authors use concepts from the MeSH (*Medical Subject Headings*) thesaurus, which is part of UMLS. Both for GermaNet and UMLS, two annotators conducted the sense-tagging. The medical corpus including all sense annotations is freely available for download.[5]

In the study by Broscheit et al. [2010], a lexical sample of 40 ambiguous word lemmas (of which 6 are adjectives, 18 are nouns, and 16 are verbs) was selected by translating words taken from the English SensEval-2 test set data. At least 20 occurrences of each of these lemmas in the *deWaC* corpus (the German part of the WaCky corpora [Baroni et al., 2009]) were manually annotated with GermaNet senses. Altogether there were 1 154 annotated occurrences. Recently, the sense annotations in this corpus have been made publicly available.[6]

What can be noted is that three GermaNet sense-annotated corpora exist [Raileanu et al., 2002; Saito et al., 2002; Broscheit et al., 2010] of which two are freely available [Raileanu et al., 2002; Broscheit et al., 2010]. All three corpora rely on old GermaNet versions[7] which stem from the time before persistent database identifiers were introduced to GermaNet (see Appendix B). This means that a unique and reliable identification of senses throughout dif-

[1] The only exception are 170 word tokens that were annotated by all five annotators in order to analyze inter-annotator agreement.

[2] `http://muchmore.dfki.de/`

[3] `http://www.nlm.nih.gov/research/umls/`

[4] Albeit this paragraph reports on sense-annotated corpora for German, it should not be concealed that the corpus was also annotated with 70 ambiguous English UMLS terms.

[5] `http://muchmore.dfki.de/resources3.htm`

[6] `http://projects.cl.uni-heidelberg.de/dewsd/`

[7] The most recent of the studies, i.e., Broscheit et al. [2010], used GermaNet 5.1 which was released in April 2008.

ferent GermaNet versions is not possible and thus, unfortunately, these corpora are not compatible with recent versions of GermaNet. However, since the sense annotations in the deWac corpus have recently been made publicly available, they have been updated in the context of this thesis (see Section 6.5) to the most recent version of GermaNet in order to be used in the WSD experiments in Chapters 7 and 8.

The most recent papers on the creation of a manually GermaNet sense-annotated corpus were published by Henrich and Hinrichs [2013, 2014]. They describe the manual sense annotation of almost 18 000 occurrences of a lexical sample of 109 word lemmas (30 nouns and 79 verbs) in the TüBa-D/Z treebank. This sense-annotated corpus is the only GermaNet sense-annotated corpus where the sense inventory was updated during the annotation process. The corpus is freely available – as part of the treebank. The sense-annotated TüBa-D/Z has been constructed as part of this dissertation (see Section 5.3) and is used throughout the WSD experiments in Chapters 7 and 8.

Further details on several GermaNet sense-annotated corpora are given in Section 5.4 below, which presents a detailed comparison of the sense-annotated corpora by Raileanu et al. [2002] and Broscheit et al. [2010] with the two sense-annotated corpora constructed in this chapter.

5.1.3 Automatically Sense-Annotated Corpora

Compared to the large amount of manually sense-annotated corpora, there are very few previous attempts to automatically harvest corpus data for the purpose of constructing a sense-annotated corpus.

One of the first attempts at automatically creating sense-annotated data is the semi-supervised method developed by Yarowsky [1995]. Yarowsky first collected all example sentences that contain a polysemous word from a very large corpus of about 460 million words spanning news articles, scientific abstracts, spoken transcripts, and novels. In a second step, a small number of examples that are representative for each of the senses of a polysemous target word were selected from this corpus. These representative examples were manually sense-annotated and then fed into a decision-list supervised WSD

algorithm as a seed set for iteratively disambiguating the remaining examples collected in step 1. The selection and annotation of the representative examples in Yarowsky's approach was performed completely manually and therefore limited to the amount of data that can reasonably be annotated by hand. The main difference between Yarowsky's approach and the approach described in this chapter is that the first one relies on a manually sense-annotated seed sample to harvest further examples whereas the latter one relies on a mapping between the sense inventory and a second resource to automatically harvest sense-annotated data.

Leacock et al. [1998], Mihalcea and Moldovan [1999], and Agirre and Lopez de Lacalle [2004] proposed a set of methods for automatic harvesting of large corpora and web data for the purposes of creating sense-annotated corpora. In the first study of this kind, Leacock et al. [1998] developed a method based on WordNet and 'monosemous relatives' with the goal of creating unsupervised training examples for a statistical WSD classifier. The approach worked as follows: in order to harvest corpus examples for a polysemous word, WordNet relations such as synonymy, hypernymy, hyponymy, and co-hyponymy were inspected for the presence of unambiguous words, i.e., words that appear only in exactly one synset. Their system first retrieved monosemous synonyms and monosemous daughter collocations containing the polysemous word as its head, if available, before it fell back on other types of monosemous relatives such as hypernyms or co-hyponyms. The examples found for these monosemous relatives in the San José Mercury News, a corpus of about 30 million words, were then sense-annotated with the particular sense of its ambiguous word relative.

Agirre and Lopez de Lacalle [2004] applied the monosemous relatives method to acquire sense-annotated examples from the World Wide Web. They used monosemous hypernyms, direct and indirect hyponyms, and co-hyponyms to query the search engine Google. Sense-specific example sentences were extracted from the search snippets returned by Google.

In order to increase coverage of the monosemous relatives approach, Mihalcea and Moldovan [1999] developed a gloss-based extension, which relied on information from the WordNet definition of the sense in question for all those

cases where a monosemous relative was not contained in the WordNet dataset or did not result in enough acquired examples. Mihalcea and Moldovan [1999] retrieved example sentences also by querying the World Wide Web. Their procedure starts by querying the search engine AltaVista on monosemous synonyms. They did not consider other monosemous relatives such as hypernyms or co-hyponyms since these produced less representative examples. If there were no monosemous synonyms available or if there were too few resulting examples, they formulated queries representing simplified versions of synset definitions. Finally, their approach produced sense-specific examples by replacing the search term, i.e., the monosemous synonym or the shortened definition, by the polysemous word.

By focusing on web-based data, the work by Mihalcea and Moldovan [1999] and Agirre and Lopez de Lacalle [2004] resembles the research described in the present chapter. However, the underlying harvesting methods differ. While the approach in this chapter relies on a wordnet to Wiktionary mapping, their approaches (also including the initial study by Leacock et al. [1998]) all rely on the monosemous relatives heuristic.

Compared to the approaches described so far, the study by Santamaría et al. [2003] is closest in spirit to the approach presented in this chapter. It also relies on an automatic mapping between wordnet senses and a second web resource. While the approach presented in this chapter is based on automatic mappings between GermaNet and Wiktionary, their mapping algorithm maps WordNet senses to web directories from the Open Directory Project (ODP). Since these ODP directories contain natural language descriptions of websites relevant to the directory in question, this textual material can be used for harvesting sense-specific examples. The ODP project also covers German so that, in principle, their harvesting method could be applied to German in order to collect German sense-tagged data.

Previous work on the automatic construction on sense-annotated corpora for German include the paper published by Henrich et al. [2012b]. It describes an automatic method for creating a domain-independent sense-annotated corpus harvested from the web. This automatic method relied on the mapping between GermaNet and Wiktionary (described in Chapter 4) to harvest sense-

specific example sentences from Wiktionary itself and additional textual materials from other web-based textual sources such as Wikipedia and online newspaper materials. The data obtained by this method have resulted in the German WebCAGe resource, which is freely available online. WebCAGe has been created as part of this dissertation (see Section 5.2 for a detailed description) and is used in the WSD experiments in Chapters 7 and 8.

The studies by Henrich et al. [2012a] and Henrich et al. [2012c] applied WebCAGe's automatic harvesting method to other resources. In the study described in Henrich et al. [2012a], the authors first created a mapping between GermaNet and Wikipedia in order to harvest GermaNet sense-annotated materials from Wikipedia articles. On the basis of this Wikipedia–GermaNet mapping, GermaNet sense-specific word occurrences were extracted from Wikipedia articles. Henrich et al. [2012c] applied the automatic harvesting method to English. They took the existing mapping between WordNet and Wiktionary provided by Meyer and Gurevych [2011] as a basis to harvest sense-specific example sentences from the English Wiktionary. Since the latter study relied on a WordNet–Wiktionary mapping, the underlying sense inventory for the sense-annotated data is the Princeton WordNet.

5.2 Automatically Constructed WebCAGe

This section reports on the (semi-)automatic creation of the WebCAGe sense-annotated corpus for German. WebCAGe stands for *Web-Harvested Corpus Annotated with GermaNet Senses*. Since WebCAGe's text harvesting and sense annotation is done automatically, with a manual post-correction in order to ensure high quality, the approach is referred to as *semi-automatic*.

To date, sense-annotated corpora have typically been constructed manually, making the creation of such resources expensive and the compilation of larger data sets difficult, if not completely infeasible. This chapter explores an alternative to manual annotation and investigate automatic means of creating sense-annotated corpora.

An automatic method for creating a sense-annotated corpus should ideally

maximize both quality and quantity of the automatically generated data. On the one hand, the quality of an automatically generated sense-annotated corpus should be high enough to be usable as is or with a minimal amount of manual post-correction. On the other hand, the resulting sense-annotated materials (i) should be non-trivial in size, (ii) should be as domain-independent as possible, and (iii) should be freely available for other researchers.

The method presented in this section satisfies all of the above criteria and relies on the following resources as input: (i) a sense inventory and (ii) a mapping between the sense inventory in question and a web-based resource such as Wiktionary or Wikipedia. For WebCAGe, the sense inventory is taken from GermaNet, and the web-harvesting relies on the mapping of GermaNet and Wiktionary. While the present section focuses on one particular language, the method as such is language-independent.

5.2.1 Creation of a Web-Harvested Corpus

The starting point for creating WebCAGe is the mapping of GermaNet senses to Wiktionary sense definitions as described in Chapter 4. The original purpose of this mapping was to automatically add Wiktionary sense descriptions to GermaNet. However, the alignment of these two resources opens up a much wider range of possibilities for data mining community-driven resources such as Wikipedia and web-generated content more generally. It is precisely this potential that is fully exploited for the creation of the WebCAGe sense-annotated corpus.

Figure 5.1 illustrates the existing GermaNet–Wiktionary mapping using the example word *Bogen*. The polysemous word *Bogen* has three distinct senses in GermaNet which directly correspond to three separate senses in Wiktionary.[1] Each Wiktionary sense entry contains a definition and one or more example sentences illustrating the sense in question. Since the target word (rendered in Figure 5.1 in boldface) in the example sentences for a particular Wiktionary sense is linked to a GermaNet sense via the sense mapping of GermaNet with

[1]Note that there are further senses in both resources not displayed here for reasons of space.

Wiktionary, the example sentences are automatically sense-annotated and can be included as part of WebCAGe.

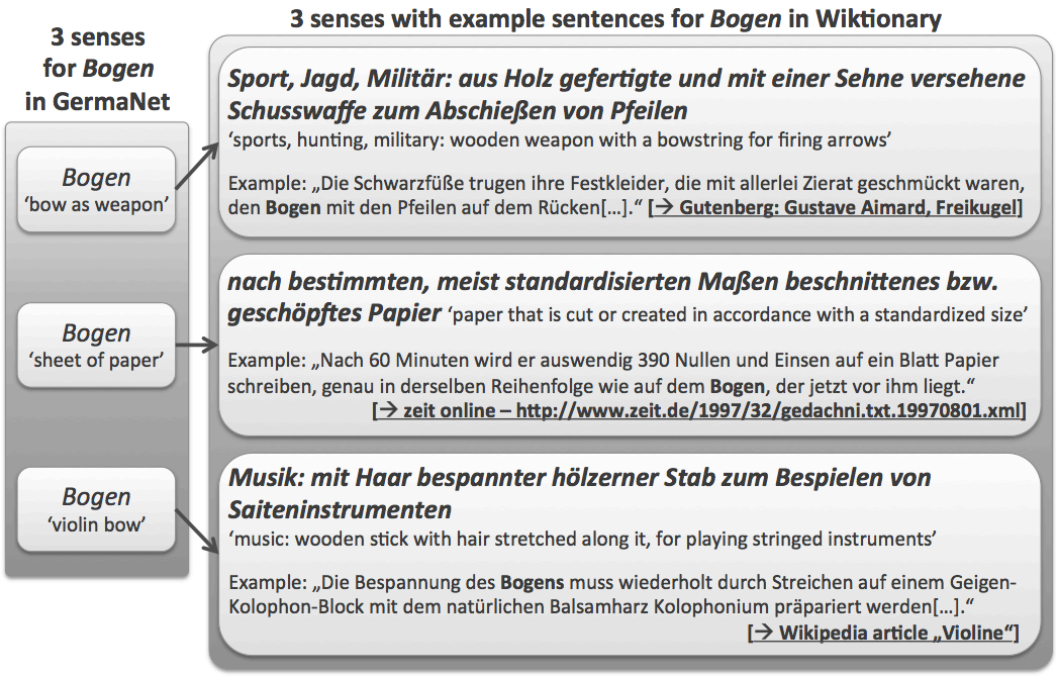

Figure 5.1: Sense mapping of GermaNet and Wiktionary using the example of *Bogen*.

Wiktionary example sentences are often linked to external references, including sentences contained in the German Gutenberg text archive[1] (see link in the topmost Wiktionary sense entry in Figure 5.1), Wikipedia articles (see link for the third Wiktionary sense entry in Figure 5.1), and other textual sources (see the second sense entry in Figure 5.1). It is precisely this collection of heterogeneous material that can be harvested as additional material for WebCAGe by following the links to Wikipedia, the Gutenberg archive, and other web-based materials. The external webpages and the Gutenberg texts are obtained from the web by a web-crawler that takes URLs as input and outputs the texts of the corresponding web sites.[2] The Wikipedia articles are obtained by the open-source Java Wikipedia Library JWPL[3] [Zesch

[1] `http://gutenberg.spiegel.de/`
[2] Thanks to Yannick Versley and Yana Panchenko for making available their web-crawler.
[3] `http://www.ukp.tu-darmstadt.de/software/jwpl/`

et al., 2008]. Since the links to Wikipedia, the Gutenberg archive, and other web-based materials also belong to particular Wiktionary sense entries that in turn are mapped to GermaNet senses, the target words contained in these materials are automatically sense-annotated.

Notice that the target word often occurs more than once in a given text. In keeping with the widely used heuristic of 'one sense per discourse' [Gale et al., 1992b], multiple occurrences of a target word in a given text are all assigned to the same GermaNet sense. An inspection of the annotated data shows that this heuristic has proven to be highly reliable in practice. It is correct in 99.96% of all target word occurrences in the Wiktionary example sentences, in 96.75% of all occurrences in the external webpages, and in 95.62% of the Wikipedia files.

WebCAGe is developed primarily for the purpose of the word sense disambiguation task. Therefore, only those target words that are genuinely ambiguous are included in this resource. Since WebCAGe uses GermaNet as its sense inventory, this means that each target word has at least two GermaNet senses, i.e., belongs to at least two distinct synsets.

The GermaNet–Wiktionary mapping is not always one-to-one. Sometimes one GermaNet sense is mapped to more than one sense in Wiktionary (as described in Section 4.4). Figure 5.2 illustrates such a case. For the word *Archiv* each resource records three distinct senses. The first sense ('data repository') in GermaNet corresponds to the first sense in Wiktionary, and the second sense in GermaNet ('archive') corresponds to both the second and third senses in Wiktionary. The third sense in GermaNet ('archived file') does not map onto any sense in Wiktionary at all. As a result, the word *Archiv* is included in the WebCAGe resource with precisely the sense mappings connected by the arrows shown in Figure 5.2. The fact that the second GermaNet sense corresponds to two sense descriptions in Wiktionary simply means that the target words in the examples are both annotated by the same sense. Furthermore, note that the word *Archiv* is still genuinely ambiguous since there is a second (one-to-one) mapping between the first senses recorded in GermaNet and Wiktionary. However, since the third GermaNet sense is not mapped onto any Wiktionary sense at all, WebCAGe does not contain any example sentences for this particular GermaNet sense.

Figure 5.2: Sense mapping of GermaNet and Wiktionary using the example of *Archiv*.

The following section describes how the target words within these textual materials can be automatically identified.

5.2.2 Automatic Detection of Target Words

For highly inflected languages such as German, target word identification is more complex compared to languages with an impoverished inflectional morphology, such as English, and thus requires automatic lemmatization. Moreover, the target word in a text to be sense-annotated is not always a simplex word but can also appear as subpart of a complex word such as a compound. Since the constituent parts of a compound are not usually separated by blank spaces or hyphens, German compounding poses a particular challenge for target word identification. Another challenging case for automatic target word detection in German concerns particle verbs such as *ankündigen* 'announce'. Here, the difficulty arises when the verbal stem (e.g., *kündigen*) is separated from its particle (e.g., *an*) in German verb-initial and verb-second clause types.

As a preprocessing step for target word identification, the text is split into individual sentences and tokenized with the help of the sentence detector and

the tokenizer of the Apache OpenNLP tool suite[1]. Lemmatization is performed by TreeTagger [Schmid, 1994]. Further, compounds are split by using BananaSplit[2]. Since the automatic lemmatization and compound splitting are not 100% accurate, target word identification also utilizes the full set of inflected forms for a target word whenever such information is available. As it turns out, Wiktionary can often be used for this purpose as well since the German version of Wiktionary often contains the full set of word forms in tables[3] such as the one shown in Figure 5.3 for the word *Bogen*.

Kasus	Singular	Plural 1	Plural 2
Nominativ	**der Bogen**	**die Bogen**	**die Bögen**
Genitiv	des Bogens	der Bogen	der Bögen
Dativ	dem Bogen	den Bogen	den Bögen
Akkusativ	den Bogen	die Bogen	die Bögen

Figure 5.3: Wiktionary inflection table for *Bogen*.

Figure 5.4 shows an example of a sense-annotated article for the target word *Bogen* 'violin bow'. The article is an excerpt from the Wikipedia article *Violine* 'violin', where the target word (surrounded by gray boxes) appears many times. Only the first three and the 12th occurrences shown in the figure (marked with `1`, `2`, `3`, and `12` on the left) exactly match the word *Bogen* as is. All other occurrences are either the plural form *Bögen* (`5` and `8`), the genitive form *Bogens* (`9` and `13`), part of a compound such as *Bogenstange* (`4`, `10` and `11`), or the plural form as part of a compound such as in *Fernambukbögen* and *Schülerbögen* (`6` and `7`).

A textual representation of this sense-annotated Wikipedia article is shown in Figure 5.5. The figure shows an excerpt of the XML data format in which WebCAGe is made available. Each target word occurrence is annotated with an XML `<head>` element. The information for each occurrence of a target word consists of the GermaNet sense(s), i.e., the lexical unit identifier(s) (XML

[1] `http://opennlp.apache.org/`

[2] `http://niels.drni.de/s9y/pages/bananasplit.html`

[3] The inflection table cannot be extracted with the Java Wikipedia Library JWPL. It is rather extracted from the Wiktionary dump file.

Figure 5.4: Wikipedia article *Violine* 'violin' with highlighted occurrences of target word *Bogen* 'violin bow'.

attribute `luids`), the lemma of the target word (XML attribute `lemma`), and the GermaNet word class information (attribute `POS`), i.e., `a` for adjectives, `n` for nouns, and `v` for verbs.

5.2.3 Evaluation

In order to assess the effectiveness of this approach, this section examines the overall size of WebCAGe and the relative size of the different text collections (see Table 5.1), and it presents a precision- and recall-based evaluation of the algorithm that is used for automatically identifying target words in the harvested texts (see Table 5.2). Below, Section 5.4 compares WebCAGe to other sense-annotated corpora for German.

Table 5.1 shows that Wiktionary (7644 tagged word tokens, column *Wiktionary examples*) and Wikipedia (1 732, column *Wikipedia articles*) contribute by far the largest subsets of the total number of tagged word tokens (10 750) compared with the tokens from the external webpages (589, column *external pages*) and the Gutenberg texts (785 tagged word tokens, column *GB texts*).

```
<corpus lang="de">
<text id="nomen_Bogen_13522" src="https://de.wikipedia.org/wiki/Violine"
title="Violine">

[...] Der <head id="3" luids="19087" lemma="Bogen" pos="n">Bogen</tag>
besteht häufig aus dem Rotholz Pernambuk (Fernambuk). Gutes Pernambuk
ist gerade gewachsen und die Fasern verlaufen parallel, die <head id="4"
luids="19087" lemma="Bogen" pos="n">Bogen</tag>stange kann besonders
dünn gearbeitet werden und weist eine ideale Elastizität auf. Das Holz
eignet sich somit besonders für qualitativ hochwertige <head id="5"
luids="19087" lemma="Bogen" pos="n">Bögen</tag>. Da das Vorkommen der
Holzart begrenzt ist, haben Fernambuk<head id="6" luids="19087"
lemma="Bogen" pos="n">bögen</tag> einen entsprechen hohen Preis.
Einfachere Schüler<head id="7" luids="19087" lemma="Bogen"
pos="n">bögen</tag> sind meist aus Brasilholz gefertigt. Heute werden,
auch von Berufsgeigern, zunehmend <head id="8" luids="19087"
lemma="Bogen" pos="n">Bögen</tag> aus Kohlefaser (Karbonfiber)
verwendet.

Am unteren Ende des <head id="9" luids="19087" lemma="Bogen"
pos="n">Bogens</tag> befindet sich der sogenannte Frosch aus Ebenholz,
meist verziert mit einer runden Perlmutt-Einlage. [...]

</text>
</corpus>
```

Figure 5.5: Excerpt from Wikipedia article *Violine* 'violin' tagged with target word *Bogen* 'violin bow'.

Table 5.1: Current size of WebCAGe.

	POS	Wiktionary examples	External pages	Wikipedia articles	GB texts	All texts
Number of tagged word tokens	Adj.	575	31	79	28	713
	Nouns	4103	446	1643	655	6847
	Verbs	2966	112	10	102	3190
	All POS	7644	589	1732	785	10750
Number of tagged sentences	Adj.	565	31	76	26	698
	Nouns	3965	420	1404	624	6413
	Verbs	2945	112	10	102	3169
	All POS	7475	563	1490	752	10280
Total number of sentences	Adj.	623	1297	430	65030	67380
	Nouns	4184	9630	6851	376159	396824
	Verbs	3087	5285	263	146755	155390
	All POS	7894	16212	7544	587944	619594

These tokens belong to 2 607 distinct polysemous words contained in GermaNet, among which there are 211 adjectives, 1 499 nouns, and 897 verbs. On average, these words have 2.9 senses in GermaNet (2.4 for adjectives, 2.6 for nouns, and 3.6 for verbs).

For the purpose of the present evaluation, a precision and recall analysis is

conducted for all text types separately for the three word classes of adjectives, nouns, and verbs. Table 5.2 shows that precision and recall for all three word classes that occur for Wiktionary examples, external webpages, and Wikipedia articles lies above 92%. The only sizeable deviations are the results for verbs that occur in the Gutenberg texts. Apart from this one exception, the results in Table 5.2 prove the viability of the proposed method for automatic harvesting of sense-annotated data. The average precision for all three word classes is of sufficient quality to be used as-is if approximately 2-5% noise in the annotated data is acceptable. In order to eliminate such noise, manual post-editing is required. However, such post-editing is within acceptable limits: it took an experienced research assistant a total of 25 hours to hand-correct all the occurrences of sense-annotated target words and to manually sense-tag any missing target words for the four text types.

Table 5.2: Evaluation of the algorithm of identifying the target words.

	POS	**Wiktionary examples**	**External webpages**	**Wikipedia articles**	**Gutenberg texts**
Precision	Adj.	97.70%	95.83%	99.34%	100%
	Nouns	98.17%	98.50%	95.87%	92.19%
	Verbs	97.38%	92.26%	100%	69.87%
	All POS	97.32%	96.19%	96.26%	87.43%
Recall	Adj.	97.70%	97.22%	98.08%	97.14%
	Nouns	98.30%	96.03%	92.70.%	97.38%
	Verbs	97.51%	99.60%	100%	89.20%
	All POS	97.94%	97.32%	93.36%	95.42%

5.2.4 Future Directions

The approaches of Yarowsky [1995], Leacock et al. [1998], Mihalcea and Moldovan [1999] and Agirre and Lopez de Lacalle [2004] (see descriptions in Section 5.1.3) provide interesting directions for further enhancing the WebCAGe resource. It would be worthwhile to use the automatically harvested sense-annotated examples as the seed set for Yarowsky's [1995] iterative method for creating a large sense-annotated corpus. Another fruitful direction for further automatic expansion of WebCAGe is to use the heuristic

of monosemous relatives used by Leacock et al. [1998], by Mihalcea and Moldovan [1999], and by Agirre and Lopez de Lacalle [2004]. However, these matters have to be left for future research.

In order to validate the resource and language independence of the automatic harvesting approach that was developed to create WebCAGe, the method was also applied to other resources. A precondition for such an experiment is an existing mapping between the sense inventory in question and a web-based resource such as Wiktionary or Wikipedia. In the study described in Henrich et al. [2012a], the authors first create a mapping between GermaNet and Wikipedia in order to harvest German sense-annotated materials from Wikipedia articles. In the study described in [Henrich et al., 2012c], the harvesting method was applied to English, taking the existing mapping between the Princeton WordNet and the English version of Wiktionary provided by Meyer and Gurevych [2011] as a basis. The results of these experiments [Henrich et al., 2012a,c] confirm the general applicability of WebCAGe's harvesting approach for automatically creating sense-annotated data to other resources and languages.

5.3 Manually Sense-Annotated TüBa-D/Z

This section describes the manual sense annotation of a selected set of lemmas in the TüBa-D/Z treebank [Telljohann et al., 2004, 2012]. The sense inventory used for tagging word senses is taken from GermaNet. The underlying textual resource, the TüBa-D/Z treebank, is a German newspaper corpus already semi-automatically enriched with high-quality annotations at various levels of language including parts of speech, morphology, syntactic constituency, etc. Subsection 5.3.1 describes all annotation layers in detail. The use of treebank data is motivated by the following considerations:

1. The grammatical information contained in a treebank makes it possible to utilize a much richer feature set for automatic WSD compared to sense-annotated training data that otherwise contain little or no linguistic annotation [Fellbaum et al., 2001; Chen and Palmer, 2009]. This

is particularly useful for automatic WSD of verbs where the syntactic structure in which a verb occurs is often highly predictive of different word senses.

2. Since the TüBa-D/Z is based on a newspaper corpus, this ensures a broad coverage of topical materials such as politics, economy, society, environmental issues, sports, arts and entertainment. This broad coverage of topics also makes it possible to obtain reliable information about the relative frequency of different senses of a given word.

Manual sense annotation of the TüBa-D/Z is along the lines of most other sense-annotated corpus projects (see Section 5.1). In several related annotation projects, there was a special interest in the sense annotation of verbs. For example, there was a lexical sample task at SensEval-2 which was dedicated to verbs [Fellbaum et al., 2001; Palmer et al., 2001] and in the all-words sense-annotated corpus SemCor, there is an additional set of 166 Brown Corpus files in which only verbs were sense-annotated [Miller et al., 1993; Landes et al., 1998]. This motivates putting a special emphasis on the systematic selection and annotation of verbs in the TüBa-D/Z. What sets this work apart from related work is the systematic selection of verb lemmas to be manually sense-annotated (see Section 5.3.2 below). This selection is determined by the goal of analyzing the influence of syntax and semantics on automatic WSD, which is particularly interesting for verbs where the syntactic structure in which a verb occurs is often highly predictive of different word senses.

5.3.1 Linguistic Annotations in the Treebank

The *Tübingen Treebank of Written German* (TüBa-D/Z) is a German newspaper corpus with high-quality annotations at various levels of language including parts of speech, morphology, and syntactic constituency. Textual materials from the daily newspaper 'die tageszeitung' (taz)[1] are semi-automatically enriched with linguistic annotations. That is, automatically pre-annotated data

[1] `http://www.taz.de/`

are manually corrected with the help of the graphical tool Annotate [Plaehn, 1998].

The TüBa-D/Z is the largest manually or semi-automatically annotated treebank for German. The work on the treebank is still in progress, which mainly includes adding textual materials. The TüBa-D/Z is freely available for academic use.[1]

Each newspaper article is split into paragraphs and sentences, and each sentence in turn into tokens. The most recent release of the TüBa-D/Z that is available at the time when the manual sense annotation started is release 8.0. It contains 1 365 642 tokens occurring in 75 408 sentences that are taken from 3 356 newspaper articles.[2]

Example (2) shows a sample sentence from the treebank with its English translation.

(2) *Wegmanns haben sich einen Hund zugelegt.*[3]
('The Wegmanns have adopted a dog.')

Figure 5.6[4] shows the same sentence with several layers of linguistic annotation. The upper part of the figure shows the syntactically annotated tree structure, below are word level annotations.

The treebank includes various linguistic annotation layers, several of which can be found in Figure 5.6. The following list details all annotations that matter for the WSD experiments in this thesis – especially as input to machine learning features in the context of supervised WSD experiments (see Chapter 8):

Sentence segmentation All newspaper articles in the TüBa-D/Z are separated into individual sentences. The example in Figure 5.6 represents

[1] See `http://www.sfs.uni-tuebingen.de/en/ascl/resources/corpora/tueba-dz.html` for the details on licensing the treebank resource and a Java API to access the data programmatically.

[2] Note that release 8.0 of the TüBa-D/Z is the most recent release available at the time when the manual sense annotation started. However, the treebank version used for the WSD experiments in Chapters 7 and 8 is release 9.1 – as described in Section 6.4.

[3] Sentence 60 611 from TüBa-D/Z 9.1.

[4] In this thesis, all syntactic tree visualisations such as the one shown in Figure 5.6 are created with the export function of the treebank search tool TIGERSearch [Lezius, 2002].

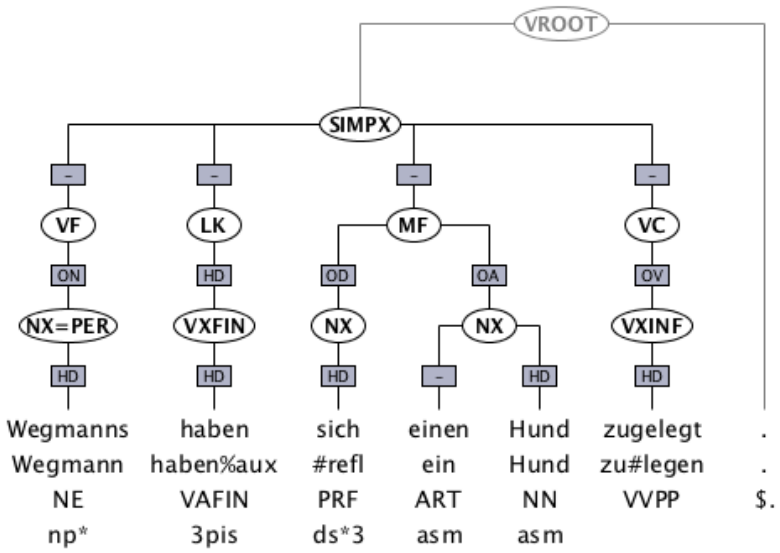

Figure 5.6: An annotated example sentence from the TüBa-D/Z.

one sentence, labeled with *SIMPX* on the highest level of the syntax tree.[1]

Tokenization Sentences, in turn, are split into individual tokens, shown in the example figure in the first line of text below the syntax tree.

Lemmatization Each token is lemmatized, as shown in the second line of text in Figure 5.6. This annotation layer encodes additional information on various linguistic phenomena – three of which are shown in the example sentence: auxiliary verbs are represented by *%aux* at the end of the lemmas (see lemma *haben%aux* for the second token in the example sentence). Reflexive personal pronouns are annotated with *#refl* (see the annotation of the third token *sich*). Separable verb particles are marked with a hash symbol (*#*, see the particle verb *zu#legen* in the example sentence).

Parts of speech Tokens are annotated by their parts of speech (POS) using the Stuttgart-Tübingen-TagSet (STTS) [Schiller et al., 1999]. The STTS

[1] Actually, the node marked with *VROOT* is even above the *SIMPX* node. It integrates all elements of the sentence, including punctuation.

consists of 54 hierarchically structured tags. That is, the first characters of each tag encode one of the 11 main word classes of nouns (*N*), verbs (*V*), articles (*ART*), adjectives (*ADJ*), pronouns (*P*), etc. Depending on the main word class, these tags are subclassified to encode word class specific information. For the main word class of nouns, for example, the tagset distinguishes appellative nouns (tag *NN*) from proper nouns (tag *NE*). For adjectives, it distinguishes between attributive adjectives (tag *ADJA*) and adverbial and predicative adjectives (tag *ADJD*), to give but two examples.

The STTS tags for punctuation marks start with a dollar symbol (*$*). That is, '*$.*' for sentence-final punctuation, '*$,*' for commas, and '*$(*' for other sentence-internal punctuation.

The third text line in Figure 5.6 shows the POS tags for each token in the example sentence, where *NE* stands for proper noun, *VAFIN* for auxiliary, finite verb, *PRF* for reflexive personal pronoun, *ART* for definite or indefinite article, *NN* for appellative noun, *VVPP* for past participle, main verb, and '*$.*' for sentence-final punctuation.

Inflectional morphology The treebank annotates inflectional morphology such as case, number, gender, person, mood, and tense. The kind of morphological information available differs for each word class. The TüBa-D/Z stylebook specifies which morphological information is annotated for which parts of speech. For example, it encodes case, number, and gender for nouns and (attributive) adjectives, whereas it encodes person, number, mood, and tense for finite verbs.

The last line in the example figure represents the annotation of inflectional morphology. The labels are abbreviations, where a star (*) marks underspecified information: *np** stands for nominative/plural/ underspecified number, *3pis* for *third person/plural/indicative/present tense*, *ds*3* for *dative/singular/underspecified gender/third person*[1], and

[1]There is an error in TüBa-D/Z's morphological annotation of sentence 60 611: token *sich* needs to be annotated with *dp*3* instead of *ds*3*, i.e., *plural* instead of *singular*. This will be corrected in the upcoming release 10.0 of the treebank. However, since this error

asm for *accusative/singular/masculine*.

Phrases Phrase labels – such as *NX* which stands for noun phrase, *ADJX* for adjectival phrase, *VXFIN* for finite verb phrase, and *VXINF* for non-finite verb phrase – form the fundamental structural elements of TüBa-D/Z's syntax trees. In Figure 5.6, phrase labels are surrounded by ovals. Note that not all labels surrounded by ovals are phrase labels – see the annotation of topological fields below.

Topological fields On the highest syntactical level, sentences in the TüBa-D/Z treebank are structured according to topological sequences [Herling, 1821; Drach, 1937; Höhle, 1986] which are widely used in German syntax to characterize word order regularities among different clause types. In German sentences, verbal elements, i.e., the *Linke Satzklammer* 'left sentence bracket' (*LK*) and the *Verbkomplex* 'verb complex' (*VC*), divide the sentence into *Vorfeld* 'initial field' (*VF*), *Mittelfeld* 'middle field' (*MF*), and *Nachfeld* 'final field' (*MF*). In the example figure, topological fields are also surrounded by ovals. Depending on the position of the finite verb, the topological structure distinguishes three sentence types: *verb-initial*, *verb-second*, and *verb-final*. The example sentence represents a verb-second type.

Grammatical functions and syntactic constituency Tokens, phrases, topological fields, and sentences can have edge labels (the labels in boxes with a grey background in the example figure) which specify grammatical functions. On the phrase level, *HD* denotes the head of the phrase which can be realized by a single token or a phrase.

The annotation of phrases with labels – such as in Figure 5.6 the *ON* which stands for nominative object, the *OD* which stands for dative object, the *OA* which stands for accusative object, or the *OV* which stands for verbal object – annotates the syntactic constituency structure

does not affect the description of TüBa-D/Z's annotation layers in this thesis, the sentence is well suited for illustrating all annotations, because it is simple yet does it contain various annotation phenomena.

of a sentence. These complements represent arguments of a verb such as accusative, dative, genitive, prepositional, or adverbial objects. The constituent structure is annotated below the level of topological fields. That is, phrases whose edge labels specify their grammatical function are attached to a topological field. Due to a relatively flexible word order in German, the structural variety of constituents within topological fields is large.

Named entities Named entities are classified into the five subclasses of organisation (*ORG*), person (*PER*), location (*LOC*), geo-political entity (*GPE*), and other (*OTH*). To denote a phrase as a named entity, the node label is assigned one of these named entity classes. In the example sentence, the nominal phrase *Wegmanns* is classified as a named entity of type *person* (by assigning *PER* to the corresponding node label *NX*).[1]

Referential relations The TüBa-D/Z contains eight types of referential relations. Besides the two most prominent referential relations – i.e., coreference and anaphora – the treebank includes cataphoric relations, expletives, bound relations, split antecedents, instances, and inherent reflexives.[2]

Details about all treebank annotations, except for the referential relations, are documented in the TüBa-D/Z stylebook [Telljohann et al., 2012]. The annotation of referential relations is specified in Naumann [2007].

5.3.2 Selection of Words to be Sense-Annotated

The sense annotation in the TüBa-D/Z is geared toward the lexical sample task in WSD (as in many existing corpora, including Kilgarriff [1998a], Palmer et al. [2001], Raileanu et al. [2002], Mihalcea et al. [2004], Broscheit et al. [2010], and

[1] Named entities are annotated on the phrase level, as the example illustrates. Additionally, on a different level of annotation, the part-of-speech tag for each individual token included in the phrase is often (but does not necessarily have to be) annotated with the POS tag *NE*.

[2] From this annotation layer only information on expletives is used in the WSD experiments.

Passonneau et al. [2012]), rather than toward the all-words task. The decision against sense annotation of all words of running text in a selected subcorpus is motivated by the requirements of machine learning. Such data are useful for training (semi-)automatic machine learning models only if there are sufficiently many instances for each item to be classified. Due to limitations of how much text can reasonably be annotated manually in an all-words, sense-annotated corpus, the resulting numbers of instances for each token are not of sufficient frequency for machine-learning applications. The selection of lemmas to be sense-annotated in the TüBa-D/Z was guided by the following criteria:

1. The selected lemmas have at least two senses in GermaNet and occur at least 16 times in the TüBa-D/Z (release 8.0).

2. The sample as a whole represents a good balance of frequencies and number of distinct word senses.

3. The selected words include both nouns and verbs so as to be able to compare and evaluate the effectiveness in WSD of structured linguistic information present in treebanks across the two word classes.

4. For verbs, the selected lemmas display different degrees of correlations between word senses and verbal frames.

As a result of the above criteria, a total of 109 lemmas (30 nouns and 79 verbs) were selected for manual sense annotation. Table 5.3 provides an overview of the entire lexical sample annotated in the TüBa-D/Z. All numbers in this chapter (and, thus, in this table) refer to release 8.0 of the treebank.[1]

The nouns are chosen by frequency and polysemy so as to be able to analyze the impact of different amounts of instances in the training data and different degrees of polysemy on automatic WSD. Altogether, the 30 nouns occur 7 538 times in the TüBa-D/Z 8.0 – at least 22 times and at most 1 427 times. On average, there are 251 occurrences per noun lemma. The average polysemy

[1]Since the sense-annotated corpus used as a gold standard throughout later chapters (see Section 6.4 for a description) is based on the newer release 9.1, it is about 13% larger than the one described here.

(number of senses in GermaNet) is 3.97 for the annotated nouns, ranging from 2-7 senses.

Table 5.3: Quantitative statistics of sense-annotated words.

	Nouns	**Verbs**
Total number of annotated word lemmas	30	79
Total number of occurrences in TüBa-D/Z	7 538	7 967
Frequency range (occurrences/lemma)	22–1427	16–710
Average frequency (occurrences/lemma)	251	101
Polysemy range (senses/lemma)	2–7	2–14
Average polysemy (senses/lemma)	3.97	2.84

The 30 selected lemmas for nouns are listed in Table 5.4 in decreasing order of their number of occurrences in the TüBa-D/Z (column $F\downarrow$). Column *GN* contains the noun's number of senses in GermaNet. Albeit inter-annotator agreement (columns *IAA* and κ) of the manual sense annotation is discussed in Section 5.3.4 below, the values are already listed in order not to replicate the table at that point.

Table 5.4: 30 selected nouns to be sense-annotated. Abbreviations: $F\downarrow$ (frequency, decreasing order), *GN* (number of senses in GermaNet), *IAA* (inter-annotator agreement), κ (Cohen's kappa).

Nouns	**F↓**	**GN**	**IAA**	κ	**Nouns**	**F↓**	**GN**	**IAA**	κ
Frau	1427	3	98.7	96.0	Spur	85	5	84.7	73.1
Mann	980	3	98.8	94.7	Anschlag	81	5	95.9	61.4
Land	962	7	97.9	95.5	Bein	78	3	97.4	-1.3
Haus	668	5	85.8	60.7	Karte	75	4	99.6	100
Partei	606	3	97.3	62.6	Runde	75	6	92.0	88.1
Grund	404	5	99.8	96.9	Sender	72	5	83.8	60.8
Stunde	367	4	98.1	92.8	Stuhl	49	3	98.0	100
Mal	250	2	100	–	Gewinn	46	3	95.7	89.2
Stimme	250	3	98.0	96.0	Ausschuss	45	2	100	–
Kopf	228	6	97.8	84.0	Bestimmung	40	6	90.8	79.2
Band	150	5	98.7	96.9	Überraschung	39	3	96.6	93.0
Tor	125	4	100	100	Teilnahme	31	3	98.9	–
Freundin	115	3	97.1	95.9	Abfall	24	4	100	100
Höhe	112	2	65.8	11.3	Kette	23	4	73.9	62.5
Fuß	109	3	99.1	79.7	Abgabe	22	5	100	100

For verbs, 7 967 verb occurrences are annotated with the senses of 79 verb lemmas (see Table 5.3). The average occurrence per verb lemma is 101 with the least frequent verb occurring 16 times, the most frequent one 710 times. The average polysemy is 2.84, with the most polysemous verb showing 14 senses in GermaNet.

Table 5.5 lists the 79 selected lemmas for verbs – again ordered by their frequency in the TüBa-D/Z (column $F\downarrow$) – with their number of senses (column GN) and inter-annotator agreement (columns IAA and κ). Compared to the noun's table, the table for verbs additionally denotes the degrees of correlation between the verb's word senses and frames (column C). The classification of verbs into the four different correlation classes 1 to 4 is described in the following.

The selection of verbs is guided by different degrees of correlations among word senses and verbal frames. The syntactic structure in which a verb occurs is often highly predictive of different word senses. The German verb *enthalten* is a case in point. It has two distinct word senses of 'to contain' and 'to abstain', which correspond directly to two distinct verbal frames. The former requires a verbal frame with a nominative and with an accusative object, i.e., frame *NN.AN* in the GermaNet notation, as in sentence (3), while the latter requires a reflexive pronoun as its object as in sentence (4) with frame *NN.AR*.[1]

(3) *[Das Medikament]^{NN} enthält Alkohol^{AN}.*
('The medicine contains alcohol.')

(4) *Ich^{NN} enthalte mich^{AR} eines Urteils.*
('I abstain from passing judgment.')

Verbs differ, however, in the degree of correlation between word senses and verbal frames, ranging from total correlation to complete lack of correlation. While *enthalten* (described above) is an example for perfect correlation, the German verb *begrüßen* is an example of the latter kind. Its senses of 'to greet someone' and 'to have a positive attitude toward something' both have a verbal frame with a nominative and accusative noun phrase, i.e., *NN.AN*.

[1]Most examples in this subsection are shortened versions of GermaNet's example sentences. The notation of verbal frames in GermaNet is explained in Section 3.7.

Table 5.5: 79 selected verbs to be sense-annotated. Abbreviations: $F\downarrow$ (frequency, decreasing order), GN (number of senses in GermaNet), C (correlation class), IAA (inter-annotator agreement), κ (Cohen's kappa).

Verbs	F↓	GN	C	IAA	κ
heißen	709	4	2	94.0	90.9
gelten	443	5	2	97.7	96.2
erhalten	346	4	3	89.2	76.4
setzen	341	14	4	79.5	75.4
sitzen	303	7	3	92.4	85.6
fragen	294	2	1	99.7	99.0
aussehen	216	2	2	92.1	75.8
reden	195	3	3	80.0	45.8
sterben	189	2	3	98.4	79.2
ankündigen	187	2	1	100	100
verkaufen	173	5	4	92.5	75.8
unterstützen	160	2	3	95.7	38.1
bedeuten	159	3	1	98.1	79.2
leisten	152	3	2	90.3	83.1
bauen	151	3	2	95.4	83.2
verurteilen	150	2	3	96.0	86.6
reichen	137	4	4	95.6	91.8
verschwinden	126	2	3	73.8	47.4
geschehen	123	2	1	98.4	49.5
gründen	118	4	2	99.2	93.0
präsentieren	118	2	1	98.9	98.0
freuen	110	2	1	90.9	64.0
informieren	109	2	1	98.2	93.0
herrschen	103	2	3	99.0	90.4
verdienen	103	2	2	100	100
aufrufen	96	2	2	99.0	66.3
demonstrieren	96	3	2	87.5	77.5
holen	93	6	4	74.2	65.4
weitergehen	87	2	3	98.9	88.3
verfolgen	85	6	3	91.8	89.0
versichern	85	3	1	96.5	80.9
enthalten	84	2	1	100	100
liefern	83	4	4	88.0	82.5
vorliegen	83	2	1	95.0	84.9
besitzen	82	2	3	92.7	84.2
drängen	76	3	2	88.2	81.7
erweisen	72	3	1	97.2	84.6
existieren	72	2	3	98.6	0.0
behandeln	71	4	3	85.6	78.0
begrüßen	68	2	3	98.5	96.2
betragen	67	2	1	100	–
beraten	65	3	2	90.8	81.4
beschränken	61	2	1	96.7	93.2
widmen	57	2	1	100	100
empfehlen	54	3	2	100	100
gestalten	54	2	1	96.3	78.0
entziehen	53	3	2	96.2	92.6
merken	53	2	1	98.1	89.9
engagieren	49	2	1	100	100
bekennen	47	2	1	97.9	95.7
wundern	46	2	1	97.8	94.5
auffallen	45	2	1	97.8	95.1
rücken	43	2	1	100	100
raten	42	2	1	100	100
bedenken	41	3	2	97.6	94.0
gestehen	40	2	3	90.0	80.0
berufen	35	2	1	100	100
klappen	35	3	2	100	100
kündigen	35	2	1	71.4	42.9
zugehen	35	6	4	94.3	87.6
erweitern	34	2	1	97.1	84.1
versammeln	33	2	1	97.0	93.9
ärgern	33	2	1	100	100
befassen	32	2	1	96.9	86.0
trauen	32	4	1	92.2	91.1
vollziehen	32	2	1	96.9	93.4
zurückgeben	32	2	1	100	100
verstoßen	31	2	1	100	100
einschränken	30	2	1	100	100
beschweren	27	3	2	100	100
identifizieren	27	3	2	92.6	88.3
nützen	27	2	1	100	–
stehlen	26	2	1	100	100
vorschreiben	26	2	3	96.2	0.0
kleben	24	2	1	87.5	74.3
verdoppeln	23	2	1	100	100
fressen	22	3	4	72.7	54.3
wiedergeben	21	3	4	76.2	40.0
verlesen	16	2	1	100	–

The inclusion of different degrees of correlations between word senses and verbal frames for verbs to be sense-annotated in TüBa-D/Z makes it possible

to systematically assess the impact of information about syntax (e.g., verbal frames) and of lexical semantics (e.g. the collocational behavior of the target lemma) on machine-learning models for WSD. Ideally, features encoding verbal frame information should suffice for verbs such as *enthalten*, while for WSD of verbs such as *begrüßen* they carry no weight whatsoever. In order to provide a fine-grained spectrum of possible degrees of correlations between word senses and frames for verbs, the verbs selected for sense annotation fall into four distinct classes:

Class 1 All verbs in this class have distinct verbal frames for their word senses, such as the above-illustrated example of *enthalten* (see Table 5.6).

Table 5.6: Example for sense/frame correlation class 1.

enthalten	**Frame**	**Example sentence**
to contain	NN.AN	*[Das Medikament]^{NN} enthält Alkohol^{AN}.* 'The medicine contains alcohol.'
to abstain (from)	NN.AR	*Ich^{NN} enthalte mich^{AR} eines Urteils.* 'I abstain from passing judgment.'

Class 2 This class contains verbs where at least one sense has a verbal frame distinct from the frames for the other senses. An example is given in Table 5.7 for the word *entziehen*. The first two senses 'withdraw' and 'extract' both have the same verbal frame of a nominative, an accusative, and a dative object, while the third sense 'to shirk doing sth.' is distinguished by requiring a nominative, a dative object, and a reflexive pronoun.

Table 5.7: Example for sense/frame correlation class 2.

entziehen	**Frame**	**Example sentence**
to withdraw	NN.DN.AN	*Sie^{NN} entzog ihm^{DN} [das Taschengeld]^{AN}.* 'She withdrew his pocket money.'
to extract	NN.DN.AN	*Salz^{NN} entzieht [dem Körper]^{DN} Wasser^{AN}.* 'Salt extracts water from the body.'
to shirk doing sth.	NN.DN.AR	*Er^{NN} entzog sich^{AR} [der Arbeit]^{AN}.* 'He shirked the work.'

Class 3 All verbs in this class share the same verbal frames for all of their senses, as the above given example of *begrüßen* where both senses have a frame with a nominative and an accusative noun phrase (see Table 5.8).

Table 5.8: Example for sense/frame correlation class 3.

begrüßen	**Frame**	**Example sentence**
to greet so.	NN.AN	*Sie^{NN} grüßte ihn^{AN} freundlich.* 'She greeted him friendly.'
positive attitute	NN.AN	*[Der Chef]^{NN} begrüßt [diese Lösung]^{AN}.* 'The boss welcomes this solution.'

Class 4 This class comprises verbs that do not fall into any of the classes 1-3. It contains for example the verb *liefern* with altogether four senses. As illustrated in Table 5.9, two senses share the same nominative/accusative frame and the other two senses have an additional dative object. Hence this verb does not belong to any of the classes 1-3.

Table 5.9: Example for sense/frame correlation class 4.

liefern	**Frame**	**Example sentence**
to supply	NN.AN	*[Der Boden]^{NN} liefert [gute Erträge]^{AN}.* 'The soil supplies good yields.'
to provide	NN.AN	*Sie^{NN} liefert [genügend Beweise]^{AN}.* 'She delivers enough evidence.'
to have a fight with sb.	NN.DN.AN	*Sie^{NN} liefern sich^{DN} [einen Kampf]^{AN}.* 'They have a fight.'
to deliver	NN.DN.AN	*Sie^{NN} liefern uns^{DN} [die Ware]^{AN} per Post.* 'They deliver the goods to us by post.'

The different degrees of correlation between word senses and frames determine the systematic selection of verbs to be sense-annotated in the TüBa-D/Z. Only verbs that have at least two senses in GermaNet are considered. Altogether, GermaNet contains 2 739 verb lemmas with 7 680 senses[1], i.e., on average 2.8 senses per lemma. An inspection of these verb lemmas resulted

[1]Numbers are calculated on GermaNet 8.0, including only non-artificial GermaNet senses.

in the distribution of the four sense/frame correlation classes in GermaNet as documented in the upper part of Table 5.10 denoted as *All GN verbs*. That is, about 22.6% of all non-articifial, polysemous verbs are classified into class 1, 25.7% into class 2, 47.6% into class 3, and the remainder of 5.1% falls into class 4.

Table 5.10: Distribution of the four sense/frame correlation classes.

	Correlation class	**1**	**2**	**3**	**4**	**Total**
All GN verbs	Total lemmas	592	703	1 303	141	2 739
	Lemmas per class (in %)	21.6	25.7	47.6	5.1	100%
	Number of senses	1 215	2 669	3 039	757	7 680
	Average polysemy	2.05	3.8	2.3	5.4	2.8
	Average frequency	9	68	15	74	23
Annotated verbs	Total annotated lemmas	38	17	16	8	79
	- in representative sample	8	9	16	8	41
	- in *high correlation* sample	30	8	0	0	38
	Average polysemy	2.13	3.06	2.88	5.62	2.84
	Average frequency	65	145	132	114	101
	Total annotated occurrences	2 476	2 466	2 116	912	7 950

The total set of annotated verbs is divided into two nearly equally sized samples. One sample that is representative and well-balanced according to the class distributions in GermaNet (as shown in the upper part of Table 5.10) and a second sample that contains verbs with a high correlation between word senses and frames (i.e., verbs from classes 1 and 2). The motivation behind this distinction into two samples is to facilitate the evaluation of WSD algorithms that use verbal frame information for disambiguation. If a WSD algorithm fails to perform well on the 'high correlation' sample, this would provide evidence that such an algorithm has major drawbacks and will not work for the representative, well-balanced sample at all. Furthermore, a comparison on how machine learning algorithms perform on the two samples is highly interesting. That is, to test the deviation in performance and learn about how powerful such an algorithm can be for verbs that show a high correlation between word senses and verbal frames. The implementation, evaluation, and discussion of such WSD experiments are topic of Chapter 8, in particular, of Subsection 8.3.4.

The first sample should not be biased toward specific sense/frame correlation classes. The verbs in this sample should rather be fairly distributed over the different classes insofar that they are distributed 'representatively' according to the ratios of how many verbs in total occur in GermaNet for which of the different classes. The amount of lemmas per correlation class included in this lexical sample are listed in the second row denoted as *in representative sample* in the lower part of Table 5.10.

The second sample contains verbs from those classes that allow a clear distinction between different word senses with the help of their frames. That is, those verbs where an automatic WSD algorithm based on verbal frame information is expected to return good results since using the verbal frames should allow distinguishing different senses of a word reliably. This means that this second sample contains verbs from classes 1 and 2 (see row *in high correlation sample* in the lower part of Table 5.10).

5.3.3 Annotation Process

In order to assure good quality of the manual sense annotation and to calculate inter-annotator agreement (see Section 5.3.4), sense annotation is independently performed by two annotators[1] (native German computational linguists) for all word lemmas and occurrences. The annotators have the possibility to indicate problematic word occurrences with comments to be discussed separately.

The manual annotation is performed lemma-by-lemma (as in many related annotation projects, for example, Kilgarriff [1998a], Fellbaum et al. [2001], Saito et al. [2002], and Passonneau et al. [2012]), i.e., an annotator first takes a look at all senses of a word in GermaNet and then – having in mind all possible senses – annotates each occurrence of that word in the TüBa-D/Z with the corresponding sense from GermaNet.

For each occurrence of a word in the treebank, the annotators are supposed to select exactly one GermaNet sense from the list of available word senses, if

[1]The two annotators are both familiar with GermaNet.: one annotator is Valentin Deyringer, a student assistant of the GermaNet project, and the other annotator is Verena Henrich, this thesis' author.

possible. Since it is not always possible to select exactly one sense, i.e., when it is either unclear or undecidable which of two senses is illustrated or none of the senses is plausible, the annotation guidelines allow the assignment of multiple senses or no sense for a word occurrence. The need to annotate more than one sense does not arise very often.[1] This confirms both the results of Raileanu et al. [2002] who annotated only 79 out of 2 421 occurrences with multiple senses and the findings of Véronis [1998, page 6] that "the average number of senses [...] is not very high, which shows that annotators have a tendency to avoid multiple answers."

By contrast, annotators choose more often no sense from the list available.[2] Most of these cases are idiomatic expressions or figurative meanings where it is not obvious which sense to choose. For example, the idiomatic expression *jdm./etw. das Wort reden* ('to put the case for sb./sth.') occurs several times in the TüBa-D/Z. For example in the phrase:

(5) *Nawrocki, der immer privat organisierten Olympischen Spielen das Wort geredet hat, ...*[3]
('Nawrocki, who always put the case for privately organized Olympic Games, ...')

Since for this example annotators cannot produce one of the three senses for *reden*, i.e., (i) 'to give a speech', (ii) 'to talk', or (iii) 'denoting the way of how to speak', this occurrence is marked as *idiomatic* and not annotated with a sense from GermaNet.

An experienced lexicographer[4], who is a native speaker of German and who has been the main responsible expert for the lexicographic extension of GermaNet for several years, supervises the two annotators. In an adjudication step, the expert goes through all occurrences, where the two annotators either do not agree or at least one of them had a comment, and resolves disagreements. This procedure of conducting independent annotations with an adjudication step afterwards is along the lines with most other sense-annotation projects,

[1] Only 2 out of the more than 15 000 occurrences are annotated with two senses.
[2] 102 out of the more than 15 000 occurrences are not assigned a GermaNet sense.
[3] Part of sentence 24 016 from TüBa-D/Z 9.1.
[4] Namely, Reinhild Barkey.

including for example Kilgarriff and Rosenzweig [2000], Fellbaum et al. [2001], and Passonneau et al. [2012].

Where annotators found during the annotation process that a sense is missing from GermaNet, GermaNet is updated to include that sense. If the TüBa-D/Z contains occurrences of senses for the selected lemmas that are currently not covered by GermaNet, the two annotators indicate for these occurrences that a sense is missing in GermaNet. For example, the noun *Mann* had the following two senses in GermaNet 7.0: (i) 'man' in the general sense of an adult male person, and (ii) 'husband' in the more specific sense of a married man. In sentence (6), the noun *Mann* is used as a 'unit for counting manpower'.

(6) *Er will die Personalstärke der Bundeswehr auf 270.000 Mann reduzieren.*[1]
('He wants to reduce the manning level of the German Armed Forces to 270,000 men.')

The lexicographic expert decides whether to add a missing sense to GermaNet. In the case of *Mann*, the mentioned *counting unit* sense has been included.[2] The subsequent update of the sense inventory during the sense annotation process brings about mutual benefits both for the sense inventory which is being extended and for the sense-annotated corpus which profits from a feasible sense inventory. Such an update is common practice for all those annotation projects where the sense-annotated corpus is being created by the same research group which maintains the sense inventory (e.g., Miller et al. [1993], Kilgarriff [1998a], Palmer et al. [2001], and Passonneau et al. [2012]).

The annotation process is supported by an online tool[3] that shows for each word occurrence in the TüBa-D/Z both the context (highlighting the word itself in bold) and the list of possible GermaNet senses from which the correct one has to be manually selected. Figure 5.7 presents a screenshot with three annotated occurrences for the lemma *Fuß* 'foot'.

[1] Sentence 10 692 from TüBa-D/Z 9.1.

[2] In GermaNet 8.0

[3] This tool was developed by Yannick Versley for the purpose of previous annotation tasks in the TüBa-D/Z treebank such as the annotation of explicit and implicit discourse relations [Gastel et al., 2011]. Yannick made it possible to reuse the tool for WSD annotation.

Figure 5.7: Screenshot of the online annotation tool showing three occurrences of the noun *Fuß*.

GermaNet contains three senses for the noun *Fuß*. They are listed (including the corresponding GermaNet identifiers) as (i) *[35740_Körperteil]* 'part of the body', (ii) *[46238_Maßeinheit]* 'unit of measurement', referring to *foot* as a unit of length, and (iii) *[9370_Artefakt]* 'artifact', representing the base of objects such as furniture. Clicking on a sense label easily allows selecting and deselecting the corresponding sense. Selected senses are highlighted in bold. In the example screenshot in Figure 5.7, the first occurrence is annotated with the *artifact* sense, whereas the second and third occurrences are each annotated with the *part of the body* sense. The *comment* text fields allow to indicate problematic word occurrences and to mark occurrences as *idiomatic* or *transferred meaning*. The third occurrence in the figure, for example, contains the

comment *übertragen* 'transferred meaning'.

The context presented to the annotators consists of the complete sentence in which the word occurs plus the preceding sentence. The hyperlinked sentence numbers (for example, *s19412*, *s19684*, etc.) allow easy inspection of the whole newspaper article in case the immediate context is too narrow. In addition, the annotators have access to all linguistic annotations in the TüBa-D/Z treebank and to all sense-related information in GermaNet.

5.3.4 Inter-Annotator Agreement

An inter-annotator agreement (IAA) score is calculated to assess the reliability of the manual sense annotations. The calculated percentage of IAA accounts for partial agreement using the Dice coefficient [Véronis, 1998]. The overall percentage of agreement, which is obtained by averaging the Dice coefficient for all annotated occurrences of the word class in question, is 96.4% for nouns and 93.7% for verbs. This corresponds to Cohen's kappa κ [Cohen, 1960] values of 85.4 and 82.4 for nouns and verbs, respectively.[1]

The agreement for each of the 30 nouns is documented in columns *IAA* and κ in Table 5.4 (in Section 5.3.2 above). With the two exceptions of *Höhe* (65.8%) and *Kette* (73.9%), the calculated IAA values for all other nouns are at least above 80%, mostly even above 90%. The explanation for the low agreement of the noun *Höhe* is due to the semantic distinction of the two word senses in GermaNet which are very fine-grained and turned out to be difficult to distinguish during the manual annotation process. The reason for a low performance for *Kette* stems from a subsequent restructuring of the sense inventory during the annotation process. A revision of senses in GermaNet has been performed after one annotator had already tagged 5 out of 23 occurrences (which constitutes already more than 20%) with a sense of *Kette* that has been deleted. The tagging by the second annotator is conducted on the already

[1]Since Cohen's kappa does not allow multiple categories (i.e., multiple senses) for a word, the technique by Raileanu et al. [2002] of ignoring all words where one of the two annotators selected more than one sense is followed. In this study, there are 60 such occurrences for nouns and 17 for verbs. Furthermore, when both annotators always pick one sense for all occurrences of a lemma, the kappa coefficient is not informative and those occurrences are ignored in calculating the reported average.

revised set of senses and thus the deleted word sense is never chosen.

The kappa coefficients (column κ in Table 5.4) show a much higher deviation compared to the percentages of IAA. Here, the two by far worst results are obtained for *Höhe* (11.3) and *Bein* (-1.3), while all other kappa values lie above 60. The low κ for *Höhe* was to be expected as a result of an already low percentage of IAA. The explanation for a negative value for κ is that there is even less agreement between the annotators than an agreement by chance would be. The reason for the negative kappa value for *Bein* is due to the skewed distribution of annotated senses, i.e., the same predominant sense is assigned to 77 out of 78 occurrences. The agreement by chance is thus nearly 1 and a deviation in the manual annotation influences the calculated coefficient enormously. For lemmas where both annotators always pick the same sense for all occurrences, Cohen's kappa is not informative. It is technically not possible to calculate the coefficient for these lemmas, because the agreement by chance is 1, which would result in a division by zero. This is the reason why there are no κ values for the three nouns *Mal*, *Ausschuss*, and *Teilnahme*.

A detailed inspection of the IAA for single words did not show a correlation between the IAA and the polysemy of a noun. The Pearson correlation coefficient [Pearson, 1896] between the IAA and the number of senses a noun has is -0.03, with a p-value of 0.88, i.e., without statistical significance. For example, the most problematic nouns mentioned above show different numbers of senses, i.e., 2 for *Höhe*, 3 for *Bein*, and 4 for *Kette*. Further, the reasons for the annotator disagreement are diverse and obviously not connected to the polysemy of a word, i.e., unclear sense distinction, skewed distribution of annotated senses, and subsequent update of the sense inventory, respectively. The other way around, the IAA values for the most polysemous nouns, i.e., 97.9% for *Land* (7 senses), 97.8% for *Kopf*, 92.0% for *Runde*, and 90.8% for *Bestimmung* (6 senses each) are comparable to the average of 96.4% for all annotated nouns. For nouns, this finding that there is no obvious correlation between the IAA and a word's polysemy corroborates the results reported by Fellbaum et al. [2001] on sense-annotating the Penn Treebank with senses from WordNet.

For each of the 79 verbs, the percentage of inter-annotator agreement (col-

umn *IAA*) and Cohen's kappa (column κ) are listed in Table 5.5 (in Section 5.3.2 above). In general, the inter-annotator agreement for verbs is slightly lower than for nouns. However, similar to nouns, the calculated IAA values for most verbs are at least above 80%, mostly even above 90%. The few exceptions with a higher disagreement are *verschwinden* with 73.8%, *holen* with 74.2%, *kündigen* with 71.4%, *fressen* with 72.7%, and *wiedergeben* with 76.2%. For most of them (i.e., for *verschwinden*, *holen*, *kündigen*, and *wiedergeben*), the difficulty is mainly caused by a fine-grained distinction of senses which make an unambiguous annotation difficult. This detrimental effect for very fine-grained word senses was already observed by Palmer et al. [2006, page 97]. They report an improvement in the inter-annotator agreement from 71.3 to 82% for the same SensEval-2 lexical sample task when more coarse-grained verb senses are used instead of the fine-grained distinctions taken from WordNet 1.7. In the case of *holen*, an additional complexity arises due to the addition of two new word senses during the annotation process. For the verb *fressen*, most disagreements occur for transferred usages of the verb.

For the same reasons that cause a lower percentage of IAA, it was expected for those verbs to yield lower κ scores, which turned out to be true (i.e., *verschwinden* (47.4), *holen* (65.4), *kündigen* (42.9), *fressen* (54.3), and *wiedergeben* (40.0)). The explanation for kappa coefficients of 0.0 for the two verbs *existieren* and *vorschreiben* is a skewed distribution of annotated senses. Both for *existieren* and for *vorschreiben*, all except one occurrence (by one of the two annotators) are assigned the same word sense. This results in an agreement by chance of nearly 1 which in turn results in a very low kappa coefficient.[1] For the verbs *betragen*, *nützen*, and *verlesen* no κ values are given because both annotators always picked one sense for all occurrences and thus the coefficient is not informative for those lemmas.

For verbs, the observation for the TüBa-D/Z sense annotation differs from Fellbaum et al.'s [2001] finding that there is no obvious correlation between the IAA and a word's polysemy when sense-annotating the Penn Treebank.

[1]Since the chance agreements for these cases are nearly 1, Cohen's kappa is basically not informative for those cases, but since it is not exactly 1, it is technically possible to calculate the coefficient.

For the sense annotation in the TüBa-D/Z, the Pearson correlation coefficient between the inter-annotator agreement and the polysemy of a verb is -0.39. The coefficient's absolute value is not remarkably high[1] to claim a strong correlation, but there is at least a higher correlation than for nouns, and, with a p-value smaller than 0.001, the correlation for verbs is statistically significant.

Overall, the reported percentage of IAA is very high. The values are comparable to the agreement statistics reported in Raileanu et al. [2002] for their work in creating a German sense-annotated corpus. The observed agreement values are much higher than those observed for English. Véronis [1998], for example, observes a pairwise Dice coefficient of 73% for nouns and 63% for verbs. Palmer et al. [2006] report an inter-annotator agreement of 71.3% for the English verb lexical sample task for SensEval-2. The reason for much higher IAA values for German than for English is the different number of distinct senses: an average of 3.97 for German nouns and 2.84 for German verbs (see the last column in Table 5.3) as opposed to an average of 7.6 for English nouns and 12.6 for English nouns in the case of Véronis [1998, Table 3].

5.4 Comparison of WebCAGe and TüBa-D/Z

The purpose of this section is to contrast the two sense-annotated corpora that were described in this chapter, to analyze advantages and disadvantages of each of them, and to compare them with existing German sense-annotated corpora. The main commonalities between WebCAGe (see Section 5.3), the sense-annotated TüBa-D/Z (see Section 5.2), and the two other sense-anntated corpora available for German constructed by Raileanu et al. [2002] and Broscheit et al. [2010][2] are that they all (i) use GermaNet as the sense inventory, (ii) follow the lexical sample variant, i.e., several occurrences of a closed set of lemmas are sense-annotated, and (iii) are freely available to the research community[3].

[1]Since 'not remarkably high' describes the absolute value of the correlation coefficient, it means that the coefficient is 'not remarkably distinct from zero'.

[2]Note that Broscheit et al. [2010] annotate the German deWaC corpus and that these sense annotations are updated to the most recent version of GermaNet (see Section 6.5 for details) and reused for the WSD experiments in Chapters 7 and 8.

[3]Note that, unfortunately, only parts of WebCAGe are freely available for download due to legal restrictions on some of the underlying textual materials.

Table 5.11 summarizes several properties of the four sense-annotated corpora. Both WebCAGe and the sense-annotated TüBa-D/Z are considerably larger than the other two sense-annotated corpora. WebCAGe has by far the highest diversity of word lemmas annotated whereas TüBa-D/Z has most overall sense-annotated occurrences. It is important to keep in mind, though, that WebCAGe is the result of an automatic harvesting method, whereas the other resources were manually constructed. This automatic method can constitute a viable alternative to the labor-intensive manual method only if it is of sufficiently high quality that it can be used as is or can be further improved with minimal manual post-editing. WebCAGe's method proved high quality and, additionally, all automatic sense annotations were manually post-corrected. It should be noted that the development time and thus the costs for the construction of an automatically harvested corpus is much less compared to the manual sense annotation of a comparable amount of occurrences.

Table 5.11: Comparing WebCAGe and the sense-annotated TüBa-D/Z to other sense-tagged corpora of German.

		WebCAGe	**Annotated TüBa-D/Z**	**Broscheit et al. 2010**	**Raileanu et al. 2002**
Sense tagged word lemmas	Adj.	211	0	6	0
	Nouns	1 499	30	18	25
	Verbs	897	79	16	0
	All POS	2 607	109	40	25
Total # of tagged word tokens		10 750	15 505	1 154	2 421
Average frequency		4	142	29	97
Average polysemy		2.9	ca. 3	4.8	3.3
Domain independent		yes	yes	yes	medical domain
Underlying texts		web data	newspaper	web data	scientific abstracts
Additional annotations		none	lemma, POS, morph., syntax, coreference, etc.	lemma, POS	lemma, POS, morph., syntax, UMLS senses
Creation type		automatic	manual	manual	manual
Update of sense inventory		no	yes	no	no

Although the average polysemy of the sense-annotated lemmas is similar, i.e., it is between 2.9 and 3.3 for three out of four corpora (with one exception of 4.8), the average frequencies show a much higher deviation – ranging from 4 for WebCAGe to 142 for TüBa-D/Z (see Table 5.11). This reveals the major advantage of the sense-annotated TüBa-D/Z compared to the other corpora, especially compared to WebCAGe. The explanation why there are apparently few annotated occurrences per word lemma in WebCAGe is due to the automatic creation approach. The WebCAGe harvesting method produces sense-annotated examples for a very large number of lemmas, but only a few examples for each lemma. In contrast, the sense annotation in the TüBa-D/Z comprises an even larger number of occurrence for a much smaller set of lemmas. If the purpose of the corpus is to test the scalability of a WSD algorithm on as many distinct lemmas as possible, this favors WebCAGe. If the purpose is to train machine learning models, where there is a need of sufficiently many instances for each item to be classified, TüBa-D/Z is clearly more suitable.

The main reason why two sense-annotated corpora have been constructed in the present chapter is to experiment with the impact of the two kinds of corpora on automatic WSD (see Chapters 7 and 8). Different genres and different text types are expected to be less relevant for WSD than, for example, the availability of linguistic annotations, distinct motivations of which lemmas are sense-annotated, and different frequencies of annotated occurrences per lemma.

For many natural language processing applications – not only for WSD – it is useful to know which senses are more frequent than others and which ones are the most frequent ones for a lemma. While it is possible to extract this information from TüBa-D/Z[1], it is not possible for WebCAGe, because the harvesting method returns a very skewed number of annotated senses and lemmas and does not annotate all subsequent occurrences of a lemma in a coherent text.

Finally, note that the sense-annotated TüBa-D/Z is the only GermaNet sense-annotated corpus where the sense inventory was updated during the

[1]At least for general language taken from newspaper texts.

annotation process to ensure completeness.

5.5 Conclusion and Continuing Work

The current version of GermaNet has not been used in sense-annotation projects other than described in this thesis. This chapter has described the construction of two sense-annotated corpora using the most current version of GermaNet with the goal of providing a gold standard for the development and evaluation of word sense disambiguation systems. The two corpora follow two radically different construction approaches and have different strengths and weaknesses. Both are freely available to the research community.[1] The obvious next steps after constructing these sense-annotated corpora include:

- Experiments with word sense disambiguation algorithms for German using the newly created sense-annotated corpora as a gold standard.
- Comparison and impact on WSD of the two kinds of sense-annotated corpora (different genres, different text types, different linguistic annotations available, different underlying text quality, different motivation of which lemmas are sense-annotated, different numbers of annotated occurrences per lemma) on different WSD algorithms.
- The implementation, evaluation, and discussion of automatic WSD using contextual features in order to investigate on the influence of syntax and semantics on automatic WSD for verbs.

These topics are addressed in Chapters 7 and 8.

[1] See `http://www.sfs.uni-tuebingen.de/en/webcage.shtml` and `http://www.sfs.uni-tuebingen.de/en/ascl/resources/corpora/sense-annotated-tueba-dz.html`

Chapter 6

Gold Standard Corpora

This chapter describes the three sense-annotated corpora used as gold standards for the word sense disambiguation experiments in Chapters 7 and 8. To allow a fair and comparable evaluation of automatic WSD systems on different corpora, the sense inventory used for sense annotation must match for all corpora. Since the sense annotations in the gold standard corpora are each based on the most recent version of GermaNet that was available when their creation commenced, they all rely on different GermaNet releases:

(i) The web-harvested corpus WebCAGe is (semi-)automatically annotated with senses from GermaNet's release 6.0– see Section 5.2.

(ii) Manual sense annotation of the TüBa-D/Z treebank, described in Section 5.3, relies on GermaNet's version 8.0.

(iii) Broscheit et al. [2010] manually annotated a lexical sample in the deWaC corpus with senses from GermaNet 5.1.

In order to make the experimental WSD results for the three corpora comparable and up-to-date, all sense annotations are updated to the most recent version of GermaNet available at this point of the dissertation – which is release 9.0, as of April 2014 (see Section 3.9). While knowledge-based word sense disambiguation experiments (Chapter 7) can and are evaluated on all available annotations, supervised WSD (Chapter 8) requires both a separate test set for evaluation as well as a certain amount of annotated instances per lemma for

training. Supervised WSD experiments are evaluated on a subset of annotations that fulfill certain criteria. Section 6.1 explains these requirements and the division of gold standard corpora into training and test sets.

A difficult issue for evaluating WSD systems is that of annotations with multiple senses or with no sense. The treatment of such annotations in the context of this work is explained in Section 6.2.

Sections 6.3, 6.4, and 6.5 of this chapter present the updated versions of the sense-annotated corpora WebCAGe, TüBa-D/Z, and deWaC, respectively. The stated corpus versions serve as gold standards for WSD experiments in later chapters. All corpora are annotated for distinct lexical samples of selected words. The corpus descriptions include statistics on the number of lemmas included in the sense-annotated lexical samples and on the total number of annotated token occurrences for each lemma. Since the annotations employed for supervised systems have to fulfill certain criteria (as described in Section 6.1), for each corpus a separate subsection reports on the annotations included in the gold standards used for supervised WSD.

Furthermore, during the WSD experiments all three corpora are automatically enriched with linguistic information including sentence segmentation, tokenization, and lemmatization. These automatic linguistic annotations are detailed in Section 6.6.

6.1 Creating Training and Test Sets

Supervised approaches to WSD utilize and, if necessary, adapt supervised learning methods to solve the task of assigning the correct sense to a word. They rely on existing training corpora whose words are already annotated with senses from a given sense inventory. While supervised WSD systems typically obtain far better results than knowledge-based systems, they presuppose the availability of training data [Màrquez et al., 2006; Navigli, 2009].

There are two main approaches to evaluate the performance of supervised algorithms: evaluating by cross-validation on all available data or evaluating on a separate, unseen test set (see Subsection 2.2.2). To meaningfully and

accurately estimate an algorithm's ability to generalize, testing on unseen data is crucial. Therefore, some annotations (referred to as the *training set*) are used for training a supervised system; another distinct portion of the annotations (the *test set*) are held back while tuning the system on the training data, and later used to obtain final evaluation results. [Palmer et al., 2006; Witten et al., 2011]

Furthermore, it is important that training and test sets are representative samples of the data. This can be achieved by proportionally stratifying all available annotations. That is, rather than dividing the annotations in the order of their occurrence or completely randomly, training and test portions are proportionally stratified, which means that the overall proportionality of class occurrences in the dataset is preserved. Thus, proportional stratification guarantees – to a certain degree – representative samples. [Witten et al., 2011]

As a drawback of separate, non-overlapping training and test sets – particularly prevalent when the amount of annotated data is limited – the evaluation results strongly depend on the specific split of the data into training and test samples [Refaeilzadeh et al., 2009]. Although evaluation by (repeated) cross-validation on all available data could help to lower this variance problem, the supervised experiments in Chapter 8 are evaluated on a completely unseen test set. This evaluation procedure primarily allows a more realistic estimate of the system's ability to continue to generalize after several experiments with distinct classifiers, parameters, and features on the training set.[1] Further, this procedure is in line with most related works on supervised disambiguation of a lexical sample, e.g., Hoste et al. [2002a] and Mohammad and Pedersen [2004], and used for many SensEval and SemEval tasks [Kilgarriff and Rosenzweig, 2000; Edmonds and Cotton, 2001; Mihalcea and Edmonds, 2004; Agirre et al., 2007]. It also follows the WSD studies that make use of these SensEval/ SemEval datasets, including Lee and Ng [2002], Dinu and Kübler [2007], and de Oliveira et al. [2011] – to name only a few.

The three German gold standard corpora available for this dissertation are

[1] In order to judge the impact of the two evaluation procedures and to analyze how much the evaluation results differ for the two procedures, Appendix D compares WSD results obtained by cross-validation with results obtained on a separate, unseen test set and proves the viability of the proposed evaluation procedure.

divided into training and test sets by following a 2:1 ratio. The use of a 2:1 ratio is recommended by Palmer et al. [2006, pages 77f.] for evaluations of the WSD tasks. Since this ratio is, for example, employed by the English lexical sample tasks in SensEval-2 [Kilgarriff, 2001] and SensEval-3 [Mihalcea et al., 2004], it is, consequently, also employed by all works using these datasets, including Lee and Ng [2002], Escudero Bakx [2006], and de Oliveira et al. [2011].

Ideally, there should be a lot of sense-annotated occurrences available for each lemma – more specifically, for each word sense. [Màrquez et al., 2006] In practice, only a restricted amount of annotations exist – for some languages and lemmas there are more, for others there are less. In order to assure a minimum level of meaningfulness when experimenting with supervised methods, a critical mass of annotated material is required per lemma. This amount is specified by three criteria:

(c-i) Evaluation of supervised WSD systems is pointless for lemmas for which there is only one sense annotated, because the system would be trained only on one sense, i.e., it can classify only one sense, and the results can thus be evaluated only for this one sense available in the test set.

(c-ii) To ensure both that a supervised classifier can potentially learn a word sense and that a supervised system is potentially evaluated for a word sense, there should be for each word sense at least one annotated occurrence in the training set and at least one occurrence in the test set, respectively. That is, it makes sense to consider senses for which at least two annotations exist which can then be divided into distinct training and test sets. For technical reasons, this minimum is increased to three, because the method for stratified splitting all annotations into training and test sets from the Weka machine learning tool suite [Hall et al., 2009] – used for the supervised experiments in this thesis (see Section 8.2) – requires at least three annotations per sense to ensure the inclusion of at least one annotation per sense in both the training and the test set.

(c-iii) If all annotations are divided into training and test sets to allow the evaluation on a final test set that has been held back, there should be a

minimum number of annotated occurrences of at least 15 (counting only those senses that fulfill criterion c-ii) to ensure that, for a 2:1 split, there are at least five annotations in the test set. Although five annotations in the test set are already borderline, they constitute the lowest acceptable threshold. A minimum of five annotations divides the result scale into a maximum size of 20 percent per scale unit, below which random chance would predominate any meaningful evaluation result. For several lemmas, more annotations are available and, thus, a more fine-grained granularity range is given. Although such a more fine-grained granularity range would generally be preferred for all lemmas, it has to be counter-balanced with the manual annotation effort.

More formally, let T denote the set of tokens ($t \in T$) in the corpus, L the set of lemmas ($l \in L$) in GermaNet, s the set of senses ($s \in S$) in GermaNet, and let $A : \{\langle T, L, S\rangle\}$ denote each $\langle t, l, s\rangle \in A$ an annotation of a token (t) with its lemma (l) and sense (s).[1] The set of annotations which fulfill the three above-itemized criteria c-i to c-iii are then identified as

$A' : \{\langle T, L, S\rangle\} \subseteq A$ where

$$\begin{aligned} A' = \{\langle t, l, s\rangle \ \in A \mid \exists_{\langle t', l', s'\rangle \in A}[l = l' \wedge s \neq s'], \\ |\{\langle t', l', s'\rangle \in A \mid l = l' \wedge s = s'\}| \geq 3, \\ |\{\langle t', l', s'\rangle \in A \mid l = l'\}| \geq 15\} \end{aligned} \tag{6.1}$$

Supervised WSD experiments in Chapter 8 are evaluated only on those annotations where all three criteria are met, i.e., that fulfill Equation (6.1). That is, those annotations where a lemma has at least two word senses with at least three annotated occurrences each, and where a lemma has at least 15 annotated occurrences for those senses with at least three occurrences.

To build training and test sets, all annotated occurrences are extracted for which the corresponding lemmas and senses fulfill all three of the specified criteria. The Weka machine learning tool suite is employed to create a stratified

[1] Also see the explanation in Section 6.2 on how tokens annotated with more than one sense or no sense from GermaNet are treated.

sample of these remaining annotations per lemma into two-thirds training set and one-third test set.

Note, however, that the criteria established in this section apply only to the evaluation of supervised systems (Chapter 8), whereas knowledge-based word sense disambiguation experiments (Chapter 7) can and are evaluated on all available annotations.[1]

6.2 Treatment of Annotations with No Sense or Multiple Senses

While most annotations assign exactly one GermaNet sense to a token, a few tokens are annotated with more than one sense or no sense from GermaNet. There is very little if any information in the WSD literature on how to treat such annotations. To overcome the lack of a standard procedure, this section describes the treatment of annotations with multiple senses or no sense in the context of this work. Table 6.1 overviews these annotations for the three corpora WebCAGe, TüBa-D/Z, and deWaC. It first lists the total numbers in the overall available corpus annotations. It then shows the corresponding numbers in the subset of annotations used for evaluating supervised WSD systems. Note that the reason for why the numbers strongly differ for the three corpora is due to distinct sets of annotated lemmas in each corpus.

There are only 164 tokens (144 in the subset used for supervised evaluation) for which no GermaNet sense is annotated: 0 in WebCAGe, 107 in the TüBa-D/Z (out of 17 910 annotations, see Subsection 6.4.2), and 57 in deWaC (out of 1 083 annotations, see Subsection 6.5.3). These cases occur in the TüBa-D/Z and deWaC, for example, for idiomatic expressions or figurative meanings where it is not obvious from the context which sense to choose. The reason why there are no such annotations in WebCAGe is due to its automatic harvesting method (see Section 5.2). That is, WebCAGe harvests tokens in context only

[1]The overall sense annotations available in each of the three corpora WebCAGe, TüBa-D/Z, and deWaC is documented in Subsections 6.3.1, 6.4.2, and 6.5.3, respectively, while the amount of sense annotations included in the 'supervised' gold standards for each of the three corpora is documented in Subsections 6.3.2, 6.4.3, and 6.5.4.

Table 6.1: Annotations with no sense or multiple senses.

		No Sense	**Multiple Senses**
Overall annotations	WebCAGe	0	753
	TüBa-D/Z	107	23
	deWaC	57	0
Data for supervised WSD	WebCAGe	0	196*
	TüBa-D/Z	93	22*
	deWaC	51	0

*Only the first listed sense in these annotations is considered, see description below.

for those Wiktionary entries which have mappings to GermaNet senses, and thus it does not include tokens which are not annotated with any GermaNet sense.

In order to account for the fact that there exist contexts for which a token cannot be assigned a GermaNet sense (in TüBa-D/Z and deWaC), the *no sense* annotations are not ignored but regarded as a separate class. For both knowledge-based and supervised systems, this treatment is straightforward and does not require any modification of the WSD algorithms or evaluation procedure.

By contrast, tokens with more than one assigned sense are more difficult to handle. There are several plausible ways in which to deal with them. The fairest approach would be (i) to give partial credit to the WSD system if it disambiguates some but not all of the annotated word senses [Resnik and Yarowsky, 1997, 1999]. Other approaches could be (ii) to simply ignore these annotations in the evaluation or (iii) to score a system already as correct if it assigns only one of the multiple senses. Yet other alternatives could modify the annotations insofar as (iv) all but one sense is removed from the gold standard or (v) the original annotation in the gold standard is replaced by multiple annotations, one for each sense. Although alternatives (ii) through (v) are easier to implement, method (i) would be desirable because it gives a fairer estimate of a WSD system's performance.

Since the system for knowledge-based WSD in Chapter 7 is implemented

completely from scratch, the evaluation can account for partially correct sense disambiguation using the Dice coefficient [Véronis, 1998]. The problem with the supervised WSD environment in Chapter 8 is that the underlying machine learning tool suite Weka does not support such *multilabel* classification [Witten et al., 2011], and thus evaluation by procedure (i) is not possible. A careful weighing of alternatives (ii) to (v) proved (iv) to be a suitable alternative – for several reasons:

- Simply ignoring multiple annotations (approach (ii)) is not feasible due to the considerable number of annotations with multiple senses in WebCAGe. Although the TüBa-D/Z has only 23 such multi-sense annotations and deWaC does not include any, in WebCAGe, 753 tokens are annotated with more than one sense (out of 10 402 annotations in total, see Subsection 6.3.1).[1]

- The reason why approach (iii) is not adopted is because it would produce slightly higher evaluation results than would be genuinely correct.

- By contrast, approach (v) would slightly worsen the results because it produces duplicate contexts with distinct sense annotations which negatively affect supervised learning methods.

- Finally, the disadvantage of approach (iv) is that it artificially reduces the gold standard annotations which might lower some and improve other results quasi randomly. Whether the results for an annotation is accidentally improved, left the same, or worsened depends on the concrete annotation instance and on its resemblance to other annotations for the same lemma. That is, when all but the first annotated sense are ignored (approach (iv)), the overall results are indirectly averaged. Compared to the other alternatives, approach (iv) appears fairest overall and its drawbacks seem least severe.

[1]The explanation for why there are considerably many multi-sense annotations in WebCAGe is, again, due to the automatic creation approach. Whenever the mapping between GermaNet and Wiktionary connects more than one GermaNet sense with the same entry in Wiktionary, the harvested texts are annotated with all GermaNet senses in question.

In short, knowledge-based WSD experiments in Chapter 7 handle annotations with multiple labels by giving partial credit using the Dice coefficient (approach (i)), while supervised WSD experiments in Chapter 8 handle such annotations by ignoring all but the firstly listed sense (approach (iv)).

6.3 Updating to WebCAGe 3.0

Section 5.2 explains the web-harvesting and (semi-)automatic sense annotation of WebCAGe. Updating its sense inventory from GermaNet 6.0 to GermaNet 9.0 mainly affects annotations (i) of which the senses do not exist in the new release anymore or (ii) where the set of senses has otherwise changed for a specific lemma. With the help of persistent database identifiers that reliably identify word senses in GermaNet across releases (since release 5.2, see Appendix B), all relevant annotations can easily be identified automatically.

Those annotations that fall in class (i) are eliminated without replacement. Therefore, the updated WebCAGe contains a few less annotated tokens compared to its previous version.

All annotations in class (ii) need to be inspected individually. For each of them it has to be decided manually whether the newly added or removed senses reflect independently added or removed senses that are completely new or removed, respectively, or whether the new set of senses involves a merging or splitting of senses compared to the previously available set of senses. Depending on this manual inspection, each annotation is updated (or removed) appropriately.

The WebCAGe resource is made freely available online.[1]

6.3.1 WebCAGe 3.0 Overall Statistics

The version of WebCAGe which is compatible with GermaNet 9.0 is identified as 3.0. This version is used throughout the WSD experiments. Table 6.2 shows WebCAGe 3.0's statistics on sense-annotated lemmas and tokens.

[1] `http://www.sfs.uni-tuebingen.de/en/ascl/resources/corpora/webcage.html`

Table 6.2: Overall sense annotation statistics of WebCAGe 3.0.

	Adj.	**Nouns**	**Verbs**	**All POS**
Total # of annotated word lemmas	212	1 537	959	2 708
- in Wiktionary examples	212	1 537	959	2 708
- in external webpages	11	85	35	131
- in Wikipedia articles	2	43	3	48
- in Gutenberg texts	2	34	18	54
Total # of tagged word tokens	694	6 522	3 186	10 402
- in Wiktionary examples	560	4 090	2 962	7 612
- in external webpages	30	424	112	566
- in Wikipedia articles	79	1 381	10	1 470
- in Gutenberg texts	25	627	102	754
Frequency range (occurrences/lemma)	1–83	1–185	1–45	1–185
Average frequency (occurrences/lemma)	3	4	3	4
Polysemy range (senses/lemma)	2–10	2–11	2–26	2–26
Average polysemy (senses/lemma)	2.5	2.8	3.7	3.1

Altogether, 10 402 occurrences of 2 708 lemmas (212 adjectives, 1 537 nouns, and 959 verbs) are sense-annotated in WebCAGe 3.0. That is, the average occurrences for an annotated lemma is 4. The explanation for why there are apparently few annotated occurrences per word lemma is due to the automatic creation approach. The WebCAGe harvesting method produces sense-annotated examples for a very large number of lemmas, but for only a few examples for each lemma.

For all word classes, the numbers of annotated word tokens from WebCAGe's four text types – i.e., Wiktionary example sentences, Wikipedia articles, Gutenberg texts, and external webpages – add up to the total numbers of tagged occurrences. By contrast, the numbers of annotated word lemmas do not add up to the total numbers, because several lemmas occur in more than one text category. The reason why the numbers of lemmas annotated in the Wiktionary example sentences equal the total numbers of lemmas is due to

WebCAGe's harvesting method. That is, only those lemmas for which there is an appropriate example sentence available in Wiktionary were considered.

6.3.2 WebCAGe Gold Standard for Supervised WSD

Although WebCAGe contains more than 10 000 sense annotations of more than 2 700 lemmas, only 1 761 annotated occurrences of 43 lemmas (3 adjectives, 33 nouns, and 7 verbs) qualify for evaluating supervised WSD. Table 6.3 shows the statistics of WebCAGe 3.0's remaining sense-annotated lemmas and tokens that fulfill the three criteria formulated in Section 6.1. This remaining set of WebCAGe annotations is later used as a gold standards for supervised WSD experiments in Chapter 8.

Table 6.3: WebCAGe 3.0 sense annotation subset for supervised WSD.

	Adj.	**Nouns**	**Verbs**	**All POS**
Total # of annotated word lemmas	3	33	7	43
Total # of tagged word tokens	128	1 471	162	1 761
- in training set	87	992	110	1 189
- in test set	41	479	52	572
Frequency range (occurrences/lemma)	17–83	15–184	16–45	15–184
Average frequency (occurrences/lemma)	43	45	23	41
Polysemy range in GermaNet (senses in GermaNet/lemma)	2–3	2–5	2–3	2–5
Average polysemy in GermaNet (senses in GermaNet/lemma)	2.3	2.6	2.1	2.5
Polysemy range of occurring words (occurring senses/lemma)	2–2	2–5	2–3	2–5
Average polysemy of occurring words (occurring senses/lemma)	2.0	2.2	2.1	2.2

With the help of Weka, all these remaining annotations are stratified into training and test sets – following a 2:1 ratio. That is, by definition, the numbers of annotated word tokens in the training and test sets add up to the total numbers of annotated word tokens. The specific numbers contained in the training and test sets are also stated in Table 6.3.

Table C.1 in Appendix C lists all lemmas contained in the supervised gold standard. It states for each lemma its overall number of annotations and its numbers of annotations in the training and test sets. The table provides details on the percentage distributions of all occurring senses. The percentage of the most frequently recorded sense is later used as a baseline for WSD. It is comparable to the most frequent sense baseline, i.e., when the WSD system always assigns the sense which has most occurrences in the annotations.

6.4 Sense-Annotated TüBa-D/Z Treebank

The TüBa-D/Z treebank [Telljohann et al., 2004, 2012] is a German newspaper corpus with high-quality annotations at various levels of language including parts of speech, morphology, and syntactic constituency (see Subsection 5.3.1 in the previous chapter). In the context of this dissertation, a selected set of lemmas is manually sense-annotated in the treebank – as described in Section 5.3. For the word sense disambiguation experiments in later chapters, the most recent version of the TüBa-D/Z (that is, release 9.1 as of December 2014) is used; and all sense annotations are updated to the most recent GermaNet release 9.0. Section 6.4.1 outlines these updates, Section 6.4.2 summarizes the statistics on the updated sense annotations, and Section 6.4.3 reports on the corresponding gold standard used for supervised word sense disambiguation experiments.

6.4.1 Updating to TüBa-D/Z 9.1

Updating the sense annotations in the TüBa-D/Z to the newest release of GermaNet differs from updating the annotations in WebCAGe in three main ways:

- It is not only the sense inventory which is available in a newer version, it is also the underlying textual resource which has been extended. In the most recent version of the treebank (that is, release 9.1 as of December 2014), all sense annotations are updated to the most recent GermaNet release 9.0.

- If the annotators found occurrences of word senses that were not covered by GermaNet, the lexicographic expert decided whether to add those senses to GermaNet (see Section 5.3). That is, the sense inventory is appropriately updated with missing word senses during the manual sense annotations in the TüBa-D/Z. The foundation for a potentially good fit with the inventory is provided. Furthermore, since the sense annotation in the TüBa-D/Z has been performed recently, there are only a limited number of lemmas for which the set of senses in GermaNet has fundamentally changed after this annotation.

- The TüBa-D/Z sense annotation is initially performed on release 8.0 of the treebank – as presented in Section 5.3. From the beginning, all prepared textual materials that were supposed to be included in TüBa-D/Z releases 9.0 and 9.1 were manually annotated with word sense information. Only when the treebank was ready to be released could the sense annotation be finished, because only then all linguistic information was available to identify missing or superfluous annotations. That is, the prepared textual materials for inclusion in the new treebank release did not contain reliable lemma and part-of-speech information, which is required to reliably identify all occurrences of a lemma with the demanded word class.

In conclusion, due to previous work during the initial annotations, the upgrade of the sense annotation in the TüBa-D/Z mainly affects annotations which occur in the new textual materials. Therefore, the new TüBa-D/Z release contains more annotations than the version described in Section 5.3. Since the update of the sense inventory has been an ongoing process until recently, there are not many annotated lemmas for which the set of senses in GermaNet has changed, and, thus, only minor work is needed to update already existing annotations.

6.4.2 TüBa-D/Z 9.1 Overall Statistics

Table 6.4 overviews the statistics on sense-annotated lemmas and tokens in the TüBa-D/Z 9.1. This treebank release contains about 10 000 additional sentences, which results in about 2 400 additional sense-annotated tokens compared to its version 8.0 described in Section 5.3.

Table 6.4: Overall sense annotation statistics of TüBa-D/Z 9.1.

	Nouns	**Verbs**	**All POS**
Total # of annotated word lemmas	30	79	109
Total # of tagged word tokens	8 803	9 107	17 910
Frequency range (occurrences/lemma)	24–1 699	21–801	21–1 699
Average frequency (occurrences/lemma)	293	115	164
Polysemy range (senses/lemma)	2–7	2–14	2–14
Average polysemy (senses/lemma)	4.1	2.8	3.2

The sense annotation in the TüBa-D/Z comprises a very large number of occurrence for a small set of lemmas. Altogether, there are 17 910 occurrences of 109 lemmas (30 nouns and 79 verbs) sense-annotated in TüBa-D/Z 9.1. The average frequency for an annotated lemma is 164. This high frequency constitutes the major advantage of the sense-annotated TüBa-D/Z compared to the other available corpora.

The 30 nouns occur 8 803 times in the treebank – at least 24 times and at most 1 699 times. On average, there are 293 occurrences per noun lemma. The average polysemy (number of senses in GermaNet) is 4.1 for the annotated nouns, ranging from 2–7 senses.

For verbs, 9 107 verb occurrences are annotated with the senses of 79 verb lemmas. The average occurrence per verb lemma is 115 with the least frequent verb occurring 21 times, the most frequent one 801 times. The average polysemy is 2.8, with the most polysemous verb showing 14 senses in GermaNet.

Appendix C documents for each of the 109 lemma details such as the number of sense-annotated occurrences, the number of senses in GermaNet, and the number of senses occurring at least once in the treebank – see Table C.2.

6.4.3 TüBa-D/Z Gold Standard for Supervised WSD

Table 6.5 shows that 16 738 – from the total of 17 910 – sense-annotated tokens fulfill the criteria formulated in Section 6.1 and are thus included in the TüBa-D/Z gold standard used for evaluating supervised WSD systems. These annotations belong to 92 lemmas (24 nouns and 68 verbs).

Table 6.5: TüBa-D/Z 9.1 sense annotation subset for supervised WSD.

	Nouns	**Verbs**	**All POS**
Total # of annotated word lemmas	24	68	92
Total # of tagged word tokens	8 198	8 540	16 738
- in training set	5 474	5 711	11 185
- in test set	2 724	2 829	5 553
Frequency range (occurrences/lemma)	22–1 699	24–799	22–1 699
Average frequency (occurrences/lemma)	342	126	182
Polysemy range in GermaNet (senses in GermaNet/lemma)	3–7	2–14	2–14
Average polysemy in GermaNet (senses in GermaNet/lemma)	4.3	2.9	3.3
Polysemy range of occurring words (occurring senses/lemma)	2–7	2–9	2–9
Average polysemy of occurring words (occurring senses/lemma)	3.4	2.6	2.8

Weka's implementation for stratification is used to divide all annotations into 66% training and 33% test set. The overall counts contained in the training and test sets are stated in Table 6.5. Lemma-specific numbers of annotations available in the training and test sets are given in Table C.3 (Appendix C). The table in the appendix also provides details on the percentage distributions of all occurring senses. The percentage of the most frequently recorded sense is used in the WSD experiments as the most frequent sense baseline.

6.5 Sense-Annotated deWaC

The more gold standard annotations available, the more reliable and meaningful the evaluation of word sense disambiguation experiments – especially for languages such as German where the availability of sense-annotated materials is restricted. For this reason, the sense annotations in the deWaC corpus are employed in addition to the other two corpora.

This gold standard consists of a subset of sentences extracted from the deWaC corpus which are manually annotated with senses from GermaNet. The deWaC corpus is the German part of the WaCky corpora [Baroni et al., 2009]. It is constructed by harvesting from the web with the goal of creating a corpus of diverse contents and genres. Therefore, a variety of websites with the German domain ending *.de* are crawled for basic German vocabulary. Altogether, deWaC consists of 1.7 billion words. The TreeTagger [Schmid, 1994] was applied to enrich tokens with their lemmas and part-of-speech tags.

In the context of their WSD study, Broscheit et al. [2010] selected a lexical sample of 40 ambiguous word lemmas (including 6 adjectives, 18 nouns, and 16 verbs) by translating words taken from the English SensEval-2 test set data [Palmer et al., 2001]. They manually annotated a total of 1 154 occurrences in the deWaC corpus with senses from GermaNet 5.1 – at least 20 occurrences for each lemma in the lexical sample. These sense-annotations in deWaC have been made publicly accessible[1] only during the compilation of the two sense-annotated corpora described in Chapter 5.

6.5.1 Reasons for Choosing deWaC

Besides the main motivation that more gold standard annotations support the meaningfulness of WSD experiments, there are three additional reasons why the sense-annotated deWaC also serves as a gold standard in the WSD experiments in Chapters 7 and 8: firstly, to allow a comparison and to study the influence of three different gold standards on word sense disambiguation experiments. Secondly, the deWaC sense-annotations are now freely available.

[1] Thanks to Anette Frank and Simone Paolo Ponzetto for making available their deWaC sense annotations – `http://projects.cl.uni-heidelberg.de/dewsd/`.

Thirdly, the corpus is supposed to be domain-independent.

The only other existing and available GermaNet-annotated corpus would be the medical corpus obtained from scientific abstracts from the Springer Link website. It was sense-annotated by Raileanu et al. [2002] in the context of the MuchMore project[1]. The preference of deWaC over this medical Springer corpus is mainly due to the GermaNet versions used for sense annotation. Since the medical Springer corpus was created before 2002, it uses a rather old version of GermaNet. Although their paper [Raileanu et al., 2002] does not state a specific version of GermaNet, they report the size of GermaNet as 16 000 words. Considering that GermaNet's release 9.0 contains more than 110 000 lemmas, i.e., about seven times as many, their version would be difficult to map onto the current GermaNet version and every lemma would probably need to be re-annotated. The GermaNet version used for annotating deWaC is clearly newer and, thus, obviously easier to map to the current release of GermaNet.

6.5.2 Reuse of an Existing Sense-Annotated Corpus

In a similar way to updating WebCAGe's sense annotation inventory, adjusting the sense annotations in deWaC to GermaNet 9.0 mainly affects annotations where the set of senses for a specific lemma has changed from GermaNet release 5.1 to release 9.0. Unlike WebCAGe's initial sense annotations, the sense annotations in deWaC do not yet possess persistent database identifiers that reliably identify word senses in GermaNet across releases. These identifiers are available only since release 5.2 (see Appendix B). Thus, it is more difficult to reliably identify or match all senses in question automatically. Finally, all annotations are revisited and, if necessary, re-annotated with the appropriate senses from GermaNet 9.0.

The revised deWaC sense annotations have slightly decreased in number compared to the original version. This is mainly due to leaving out three lemmas with all their annotations. One of these lemmas is left out because it has only one sense in GermaNet. Another lemma, an adjective, is taken

[1] `http://muchmore.dfki.de/`

out because most of its occurrences represented non-inflectible tokens such as particles or adverbs rather than adjectival occurrences. For the third left-out lemma, the set of senses had been completely revised in GermaNet and it was not possible to easily map the senses from the different releases onto each other. Furthermore, a few annotations are removed if their lemma or word class in the specific context does not correspond to a GermaNet entry, because the task of recognizing lemma and POS is not tackled in the scope of this dissertation.

The contexts of the original deWaC sense annotations provided online[1] by Broscheit et al. [2010] are restricted to sentential phrases only. Since such restricted contexts pose a specific challenge to WSD, which is not the desired research goal to tackle in this thesis, the contexts of the revised version of the deWaC sense annotations are enlarged. These larger contexts are extracted from the proper deWaC corpus, which is also freely available online[2].

6.5.3 deWaC Overall Statistics

The GermaNet 9.0 sense annotations in deWaC, which are used for WSD experiments, comprise 1 083 occurrences of 37 lemmas: 90 annotations for the 4 adjectives, 385 annotations for the 18 nouns, and 608 annotated tokens for the 15 verbs – see Table 6.6. This implies an average of 29 annotations per lemma.

Details for each of the 37 lemmas – including the number of sense-annotated tokens, the number of senses in GermaNet, and the number of senses with at least one annotation – are listed in Table C.4 in Appendix C.

6.5.4 deWaC Gold Standard for Supervised WSD

From the deWaC sense annotations, a gold standard for evaluating supervised WSD systems is formed analogously to WebCAGe and TüBa-D/Z – see Sections 6.3.2 and 6.4.3, respectively. Table 6.7 shows the subset of annotations

[1] See Footnote 1 on page 160 for the link to the original deWaC sense annotations.

[2] *The Web-As-Corpus Kool Yinitiative* (short: *WaCky*) project group [Baroni et al., 2009] provides large web corpora for several languages, including the German deWaC corpus, at `http://wacky.sslmit.unibo.it`.

Table 6.6: Overall sense annotation statistics of deWaC.

	Adj.	**Nouns**	**Verbs**	**All POS**
Total # of annotated word lemmas	4	18	15	37
Total # of tagged word tokens	90	385	608	1 083
Frequency range (occurrences/lemma)	20–30	20–30	18–127	18–127
Average frequency (occurrences/lemma)	23	21	41	29
Polysemy range (senses/lemma)	2–9	2–11	3–26	2–26
Average polysemy (senses/lemma)	5.5	3.9	7.9	5.7

included in this 'supervised' gold standard. Altogether, there are 903 annotations, belonging to 31 lemmas (4 adjectives, 15 nouns, and 12 verbs), which fulfill the criteria formulated in Section 6.1 and are thus employed for evaluating supervised WSD experiments. With Weka, the annotations per lemma are stratified into two-thirds training data and one-third test data. The table also lists the total counts of annotated tokens contained in the training and test sets.

Lemma-specific counts of annotated tokens included in the training and test sets are given in Appendix C, Table C.5. The table in the appendix also provides details on the percentage distributions of all occurring senses. The percentage of the most frequently recorded sense is used in the WSD experiments as the most frequent sense baseline.

6.6 Automatic Linguistic Preprocessing

For both knowledge-based and supervised word sense disambiguation experiments – described in Chapters 7 and 8, respectively – all corpora are automatically preprocessed.[1] In order to identify and preprocess the context windows

[1]Technically, the automatic linguistic processing takes place during the WSD experiments (for the knowledge-based WSD experiments) and when the training files get created

Table 6.7: deWaC sense annotation subset for supervised WSD.

	Adj.	**Nouns**	**Verbs**	**All POS**
Total # of annotated word lemmas	4	15	12	31
Total # of tagged word tokens	82	315	506	903
- in training set	57	217	342	616
- in test set	25	98	164	287
Frequency range (occurrences/lemma)	19–23	18–28	18–115	18–115
Average frequency (occurrences/lemma)	21	21	42	29
Polysemy range in GermaNet (senses in GermaNet/lemma)	2–9	2–11	3–26	2–26
Average polysemy in GermaNet (senses in GermaNet/lemma)	5.5	3.9	8.9	6.0
Polysemy range of occurring words (occurring senses/lemma)	2–3	2–4	2–11	2–11
Average polysemy of occurring words (occurring senses/lemma)	2.5	2.3	3.9	3.0

used for disambiguating word senses (see Sections 7.2.1 and 8.1), the three gold standard corpora described in this chapter, i.e., deWaC (Section 6.5), TüBa-D/Z (Section 6.4), and WebCAGe (Section 6.3), are automatically split into sentences, tokenized, and lemmatized – as depicted in the following list:

Sentence segmentation With the help of the sentence detector of the Apache OpenNLP tool suite[1], all texts are split into individual sentences. The sentence detector implements a maximum entropy model to evaluate whether or not the punctuation characters '.', '!' and '?' in a given text signify the end of a sentence. The German model provided[2] is trained on the Tiger treebank [Brants et al., 2004].

Tokenization The tokenizer available on the *TreeTagger for Java* website[3]

(for the supervised experiments). Furthermore, only those sentences and tokens for which automatic linguistic annotation is required during the WSD experiments are actually processed. Thus, the three corpora are not (made) available in fully automatically preprocessed versions.

[1] `http://opennlp.apache.org/`

[2] Available at `http://opennlp.sourceforge.net/models-1.5/`.

[3] The tokenizer was proposed by Richard Eckart de Castilho and is available at `http:`

is used for tokenization. It uses Java's built-in *BreakIterator* – a very simple yet functional tokenizer which even separates characters that are not part of a word, such as symbols or punctuation marks.

Lemmatization TreeTagger [Schmid, 1994] is a part-of-speech tagger which is also able to lemmatize tokens. Since for lemmatization with TreeTagger a lexicon lookup is performed, the availability of a lexicon is required. The official parameter file for German provided on the TreeTagger website, includes such a lexicon. Further details about the TreeTagger, its probailistic tagging method, and a link to the available parameter file are provided in Subsection 8.1.1.

In order to allow a direct comparison of the WSD results for all three corpora with the same given quality of preprocessing, this automatic preprocessing is also performed on TüBa-D/Z's texts, although the treebank already contains manual sentence segmentation, tokens, and lemmas (see Section 8.1 for further explanation).

In addition to these annotations, several supervised machine learning features require part-of-speech tags, morphological information, syntactical structures, and translations to English. These automatic annotations are explained in the section on machine learning features (Subsection 8.1.1).

//code.google.com/p/tt4j/wiki/SimpleTokenizer.

Part III

Word Sense Disambiguation (WSD)

Chapter 7

Knowledge-Based Word Sense Disambiguation

Word sense disambiguation (WSD) has been a very active area of research in computational linguistics (see Chapter 2). It is the task of computationally assigning the most appropriate senses to words occurring in a text [Navigli, 2009]. In stand-alone WSD systems, these senses are taken from a predefined sense inventory such as a wordnet. Most of the work on WSD has focused on English. One of the factors that has hampered WSD research for other languages has been the lack of appropriate resources, particularly in the form of sense-annotated corpus data. Due to the lack of sense-annotated corpora for German prior to the construction of the sense-annotated corpora described in Chapters 5 and 6, there has been relatively little research on WSD for this language.[1] The purpose of this chapter and the following chapter is to help close this gap. They focus on WSD for German, utilize GermaNet as the sense inventory, and make use of the three sense-annotated corpora recently made available for this language: WebCAGe, TüBa-D/Z, and deWaC.

There are a variety of techniques for trying to solve the task of WSD – including knowledge-based, supervised, and unsupervised approaches. While the next chapter investigates WSD using supervised machine learning methods, this chapter focuses on knowledge-based WSD. In general, knowledge-based

[1] For more discussion, see Chapter 2.

techniques make use solely of available lexical resources including dictionaries or wordnets. They use clues such as selectional restrictions or similarity between two words in a knowledge base in order to tackle the WSD task.[1]

The present chapter explores a wide range of knowledge-based word sense disambiguation algorithms for German. These WSD algorithms are based on a suite of semantic relatedness measures, including path-based, information-content-based, and gloss-based methods. The set of employed semantic relatedness measures is described in Section 7.1 below.

The word sense disambiguation of a polysemous target word in a given sentence starts with a calculation of semantic relatedness for all sense combinations in question around the ambiguous target word to be disambiguated, as illustrated in Figure 7.1. That is, a relatedness measure $rel(ts, cs)$ is calculated for each sense of the target word to each sense of each word in the context. The computed relatedness values are illustrated in the table in Figure 7.1, where $rel(ts, cs)$ can be one of the employed semantic relatedness measures.

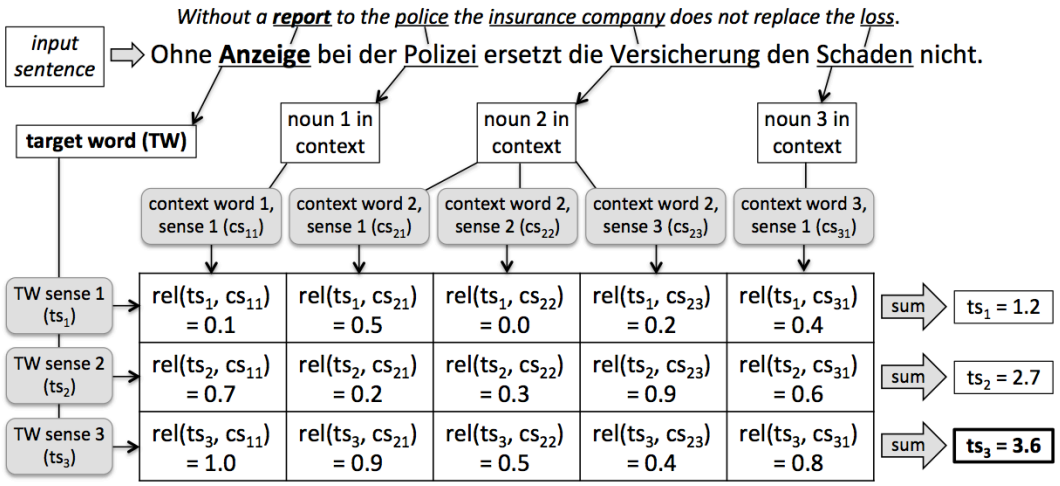

Figure 7.1: Knowlege-based algorithm for disambiguating a target word.

In a next step, all calculated scores per target word are summed and the target word sense (or senses) yielding the highest sum is returned.

Experiments on gloss-based relatedness methods also employ the newly harvested definitions from Wiktionary, which are linked to corresponding Germa-

[1]See Subsection 2.3.1 for an overview of knowledge-based WSD algorithms and related works.

Net senses via the automatic mapping between GermaNet and Wiktionary described in Chapter 4. Furthermore, since the individual relatedness algorithms produce diverse results in terms of precision that complement each other well in terms of coverage, combined algorithms are investigated and compared in performance to the individual algorithms.

In short, this chapter has the following four main goals:

(i) To apply a wide range of knowledge-based WSD algorithms to German – including the three word classes of adjectives, nouns, and verbs – since the range of methods that has thus far been applied to this language is rather limited (as described in Chapter 2).

(ii) To study the combination of knowledge-based WSD methods in a *majority voting* scheme and in a *Borda count* setup.

(iii) To study the effect of linking GermaNet with Wiktionary (Chapter 4) on WSD, which is relevant for gloss-based methods.

(iv) To evaluate and compare the performance of knowledge-based WSD methods on three heterogeneous sense-annotated corpora.

This chapter is based on an earlier published paper [Henrich and Hinrichs, 2012]. The underlying idea to use semantic relatedness measures for WSD is the same. However, the WSD experiments described in the paper use old versions of GermaNet (release 6.0) and WebCAGe (unpublished pre-version), whereas this chapter reports on experiments with the most recent versions of GermaNet (i.e., release 9.0) and WebCAGe (version 3.0). This chapter presents WSD results for two additional sense-annotated corpora, i.e., TüBa-D/Z and deWaC, which were not available in the context of Henrich and Hinrichs [2012]. Moreover, it includes the evaluation of adjectives and verbs while the paper focused solely on nouns.

The remainder of this chapter is structured as follows: Section 7.1 introduces the terminology and the measures for computing semantic relatedness. Section 7.2 describes how these measures are applied to disambiguate word senses. The results of the knowledge-based WSD experiments are presented

and discussed in Section 7.3 – separately for the three word classes of adjectives, nouns, and verbs. Finally, Section 7.4 summarizes all findings from this chapter and concludes.

7.1 Semantic Relatedness Measures

In order to be able to apply a wide range of knowledge-based WSD algorithms to German, the same suite of semantic relatedness algorithms that was previously used by Pedersen et al. [2005][1] for English WSD is reimplemented for German[2]. The basic ideas of these algorithms are summarized in the following subsections.[3] Following Pedersen et al.'s [2005] terminology, the algorithms are grouped into path-based (Subsection 7.1.2), information-content-based (Subsection 7.1.3), and gloss-based (Subsection 7.1.4). The measures in these groups are different from each other with regard to the type of information they use for computing semantic relatedness: the path-based measures rely on GermaNet's graph structure to compute the shortest path between two concepts; the information-content-based measures rely on concept probabilities, i.e., relative word frequencies in a large corpus; and the gloss-based measures make use of sense definitions to count word overlaps. Due to the different kinds of information employed, the measures have different strengths and weaknesses, which are discussed in Section 7.3 where the different types of measures are evaluated and compared in a knowledge-based word sense disambiguation setup.

[1] Ted Pedersen, Siddharth Patwardhan, Jason Michelizzi, and Satanjeev Banerjee developed the WordNet::Similarity package in Perl [Pedersen et al., 2004], which implements several measures to compute similarity between word senses in the Princeton WordNet. It is available at `http://wn-similarity.sourceforge.net/`.

[2] This suite of semantic relatedness algorithms was reimplemented for GermaNet by Anne Brock under the supervision of this thesis' author. It is made freely available online – see `http://www.sfs.uni-tuebingen.de/GermaNet/tools.shtml#SemRelAPI`.

[3] Also see the original papers cited in each subsection for detailed descriptions and Budanitsky and Hirst [2006] for an overview.

7.1.1 Terminology

Semantic relatedness and semantic similarity are two similar yet distinct concepts. In the literature [Resnik, 1995; Patwardhan et al., 2003; Pedersen et al., 2005; Budanitsky and Hirst, 2006; Zesch, 2010a], *semantic similarity* between two words is understood as the resemblance of their meanings, i.e., the likeness of two semantic concepts. *Semantic relatedness* between two concepts is often defined as their direct or indirect relationship in a taxonomy such as a wordnet. In this view, semantic relatedness is a more general term than semantic similarity, because two concepts can be related in a wordnet but yet not semantically similar.[1] For example, 'hard' and 'soft' are semantically related because a wordnet would link them with an antonomy relation, but antonyms are, by definition, semantically opposites, so the two concepts are not semantically similar. [Resnik, 1995; Patwardhan et al., 2003; Pedersen et al., 2005; Budanitsky and Hirst, 2006; Zesch, 2010a]

In the following, some definitions are made to understand the descriptions of the relatedness measures in the subsections below. In general, the measures $rel_{name}(s_1, s_2)$ compute semantic relatedness between two semantic concepts (i.e., synsets) s_1 and s_2 in GermaNet. The shortest path from synset s_1 to synset s_2 in terms of the hypernymy/hyponymy relations in the GermaNet graph is stated as $pathlength(s_1, s_2)$. As a special case, $depth(s)$ refers to the shortest path from synset s to GermaNet's root node. GermaNet's artificial root node (also see Section 3.4) is abbreviated as $GNROOT$.

The term *least common subsumer*[2] of two synsets s_1 and s_2 – expressed as $LCS(s_1, s_2)$ – is defined as the most specific (direct or indirect) hypernym that the two synsets have in common, i.e., the shared hypernym h with the largest $depth(h)$, which means the longest possible path to GermaNet's root node.

In general, the hypernymy/hyponymy relation is defined to connect semantic concepts of the same word class. Thus, all measures that rely on hypernymy/hyponymy relations to compute the shortest path or the least common subsumer of two concepts in the GermaNet graph are limited to comparing

[1] In the context of this chapter, the more general concept of semantic relatedness is usually meant, if not otherwise stated.

[2] Sometimes also *lowest common subsumer*.

synsets of the same word class.

7.1.2 Path-Based Measures

All path-based measures use the GermaNet graph structure to compute the shortest path between two concepts contained in the graph. It has to be noted, however, that even paths with the same length can be significantly different from each other. For example, paths between two specific concepts are different from paths between two general concepts. That is, two directly related, specific concepts are often very similar (e.g., 'dog' and 'poodle'), whereas two directly related, general concepts are often more distant (e.g., 'physical entity' and 'physical object'). To take these differences into account, the shortest paths need to be *normalized*. The measures described in this subsection differ from each other in the way of how they normalize these shortest paths. The first three itemized measures (*path*, *lch*, and *wup*) normalize path lengths by incorporating information on the specificity of concepts, i.e., by incorporating information on which level in the GermaNet hierarchy the concepts are located, while the fourth method (*hso*) incorporates the number of 'direction changes' for normalization. [Pedersen et al., 2005]

path: This is the simplest method used to calculate semantic relatedness between two concepts s_1 and s_2. It computes relatedness as a function of the distance between two nodes (i.e., $pathlength(s_1, s_2)$) normalized by the longest possible 'shortest path' between any two nodes in GermaNet (i.e., $MAXSHORTESTPATH$).

$$rel_{path}(s_1, s_2) = \frac{MAXSHORTESTPATH - pathlength(s_1, s_2)}{MAXSHORTESTPATH} \quad (7.1)$$

where the constant $MAXSHORTESTPATH$ is the maximum path length of all shortest paths in GermaNet. For GermaNet 9.0, $MAXSHORTESTPATH$ is calculated as 35.

lch: This method also computes the shortest path between two nodes s_1 and s_2 by the function $pathlength(s_1, s_2)$ – similarly to the previous method

but with a different normalization. Leacock and Chodorow [1998] define similarity between two concepts as the negative logarithm of the length of the shortest path between the two concepts (limited to hypernymy/hyponymy relations only) over twice the path length of the overall depth of the wordnet:[1]

$$rel_{lch}(s_1, s_2) = -\log \frac{pathlength(s_1, s_2)}{2 \times MAXDEPTH} \tag{7.2}$$

where *MAXDEPTH* is the maximal distance of all synsets from GN-ROOT. For GermaNet 9.0, *MAXDEPTH* is 20.

wup: Again, conceptual relatedness between two concepts s_1 and s_2 is computed as the shortest path length (limited to hypernymy/hyponymy relations) between the two concepts, i.e., $pathlength(s_1, s_2)$. Wu and Palmer [1994] normalize the path length by the depth of the two synsets' least common subsumer, i.e., $depth(LCS(s_1, s_2))$.

$$rel_{wup}(s_1, s_2) = \frac{2 \times depth(LCS(s_1, s_2))}{pathlength(s_1, s_2) + 2 \times depth(LCS(s_1, s_2))} \tag{7.3}$$

hso: Since the path-based relatedness measure introduced by Hirst and St-Onge [1998] is not limited to the hypernymy/hyponymy relations, it can be computed for concepts of different word classes. Hirst and St-Onge [1998] categorize relations in a wordnet as being 'upwards' (e.g., hypernymy, holonymy, is-entailed-by), 'downwards' (such as hyponymy, meronymy, entails, causes), and 'horizontal' (including is-related-to, synonymy, antonymy, has-participle, pertainymy), and refer to a change of 'direction' whenever the path between two concepts consists of two relations from distinct groups.

Hirst and St-Onge [1998] define 'strong relations' as identical concepts, concepts with a direct horizontal link, or when one word lemma of one synset is contained in one of the other synset's lemmas (presumably

[1] In the implementation, 1 is added to numerator and denominator to avoid an infinite result for the logarithm of zero, which would otherwise occur for identical concepts.

a compound). 'Medium-strong relations' refer to paths between two concepts that (i) have a maximum length of five when considering all types of relations and (ii) confirm specified patterns such as $u+$, $u+d+$, $u+h+$, $u+h+d+$, $d+$, $d+h+$, etc., where 'u' refers to upward relations, 'd' to downward relations, and 'h' to horizontal relations.[1]

In the current implementation, the relatedness value for strong relations is set to 15. Semantic relatedness between two medium-strong related concepts considers the length of the shortest path between the concepts (i.e., $pathlength(s_1, s_2)$, as before) and the number of 'direction' changes:

$$rel_{hso}(s_1, s_2) = C - pathlength(s_2, s_2) - k \times d \tag{7.4}$$

where C is the maximum value for these medium-strong relations, k the factor to scale the number of direction changes, and d the number of changes of direction in the shortest path between the two synsets. The results reported in the experiments in Section 7.3 use the default values of 10 and 1 for parameters C and k, respectively.[2]

Although *path*, *lch*, and *wup* allow arbitrarily long paths between concepts, the paths' computation considers only hypernymy/hyponymy relations. Since this type of relation always connects concepts of the same word class, these measures are restricted to synsets of the same word class. The shortest path computation in *hso* is not limited to the hypernymy/hyponymy relation. Thus, *hso* is applicable to concepts of different word classes. However, since the *hso* measure prohibits long paths between two synsets and many changes in direction, the measure is restricted to the set of synset combinations that are 'close' enough in the GermaNet graph.

7.1.3 Information-Content-Based Measures

The measures in this subsection are different from the previously described measures with regard to the type of information they use for computing se-

[1] See the paper by Hirst and St-Onge [1998] for more information.

[2] Note that both the value for strong relations and the maximum value for medium-strong relations differ from those used in the implementation by Pedersen et al. [2005] for English.

mantic relatedness. That is, while the path-based measures use GermaNet's graph structure to compute the shortest path between two concepts, the measures described in this subsection rely on concept probabilities, i.e., relative frequencies of words in a large corpus.

More specifically, the three measures in this subsection, i.e., *res*, *jcn*, and *lin*, rely on the information content (IC) of a concept in the GermaNet graph. The *information content* graduates semantic concepts from general to specific. The more specific a concept, the smaller its probability and, thus, the higher its informativeness. Conversely, the more generic or abstract a concept, the higher its probability, but the lower its IC. The IC of a semantic concept is determined by the negative logarithm of the concept's probability. [Resnik, 1995]

In the present implementation, the information content of a semantic concept is estimated by the relative frequency of the word in a large corpus, namely the *Tübingen Partially Parsed Corpus of Written German* (TüPP-D/Z) [Müller, 2004; Ule, 2004]. The TüPP-D/Z is a German newspaper corpus of 200 million words, freely available for academic use.[1]

Since there are no large German corpora available with all words sense-annotated, the frequency of a word is assigned to all synsets containing that word[2] and cumulatively to all direct and indirect hypernyms of these synsets up to GNROOT. This cumulative counting means that occurrences of more specific concepts are also added to the frequencies of more general concepts. More formally, the information content $IC(s)$ of a semantic concept s is summarized as

$$IC(s) = -\log \frac{\sum_{w \in W(s)} cumFreq(w)}{cumFreq(GNROOT)} \tag{7.5}$$

where $cumFreq(s)$ is the cumulated frequency of synset s (i.e., the frequency of s and *cumFreq*s of all hyponyms) and $W(s)$ is the set of words w in synset s.

Furthermore, all three measures employ the least common subsumer (LCS)

[1] `http://www.sfs.uni-tuebingen.de/en/ascl/resources/corpora/tuepp-dz.html`

[2] Note that this assigment of a word's frequency to all corresponding synsets follows the approach by Patwardhan et al. [2003], whereas Resnik [1995] proposes to divide the frequency by the number of synsets.

of two synsets. Since the LCS is determined by the hypernymy/hyponymy relation, which always connects concepts of the same word class, these measures are limited to comparing synsets of the same word class.

res: Resnik's [1995] measure to compute semantic similarity is the simplest IC-based measure. It assumes (i) that two concepts are more related the more information they share and (ii) that the shared information of two concepts can be quantified by the information content of two concepts' lowest common subsumer. Thus, this measure defines semantic relatedness between two concepts s_1 and s_2 to be the information content of the two concepts' least common subsumer:[1]

$$rel_{res}(s_1, s_2) = IC(LCS(s_1, s_2)) \tag{7.6}$$

More specifically, Resnik [1995] measures semantic similarity of two concepts as the negative logarithm of the probability of their least common subsumer in the graph. To use a terminology consistent with the other measures, Resnik's [1995] measure is labeled as rel_{res}, although, strictly speaking, he defines a measure to compute semantic similarity rather than relatedness.

jcn: The idea of Jiang and Conrath's [1997] measure extends Resnik's [1995] measure in that it associates the information content of two concepts' least common subsumer with the individual information contents of the two concepts. The assumption behind this measure is that two concepts are more related the smaller the difference between the IC of two concepts' LCS and the sum of the two concepts' ICs.

Jiang and Conrath [1997] define semantic distance of two concepts as the sum of the ICs of the concepts minus twice the IC of their LCS:

$$dist_{jcn}(s_1, s_2) = IC(s_1) + IC(s_2) - 2 \times IC(LCS(s_1, s_2)) \tag{7.7}$$

[1] If more than one LCS for two concepts exist, the 'most informative' one is taken, i.e., the more specific concept.

Conceptually, semantic distance in a graph is the inverse of semantic relatedness. In the implementation, the distance measure is turned into a similarity measure by subtracting it from the maximum possible 'distance' of any two concepts in GermaNet:[1]

$$rel_{jcn}(s_1, s_2) = JCNMAXDIST - dist_{jcn}(s_1, s_2) \tag{7.8}$$

where the maximum possible 'distance' in terms of the *jcn* measure is defined as

$$\begin{aligned} JCNMAXDIST &= \max(dist_{jcn}(s_1, s_2)) \\ &= \max(IC(s_1) + IC(s_2) - 2 \times IC(LCS(s_1, s_2))) \\ &= MAX\_IC + MAX\_IC - 2 \times IC(GNROOT) \\ &= MAX\_IC + MAX\_IC - 2 \times 0 \\ &= 2 \times MAX\_IC \\ &= 2 \times -\log \frac{1}{frequency\ of\ GNROOT} \end{aligned} \tag{7.9}$$

That is, *JCNMAXDIST* is twice the IC of two leaf nodes with the highest IC (MAX_IC), whose LCS is GNROOT[2]. Assuming that a leaf has the assigned default minimal frequency of 1, $MAX\_IC = -\log \frac{1}{frequency\ of\ GNROOT}$. *JCNMAXDIST* is constant for a specific GermaNet release and lemma frequency list. For the current implementation with GermaNet release 9.0 using frequencies from TüPP-D/Z, the value for *JCNMAXDIST* is approximately 37.65.

lin: Lin's [1998] measure is similar to the measure by Jiang and Conrath [1997] in that it considers the same information contents, i.e., the individual information contents of two concepts and the information content of the two concept's lowest common subsumer. However, the components interact differently. To measure similarity between two concepts s_1 and s_2,

[1] Note that this way of turning the distance measure into a relatedness measure differs from the implementation by Pedersen et al. [2004], who calculate relatedness as the inverse of the distance, i.e., $\frac{1}{dist_{jcn}(s_1, s_2)}$.

[2] The IC of GNROOT is 0.

Lin [1998] computes the IC (multiplied by 2) of the two concepts' LCS over the sum of the ICs of the two concepts:

$$rel_{lin}(s_1, s_2) = \frac{2 \times IC(LCS(s_1, s_2))}{IC(s_1) + IC(s_2)} \tag{7.10}$$

7.1.4 Gloss-Based Measures

The measures in this subsection are again different from the previously described measures with regard to the type of information they use for computing semantic relatedness. That is, the measures in this subsection use neither the GermaNet graph to calculate paths or least common subsumers nor word frequencies to calculate information contents for words. The measures in this subsection rather make use of sense definitions to count word overlaps.

Lesk [1986] introduces a word sense disambiguation algorithm assuming that words occurring together in a text tend to share a common topic. The algorithm counts the word overlaps between the definitions of an ambiguous target word and the definitions of the words in its context. Banerjee and Pedersen [2003] apply Lesk's algorithm to the task of computing semantic relatedness between synsets of WordNet. They extend the original measure by including definitions of related synsets. The underlying idea is that the more related two synsets are, the more overlaps their definitions and the definitions of corresponding related words have.[1]

The measures in this subsection calculate semantic relatedness between two concepts s_1 and s_2 by counting words shared by the 'bags of words' (BOW) of two concepts:

$$rel_{lesk}(s_1, s_2) = BOW(s_1) \cap BOW(s_2) \tag{7.11}$$

where $BOW(s)$ is a set of distinct words harvested for synset s. Word overlap methods can generally compare two words of any word class. Several options allow specifying which words to include in these bag of words representations of synsets. Since the coverage of definitions for GermaNet synsets is limited, the newly harvested definitions from Wiktionary, which are linked to

[1] See Subsection 2.3.1 for more details on the original Lesk [1986] algorithm and Banerjee and Pedersen's [2003] adapted variant.

corresponding GermaNet senses via the automatic mapping between GermaNet and Wiktionary described in Chapter 4, are also employed. Furthermore, lexical fields are examined for calculating word overlaps. Lexical fields substitute sense descriptions by encapsulating related words – as introduced in Section 4.2. Using the mapping from GermaNet to Wiktionary, such lexical fields can also be obtained for Wiktionary.

The choice of whether to include definitions[1] (see Section 3.2) or lexical fields (see Section 4.2) from Wiktionary or GermaNet results in many different bags of words. Since it would be too much to report experiments for all possible variants[2], the following list itemizes the five versions for which results are reported and discussed in Section 7.3. The main reasons for including these five variants is to be able (i) to compare the performance of glosses as opposed to lexical fields, (ii) to study the impact of bags of words constructed solely from GermaNet with those constructed solely from Wiktionary, and (iii) to study the general impact of the alignment between GermaNet and Wiktionary (i.e., all improvements over *lesk-GermaNet*).

lesk-GermaNet: All available information from GermaNet is included into the bags of words, i.e., both glosses and lexical fields from GermaNet.

lesk-Wiktionary: All available information from Wiktionary is used for counting word overlaps, i.e., both glosses and lexical fields from Wiktionary.

lesk-glosses: All glosses – both from GermaNet and Wiktionary – are included into the bags of words.

lesk-lex-fields: All lexical fields – both from GermaNet and from Wiktionary – are used for constructing the bags of words.

[1] As stated in Footnote 2 on page 58, the terms *definition*, *gloss*, and *sense description* are used interchangeably throughout this thesis.

[2] Further possible variants include any other combination of glosses/lexical fields and GermaNet/Wiktionary that is not itemized in the list. For example, variants where (i) only glosses from GermaNet, (ii) only glosses from Wiktionary, (iii) only lexical fields from GermaNet, (iv) only lexical fields from Wiktionary, (v) lexical fields from GermaNet and glosses from Wiktionary, (vi) lexical fields from Wiktionary and glosses from GermaNet, etc., are used.

lesk-all-info: All available information is used, i.e, glosses and lexical fields both from GermaNet and Wiktionary.

Besides the general choice of whether definitions or lexical fields from Wiktionary or GermaNet are included in the bags of words, there are many more parameters for the gloss-based methods. The results reported in the WSD experiments below use the following default values, which turned out to be generally (i.e., across word classes and corpora) the most reliable:

All POS The experiments described below include words of all word classes in the bags of words representations.

All words and orthographic forms from synset The bags of words include all words from a synset, i.e., all orthographic forms of all synonymous lexical units.

Compound splitting Nominal compounds are split into their constituent parts before being included in the bag of words. It turned out that information on compound constituents available in GermaNet is more reliable than external compound splitters. The reason might be that the compounds in GermaNet are manually post-corrected automatic splits – see Henrich and Hinrichs [2011] and Subsection 3.6.

Stemming and case folding To generally increase the word overlap counts, all words in the bags of words are stemmed (using the Snowball stemmer [Porter, 1980]) and converted to lower case.

Hypernyms only As with the word overlap experiments in Chapter 4, different types of related words can be included into the lexical fields. The experiments reported in the present chapter consider hypernyms only – disregarding all other relation types.

Maximum path length Hypernyms and hypernyms of hypernyms up to a maximum distance of 4 are added to the bags of words for a synset.

Minimum specificity Even hypernyms within this maximum distance of 4 are included only if they have a minimum distance of 2 to GermaNet's

root node. That is, synsets from the first two upper levels of the GermaNet hierarchy are excluded.

7.2 Semantic Relatedness for WSD

The knowledge-based word sense disambiguation of a polysemous target word in a given sentence starts with the extraction of the appropriate context words for each target word. This context window is explained in Subsection 7.2.1 below.

In a next step, for each above-described relatedness measure taken in isolation, semantic relatedness is calculated between all senses of the target word with all senses of all words in this context window, as illustrated in Figure 7.1 (on page 170) above. That is, a relatedness measure $rel(ts, cs)$ is calculated for each sense ts_i of the target word to each sense cs_{jk} of each word j in the context. The computed relatedness values are illustrated in the table in Figure 7.1. All calculated values per target word are summed and the target word sense (or senses) yielding the highest sum are returned. More formally:

$$\arg\max_{ts_i \in T} ts_i := \max_{ts_i \in T} \sum_{ts_i \in T, cs_{jk} \in C} rel(ts_i, cs_{jk}) \tag{7.12}$$

where T is the set of all target word senses and C the set containing all senses of all words in the context window (described in the next subsection).

In case of ties, i.e., when more than one sense achieves the maximum value by Equation 7.12, all corresponding target word senses are returned by the algorithm. By contrast, if no target word sense achieves a value above zero, the algorithm does not return any target word sense.

7.2.1 Context Window

The window of context contains words surrounding the target word to be disambiguated. To determine the context window, the texts around the target words are automatically split into sentences, tokenized, and lemmatized – as described in Section 6.6. Stopwords such as determiners, identified with the

help of a given list[1], are ignored. Since semantic relatedness as defined above (in Section 7.1) can be calculated only for word lemma entries in GermaNet, the context window generally contains lemmas – rather than tokens – and is restricted to lemmas covered by GermaNet. Note that this restriction is in line with Pedersen et al. [2005] who include only words with entries from the Princeton WordNet into their context window.

As also explained in Section 7.1 above, all measures that rely on hypernymy/hyponymy relations for computing semantic relatedness are limited to comparing synsets of the same word class. This applies to all information-content-based methods (*res*, *jcn*, and *lin*) and most path-based methods (*path*, *lch*, and *wup*, but not *hso*). The context windows for these measures are further restricted to words of the same word class than the target word. Word class information is taken from GermaNet.[2]

Similarly to Pedersen et al. [2005], experiments are conducted with different context window sizes – see Subsection 7.3.4 for detailed observations. In particular, windows of sizes between 1 and 50 words are explored, which are not restricted to sentence boundaries and where the target word occurs in the middlemost position, if possible. The specified sizes do not include the target word. For instance, a context window of size four includes two words to the left and two words to the right of the target word. Further experiments use a context window which consists of all words with the appropriate word class from within the same sentence than the target word. The finding by Pedersen et al. [2005] that larger context windows improve performance is generally[3] confirmed for German across word classes and corpora (also see

[1]The stopwords list is copied from the Snowball stemmer [Porter, 1980], available at `http://snowball.tartarus.org/algorithms/german/stop.txt`.

[2]In contrast to English, very few German words with identical lemma forms belong to multiple word classes. More specifically, 28 out of 110 738 lemmas in GermaNet 9.0 have entries in more than one word class. Consequently, for the purpose at hand it is absolutely sufficient to use GermaNet's word class information – even with randomly choosing the 28 ambiguous cases.

[3]A few exceptions exist for some gloss-based measures, which obtain slightly better results for the context window which consists of all words from within the same sentence than the target word. This, in turn, corroborates the findings by Vasilescu et al. [2004] that performance of such word overlap algorithms decreases for larger contexts windows. Larger contexts have generally (i.e., across word classes and corpora) turned out to be the most reliable in the WSD setup of the present chapter.

Subsection 7.3.4). Thus, the results reported in Subsections 7.3.1 to 7.3.3 use a context window size of 50 words.

7.2.2 Random Sense Baseline

The *random sense baseline* represents a lower baseline for comparison with the WSD algorithms. For each target word occurrence, the baseline randomly selects exactly one sense from the set of corresponding GermaNet senses for the target word lemma. It is generally a good indicator of the difficulty of the task at hand.

The approach pursued in this chapter of selecting one sense from the set of GermaNet senses differs from the implementation by Henrich and Hinrichs [2012], where the option of *no appropriate sense from GermaNet* is included in the set of senses from which the baseline randomly selects one entry. The inclusion of this *no sense* option reflects the annotation setup which includes cases where it is not possible to annotate a sense from GermaNet. However, since there are generally few annotations with *no sense* (see Section 6.2), the baseline implemented in this chapter, which excludes this option, achieves a considerably higher performance than the baseline by Henrich and Hinrichs [2012] – most prominently, the baseline in this chapter achieves a perfect coverage of 100%. It is thus more difficult for the WSD algorithms to beat this baseline.

As a second baseline, a *most frequent sense baseline*, i.e., when the WSD system always assigns the sense which has most occurrences in the annotations, is sometimes used as an upper baseline for WSD. The main reason why the experiments of this chapter are not directly compared to such a baseline is to keep the setup purely knowledge-based and without any training bias – as described in Subsection 7.2.4 below. However, a most frequent sense baseline is employed in Chapter 8 on WSD using supervised machine learning methods.[1] This procedure follows the approach by Kilgarriff and Rosenzweig [2000] for SensEval, who state that "baselines which use training data [such as the most

[1]Furthermore, detailed sense distributions per lemma, which are comparable to a *most frequent sense baseline*, are provided in Appendix C.

frequent sense baseline] are intended for comparison with supervised systems" [Kilgarriff and Rosenzweig, 2000, page 28].

7.2.3 Combined WSD Algorithms

It has been shown for various NLP tasks, including part-of-speech tagging [van Halteren et al., 2001; Henrich et al., 2009] and word sense disambiguation [Florian et al., 2002; Florian and Yarowsky, 2002; Klein et al., 2002], that multiple classifier systems outperform single decision systems. Further, the performance of such methods is usually better the more diverse the individual systems are [Polikar, 2006].

Several WSD algorithms used in the present chapter employ – on the one hand – rather similar knowledge: all information-content-based measures use frequency information, all path-based and IC-based measures use the GermaNet hierarchy, and all gloss-based algorithms calculate word overlaps. On the other hand, since groups of algorithms are based on different underlying ideas, they are likely to produce diverse results: the information-content-based measures utilize frequency counts which are not used for the path-based algorithms; and both the path-based and information-content-based measures directly employ the GermaNet hierarchy to calculate relatedness which is different for the gloss-based algorithms. Therefore, combining the algorithms into a joint classifier appears to be a reasonable direction to pursue.

The combined algorithms in this chapter take all individual algorithms described in Section 7.1 above as input. In order to be able to combine the values of these individual algorithms into a joint overall score, the values returned by the single relatedness measures are normalized from zero to one. The combinations in this chapter employ simple *majority voting* and *Borda count* [Polikar, 2006]:

Majority voting Each relatedness measure votes for a candidate sense of the target word. The votes are summed with an equal weight of one; and the target word sense(s) with the highest count is defined to be the overall disambiguation result.

Borda count Each relatedness measure ranks the candidate senses of the target word. The first ranked sense receives a value of $N - 1$ (where N is the amount of senses the target word has), the second ranked sense gets $N - 2$ votes, etc. Thus, the last ranked candidate sense receives a value of 0. The values are summed for each target word sense; and the sense(s) with the highest value win(s).

Several approaches can improve the performance of a combined algorithm, such as (i) selecting a subset of single algorithms to be combined (rather than combining all of the above described), (ii) maximizing the precisions of the single algorithms by thresholds, and (iii) using *weighted majority voting* (rather than *simple majority voting* without weights). The reason why these approaches are not pursued in this chapter is to keep the setup purely knowledge-based and without any training bias – as described in the following subsection.

7.2.4 Purely Knowledge-Based Setup

For the WSD algorithms at hand, precision could be maximized by thresholds that allow the algorithms to yield non-zero values only if their scores are above a certain value, which would, in effect, reflect a minimal level of confidence. Since there is an inverse relationship between recall/coverage and precision, such thresholds apparently lead to a loss in coverage and recall (see Subsection 2.2.1 for a description of the trade-off between precision and recall). However, the performance of combined algorithms can be boosted by maximizing the precision of the single algorithms – even the loss in coverage and recall of the single algorithms can be compensated in a combined algorithm as long as the coverage of the single algorithms complement one another.

In the experiments by Henrich and Hinrichs [2012], the performance of combined algorithms are boosted by maximizing the precision of the single algorithms with such thresholds. Furthermore, a combination by *weighted majority voting* outperformed *simple majority voting* and *Borda count*.

However, the main reason why the experiments in this chapter do not utilize thresholds and *weighted majority voting*, nor a most frequent sense baseline, is

to keep the setup purely knowledge-based and without any training bias. That is, determining most frequent senses and identifying optimal threshold values for the single algorithms or optimal weights for a *weighted majority voting* algorithm would either require a separate training set, which is generally very uncommon for knowledge-based systems, or would introduce a bias when the reported evaluation results are obtained on the same data that is used for adjusting thresholds and weights or for identifying most frequent senses.

7.3 Evaluating Knowledge-Based WSD

In order to evaluate the knowledge-based WSD setup, an extensive set of experiments using many different measures of semantic relatedness and two algorithms for combining those individual results is performed. Since the evaluation of WSD experiments is more reliable and meaningful the more gold standard annotations available, the evaluation in this section considers all available sense annotations for German. The experiments are run on all annotations from the three German corpora described in Chapter 6: the semi-automatically constructed, web-harvested WebCAGe (Section 6.3), the manually sense-annotated TüBa-D/Z treebank (Section 6.4), and the web-harvested and manually sense-annotated deWaC (Section 6.5).

Performance of the algorithms is measured in terms of coverage, recall, precision, and F_1 – calculated by their standard formulas as described in Subsection 2.2.1. Results are reported as overall numbers, i.e., micro-averaged over all annotated instances rather than macro-averaged over all lemmas.[1]

The following subsections report WSD results separately for the three word classes available in GermaNet: nouns, verbs, and adjectives.[2]

[1] See Section 2.2.2 for more information on micro- and macro-averages.

[2] The reason why adjectives are – against alphabetic order – reported last is that only two of the three corpora contain sense annotations for this word class while sense annotations for nouns and verbs are available in all three corpora.

7.3.1 Profiling WSD Results for Nouns

Table 7.1 gives coverage, recall, precision, and F_1-scores for all above-described knowledge-based WSD algorithms evaluated on all noun instances from the three sense-annotated corpora. The principle structure of the table divides the results for the three sense-annotated corpora by two horizontal lines: it first prints the results for WebCAGe, in the middle for TüBa-D/Z, and at the bottom for deWaC. For each corpus, Table 7.1 shows the results for one relatedness measure at a time as explained in Section 7.2, for the two combined WSD algorithms outlined in Subsection 7.2.3, and for the random sense baseline from Subsection 7.2.2. According to the classification from Section 7.1, the relatedness algorithms are grouped into path-based (*lch*, *wup*, *path*, and *hso*), information-content-based (*res*, *jcn*, and *lin*), and gloss-based (*lesk-**) – separated by horizontal lines. In general, prominently high results are highlighted in boldface, whereas unexpectedly low values are italicized.

Results for Nouns in WebCAGe

The upper part of Table 7.1 shows that – with the exceptions of *hso* and *lesk-GermaNet*, which stand out from the rest in terms of precision – the single relatedness measures all achieve similar results for recall (between 44% and 50%) and precision (between 49% and 56%), and, consequently, also for F_1 (between 48 and 51) when evaluated on all sense-annotated nouns in WebCAGe. The path-based *hso* measure achieves overall the highest precision of 64.38%. By contrast, *hso* achieves the lowest recall of 38.30%, which is the reason for an F_1-score comparable to the other algorithms. This low recall is not surprising since *hso* has a low coverage. What rather remains to be explained is why *hso* has such a low coverage. This is due to the nature of the relatedness measure: since the *hso* measure prohibits long paths between two synsets and many changes in direction, the measure returns a positive value only for a restricted set of synset combinations that are 'close' enough in the GermaNet graph. A low coverage is, thus, caused by the fact that the *hso* measure apparently does not return positive values when comparing the target words with many words from the context window.

Table 7.1: Knowledge-based WSD results for nouns.

Corpus	Method	Coverage	Recall	Precision	F_1
WebCAGe: 6 522 instances of 1 537 lemmas	*path*	89.90%	44.42%	49.41%	46.78
	lch	89.90%	46.04%	51.22%	48.49
	wup	84.65%	44.46%	52.53%	48.16
	hso	*59.49%*	*38.30%*	**64.38%**	48.03
	res	83.89%	45.03%	53.68%	48.98
	jcn	89.90%	45.92%	51.08%	48.36
	lin	83.89%	45.15%	53.83%	49.11
	lesk-GermaNet	80.34%	47.95%	**59.68%**	**53.17**
	lesk-Wiktionary	97.90%	49.85%	50.92%	50.38
	lesk-glosses	77.66%	43.82%	56.43%	49.33
	lesk-lex-fields	97.81%	49.59%	50.70%	50.14
	lesk-all-info	98.30%	50.17%	51.04%	50.60
	majority voting	98.57%	54.91%	55.70%	55.30
	Borda count	98.57%	**55.52%**	**56.32%**	**55.92**
	random sense	100.00%	42.41%	42.41%	42.41
TüBa-D/Z: 8 803 instances of 30 lemmas	*path*	100.00%	*25.45%*	*25.45%*	*25.45*
	lch	100.00%	*29.01%*	*29.01%*	*29.01*
	wup	99.99%	46.63%	46.64%	46.63
	hso	97.14%	46.89%	48.28%	47.57
	res	99.99%	*26.77%*	*26.78%*	*26.78*
	jcn	100.00%	52.77%	52.77%	52.77
	lin	99.99%	42.60%	42.60%	42.60
	lesk-GermaNet	99.98%	55.45%	55.46%	55.45
	lesk-Wiktionary	100.00%	**67.19%**	**67.19%**	**67.19**
	lesk-glosses	99.59%	52.32%	52.54%	52.43
	lesk-lex-fields	100.00%	58.72%	58.72%	58.72
	lesk-all-info	100.00%	56.67%	56.67%	56.67
	majority voting	100.00%	57.86%	57.86%	57.86
	Borda count	100.00%	**66.77%**	**66.77%**	**66.77**
	random sense	100.00%	27.22%	27.22%	27.22
deWaC: 385 instances of 18 lemmas	*path*	99.48%	38.70%	38.90%	38.80
	lch	99.48%	41.82%	42.04%	41.93
	wup	99.48%	42.08%	42.30%	42.19
	hso	96.36%	44.94%	46.63%	45.77
	res	99.48%	45.71%	45.95%	45.83
	jcn	99.48%	**49.61%**	**49.87%**	**49.74**
	lin	99.48%	**49.61%**	**49.87%**	**49.74**
	lesk-GermaNet	99.74%	40.00%	40.10%	40.05
	lesk-Wiktionary	100.00%	38.18%	38.18%	38.18
	lesk-glosses	98.44%	43.12%	43.80%	43.46
	lesk-lex-fields	100.00%	42.86%	42.86%	42.86
	lesk-all-info	100.00%	41.82%	41.82%	41.82
	majority voting	100.00%	**55.06%**	**55.06%**	**55.06**
	Borda count	100.00%	52.47%	52.47%	52.47
	random sense	100.00%	29.61%	29.61%	29.61

The gloss-based *lesk-GermaNet* measure, which uses glosses and lexical fields from GermaNet to calculate word overlaps, yields – with a value of 53.17 – overall the highest F_1-score. Since WebCAGe contains sense annotations for exactly those words for which GermaNet has harvested sense descriptions from Wiktionary, it seems surprising that *lesk-GermaNet* outperforms the other relatedness measures – especially the other gloss-based measures which all use information from Wiktionary. The explanation for this behavior lies mainly in the prominently higher precision of 59.68% for *lesk-GermaNet* which leads to a higher F_1-score.

Further, the single relatedness algorithms – again with the exception of *hso* – yield good results in terms of coverage ranging from 77.66% to 98.30%. However, the gloss-based algorithms which use glosses and lexical fields from GermaNet (*lesk-GermaNet*) or which use glosses from both GermaNet and Wiktionary (*lesk-glosses*) score lower in coverage than the other algorithms. One explanation for this lower coverage is that the glosses do not contain enough lexical material. This defect can be remedied by including lexical material in the lexical fields obtained from Wiktionary. There is a considerable jump in coverage from 80.34% for *lesk-GermaNet*, which uses only glosses and lexical fields from GermaNet to calculate word overlaps, to an almost complete coverage of 98.30% achieved by *lesk-all-info*, which makes use of lexical fields and glosses from both resources. This jump in coverage underscores the usefulness of aligning GermaNet with Wiktionary (see Chapter 4) for such gloss-based algorithms.

In general, the results obtained by the gloss-based measures lie above those of path-based and information-content-based measures: the F-score of 49.33 for *lesk-glosses*, which is the lowest result among the gloss-based measures, outperforms the F-score of 48.11 for *lin*, which is the highest result among the path-based and information-content-based measures. This finding that word overlap methods perform well corroborates the results reported by Pedersen et al. [2005] for English WSD.[1] It indicates that words from glosses or lexical

[1]Generally, it does not make much sense to compare the numbers of English WSD results from Pedersen et al. [2005] with numbers of German WSD experiments from this chapter, because the wordnets which serve as sense inventories and the sense-annotated materials

fields of the appropriate target word senses often occur in WebCAGe's context windows.

The best results are achieved by combining all single algorithms. Table 7.1 also shows the results of the experiments with the two combined algorithms using *majority voting* and *Borda count* – as described in Subsection 7.2.3. The best overall result with an F-score of 55.92 is achieved by the combination with *Borda count*. This result contradicts the previous finding of Broscheit et al. [2010] who did not obtain better results by combining individual WSD algorithms.

In general, the results presented for nouns in WebCAGe represent an updated version of the experiments reported by Henrich and Hinrichs [2012]. The results are comparable, though due to different versions for both the sense inventory and the sense-annotated corpus, they are not exactly identical.[1] That is, the WSD experiments described in the paper use old versions of GermaNet (release 6.0) and WebCAGe (unpublished pre-version), whereas this chapter reports on experiments with the most recent versions of GermaNet (i.e., release 9.0) and WebCAGe (version 3.0).

What remains to be explained is the only remarkable deviation in the results, which is for the reported random sense baselines. This deviation is due to different implementations of the baseline (see Subsection 7.2.2 for further details): in contrast with the implementation in Henrich and Hinrichs [2012], the baseline in this chapter excludes the option of *no appropriate sense from GermaNet* from the set of senses from which the baseline randomly selects one entry. This is the reason why the baseline in this chapter achieves a perfect coverage of 100%. Although the inclusion of this *no sense* option would reflect the annotation setup which includes cases where it is not possible to annotate

differ considerably (see Subsection 3.1 for a comparison of GermaNet with Princeton WordNet). The strongest difference that impacts the WSD results is probably the granularity of the sense distinctions. This is reflected in the random sense baselines when comparing these baselines from Pedersen et al. [2005] with those in the following subsections. However, it is interesting to compare general tendencies and whether or not they are in common to WSD experiments for the two languages.

[1] Furthermore, the gloss-based algorithms differ in the information used for calculating word overlaps. Only the gloss-based algorithm that uses lexical fields and glosses from GermaNet and Wiktionary is used both in the paper (abbreviated as *lesk-Ggw-Lgw*) and in this chapter (labeled as *lesk-all-info*).

a sense from GermaNet, there are generally few annotations with *no sense* (see Section 6.2). Thus, the baseline implemented in this chapter achieves a considerably higher performance than the baseline by Henrich and Hinrichs [2012] and is consequently more difficult to beat. Nevertheless, Table 7.1 shows that for the evaluation scores of recall, precision, and F_1 obtained for nouns in WebCAGe – with the only exception of *hso* in terms of recall – all algorithms outperform this random sense baseline by a wide margin.

Results for Nouns in TüBa-D/Z

The most striking finding among the results of the evaluation on all sense-annotated nouns in the TüBa-D/Z treebank – shown in the middle of Table 7.1 – is that the gloss-based *lesk-Wiktionary* measure, which uses glosses and lexical fields from Wiktionary, yields outstandingly high results: 67.19% recall, precision, and F-score at a coverage of 100%. This result is the most remarkably highest result over all three corpora that a single relatedness measure achieves. It is 12 points higher than the F-score achieved by the gloss-based measure that uses solely GermaNet information (*lesk-GermaNet*). This finding again underscores the usefulness of aligning GermaNet with Wiktionary for gloss-based algorithms.

Since not all GermaNet words with sense annotations in TüBa-D/Z contain Wiktionary sense descriptions, it seems especially surprising that *lesk-Wiktionary* performs significantly better when evaluated on the TüBa-D/Z than on WebCAGe: the F_1-score is 17 points higher for the treebank than for WebCAGe, i.e., 67.19 vs. 50.38, respectively. This comparison gives evidence that the study by Henrich and Hinrichs [2012], which evaluated knowledge-based WSD only on the WebCAGe corpus, was not biased towards artificially high results due to evaluating only words with a Wiktionary mapping.

While the single relatedness measures (with only two exceptions in terms of precision) performed similarly when evaluated on nouns in WebCAGe, there is much more variance for TüBa-D/Z. On the one hand, three of the single measures (*lch*, *path*, and *res*) perform particularly poorly with F_1-scores below

30.[1] On the other hand, all gloss-based measures reach F_1-scores above 52. The difference between the worst F_1-value of 25.45 obtained by *path* compared to the best value of 67.19 obtained by *lesk-Wiktionary* is 42. This difference is the largest gap obtained over all word classes and corpora.

As already found for nouns in WebCAGe, the results obtained by the gloss-based measures lie above the path-based and information-content-based measure results: the F-score for *lesk-glosses*, which is the lowest result among the gloss-based measures, is in the same range (between 51 and 52) with the F-score for *jcn*, which is the highest result among the path-based and information-content-based measures. It corroborates the finding by Pedersen et al. [2005] for English WSD that word overlap methods perform well, and it indicates that words from glosses or lexical fields of the appropriate target word senses often occur in TüBa-D/Z's contexts.

Since the single relatedness measure *lesk-Wiktionary* already performs extremely well, the combined measures do not help: with F-measures of 57.86 and 66.77, *majority voting* and *Borda count* decrease the overall result of 67.19 obtained by *lesk-Wiktionary*. This *lesk-Wiktionary* score is the best overall evaluation result achieved on the TüBa-D/Z data. It outperforms the best WebCAGe result, which is obtained for the *Borda count* algorithm, by 11 points (i.e., best F_1-score of 67.19 for TüBa-D/Z vs. 55.92 for WebCAGe). This fact seems particularly surprising when considering the baselines: the random sense baseline for WebCAGe has an F-score of 42.41, about 15 points higher than the score of 27.22 for TüBa-D/Z. The explanation for what looks like contradictory findings must have to do with the nature of the corpus texts. That is, WebCAGe contains several annotations for which only a restricted context is available (i.e., the example sentences harvested from Wiktionary itself are single sentences only without a larger context – see Section 5.2), whereas all of the sense annotations in TüBa-D/Z have sufficiently large contexts. Subsec-

[1] It is interesting to note that both *lch* and *path* would perform significantly better (about 15%) with the setup applied in Henrich and Hinrichs [2012], where individual results – depending on whether they lie above or below a certain threshold – get scaled up or down, respectively. As already outlined in Subsection 7.2.4 above, the main reason why the experiments in this chapter do not utilize thresholds is to keep the setup purely knowledge-based and without any training bias.

tion 7.3.4 below performs detailed experiments with different context sizes and compares the availability of contexts for all three corpora. The fact that more context is available for TüBa-D/Z than for WebCAGe is also reflected in the consistently high coverage of more than 99% for almost all WSD algorithms when evaluated on TüBa-D/Z (with the only exception of 97.14% coverage obtained by *hso*), as opposed to (still good, but nevertheless lower) coverage results below 90% for several WSD algorithms when evaluated on WebCAGe.

Results for Nouns in deWaC

The WSD results for nouns in deWaC are shown in the bottom of Table 7.1. Most single relatedness measures achieve F_1-scores between 41 and 46[1] – with little variance. The exceptions are *path* and *lesk-Wiktionary*, which perform lower than the other relatedness measures (with F_1-scores between 38 and 39), and *jcn* and *lin*, which outperform all other single algorithms with a score of 49.74.

As for TüBa-D/Z, all WSD algorithms yield consistently high coverage of more than 98%. The reason is also the same as for the treebank: sufficiently large contexts for all sense annotations in deWaC.[2] The only exception is – as is also the case for WebCAGe and TüBa-D/Z – *hso* which achieves a slightly lower but still very good coverage of 96.36%. The explanation why *hso* has a lower coverage is provided in the discussion on the results for nouns in WebCAGe above. It is mainly due to the nature of the relatedness measure, which prohibits long paths between two synsets or many changes in direction and thus returns positive values only for a restricted set of synset combinations that are 'close' enough in the GermaNet graph.

All in all, information-content-based measures produce the strongest results for nouns in deWaC, which is also reflected by the facts that the best

[1]Since the coverage is (almost) 100% for all WSD algorithms, the values for precision and recall are very similar to the F-scores – see Subsection 2.2.1 for an explanation of the relationship between the performance measures of coverage, precision, recall, and F_1.

[2]Note that the contexts of the original deWaC sense annotations provided online by Broscheit et al. [2010] are restricted to sentential phrases only. The revised version of the deWaC sense annotations used in this thesis is enriched by larger contexts extracted from the proper deWaC corpus [Baroni et al., 2009]. See Section 6.5 for more details on the original as well as the revised versions of the sense annotations in the deWaC corpus.

scoring single measures are *jcn* and *lin* and that all results obtained by the information-content-based measures lie above all path-based and gloss-based measure results. This is different from the finding for the TüBa-D/Z treebank, where gloss-based measures perform best. The real explanation for what looks like contradictory findings has to do with the kind of words contained in the contexts of the target words. That is, while words from TüBa-D/Z's contexts are often also used in glosses or lexical fields of the appropriate target word senses, words from deWaC's contexts are apparently often closely related to the appropriate target word senses in terms of the taxonomic hypernymy structure of the GermaNet graph.

With an F-score of 43.46, the best scoring word-overlap measure for deWaC is *lesk-glosses*, which uses glosses from GermaNet and Wiktionary for calculating word overlaps. This result again stands in contrast to the results from WebCAGe and TüBa-D/Z, where of the gloss-based measures, *lesk-glosses* performs worst.

As for WebCAGe, the combined algorithms outperform the single measures. *Majority voting* yields – with an F-score of 55.06 – the highest result for nouns in deWaC. However, for all single algorithms the WSD results for deWaC are significantly lower than the results for WebCAGe and TüBa-D/Z. One explanation for this lower performance is related to the lower random sense baseline (compared to the random sense baseline for WebCAGe), which suggests that the disambiguation for nouns in deWaC is more difficult than the disambiguation for nouns in WebCAGe. However, this does not explain the lower performance compared to TüBa-D/Z. There must be a further explanation, which can be related only to the nature of the corpus texts. That is, data from the web, which is the basis for the WebCAGe and deWaC corpora, is apparently noisier or for other reasons harder for WSD algorithms than newspaper texts, which are the basis for the TüBa-D/Z treebank.

The results for nouns in deWaC are comparable to previously published experiments on the same sense-annotated corpus. Broscheit et al. [2010] report F_1-scores between 41 and 49 for most of the WSD algorithms they use, including the two methods to predict most frequent sense information originally proposed by McCarthy et al. [2004] and Lapata and Keller [2007], and

a simple majority combination of single algorithms. As their best overall result, Broscheit et al. [2010] report an F-score of 55.49, which they achieve with the Personalized PageRank algorithm [Agirre and Soroa, 2009]. As for combining individual WSD algorithms, the results reported in this chapter improve in combination, which contradicts Broscheit et al.'s finding that there is no improvement. However, it is important to note that a comparison of the results from Broscheit et al. [2010] with the results of this chapter is not entirely fair. There are two main factors that influence the results. Firstly, the versions for both underlying resources, i.e., the sense inventory and the sense-annotated corpus, are different. The WSD experiments described in the paper use old versions of GermaNet (release 5.1) and the sense annotations in deWaC, whereas this chapter reports on experiments with the most recent version of GermaNet (i.e., release 9.0) and a revised version of the sense annotations (see Section 6.5 for details). Secondly, the evaluation setups differ both in the used algorithms (semantic relatedness algorithms in this chapter vs. the Personalized PageRank algorithm in the paper) as well as backoff strategies for cases where the WSD algorithms are unable to assign any word sense (the experiments reported in this chapter do not use any backoff while Broscheit et al. [2010] do, which mainly leads to a perfect coverage of 100%).

7.3.2 Profiling WSD Results for Verbs

This subsection discusses the knowledge-based sense disambiguation of verbs. Table 7.2 provides results in terms of coverage, recall, precision, and F-score – separately for the three sense-annotated corpora. Analogously to the table for nouns, two horizontal lines divide the results for WebCAGe (at the top), from those for TüBa-D/Z (in the middle), and for deWaC (at the bottom). For each corpus, Table 7.2 presents the results for the single relatedness measures, for the two combined WSD algorithms, and for the random sense baseline. Groups of relatedness measures, i.e., path-based (*lch*, *wup*, *path*, and *hso*), information-content-based (*res*, *jcn*, and *lin*), and gloss-based (*lesk-**), are separated by horizontal lines. As for nouns, remarkably high results are highlighted in boldface, whereas low values are italicized.

Table 7.2: Knowledge-based WSD results for verbs.

Corpus	Method	Coverage	Recall	Precision	F_1
WebCAGe: 3 186 instances of 959 lemmas	*path*	43.16%	17.92%	41.53%	25.04
	lch	43.16%	18.27%	42.33%	25.52
	wup	29.79%	11.77%	39.52%	*18.14*
	hso	28.81%	12.37%	42.92%	19.20
	res	29.79%	12.08%	40.57%	*18.62*
	jcn	43.16%	18.20%	42.18%	25.43
	lin	29.79%	12.27%	41.20%	*18.91*
	lesk-GermaNet	49.06%	21.44%	43.70%	28.76
	lesk-Wiktionary	91.84%	39.96%	43.51%	41.66
	lesk-glosses	59.54%	32.42%	**54.45%**	40.65
	lesk-lex-fields	90.18%	38.54%	42.74%	40.53
	lesk-all-info	92.34%	41.37%	44.80%	**43.02**
	majority voting	95.54%	42.50%	44.48%	43.47
	Borda count	95.54%	**44.95%**	**47.04%**	**45.97**
	random sense	100.00%	35.56%	35.56%	35.56
TüBa-D/Z: 9 107 instances of 79 lemmas	*path*	99.99%	37.58%	37.58%	37.58
	lch	99.99%	39.34%	39.35%	39.35
	wup	98.86%	41.93%	42.42%	42.18
	hso	94.98%	39.99%	42.10%	41.02
	res	98.86%	40.33%	40.80%	40.56
	jcn	99.99%	**55.64%**	**55.64%**	**55.64**
	lin	98.86%	44.46%	44.97%	44.72
	lesk-GermaNet	98.43%	39.45%	40.08%	39.77
	lesk-Wiktionary	92.12%	45.24%	49.11%	47.10
	lesk-glosses	95.18%	37.39%	39.28%	38.31
	lesk-lex-fields	94.29%	46.72%	49.55%	48.10
	lesk-all-info	99.91%	48.95%	48.99%	**48.97**
	majority voting	100.00%	51.60%	51.60%	51.60
	Borda count	100.00%	**52.10%**	**52.10%**	**52.10**
	random sense	100.00%	36.17%	36.17%	36.17
deWaC: 608 instances of 15 lemmas	*path*	100.00%	22.86%	22.86%	22.86
	lch	100.00%	22.04%	22.04%	22.04
	wup	99.67%	19.24%	19.31%	19.28
	hso	99.34%	17.27%	17.38%	17.33
	res	99.67%	15.79%	15.84%	*15.82*
	jcn	100.00%	22.70%	22.70%	22.70
	lin	99.67%	17.43%	17.49%	17.46
	lesk-GermaNet	98.85%	24.51%	24.79%	24.65
	lesk-Wiktionary	97.20%	25.99%	26.73%	**26.36**
	lesk-glosses	98.85%	21.38%	21.63%	21.51
	lesk-lex-fields	100.00%	21.88%	21.88%	21.88
	lesk-all-info	100.00%	22.86%	22.86%	22.86
	majority voting	100.00%	25.66%	25.66%	25.66
	Borda count	100.00%	**27.80%**	**27.80%**	**27.80**
	random sense	100.00%	13.65%	13.65%	13.65

Results for Verbs in WebCAGe

The upper part of Table 7.2 shows that, especially when compared to the evaluation on nouns in WebCAGe, the WSD algorithms all achieve very heterogeneous results for the four performance measures of coverage, recall, precision, and F_1-score when evaluated on all sense-annotated verbs in WebCAGe. The *lesk-glosses* measure achieves overall the highest precision of 54.45%, whereas *lesk-all-info* yields among the single relatedness measures the best F-score of 43.02. By contrast, *wup*, *lin*, and *res* achieve the lowest F-scores – between 18 and 19.

A comparison of the results for the three different types of relatedness measures shows that gloss-based measures (with F-scores between 28 and 43) outperform path-based and information-content-based measures (with F-scores between 18 and 26). This behavior corroborates the findings for nouns in WebCAGe and TüBa-D/Z as discussed in Subsection 7.3.1 above. It proves that words from glosses or lexical fields of the appropriate target word senses often occur in WebCAGe's contexts.

The most striking observation among the results for verbs in WebCAGe is that the alignment between GermaNet and Wiktionary considerably improves the WSD performance: with F-scores between 40 and 43, the gloss-based measures that employ information from Wiktionary (*lesk-Wiktionary*, *lesk-glosses*, *lesk-lex-fields*, and *lesk-all-info*) clearly outperform the gloss-based measure that merely employs information from GermaNet (*lesk-GermaNet*) and which achieves an F-score of 28.76. Further, the individual relatedness measures that do not use information from Wiktionary (including all path- and IC-based measures and *lesk-GermaNet*) achieve coverage between 28% and 49%, whereas those relatedness measures that employ Wiktionary achieve considerably higher coverage, between 59% and 92%. These findings underscore the usefulness of aligning GermaNet with Wiktionary for such gloss-based algorithms and explain why *lesk-all-info* yields the best F-score among the single relatedness measures.

Even better results than for *lesk-all-info* are achieved by combining all single algorithms: *majority voting* achieves an F-score of 43.47 and *Borda*

count achieves the best overall result with an F-score of 45.97.

In comparison to the evaluation on nouns in WebCAGe (see Table 7.1 above), the performance for verbs is considerably lower: for path-based and information-content-based measures, the F-values are between 21 and 30 points lower, whereas for gloss-based measures, the F-scores are mostly about 8–10 points lower – with the only exception of *lesk-GermaNet*, where the F-score for verbs is 24 points lower. The difference in performance of the combined measures is between 10 and 11 points for *majority voting* and *Borda count*. Since the random sense baseline is a good indicator of the difficulty of the task at hand, a lower random sense baseline for verbs indicates that the task of disambiguating verbs is more difficult that the task of disambiguating nouns. The obvious explanation for a lower random sense baseline is a higher polysemy. In the present case, the lower performance cannot be explained solely by the random sense baseline, because it is only 6 points lower for verbs than for nouns. Rather, low coverage for verbs causes low recall, which in turn heavily influences the F_1-scores. However, there must be a further explanation related to the word class of verbs. That is, semantic relatedness measures are little suitable for disambiguating verb senses. This finding generally corroborates the results reported by Pedersen et al. [2005] for English WSD.

According to Pedersen et al. [2005, page 27], the low disambiguation performance of path-based and IC-based measures for verbs was to be expected because these measures were originally meant for nouns rather than for verbs. Their explanation is that since the WordNet hierarchies for verbs are very shallow compared to the hierarchies for nouns, path-based and information-content-based measures are much more effective for nouns while more of an experimental nature when applied to verbs. Since the verb hierarchies in GermaNet have similar density as those for English verbs (as compared to those of German and English nouns), this suggests that Pedersen et al.'s [2005] explanation also holds true for German WSD.

Overall, the analogy that path-based and IC-based measures perform poorly while most gloss-based measures perform well above random guessing confirms the findings by Pedersen et al. [2005].

Results for Verbs in TüBa-D/Z

As for verbs in WebCAGe and for nouns in TüBa-D/Z, the single relatedness measures achieve heterogeneous results when evaluated on verbs in the treebank – shown in the middle of Table 7.2. The range begins with 37.58 as the lowest F-score, achieved by the *path* measure, and goes up to the highest result of 55.64, which is obtained by *jcn*. For all evaluation scores, i.e., in terms of coverage, recall, precision, and F_1, obtained for verbs in TüBa-D/Z, all WSD algorithms outperform the random sense baseline by a wide margin.

Table 7.2 shows that, although the baselines for verbs in TüBa-D/Z and for verbs in WebCAGe are with F-scores of 35.56 and 36.17 very similar, the results for verbs in TüBa-D/Z outperform the results for verbs in WebCAGe for most WSD methods. The explanation for this behavior must have to do with the nature of the corpus texts – as already outlined in the discussion of results for nouns above. That is, WebCAGe contains several annotations for which only a restricted context is available (i.e., the example sentences harvested from Wiktionary itself are single sentences only without a larger context – see Section 5.2), whereas all of the sense annotations in TüBa-D/Z have sufficiently large contexts. The availability of context in the individual corpora is analyzed in Subsection 7.3.4 below. The large contexts available in the TüBa-D/Z is also reflected in the consistently high coverage of more than 92% for all WSD algorithms when evaluated on verbs in TüBa-D/Z, as opposed to considerably lower coverage between 28% and 92% for the semantic relatedness algorithms when evaluated on verbs in WebCAGe.

The highest F-score of 48.97 among the gloss-based measures, which is the second highest score among all semantic relatedness measures, is achieved by using lexical fields and glosses both from GermaNet and Wiktionary to calculate word overlaps (i.e., by the *lesk-all-info* measure). *Lesk-all-info* outperforms the result obtained by using lexical fields and glosses solely from GermaNet to calculate word overlaps (i.e., *lesk-GermaNet*) by 9 points. This finding – once again – underscores the usefulness of aligning GermaNet with Wiktionary.

Since the single relatedness measure *jcn* already performs very well, the

combined measures do not help: with F-scores of 51.60 and 52.10, *majority voting* and *Borda count* decrease the overall result of 55.64 obtained by *jcn*.

Results for Verbs in deWaC

The bottom part of Table 7.2 shows the evaluation results for verbs in deWaC. All WSD algorithms produce very poor results: the single relatedness measures achieve F_1-scores between 16 and 26 – without any exceptions. The lowest result of 15.82 is obtained by *res*, and the highest result of 26.36 by *lesk-Wiktionary*. Even the combination with *Borda count* is – with a score of 27.80 – only slightly higher than the highest result obtained by a singe relatedness measure.

Overall, these results are the worst results achieved among all corpora and word classes. The most obvious explanation for this behavior is the high polysemy for verbs in deWaC, which is, with an average of 7.9, by far the highest among all corpora and word classes (see Chapter 6). The conclusion is that none of the knowledge-based algorithms – not even the combined classifiers – is suited for disambiguating verbs in deWaC. The only positive finding concerning the disambiguation of verbs in deWaC is that – despite their bad performance – all WSD algorithms outperform the random sense baseline for all evaluation scores, i.e., in terms of coverage, recall, precision, and F_1, as shown in the last row in Table 7.2.

A direct comparison to previously published experiments on the same sense-annotated corpus, which was possible for nouns (see Subsection 7.3.1), is not possible for verbs, because Broscheit et al. [2010] report results for *nouns only* and for *all words* (including adjectives, nouns, and verbs), but not for *verbs only*. However, the large drop in performance reported by Broscheit et al. [2010] for all word classes compared to the results for only nouns implies that the performance for the disambiguation of both adjectives and verbs is much worse than the performance for nouns. Even if such a direct comparison is not entirely fair for several reasons (also see the discussion for nouns in deWaC above), the tendency that the disambiguation of verbs is much more difficult than for nouns corroborates the results by Broscheit et al. [2010] on German

WSD.

7.3.3 Profiling WSD Results for Adjectives

The knowledge-based disambiguation of adjectives differs from the experiments by Pedersen et al. [2005] for English: in particular, path-based and information-content-based algorithms are available only for German adjectives but not for English ones (when using Princeton WordNet to calculate semantic relatedness). The reason is that the Princeton WordNet does not structure adjectives hierarchically (as explained in Subsection 3.1), which precludes the use of relatedness measures based on hypernymy paths. By contrast, GermaNet encodes adjectives comparable to nouns and verbs in a hypernymy hierarchy, and, thus, allows experimenting with the same set of relatedness measures as for the other two word classes.

Table 7.3 presents the evaluation results for the knowledge-based sense disambiguation of German adjectives. The structure of the table follows the table structures for nouns and verbs above – with the exception that it includes results only for WebCAGe and deWaC, the two corpora that contain sense annotations for adjectives.[1] For both corpora, Table 7.2 lists coverage, recall, precision, and F-score for each of the single relatedness measures (grouped by type), for the two combined WSD algorithms, and for the random sense baseline. As before, very high and very low results are emphasized in boldface and in italic, respectively.

Results for Adjectives in WebCAGe

The two most striking findings among the WSD results of the evaluation on adjectives in WebCAGe – shown in the upper part of Table 7.3 – are (i) the strong positive impact of linking GermaNet with Wiktionary and (ii) the great improvement when combining individual WSD methods.

With respect to the observation (i), while the gloss-based method that merely uses information from GermaNet (i.e., *lesk-GermaNet*) achieves only

[1] The TüBa-D/Z treebank contains sense annotations for nouns and verbs only.

Table 7.3: Knowledge-based WSD results for adjectives.

Corpus	Method	Coverage	Recall	Precision	F_1
WebCAGe: 694 instances of 212 lemmas	*path*	50.14%	19.74%	39.37%	*26.30*
	lch	50.14%	28.82%	57.47%	38.39
	wup	36.46%	24.93%	**68.38%**	36.54
	hso	41.07%	23.92%	58.25%	33.91
	res	35.73%	23.78%	**66.53%**	35.03
	jcn	50.14%	20.03%	39.94%	*26.68*
	lin	35.73%	24.64%	**68.95%**	36.31
	lesk-GermaNet	32.13%	*19.16%*	59.64%	29.01
	lesk-Wiktionary	93.37%	41.35%	44.29%	42.77
	lesk-glosses	51.59%	32.85%	**63.69%**	43.35
	lesk-lex-fields	91.21%	40.78%	44.71%	42.65
	lesk-all-info	93.95%	**42.65%**	45.40%	**43.98**
	majority voting	96.11%	54.90%	57.12%	55.99
	Borda count	96.11%	**55.19%**	**57.42%**	**56.28**
	random sense	100.00%	44.81%	44.81%	44.81
deWaC: 90 instances of 4 lemmas	*path*	100.00%	22.22%	22.22%	22.22
	lch	100.00%	*18.89%*	*18.89%*	*18.89*
	wup	100.00%	25.56%	25.56%	25.56
	hso	92.22%	26.67%	28.92%	27.75
	res	100.00%	26.67%	26.67%	26.67
	jcn	100.00%	25.56%	25.56%	25.56
	lin	100.00%	21.11%	21.11%	21.11
	lesk-GermaNet	86.67%	18.89%	21.79%	20.24
	lesk-Wiktionary	100.00%	**40.00%**	**40.00%**	**40.00**
	lesk-glosses	100.00%	23.33%	23.33%	23.33
	lesk-lex-fields	100.00%	**40.00%**	**40.00%**	**40.00**
	lesk-all-info	100.00%	**40.00%**	**40.00%**	**40.00**
	majority voting	100.00%	28.89%	28.89%	28.89
	Borda count	100.00%	32.22%	32.22%	32.22
	random sense	100.00%	18.89%	18.89%	18.89

an F-score of 29.01, all gloss-based methods that employ Wiktionary information outperform this score by at least 13.5 points. The main reason for the low performance for *lesk-GermaNet* is its lower coverage, which indicates that GermaNet glosses do not contain enough lexical material for an effective use of a word overlap method. This defect can be remedied by including lexical material in the glosses and lexical fields obtained from Wiktionary, which causes a considerable jump in coverage and therefore also in the overall F-measure.

Generally, with F-scores between 42 and 44, the gloss-based methods which

employ Wiktionary information (*lesk-Wiktionary*, *lesk-glosses*, *lesk-lex-fields*, and *lesk-all-info*) clearly outperform the individual relatedness measures which do not use information from Wiktionary (including all path- and IC-based measures and *lesk-GermaNet*) and which achieve F-scores between 26 and 38. This finding once more underscores the usefulness of aligning GermaNet with Wiktionary for word overlap algorithms and explains why the gloss-based method that uses all information from GermaNet and Wiktionary (i.e., *lesk-all-info*) again performs best among all single relatedness measures. It achieves an F-score of 43.98.

The main reason for this performance improvement is considerably greater coverage. That is, the individual relatedness measures that do not use information from Wiktionary achieve coverage between 32% and 50%, whereas those relatedness measures that employ Wiktionary achieve considerably higher coverage, between 51% and 94%. The fact that these improvements are especially remarkable for adjectives and verbs in WebCAGe (less for nouns in WebCAGe) has to do with the GermaNet resource – especially with its hypernymy structure. Due to a more elaborated noun hierarchy and less dense hierarchies for adjectives and verbs, the path-based and information-content-based measures are less effective for adjectives and verbs – as already discussed in Subsection 7.3.2 for verbs in WebCAGe.

With respect to the observation (ii), the fact that the combined algorithms outperform the individual methods by a wide margin is not surprising since several semantic relatedness measures achieve considerably high precision. These high precision values compensate the extremely low coverage results and lead to great overall improvements when combining individual methods. The highest overall F-score of 56.28 is achieved by the combination with *Borda count*. Compared to the results obtained for the other two word classes in WebCAGe, this score is comparable to the best result for nouns in WebCAGe (which is 55.92) while much better than the best result for verbs in WebCAGe (which is 45.97). Furthermore, the combined methods outperform the random sense baseline for all performance measures.

In general, precision above 63% (for several individual methods when evaluated on adjectives in WebCAGe) is outstandingly high among the results of

all corpora and word classes. That is, for adjectives, nouns, and verbs (see Tables 7.3, Tables 7.1, and 7.2) hardly any method achieves a comparably high precision. On the other hand, the most obvious problem for adjectives in WebCAGe is low coverage, and therefore also low recall, which depends on coverage. However, the outstandingly high precision compensates for the low coverage and recall insofar as the combined algorithms achieve overall competitive F-scores above 55.

Results for Adjectives in deWaC

The WSD results for adjectives in deWaC are shown in the bottom of Table 7.3. As already found for verbs in deWaC, none of the knowledge-based algorithms – not even the combined classifiers – is suited for disambiguating adjectives in deWaC. All algorithms produce very poor results: most single relatedness measures achieve F_1-scores between 18 and 27 – with little variance. The exceptions are the three gloss-based measures *lesk-Wiktionary*, *lesk-lexfields*, and *lesk-all-info*, which outperform all other algorithms (including the combination algorithms) with a still very low score of 40. Again (as for verbs in deWaC), the most obvious explanation for this low performance is the relatively high polysemy for adjectives in deWaC and the general inapplicability – as already discussed above – of path-based and information-content-based relatedness measures for adjectives and verbs.

As for verbs in deWaC, a direct comparison to previously published experiments on adjectives in deWaC is not possible, because Broscheit et al. [2010] do not report separate results for adjectives. However, as explained in the above discussion on verbs in deWaC, the results reported by Broscheit et al. [2010] implies that the performance for the disambiguation of adjectives and verbs is much worse than the performance for nouns. The finding that the disambiguation of adjectives is much more difficult than for nouns corroborates the results by Broscheit et al. [2010] on German WSD.

With the setup applied in Henrich and Hinrichs [2012], where individual results – depending on whether they lie above or below a certain threshold – get scaled up or down, respectively, the WSD algorithms would perform sig-

nificantly better on adjectives in deWaC. However, as already outlined in Subsection 7.2.4 above, the main reason why the experiments in this chapter do not utilize thresholds is to keep the setup purely knowledge-based and without any training bias.

Despite the poor overall results, Table 7.3 reveals three positive findings concerning the disambiguation of deWaC adjectives. Firstly, most WSD algorithms outperform the 18.89 F-score of the random sense baseline. Secondly, all except two algorithms achieve a perfect coverage of 100% – with the minor exceptions of *hso* and *lesk-GermaNet*, though they still achieve good coverage of 92.22% and 86.67%, respectively. This is much different in comparison to the coverage of adjectives in WebCAGe, which is significantly lower for most relatedness measures. Thirdly, and most importantly, the alignment of GermaNet and Wiktionary has a very large positive impact on the WSD results. While *lesk-GermaNet* achieves only an F-score of 20.24, the gloss-based methods that employ Wiktionary information (particularly *lesk-Wiktionary*, *lesk-lex-fields*, and *lesk-all-info*) double the result to a score of 40. This large improvement is one of the most striking findings among the results of all corpora and word classes that the alignment with Wiktionary reveals.

7.3.4 Profiling Context Window Sizes

As mentioned in Subsection 7.2.1 above, knowledge-based WSD experiments are conducted with context windows between 1 and 50 word tokens. Figure 7.2 shows the results for these context window sizes for the overall consistently best performing measure from the previous three subsections, i.e., for the combined algorithm *Borda count*. The figure plots the performance results in terms of the F_1-score (on the y-axis) for each word class and gold standard corpus (the distinct curves) for several context window sizes (on the x-axis). Note that the x-axis is skewed in that it includes single steps for context windows between 1 and 15 word tokens, but 5-word steps for context windows between 15 and 50 tokens.

Figure 7.2 shows clearly that larger context windows improve disambiguation performance across word classes and gold standard corpora. This finding

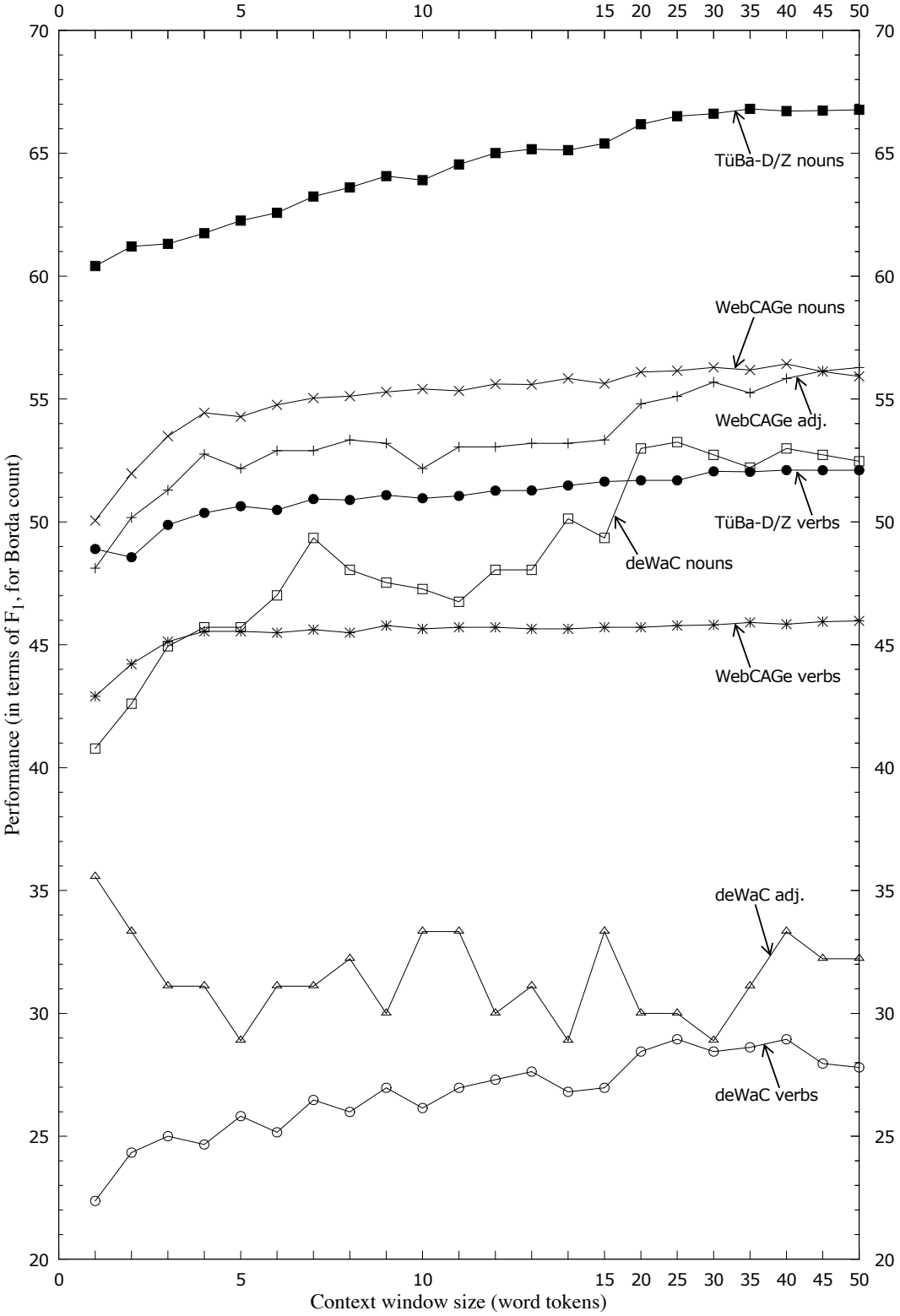

Figure 7.2: Impact of context window size on knowlege-based WSD.

corroborates the results by Pedersen et al. [2005] on English WSD. For all except one word class–corpus combination, the plotted curves in Figure 7.2 generally raise consistently: some results improve significantly, others improve only minimally or with some local fluctuations, but they clearly improve with larger context window sizes. The only exception are adjectives in the deWaC corpus, for which the performance is strongly varying for different window sizes. The explanation for this unsteady curve is that there are only few sense-annotated adjectives in the deWaC gold standard.

Since the tendency that larger context windows improve disambiguation performance across word classes and gold standard corpora is clear (with only one exception), the results reported in the previous three subsections (Subsections 7.3.1 to 7.3.3) used a context window of 50 tokens. More specifically, they use a window of up to 50 lemmatized tokens from the context surrounding the target word. Since the semantic relatedness measures can be calculated only for word lemma entries in GermaNet, the context window contains only lemmas covered by GermaNet. As described in Subsection 7.2.1 above, the 50-token context windows are not restricted to sentence boundaries and the target word occurs in the middlemost position, if possible.

However, for those annotations where only a restricted context is available, the context windows include less than 50 word lemmas. These restricted contexts are particularly frequent for WebCAGe, where most example sentences harvested from Wiktionary itself (which constitute about 73% of all sense annotations in WebCAGe – see Subsection 6.3.1, Table 6.2) are single sentences only without a larger context. For altogether 73.8% of all sense annotations in WebCAGe, the context window contains less than the desired amount of 50 lemmas. The average number of words contained in WebCAGe's context windows is 16.1 only (12.0 for adjectives, 21.3 for nouns, and 6.4 for verbs). These restricted contexts cause lower coverage for the WSD experiments reported for the WebCAGe corpus in the previous subsections.

By contrast, all sense annotations in TüBa-D/Z and deWaC have sufficiently large contexts. In the TüBa-D/Z, only 6.3% of all sense annotations contain context windows with less than 50 tokens. This corresponds to an average of 48.8 words for the treebank's context windows. The available con-

text window sizes for deWaC are comparable to those in the TüBa-D/Z. That is, 4.8% of the sense annotations in deWaC have a context window with less than 50 tokens, which corresponds to an average of 49.1 words in the context windows. The sufficiently large contexts available for TüBa-D/Z and deWaC is reflected in the consistently high coverage for the WSD experiments on these two corpora in the previous subsections.

7.4 Summary and Conclusion

The present chapter has explored a wide range of knowledge-based WSD algorithms for German. These algorithms are based on semantic relatedness measures, including path-based, information-content-based, and gloss-based methods. In addition, combined algorithms have been investigated and compared in performance to the individual algorithms. WSD experiments have been run on all annotations from the three corpora WebCAGe, TüBa-D/Z, and deWaC – for the three word classes of adjectives, nouns, and verbs.

The evaluation in Section 7.3 has shown that random sense baselines are outperformed by most algorithms for all sense-annotated corpora and word classes. Table 7.4 summarizes the eight different evaluation results, i.e., for three word classes evaluated on three sense-annotated corpora (excluding adjectives from the TüBa-D/Z for which there are no sense annotations available). For all sense-annotated corpora, it lists – separately for each word class – best and worst results in terms of F-scores[1] together with the corresponding WSD algorithms that achieved the respective scores. The table also indicates whether or not the combination of single relatedness measures has improved the overall WSD results and in what way the alignment of GermaNet with Wiktionary has an effect on the performance.

Among the three word classes considered, the performance of knowledge-based WSD algorithms is better for nouns than for adjectives and for verbs. In general, while semantic relatedness measures perform well for nouns, they are little suitable for disambiguating adjective and verb senses. This finding

[1]F-score values are rounded to the nearest integers.

Table 7.4: Summary of the knowledge-based evaluation results.

	POS	Best F-scores	Worst F-scores	Combination	Wiktionary alignment
WebCAGe	Adj.	56 *Borda count* 56 *majority voting* 44 *lesk-all-info*	26 *jcn* 27 *path* 29 *lesk-GermaNet*	helps a lot*	helps a lot*
	Nouns	56 *Borda count* 55 *majority voting* 53 *lesk-GermaNet*	47 *path* (generally path- and IC-based)	helps†	only in terms of coverage§
	Verbs	46 *Borda count* 43 *majority voting* 43 *lesk-all-info* (gen. gloss-based)	18 *wup* 19 *res*, *lin*, *hso* 25 *path*, *jcn* 26 *lch*	helps†	helps a lot*
TüBa-D/Z	Nouns	67 *lesk-Wiktionary* 67 *Borda count* (gen. gloss-based)	25 *path* 27 *res* 29 *lch*	does not help‡	helps a lot*
	Verbs	56 *jcn* 52 *Borda count* 52 *majority voting* 49 *lesk-all-info*	38 *path* 38 *lesk-glosses* 39 *lch* 40 *lesk-GermaNet*	does not help‡	helps a lot*
deWaC	Adj.	40 *lesk-Wiktionary* 40 *lesk-lex-fields* 40 *lesk-all-info*	19 *lch* 20 *lesk-GermaNet* 21 *lin*	no help at all◇	helps a lot*
	Nouns	55 *majority voting* 52 *Borda count* (gen. IC-based)	38 *lesk-Wiktionary* 39 *path* 40 *lesk-GermaNet*	helps a lot*	does not help‡
	Verbs	28 *Borda count* 27 *lesk-Wiktionary* 26 *majority voting*	16 *res* 17 *hso*, *lin* 19 *wup*	helps†	helps†

**helps a lot* := improvement by a wide margin.
†*helps* := marginal to minimal improvement only.
‡*does not help* := marginal to minimal decrease.
◇*no help at all* := decrease by a wide margin.
§*only in terms of coverage* := when considering individual gloss-based algorithms, the F-scores have decreased, but coverages improved.

generally corroborates the results reported by Pedersen et al. [2005] for English WSD. A reason might be that the noun hierarchy in GermaNet (as in WordNet) is denser than for the other two word classes.

As summarized in Table 7.4, the best performing WSD algorithms for adjectives and verbs are solely gloss-based and combined algorithms. According to Pedersen et al. [2005, page 27], a low disambiguation performance of

path-based and IC-based measures for verbs was to be expected because these measures were originally meant for nouns rather than for verbs. Their explanation is that since the WordNet hierarchies for verbs are very shallow compared to the hierarchies for nouns, path-based and information-content-based measures are much more effective for nouns while more of an experimental nature when applied to verbs. Since the adjective and verb hierarchies in GermaNet have similar density as those of English verbs (when compared to German and English nouns), this suggests that Pedersen et al.'s [2005] explanation also holds true for German WSD. The knowledge-based disambiguation of adjectives differs from the experiments by Pedersen et al. [2005] for English since path-based and information-content-based algorithms are available only for German adjectives but not for English ones (when using Princeton WordNet to calculate semantic relatedness). The reason is that the Princeton WordNet does not structure adjectives hierarchically (as explained in Section 3.1), which precludes the use of relatedness measures based on hypernymy paths. By contrast, GermaNet encodes adjectives in a manner comparable to nouns and verbs in a hypernymy hierarchy, and, thus, allows experimenting with the same set of relatedness measures as for the other two word classes.

In seven out of the eight evaluation results in Table 7.4, the combination by *Borda count* is among the best performing algorithms; and in four of the eight results, *Borda count* achieves the very best scores. The only exception is adjectives in deWaC, for which a combination does not help at all. In six out of the eight evaluation results, the other combined algorithm (*majority voting*) is the algorithm that occurs second most commonly in the list of best performing algorithms. No other method is so often among the best performing algorithms. This finding proves that combined methods – in particular *Borda count* – generally perform stronger and more consistent than single WSD algorithms. This result contradicts the previous finding of Broscheit et al. [2010] who did not obtain better results by combining individual WSD algorithms.

The alignment between GermaNet and Wiktionary has an even more remarkable positive impact on the performance of gloss-based WSD algorithms than the combination of algorithms. Although it depends on the exact evaluation setup (i.e., which word class is considered in which corpus), the use of the

aligned Wiktionary information considerably improves the WSD performance. This answers in the affirmative one of the leading questions for the research reported here, namely, whether the GermaNet–Wiktionary mapping improves the performance of knowledge-based WSD.

Although GermaNet's coverage of definitions – even with the harvested definitions from Wiktionary – is far from complete, the gloss-based measures have generally outperformed path-based and information-content-based measures. This finding that word overlap methods perform well corroborates the results reported by Pedersen et al. [2005] for their experiments on English WSD. It indicates that words from glosses or lexical fields of the appropriate target word senses often also occur as context words.

Comparing the results for the three sense-annotated corpora, the WSD results obtained for the TüBa-D/Z treebank are the best, the results obtained for WebCAGe second, while the results for deWaC are the worst. The most apparent reason for the bad performance on the deWaC corpus is the much higher polysemy of annotated words, which makes the sense disambiguation more difficult. That is, while the average polysemy of sense-annotated words in WebCAGe and TüBa-D/Z is 3.1 and 3.2, respectively, the average polysemy of sense-annotated words in deWaC is 5.7 and thus much higher. Another reason for a better performance when evaluated on the TüBa-D/Z is related to the nature of the corpus texts: data from the web – which is the basis for the WebCAGe and deWaC corpora – is apparently noisier or for other reasons harder for WSD algorithms than newspaper texts – which is the basis for the TüBa-D/Z.

The largest deviation when comparing the three corpora are coverage results, which are particularly low for several (particularly path-based and information-content-based) WSD algorithms when evaluated on WebCAGe. This is hardly surprising when considering the nature of the corpus texts. That is, WebCAGe contains several annotations for which only a restricted context is available (i.e., the example sentences harvested from Wiktionary itself are single sentences only without a larger context – see Section 5.2), whereas all of the sense annotations in TüBa-D/Z and in deWaC have sufficiently large contexts. The availability of context was analyzed in Subsection 7.3.4.

In general, most algorithms perform better than random guessing. Apart from the low results obtained for adjectives and verbs in deWaC, the evaluation results in this chapter prove the viability of the proposed knowledge-based approach for German word sense disambiguation.

In order to boost the performance of such a knowledge-based WSD system, natural next steps would be (i) to experiment with different parameter settings of the algorithms used in this chapter, (ii) to employ different corpora for obtaining word frequency for calculating information content – similarly to the experiments by Patwardhan et al. [2003] – and (iii) to implement further state-of-the-art WSD algorithms such as the Personalized PageRank algorithm [Agirre and Soroa, 2009], which yielded the best results for German WSD reported by Broscheit et al. [2010][1]. Further, a very simple way to achieve perfect coverage and therefore also to improve the disambiguation results is to use a backoff strategy – such as choosing a sense at random for cases where the WSD algorithms are unable to assign any word sense – which has shown to work well in the experiments by Broscheit et al. [2010] and Miller et al. [2012]. Many existing knowledge-based WSD systems suffer from the problem of sparse knowledge (see Subsection 2.3.1), so it seems worthwhile to try Miller et al.'s [2012] approach to overcome this problem by expanding contexts and word senses with distributionally similar words.

However, even if the knowledge-based approach can be improved, results can be boosted only to a certain extent. Since supervised systems generally perform much better than knowledge-based approaches to WSD [Kilgarriff and Rosenzweig, 2000], the following chapter focuses on WSD experiments using supervised machine learning methods. While the knowledge-based results for disambiguating nouns reported in this chapter are already good, it is even more important to improve the very low results for adjectives and verbs. Therefore, the following chapter explores several features that represent syntactic properties. It shows that – for all word classes – a supervised system performs far better than the knowledge-based approach presented in the present chapter.

[1]However, the study by Henrich and Hinrichs [2012], for example, has found that the use of the Personalized PageRank algorithm performs considerably lower than any of the combined algorithms and many of the single algorithms in their setup.

Chapter 8

WSD Using Supervised Machine Learning Methods

As already pointed out in the introduction of the previous chapter, the purpose of this chapter is to help close the gap of sparse research on German word sense disambiguation. While the previous chapter investigated knowledge-based WSD, this chapter focuses on German WSD using supervised machine learning methods. In general, supervised approaches to WSD adapt supervised machine learning methods to solve the task of assigning the correct sense to a word.

The task at hand is considered as a classification problem, where the *class* that needs to be predicted is the corresponding word sense (from GermaNet) for a given context. In order to learn how to predict the corresponding word senses for unseen words, supervised machine learning methods rely on corpora whose words are already annotated with senses from a given sense inventory. Each such annotated word occurrence is denoted as an *instance*. A certain amount of sense annotations serves for training a supervised method how to predict the correct senses for unseen word occurrences; and another set of sense annotations is used to evaluate the performance of the automatic disambiguation prediction (see Chapter 6 for a description of sense-annotated corpora for German, especially Section 6.1 on training and test sets for supervised WSD systems).

The contexts of these annotated words provide linguistic clues specific to particular senses – such as morphological information for the target word, structural information from the sentence, or co-occurring words or word classes. Supervised WSD systems use these clues – referred to as *features* – in order to disambiguate between word senses. Thus, sense-annotated word instances are represented by corresponding features. The concrete features employed for the WSD experiments in this chapter are described in Section 8.1 below.

Given a set of classified instances, i.e., instances with assigned word senses and corresponding features, a classification algorithm learns how to predict the corresponding word sense for an unseen instance. For the experiments in this chapter, the classification algorithms – also called *classifiers* – are taken from the Weka machine learning tool suite [Hall et al., 2009] (see Section 8.2 for more details). Experiments include rule-based methods, instance-based methods, probabilistic methods, support vector machines, and combined approaches – as described in Section 8.2 below.

In short, this chapter has the following four main goals:[1]

(i) To apply a wide range of supervised WSD algorithms to German – including the three word classes of adjectives, nouns, and verbs – since the range of methods that has thus far been applied to this language is rather limited (as described in Chapter 2).

(ii) To study the impact of several heterogeneous machine learning features on the automatic disambiguation of German word senses.

(iii) To investigate the influence of syntax and semantics on WSD which is particularly interesting for verbs where the syntactic structure in which a verb occurs is often highly predictive of different word senses.

(iv) To evaluate and compare the performance of supervised WSD algorithms on three heterogeneous sense-annotated corpora.

The following Section 8.1 outlines the variety of employed machine learning features. The Weka machine learning tool suite and the set of supervised

[1]Note that goals (i) and (iv) are analogous to the goals of the previous chapter on knowledge-based WSD for German.

machine learning classifiers that are used for the WSD experiments are described in Section 8.2. The evaluation of the supervised WSD experiments in Section 8.3 focuses on several aspects, including a comparison of several heterogeneous machine learning classifiers, a detailed analysis of the implemented machine learning features, and an investigation of the influence of syntax and semantics on the disambiguation performance for verbs.

8.1 Machine Learning Features

The contexts of the ambiguous words to be disambiguated provide linguistic clues specific to particular senses. Supervised WSD systems use these clues – referred to as *features* – in order to disambiguate between word senses. A *feature* – often synonymously referred to as an *attribute*[1] – is a distinct bit of information that encodes linguistic clues from the context of a target word [McCarthy, 2009], such as morphological information for the target word, structural information from the sentence, or co-occurring words or word classes. The value of a feature is specific to the word instance it represents.[2]

This section describes the features implemented for the WSD experiments in this chapter. The large amount and variety of implemented features sets the presented work apart from related work. The main reason for putting so much effort into the implementation of features is due to the experience of previous works: Yarowsky and Florian [2002], inter alia, found that the impact

[1] In the machine learning terminology, which is adopted for this thesis, the terms *feature* and *attribute* are often used synonymously while the term *feature* is prevalent (see, for example, Weiss and Kulikowski [1991, page 5] or Màrquez et al. [2006, page 169]). Even in the official Weka book [Witten et al., 2011, page 49] the two terms are sometimes used interchangeably although there the term *attribute* is used predominantly. Note however, that the expressions *feature* and *attribute* are distinguished by some researchers; in that case attributes generally define properties and their set of possible values while features are concrete realizations of attribute-value pairs assigned to instances [Kohavi and Provost, 1998, page 271].

[2] Supervised classification in Weka requires training and test data to conform to a certain data format, in which instances are listed with appropriate feature values. Weka supports two types of input formats. In this implementation, the *Attribute-Relation File Format* (*ARFF*) is used. The set of features is shared by all instances in such a file, but the concrete value of a feature depends on the corresponding instance (i.e., sense annotation) it represents. [Witten et al., 2011]

of features on the performance of WSD systems is significantly greater than the impact of the applied classification algorithms; and Lee and Ng [2002] experienced better performance when combining several types of features rather than employing only one single knowledge source.

Table 8.1 gives an overview of all implemented features. The leftmost column (*Grp.*) groups the features depending on the type of information they employ, such as information on the word's surface forms, co-occurring lemmas, part-of-speech, or morphology. The third column denotes the features' data type as either *nominal* (abbreviated as *nom.*), *numeric* (abbreviated as *num.*), *boolean* (abbreviated as *bool.*), or *string*.[1]

Several features are applicable to all word classes, while other features are applicable to certain word classes only. The information on which features are implemented for which word classes is specified in column *Applicable for POS*. Finally, the last column gives example values for a feature in question.[2]

Table 8.1: Machine learning features.

Grp.	Feature name	Type	Applicable for POS	Example values
Surface	word_form	string	all	Fuß, Füßen, Fuße, ...
	last_3_chars			Fuß, ßen, uße, üße, ...
	last_2_chars			uß, en, ße, es, ...
	separated_particle	bool.	verbs	yes, no

Continued on next page

[1]For technical reasons, only nominal and numeric feature types are used in the actual WSD experiments. Therefore, boolean- and string-typed features are represented as nominal. That is, boolean features are represented as nominal features with the two values *true* and *false*, while string features are converted to nominal features after collecting all input data. The difference between nominal and string types (in Weka) is that nominal features have a predefined value set (which is the same for all lemmas) while string features can be arbitrary strings, which might be different for each lemma. Once all features are collected, such string features can easily be converted into nominal.

[2]Example values are sometimes abbreviated for reasons of space.

Table 8.1 Machine learning features (continued)

Grp.	Feature name	Type	Applicable for POS	Example values
Context lemmas	ctx_lemmas_sent_bool ctx_lemmas_sent_6_bool ctx_lemmas_50_bool	bool.	all	yes, no
	ctx_lemmas_sent_num ctx_lemmas_sent_6_num ctx_lemmas_50_num	num.	all	<number>
POS	pos_1_left pos_2_left pos_3_left pos_1_right pos_2_right pos_3_right	nom.	all	one STTS POS tag, e.g.: ADJA, ADJD, ADV, ART, APPR, CARD, FM, ITJ, KON, NN, NE, PDS, PIS, PRF, ...
	pos	nom.	adj., verbs	ADJA, ADJD, ...
	verbs_pos	string	adj., nouns	VMFIN_VAINF, ...
	verbs_pos_ignoring_aux			VVINF_VMFIN, ...
Morphological	morph_number	nom.	all	singular, plural
	morph_case	nom.	adj., nouns	nom., gen., dat., acc.
	morph_gender			masc., fem., neuter
	morph_person	nom.	verbs	first, second, third
	morph_mood			indicative, subjunc.
	morph_tense			present, past
Context details	sentence_length	num.	all	<number>
	adjective	bool.	nouns	yes, no
	article	nom.	nouns	def., indef., mein, ...
	adposition	string	nouns	zu, in, bei, mit, an, ...
	verbs	string	adj., nouns	wollen%aux_holen...
	verbs_ignoring_aux			holen, hören, malen...
	auxiliary_verb	string	verbs	sein, haben, sollen, ...
Sentence structure	sentence_type	nom.	all	initial, second, final
	head passive	bool.	all	yes, no
	nx_length	num.	nouns	<number>
	part_of_conjunction	bool.	nouns	yes, no
	grammatical_function	nom.	nouns	OA, ON, OD, OPP...

Continued on next page

Table 8.1 Machine learning features (continued)

Grp.	Feature name	Type	Applicable for POS	Example values
Constituent structure	has_ON has_OA has_OD has_OG has_FOPP has_OPP has_OS has_OV has_OADJP has_OADVP has_PRED has_ES	bool.	verbs	yes, no
Verbal frames	NE_confidence DR_confidence AR_confidence AN_confidence DN_confidence GN_confidence AZ_confidence AI_confidence NG_confidence DS_confidence FSo_confidence FSw_confidence FS_confidence PP_confidence Pp_confidence NN_confidence	num.	verbs	<number>
Other	headline named_entity	bool.	all	yes, no
	translation	string	all	mrs, woman, wife, ...

The features all employ different kinds of linguistic information (i.e., knowledge sources), such as information on surface forms of words, co-occurring lemmas, POS tags, morphology, or sentence structures. The sources for this

information are twofold: (i) for the TüBa-D/Z treebank, manual linguistic annotations as described in Section 5.3.1 serve as input, while (ii) for all three corpora (TüBa-D/Z, WebCAGe, and deWaC) automatic annotation tools are employed. There are two main reasons why the automatic annotation is also performed on TüBa-D/Z's texts, although the treebank already contains manual annotation for all linguistic phenomena in question. Firstly, to allow a direct comparison of the WSD results for all three corpora with the same compilation of linguistic details. Secondly, to study the influence of linguistic annotation quality (manual versus automatic) on the WSD results.

The automatic annotations are outlined in the following subsection, before the subsequent subsections describe all features in detail.

8.1.1 Automatic Linguistic Preprocessing

For both knowledge-based and supervised word sense disambiguation experiments, the three gold standard corpora are automatically split into sentences, tokenized, and lemmatized – as described in Section 6.6. In addition to these annotations, several supervised machine learning features require part-of-speech tags, morphological information, syntactical structures, and translations to English. Therefore, the following annotation tools are employed:

Part-of-speech tagging POS tagging is performed by the TreeTagger [Schmid, 1994]. Its probabilistic tagging method represents probabilities of tagged sequences of tokens with Markov Models, and uses binary decision trees for estimating transition probabilities. With a modified version of the ID3 algorithm [Quinlan, 1983], the decision trees are recursively constructed from training data. The Viterbi algorithm [Viterbi, 1967] is then used to decide the best tag sequence for a sequence of tokens. The official parameter file for German is trained for the STTS tagset. For the implementation, a Java wrapper around the TreeTagger is employed.[1]

[1]Helmut Schmid developed the TreeTagger. The tagger and several parameter files, including the one for German used in this work, are available at `http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/`. Richard Eckart de Castilho built a Java wrapper around TreeTagger, which is available at `http://code.google.com/p/tt4j/`.

Morphological analysis Morphological information is annotated with RFTagger [Schmid and Laws, 2008]. The tagger annotates tokens with fine-grained part-of-speech tags which encode morphological information. The fine-grained tagset is an enriched version of the STTS tagset. The morphological information available for adjectives, for example, includes *case* (*nominative*, *genitive*, *dative*, and *accusative*), *number* (*singular* and *plural*), *gender* (*feminine*, *masculine*, and *neuter*), and *degree* (*comparative*, *positive*, and *superlative*). For verbs, the tagset includes information on *person* (*1*, *2*, and *3*), *number* (*singular* and *plural*), *tense* (*past* and *present*), *mood* (*indicative* and *subjunctive*), and *type* (*auxiliary*, *modal*, or *full*) – to give but two examples.

The implementation of RFTagger is based on Hidden Markov Models. It first splits all part-of-speech tags into attribute vectors and decomposes contextual POS probabilities of the Markov Model into products of attribute probabilities. It then uses decision trees for estimating conditional probabilities of these attributes. The official RFTagger download[1] includes a German model trained on the Tiger treebank [Brants et al., 2004]. In the implementation, the Java interface to the RFTagger[2] is used.

Parsing The Berkeley Parser[3] [Petrov et al., 2006] assigns the most likely parse trees to input texts by learning probabilistic context-free grammars (PCFGs). The parser follows a split-and-merge approach. It starts by learning a PCFG from a heavily reduced initial structure. Repeatedly, non-terminals are divided into two non-terminals. For each such splitting, the likelihood loss for merging the two non-terminals (i.e., for undoing the splitting) is calculated. When this likelihood loss is little, the two non-terminals are merged. As a result, the splitting facilitates a good adaptation to the trained input and the merging allows more

[1]Helmut Schmid and Florian Laws developed RFTagger, available at `http://www.cis.uni-muenchen.de/~schmid/tools/RFTagger/`.

[2]Niels Ott and Ramon Ziai developed the Java interface to RFTagger, available at `http://www.sfs.uni-tuebingen.de/~nott/rftj-public/`.

[3]Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein developed the Berkeley Parser, available at `http://code.google.com/p/berkeleyparser/`.

compact grammars with less rules compared to models for other parsers (including Collins' and Charniak's parsers [Collins, 1999; Charniak, 2000] and the Stanford Parser [Klein and Manning, 2003]).

In order to produce parse trees comparable to the manual TüBa-D/Z treebank annotations, the Berkeley Parser is trained on the TüBa-D/Z data from release 9.0.[1] This also facilitates the parser to structure sentences according to topological sequences, which are widely used in German syntax.[2]

English translation Determining the English translation of a German word in its specific context cannot make use of one translation tool out-of-the-box. Rather, more complex preprocessing steps than for the other automatic linguistic annotations are necessary.

Step 1: Three different approaches generate lists of English candidate translations for German target words: (i) extracted from GermaNet's ILI records[3], (ii) gained from the dict.cc German–English translation list[4], and (iii) produced with an existing machine translation tool, the Microsoft Translator API[5]. The three implementations (i) to (iii) are applied to the German target words and to their synonyms for all GermaNet senses of these words. This procedure generates for each German target word a set of candidate translations.

Step 2: The sentence in which the target word occurs is translated into English with the Microsoft Translator API[6]. As part of the Microsoft Translator[7], this API offers an online translation service using statisti-

[1]Thanks to Daniël de Kok, Eyal Schejter, and Yannick Versley for their support on training the Berkeley Parser.

[2]For more details on the general idea of topological fields in the TüBa-D/Z, see Subsection 5.3.1.

[3]See Section 3.5 for a description of ILI.

[4]dict.cc is an online translation dictionary for different language pairs, including German–English. The German–English translation list is provided at `http://www1.dict.cc/translation_file_request.php?l=e`.

[5]The Microsoft Translator API is described in step 2 below.

[6]The Microsoft Translator API is available from the Windows Azure Marketplace: `http://datamarket.azure.com/dataset/bing/microsofttranslator`. The translation of up to 2 000 000 tokens per month is provided for free.

[7]See `http://api.microsofttranslator.com`.

cal machine translation. It supports several language pairs, including German–English. For the implementation, a Java wrapper is employed[1].

Step 3: With the help of the three candidate translation lists created in step 1, the translation of the German target word is identified heuristically in the English sentence translated in step 2. In the simplest case, a translated word occurs in the same position than the target word in the original sentence. That is, when the translated word at the corresponding target word position occurs in at least one of the candidate translation lists, that word is defined as the translation in the specific context.

However, due to language-specific syntactical structures and phrasal word orders, a translated sentence is seldom an exact word-for-word translation. The position of a target word in the original German sentence often differs from the position of the equivalent English word in the translated sentence. In these cases, the translated sentence is searched for matching words in the candidate translation lists from step 1 to identify potential translations of the target word in the specific context. A heuristic, which takes the distance of the potential translated word's position compared to the position of the original word into account, decides which translation to use in the specific context in cases where multiple potential candidate translations are found in the English sentence.[2]

The following subsections describe all features in detail – grouped by the type of linguistic information they employ.

[1]Jonathan Griggs developed the Microsoft Translator Java API – a Java wrapper around the Microsoft Translator API, available at `https://code.google.com/p/microsoft-translator-java-api/`.

[2]Certainly, more sophisticated machine translation approaches might improve the quality of the target word translation. However, the described implementation works well for the purpose of a translational feature.

8.1.2 Surface Features

The feature group 'surface' subsumes four features, all relying on the written surface form of the target word itself:

word_form This feature represents the surface forms of the words themselves as strings. It thus comprises information on capitalization, morphological endings, and modification of German umlauts. For the lemma *Fuß*, for example, there might be the value set of *{Fuß, Füßen, Fuße, Füße, Fußes, FÜSSE}*.

last_3_chars The last three characters of a target word serve as a surface feature which, in a way, mainly includes morphological endings. The value set for the example lemma *Fuß* contains *{Fuß, ßen, uße, üße, ßes, SSE}*.

last_2_chars Similarly, the last two characters form a separate feature. This feature clearly very often encodes the same set of information as the previous feature 'last_3_chars'. Example values for the lemma *Fuß* are *{uß, en, ße, es, SE}*.

separated_particle This boolean-valued feature indicates whether the occurrence of a separable verb appears as one word or with its particle separated. For the specific class of German separable verbs including a particle – also referred to as *particle verbs* – the particle can occur separately from the lexical verb base. For example, the particle *hoch* 'up' in a verb such as *hochladen* 'to upload' can appear with the verb as one word (see sentence (7a)) or occur separated from the verb base (as in sentence (7b)).

(7) a. *Im Webportal kann jeder Benutzer Fotos* ***hochladen****.*
('Every user can upload pictures to the web portal.')

b. *Im Webportal* ***laden*** *viele Benutzer Fotos* ***hoch****.*
('Many users upload pictures to the web portal.')

In the used gold standard corpora, separated particles can be identified by their POS tag *PTKVZ*.

Since different surface forms and endings occur for words of all word classes, the first three features listed are applicable for all relevant word classes, i.e., adjectives, nouns, and verbs. The applicability of the last feature listed is restricted to particle verbs, because in German only particle verbs can have separated particles.

8.1.3 Context Lemma Features

All features in this subsection make use of information on the occurrence of lemmas in the contexts of the target words. In order to keep the number of lemmas manageable and to ensure a certain level of meaningfulness and generalizability of the created features, only lemmas with a certain frequency are considered. In the experiments reported in Section 8.3, lemmas which occur at least three times overall in all contexts of the target words are considered. Punctuation marks, identified by their STTS POS tag starting with a dollar symbol ($)[1], and stopwords such as determiners, identified with the help of a given list[2], are ignored for all features listed below.

The features in this section are available in a numeric and in a boolean-valued variant. The numeric variant encodes the frequency with which the lemma occurs in the context of the target word under consideration, whereas the boolean variant simply states whether or not the lemma occurs at least once in the context.

Since such contextual information is available for all tokens in a corpus, the features are applicable to all word classes.

ctx_lemmas_sent_bool This boolean-valued feature encodes whether or not a lemma occurs in the same sentence as a target word.

ctx_lemmas_sent_num The numeric variant of the previous feature specifies the number of occurrences with which a lemma occurs in the sentential context of a target word.

[1]The three STTS tags for punctuation marks are '$.' for sentence-final punctuation, '$,' for commas, and '$(' for other sentence-internal punctuation.

[2]The stopwords list is copied from the Snowball stemmer [Porter, 1980], available at `http://snowball.tartarus.org/algorithms/german/stop.txt`.

ctx_lemmas_sent_6_bool This feature is similar though more restrictive than the first one in the list. It also encodes whether a lemma occurs in the context of a target word or not, but this time the considered context is not the whole sentence but a maximum window of six tokens (three on each side of the target word) within the boundaries of a sentence.

ctx_lemmas_sent_6_num Analogously, the frequency of a lemma in the restricted context window of six words is represented in the numeric variant of the feature.

ctx_lemmas_50_bool Different from the four previously listed features, the context of this feature is not restricted to tokens within the sentence in which a target word occurs. All lemmas which occur in a context window of 50 words beyond sentence boundaries are considered.

ctx_lemmas_50_num This feature represents the frequency-based version of the 50-word context feature.

Technically, there are more than six 'context lemma' features. For each lemma (excluding stopwords), which occurs overall at least three times in the specified context windows of all word instances, there is a separate feature implemented.

In general, arbitrary context window sizes could be chosen. The decision to report experiments for the above-listed context sizes supports a comparison of heterogeneous contexts windows, i.e., a very small window size of at most six tokens, a relatively variable-in-size context representing a sentence, and a large window of 50 tokens.

Probably because these 'context lemma' features have proven to perform well, are easy to obtain and applicable to all words and word classes, they are popular in the literature [Lee and Ng, 2002; Martínez et al., 2002; Mihalcea, 2002b; Kopeć et al., 2012].

8.1.4 Part-of-Speech Features

All nine part-of-speech features make use of the Stuttgart-Tübingen-TagSet (STTS) [Schiller et al., 1999], which comprises a total set of 54 POS tags. The

sentence

(8) *Information will **frei** sein, heißt es.*[1]
('Information wants to be free, it says.')

from Figure 8.1 with the target word *frei* 'free' (rendered in boldface in example sentence (8); surrounded by a dashed box in Figure 8.1) serves as an illustrating example for the features in this group.

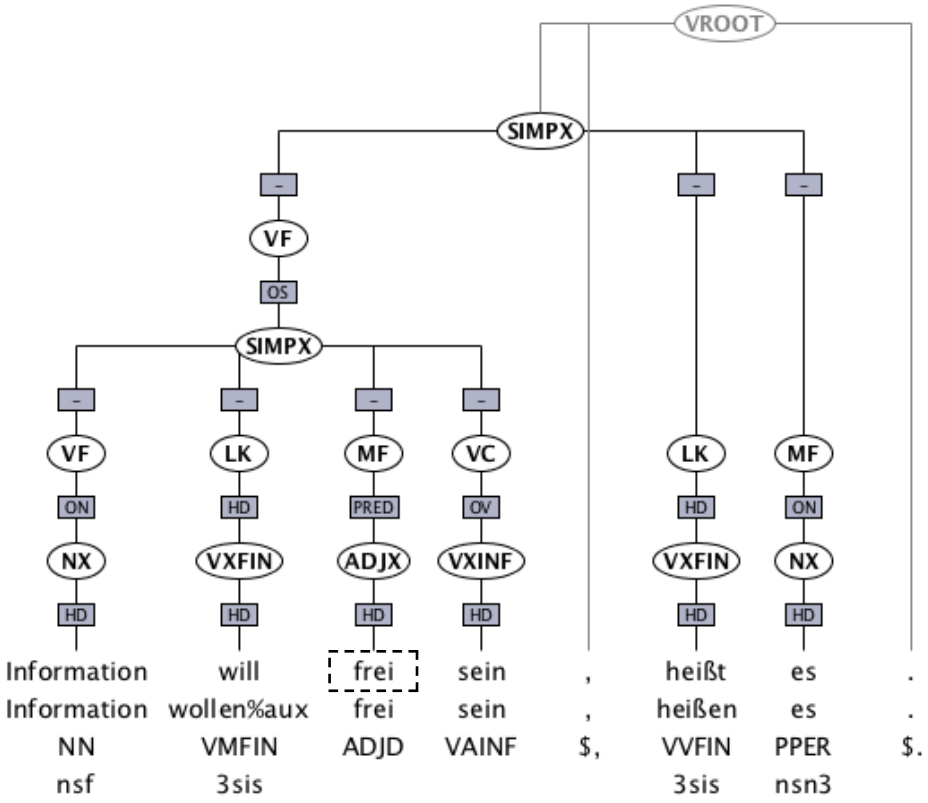

Figure 8.1: Example sentence for illustrating POS features.

Since the first six features listed below rely on contextual information, which is generally available for all tokens in a corpus, they are applicable to all word classes. Only tokens within the sentence in which the target word occurs are considered.

[1]Sentence 50 865 from TüBa-D/Z 9.1.

pos_1_left This feature encodes the STTS part-of-speech tag of the token occurring immediately to the left of the target word. For the target word *frei* in the example sentence in Figure 8.1, the extracted POS tag value of the token to the left (i.e., *will*) is *VMFIN*.

pos_2_left Analogously, this feature encodes the STTS POS tag of the token two tokens to the left of the target word. In the example, the token two to the left of the target word *frei* is *Information*, and thus the extracted POS tag is *NN*.

pos_3_left The part-of-speech tag of the token three to the left of the target word is represented by feature 'pos_3_left'. In the example sentence in Figure 8.1, there is no token three to the left which belongs to the same sentence than the target word *frei*. In this case, the value of this feature is left empty.

pos_1_right Along the lines of the first three POS features listed, which encode the POS tags of the tokens in the left context of the target word, there are three such nominal features for the right context. 'pos_1_right' represents the POS tag of the token following the target word. In the example, this is tag *VAINF*.

pos_2_right This feature encodes the part-of-speech tag of the token two tokens to the right of the target word. In the example, the value of feature 'pos_2_right' is *$,* (representing the comma).

pos_3_right The STTS tag of the token three to the right is extracted analogously, which is *VVFIN* (from token *heißt*) in the example.

pos The part of speech of the target word itself is encoded as a feature. Since the STTS tagset does not further distinguish the tags for nouns, this feature is applicable to adjectives and verbs. For adjectives, the nominal value set includes the two values *{ADJA, ADJD}* only. For the example target word *frei* from Figure 8.1, this feature has value *ADJD*. The range of values for verbs is larger. It comprises the 12 verbal STTS POS tags

starting with a 'V': *{VAFIN, VAIMP, VAINF, VAPP, VMFIN, VMINF, VMPP, VVFIN, VVIMP, VVINF, VVIZU, VVPP}*.[1]

verbs_pos This feature, which is implemented for adjectives and nouns, concatenates the part-of-speech tags of all verbs in the sentence containing the target word. Although this feature apparently concerns part-of-speech information, it makes use of the topological field to identify the tokens of which POS information is extracted. The POS tags for all tokens that are part of a verbal element in the topological model are extracted and concatenated (by underscores) in the order in which the tokens occur in the corpus. 'All tokens that are part of a verbal element in the topological model' refers to all tokens that are part of an *LK*, a *VC*, or a *VCE* field (see Subsection 5.3.1 for a description of topological fields). Including all verbal fields from the example sentence in Figure 8.1 results in the string *VMFIN_VAINF_VVFIN* (i.e., the POS tags from the three verbs *will*, *sein*, and *heißt* in the order of their occurrence in the corpus).

In the ideal case, i.e., assuming that the structural annotation is correct, only those verbal elements from the topological field are considered that actually match the target word. In the example, only the first *LK* field containing token *will* and the *VC* field containing token *sein* match with the target word *frei*. Thus, in the example, the correct value for this feature is *VMFIN_VAINF*.

verbs_pos_ignoring_aux Similarly to the previously described feature, the part-of-speech tags of verbs in the sentence containing the target word are concatenated. It is analogously implemented for adjectives and nouns. By contrast, this feature focuses on main verbs and omits auxiliary verbs.[2] For the example target word *frei*, the ideal value of this fea-

[1] Note that while these restricted feature value sets reflect the manual annotations in the TüBa-D/Z treebank, further POS tags certainly occur erroneously in the automatic annotations.

[2] In the manually annotated TüBa-D/Z, auxiliary verbs are identified by *%aux* or *%passiv* markers at the end of the lemmas (see lemma *wollen%aux* for the second token in the sentence in Figure 8.1). For the automatically annotated corpora, morphological information on

ture is *VAINF*.[1]

Related studies, including Lee and Ng [2002], [Martínez et al., 2002], Mihalcea [2002b] and Kopeć et al. [2012], also implemented the first seven above-listed 'part-of-speech' features, i.e., the POS tags of the tokens surrounding the target word (in a context of ±3 tokens) and the POS tag of the target word itself.

8.1.5 Morphological Features

Six morphological features represent information on inflectional morphology, i.e., number, case, gender, person, mood, and tense, of the target word.

morph_number Each target word contains information on its grammatical number (*singular* or *plural*) or is left *underspecified*. The nominal value set for this feature includes three entries: *{singular, plural, underspecified}*.

morph_case German morphology involves the four cases *nominative*, *genitive*, *dative*, and *accusative*, which occur for adjectives and nouns. The value set includes an *underspecified* marker: *{nominative, genitive, dative, accusative, underspecified}*.

morph_gender Each German noun belongs to one of the three grammatical genders *masculine*, *feminine*, and *neuter*. The gender of attributive adjectives is determined by the corresponding noun. The nominal values for this feature are *{masculine, feminine, neuter, underspecified}*.

morph_person Depending on the subject of a sentence, German verbs are conjugated by person: *{first, second, third, underspecified}*

morph_mood German verbs signal linguistic modality. This feature represents two moods: *{indicative, subjunctive}*.

whether a verb is a full verb or an auxiliary verb is extracted with the help of RFTagger, which is described in Subsection 8.1.1 above.

[1] As for the previous feature, the *ideal* implementation of this feature considers only verbal elements that match the target word.

morph_tense The tense of a German sentence is expressed by the verb. The feature comprises the broad distinction between *present* and *past* tense: *{present, past, underspecified}*.

For the manual TüBa-D/Z annotation, morphological information on number, case, gender, person, mood, and tense is straight-forwardly extracted from the treebank's morphologic annotation layer (see Section 5.3.1). In its stylebook, the TüBa-D/Z specifies the kind of morphological information available for each part of speech. Table 8.2 is a shortened version of the original table copied from the TüBa-D/Z stylebook [Telljohann et al., 2012]. It shows corresponding morphological information for those adjectival, nominal, and verbal POS tags that are relevant for the annotated target words and for which morphological information is available in the treebank.[1]

Table 8.2: Morphological information available for certain parts of speech.

POS	**Morphological information**	**Comments**
ADJA	case, number, gender	underspecified for gender if plural noun is underspecified, e.g. *die/np* nordhessischen /np* Grünen/np**; underspecified for invariant local descriptions, e.g. *Berliner/****; full morphology for cardinal numbers as abbreviation, e.g. *im 4./dsn Jahrhundert/dsn*
NN	case, number, gender	underspecified gender for specific nouns, e.g. *Abgeordnete* (in plural) or *Leute*
VAFIN	person, number, mood, tense	
VAIMP	number	
VMFIN	person, number, mood, tense	
VVFIN	person, number, mood, tense	
VVIMP	number	German has only second person imperative forms

[1]Those POS tags that are not included in the table are either not relevant for the annotated target words or do not contain any morphological annotation.

While this specification of which morphological information is available for which part of speech reflects the manual annotations in the TüBa-D/Z treebank, any automatic annotations are certainly less controlled. For the automatic processing, morphological information is annotated with RFTagger – as described in Subsection 8.1.1 above.

Related studies that used morphological information of the target word as features include Ng and Lee [1996] and Kopeć et al. [2012].

8.1.6 Context Detail Features

This subsection describes seven features which all use heterogeneous information from the contexts of the target words.

sentence_length The length of the sentence in which a target word occurs represents a numeric feature – available for target words of all word classes.

adjective For nouns, this boolean feature encodes whether or not an adjective belongs to the target noun. The relevant adjectives which modify nouns are attributive adjectives (identified by their POS tag *ADJA*) syntactically attached to the same nominal phrase *NX* than the target noun. Technically, the value of this feature is set to *true* if there is an adjectival phrase *ADJX* occurring in the target noun's nominal phrase.[1]

article German nouns often have a definite article (*der*/*die*/*das* 'the'), an indefinite article (*ein*/*eine* 'a'/'an'), or an attributive possessive pronoun (such as *mein(e)* 'my', *dein(e)* 'your', *unser(e)* 'our', etc.). Linguistically, a possessive pronoun is not an article, but because for German nouns there can only either be an article or an attributive possessive pronoun, this information can be encoded in one feature. In order to qualify for this feature, the token in question syntactically needs to be part of the same noun phrase *NX* than the target word.

[1]This does not consider only attributive adjectives as *true*, but – on purpose – also includes adjectival phrases containing, for example, a cardinal number (POS tag *CARD*) or an attributive indefinite pronoun (POS tag *PIDAT*).

The nominal value set of this feature is *{definite, indefinite, none, mein, dein, sein, ihr, unser, euer, Ihr}*. All eligible tokens can be identified by their POS tags (*ART* for definite or indefinite articles and *PPOSAT* for attributive possessive pronouns) in combination with their lemma (i.e., *der/die/das* versus *ein/eine* for part-of-speech tag *ART* or *mein(e)*, *dein(e)*, *sein(e)*, etc. for *PPOSAT*).

adposition By definition, a prepositional phrase *PX* in the TüBa-D/Z contains an *adposition*, i.e., a preposition, a postposition, or a circumposition. For the case that a target noun is syntactically part of a prepositional phrase, the corresponding adposition is used as the value for this feature.[1] Adpositions are identified by their POS tags, which all start with the two letters *AP* in the STTS tagset: *APPR* for prepositions (and left circumpositions), *APPRART* for prepositions with incorporated articles, *APPO* for postpositions, and *APZR* for right circumpositions. Sample values for this string feature include: *{zu, in, bei, mit, von, über, durch, aus, auf, für, nach, an, samt}*.

verbs This feature concatenates all verbal lemmas of a sentence. Similarly to the above-described feature 'verbs_pos' (see Subsection 8.1.4), which concatenates verbal POS tags, it makes use of the topological field to identify all verbal tokens. All tokens that are part of verbal field, i.e., an *LK*, a *VC*, or a *VCE*, are extracted and their lemmas concatenated (by underscores) in the order in which they occur in the corpus. Using the same example sentence as above (see Figure 8.1) with the target word *frei* 'free', this results in the string *wollen%aux_sein_heißen*. Again, the ideal case is that only those verbal parts from the topological field are considered that match the target word. Since in the example only the first *LK* field containing token *will* and the *VC* field containing token *sein* match with the target word *frei*, the ideal value for the present feature is *wollen%aux_sein*.

This feature and the next feature are both implemented for adjectives

[1] In case of multiple adpositions in a PX, especially for circumpositions, all relevant adpositions are concatenated (by underscores).

and nouns.

verbs_ignoring_aux For this feature, all main verb lemmas of a sentence are concatenated – similarly to the previous feature (which concatenates all verbs), but without auxiliary verbs.[1] For the target word *frei* in the example from Figure 8.1, the ideal value of this feature is *sein*.[2]

auxiliary_verb Different from the two previously listed features, this feature specifies the usage of an auxiliary verb given a main verb as the target word. It also makes use of the structure of the topological field to identify matching verbs. Assuming that the target word under consideration is the verb *sein* in the same example sentence from Figure 8.1, the value for this feature is *wollen*.[3] Further sample values for the feature at hand are *{sein, haben, sollen, müssen, wollen, werden, ...}*.

8.1.7 Sentence Structure Features

The following six features employ information on sentence structures:

sentence_type Depending on the position of the finite verb, the concept of topological fields distinguishes German sentences into three types: *verb-initial*, *verb-second*, and *verb-final*. The feature is available to target words of all word classes. Its nominal value set includes an underspecified marker (*none*) for elliptical or incomplete sentences: *{verb_initial, verb_second, verb_final, none}*.

head Each token in the TüBa-D/Z (and in the automatic annotation with the Berkeley Parser – see Subsection 8.1.1 above) has a direct edge label.

[1] As described for the analogous feature 'verbs_pos_ignoring_aux' encoding POS tags instead of lemmas (see Subsection 8.1.4 above), auxiliary verbs are identified by *%aux* or *%passiv* markers at the end of the lemmas in the manually annotated TüBa-D/Z. For the automatically annotated corpora, morphological information on whether a verb is a full verb or an auxiliary verb is extracted with the help of RFTagger, which is described in Subsection 8.1.1 above.

[2] As for the previous feature, the *ideal* implementation of this feature considers only verbal fields that match the target word.

[3] Again, in case of the existence of multiple auxiliary verbs in one sentence, these are underscore separated.

In most cases, this marker denotes the token either as the head (label *HD*) or as a non-head (label '-') of a phrase[1] – captured in this boolean feature called 'head'.

passive This boolean-valued feature indicates whether or not passive voice is used in the sentence in which the target word occurs. It is available to target words of all word classes. In the implementation, a sentence is marked as passive if the passive voice auxiliary verb *werden*[2] occurs together with a past participle[3].

nx_length Each noun is syntactically attached to a nominal phrase *NX*. This numeric feature encodes the number of tokens subsumed under the target noun's *NX*.

part_of_conjunction For nouns, this boolean-valued feature specifies whether the target word is directly part of a conjunction. Technically, the feature value is set to *true*, if the direct noun phrase *NX* in which the target noun occurs has the edge label *KONJ* denoting conjunctions.

grammatical_function This nominal feature specifies the closest edge label for a target noun. That is, nouns are part of noun phrases (*NX*) which are – depending on their status within the sentence – labeled with an edge label denoting their appropriate grammatical function within the sentence, such as *ON*, *OD*, *OA*, *OG*, *MOD*, *ON-MOD*, *OA-MOD*, etc. For instance, for the noun *Information* in Figure 8.1, the corresponding 'grammatical_function' feature value would be *ON*.

[1]The other two less-frequently occurring edge labels for denoting tokens, i.e., *-NE* for excluding a token from a named entity phrase and *VPT* for separated verb prefixes, are counted as non-heads in the current implementation.

[2]In the manually annotated TüBa-D/Z, the lemmas of passive voice auxiliary verbs are annotated as *werden%passiv*. For the automatically annotated corpora, morphological information on whether a verb is a full verb or an auxiliary or modal verb is extracted with the help of RFTagger.

[3]Past participles are identified by their part-of-speech tag *VVPP*.

8.1.8 Constituent Structure Features

Grammatical information contained in a treebank makes it possible to utilize features that rely on syntactic structures. This is particularly useful for the disambiguation of verbs where the syntactic structure in which a verb occurs is often highly predictive of different word senses. Fellbaum et al. [2001], Dligach and Palmer [2008] and Chen and Palmer [2009] previously showed that the incorporation of features encoding syntactic structures improves the performance of verb sense disambiguation systems for English.

In order to analyze the influence of syntax and semantics on automatic WSD for German, an extensive set of features that encode syntactic structures is implemented: features based on constituent structures are described in this subsection and features based on verbal frames are described in the next subsection (Subsection 8.1.9).

This subsection describes altogether 11 boolean-valued features that capture the syntactic constituency of verbs. Each of the features encodes the existence of one of the complement edges, e.g., *ON*, *OA*, *OD*, *OG*, *FOPP*, *OPP*, etc., specified in the TüBa-D/Z syntax.[1] These complements mostly represent objects of a verb such as accusative, dative, genitive, prepositional, or adverbial objects. In addition to these complement edges, there is one feature encoding the structural expletive, i.e., edge label *ES*.

All features use topological fields to extract complements for a verb. In the ideal case, i.e., assuming that the structural annotation is correct, only those complements from the topological field are considered that actually match the target verb.[2] Syntactic constituency features are available to verbal target words. Also see the TüBa-D/Z stylebook [Telljohann et al., 2012] for more details on each edge label.

[1] Altogether, there are 13 complement edges specified in the TüBa-D/Z stylebook [Telljohann et al., 2012, Table 3.8]. Both *VPT*, which denotes separated verb particles, and *APP*, which denotes appositions, do not specify a feature in the current implementation, because they are not relevant for the type of syntactic constituency described in this section. In order to be able to produce such constituent structures with automatic tools that are comparable to the manual TüBa-D/Z treebank annotations, the Berkeley Parser is trained on the TüBa-D/Z data – see Subsection 8.1.1 above.

[2] See the description of the 'verbs_pos' feature in Subsection 8.1.4 above for a concrete example for what is meant by *match* the verb.

has_ON This feature encodes whether or not a subject with the complement edge label *ON* (nominative object) occurs in the topological field of a verb.

has_OA This feature encodes whether or not an accusative object (edge label *OA*) is realized in the topological field of a verb.

has_OD The existence of a dative object *OD* is captured accordingly.

has_OG Whether or not a verb has a genitive complement – annotated by the *OG* edge label – is encoded by this feature.

has_FOPP Constituents marked with *FOPP* are optional prepositional objects including passivized subjects. These phrases are denoted as optional since they can be left out without resulting in ungrammatical sentences. An example is given in sentence (9).

(9) *Das is gut [für uns]*^{FOPP}.[1] ('That's good for us.')

has_OPP In contrast to *FOPP*, prepositional phrases labeled by *OPP* are mandatory complements. That is, without the *OPP* complement a sentence would be ungrammatical – as illustrated in example sentence (10).

(10) *Beide stammen nämlich [aus dem Westen]*^{OPP}.[2]
('Both are from the West.')

has_OS In examples such as in sentence (11), where subordinate clauses occupy the position of sentential objects, the sentential object is annotated with the edge label *OS*. This 'has_OS' feature encodes whether or not such a sentential object exists for the verb under investigation.

(11) *Die Spieler wissen, [worum es geht]*^{OS}.[3]
('The players know what it is about.')

[1] Sentence 946 from TüBa-D/Z 9.1.
[2] Sentence 3 169 from TüBa-D/Z 9.1.
[3] Part of sentence 735 from TüBa-D/Z 9.1.

has_OV Verbal objects are required by other verbs. Without these verbal objects – labeled as *OV* – a sentence is usually either grammatically incorrect or represents a completely different meaning. The example in sentence (12) illustrates the case that the meaning of the sentence would change without the verbal object *lesen* 'to read'.

(12) *Ich kämpfe dafür, dass wir mit den Augen lesen*^{OV} *lernen.*[1]
('I fight that we learn to read with the eyes.')

has_OADJP Adjectival objects that are required by the main verb of a sentence are annotated with the edge label *OADJP*. Without the *OADJP* constituent, the sentence would be grammatically incorrect or would have a different meaning, such as in the example sentence (13).

(13) *Die Wählerregistrierung verlief schleppend*^{OADJP}*.*[2]
('The voter registration proceeded slowly.')

has_OADVP Adverbival objects are annotated with the edge label *OADVP*. Analogously to adjectival objects, constituents annotated with *OADVP* are required by the main verb of a sentence and without them the sentence would be grammatically incorrect or would have a different meaning. An example is given in sentence (14).

(14) *Das sehen die Gewerkschaftler anders*^{OADVP}*.*[3]
('The unionists see this differently.')

has_PRED Predicates of verbs are annotated with the edge label *PRED* in the TüBa-D/Z treebank – as illustrated in example sentence (15). Typical verbs for which predicates occur are, for example, *sein*, *haben*, *scheinen*, *aussehen*, etc. (see TüBa-D/Z's handbook [Telljohann et al., 2012] for more details on when exactly the edge label *PRED* is annotated). The existence of such a *PRED* label is encoded in the 'has_pred' feature.

[1] Sentence 45 599 from TüBa-D/Z 9.1.
[2] Sentence 12 404 from TüBa-D/Z 9.1.
[3] Sentence 1 747 from TüBa-D/Z 9.1.

(15) *Jamal ist [wütend]^{PRED}.*[1] ('Jamal is angry.')

has_ES This feature specifies whether or not the topological field of the target verb includes the structural expletive *Vorfeld-es* (edge label *ES*). If the pronominal form *es* 'it' is used as a purely structural dummy element in the *Vorfeld* 'initial field' position and is not correlated with any other phrases of the sentence, it is labeled as *ES*. An example is given in sentence (16).

(16) *Es^{ES} geschieht hier nichts.*[2] ('Nothing happens here.')

Since the annotation of a sentence's constituency structure is much more difficult than other linguistic annotation such as lemmatization or part-of-speech tagging, the quality of the features described in this subsection largely differs: manual annotations are certainly more accurate than automatic parses. This behavior is confirmed and discussed in the evaluation section below (Section 8.3).

8.1.9 Verbal Frame Features

Verbal frames are captured in altogether 16 numeric features. Each of the features encodes the existence of a frame label as used in GermaNet. These frame labels mostly represent objects of a verb such as accusative, dative, genitive, prepositional, or adverbial objects. As described in Subsection 3.7, the verbal frames in GermaNet are adapted from the CELEX Lexical Database [Baayen et al., 1995]. The frame labels typically differ from the names of non-terminal symbols and dependency labels used by a parser trained on a particular treebank. Therefore the tags need to be mapped; in the present work, a manual tag mapping from GermaNet to TüBa-D/Z has been performed. The confidence scores assigned to individual frame labels have been determined by human introspection and take into account how easy it is to map a GermaNet frame label to TüBa-D/Z's annotation structure. If the GermaNet to TüBa-D/Z mapping is one-to-many for a given frame label, such a mapping may

[1] Sentence 408 from TüBa-D/Z 9.1.
[2] Part of sentence 6 663 from TüBa-D/Z 9.1.

result in loss of information. Therefore, the frame label in question receives a lower score than a label where the mapping from GermaNet to TüBa-D/Z is one-to-one and no information loss occurs.

Altogether, there are 23 frame labels specified in GermaNet (see Table 3.4) of which 15 are implemented as features in their obligatory version, i.e., *NN*, *AN*, *DN*, etc. Prepositional phrases are further implemented in their optional variant *Pp* (indicated by a lower case second letter).[1]

All implemented features in this subsection use topological fields to extract verbal frames. In the ideal case, i.e., assuming that the structural annotation is correct, only those frames from the topological field are considered that actually match the target verb.[2] Since the features in this subsection rely on similar and largely overlapping information on syntactic constituency information of verbs, the verbal frame features in this subsection are similar to the constituent structure features described in the previous subsection.[3] However, since the verbal frame features make use of more information than solely constituent structures, they can be considered as an extension of the constituent structure features described in Section 8.1.8.

All verbal frame features are by definition available to verbal target words. Since these features encode confidence scores assigned to specific frame labels, their values are numeric. Those features that appear reliable and their GermaNet frame labels are easy to map to the annotation available in the TüBa-D/Z treebank can achieve the highest confidence scores. The following list sorts the 16 verbal frame features in decreasing order of confidence.

NE_confidence The GermaNet frame label *NE* expresses the expletive sub-

[1]The adverbial frame labels (starting with a 'B', i.e., *BC*, *BD*, *BL*, etc.) do not specify features in the current implementation: on the one hand, adverbial frame labels as defined in GermaNet can be realized in many different ways in a sentence and are, thus, very complex and rather difficult to implement without an extensive analysis of annotated examples. On the other hand, no such annotated examples are available and, in general, adverbial frame labels are encoded too rarely in GermaNet so as to allow such an extensive analysis with sufficiently many instances. For these reasons, their implementation as features is left to future work.

[2]A concrete example for what is meant by *match* the verb is given in the description of the 'verbs_pos' feature in Subsection 8.1.4 above.

[3]In some cases, where the features encode exactly the same information, the verbal frame feature represents a numeric variant of its boolean-valued constituent structure feature alternative from the previous subsection.

ject *es* 'it'. In the TüBa-D/Z, all non-referential uses of the pronoun *es* are annotated with the *%expletive* marker [Naumann, 2007]. This subsumes the structural expletive *Vorfeld-es* (TüBa-D/Z edge label *ES*), but it also occurs for weather verbs and verbs with a missing agent, such as in example sentence (17).

(17) *In Schleifen geht es^{NE} bergauf.*[1]
('In serpentines it goes uphill.')

For the feature at hand, the following three criteria identify occurrences of GermaNet's *NE* frame labels: (i) the lemma has to be *es*, (ii) it has to be the subject of the sentence (annotated with edge label *ON*), and (iii) it has to have an explicit *%expletive* annotation.[2] Since these criteria are very clear, *NE* is the GermaNet frame label that can be mapped most reliably to the annotation in TüBa-D/Z.

DR_confidence Identifying reflexive pronouns in dative case (GermaNet frame label *DR*) is straight-forward: if a corresponding token exists that represents a dative object (annotated with the edge label *OD* in TüBa-D/Z) and that is a reflexive personal pronoun (annotated with the part-of-speech tag *PRF*). Sentence (18) gives an example:[3]

(18) *Wegmanns haben sich^{OD} einen Hund zugelegt.*[4]
('The Wegmanns have adopted a dog.')

AR_confidence Reflexive pronouns in accusative case are analogously straight-forward to identify: they have to represent an accusative object (edge label *OA*) and be a reflexive personal pronoun (POS tag *PRF*).

[1] Sentence 73 995 from TüBa-D/Z 9.1.

[2] Note that since these *%expletive* annotations are available only to the manually annotated treebank, the corpora that rely on automatic annotations have to satisfice with criteria (i) and (ii), which are certainly less reliable.

[3] The example was already given in Section 5.3.1 where the manual annotations in the TüBa-D/Z treebank were explained. It is repeated here for convenience. However, in Section 5.3.1 (Figure 5.6) the sentence is also visualized with its complete syntactical annotations.

[4] Sentence 60 611 from TüBa-D/Z 9.1.

AN_confidence GermaNet's frame labels distinguish between reflexive accusative objects (*AR*) and general (non-reflexive) accusative objects (*AN*). Both these frame labels are identified in the TüBa-D/Z by the *OA* annotation for accusative objects. The part-of-speech tags help distinguishing between the reflexive and non-reflexive objects (reflexive personal pronouns can be identified by their POS tag *PRF*).

For the 'AN_confidence' feature at hand, the maximum confidence score is given for occurrences of non-reflexive accusative objects (i.e., edge label *OA* occurs without POS tag *PRF*).

A problem arises, however, for reflexive uses of non-inherent reflexive German verbs such as the verb *waschen* 'to wash' in sentence (19). These cases are classified with frame label *AN* (rather than *AR*) in GermaNet, but in the TüBa-D/Z, they cannot be distinguished from inherent reflexive verbs. For this reason, occurrences of reflexive accusative objects result in a confidence score minimally above zero for frame label *AN*.

(19) *Sonst wäscht er sich*$^{OA+PRF}$ *nie.*[1]
('Otherwise he never washes himself.')

DN_confidence Analogously to the accusative case, *DN* frame labels in GermaNet are identified by dative object edge labels *OD* in TüBa-D/Z, and the problem of reflexive uses for non-inherent reflexive verbs is handled similarly.

In certain cases, dative objects can be expressed with prepositional phrases in dative case (annotated in TüBa-D/Z as facultative prepositional objects *FOPP* with appropriate dative case markers in the morphological layer), such as the prepositional phrase *mit den Daten* 'with the data' in example (20). The existence of *FOPP*'s in dative case give a much lower confidence score than a 'real' dative object.

(20) *Was [mit den Daten]*^{FOPP} *geschehe, sei nicht überprüfbar.*[2]
('What happens with the data, would not be verifiable.')

[1] Sentence 47 605 from TüBa-D/Z 9.1.
[2] Sentence 15 714 from TüBa-D/Z 9.1.

GN_confidence Genitive objects are simply identified by their edge label *OG* in TüBa-D/Z. Unlike for dative and accusative objects, the above-described problem with non-inherent reflexive verbs does not exist for German genitive objects.

AZ_confidence The existence of verbal infinitive clauses with *zu* 'to' (GermaNet frame label *AZ*) relies on annotations both from the structural as well as from the part-of-speech layer in the treebank. On the structural level, subordinated sentential objects (with edge label *OS*) are extracted. Each of these *OS* sentences qualifies for this 'AZ_confidence' feature at hand, if it (i) either contains a main verbs with *zu*-infinitive where the *zu* is realized as an infix (such tokens are identified by their part-of-speech tag *VVIZU*) or (ii) contains both a *zu* particle (identified by its POS tag *PTKZU*) and an infinitive verb (identified by its POS tag starting with 'V' and ending in 'INF', i.e., *VVINF* represents main infinitive verbs, *VAINF* represents auxiliary infinitive verbs, and *VMINF* represents modal infinitive verbs). Sentence (21) – represented in Figure 8.2 with all its annotation – gives an example for the second case (ii) as described above: assuming that the target verb is *lernen* 'to learn', the existence of frame label *AZ* (represented by *zu leben*) can be identified with high confidence.

(21) *Ich lernte [zu leben]*^{AZ}.[1] ('I learnt to live.')

AI_confidence The GermaNet frame label *AI* encodes verbal infinitive clauses. It is identified in the treebank by a verbal object (identified by its edge label *OV*) which contains an infinitive verb. As for the previously described feature, infinitive verbs are identified by their POS tags starting with 'V' and ending in 'INF', i.e., *VVINF*, *VAINF*, or *VMINF*. For *AI*'s, however, it is important that the *OV* does not contain any *zu* particles (POS tag *PTKZU*).

NG_confidence The GermaNet frame label *NG* represents additional noun phrases or adjectival phrases in nominative case besides the subject (in

[1] Sentence 36 168 from TüBa-D/Z 9.1.

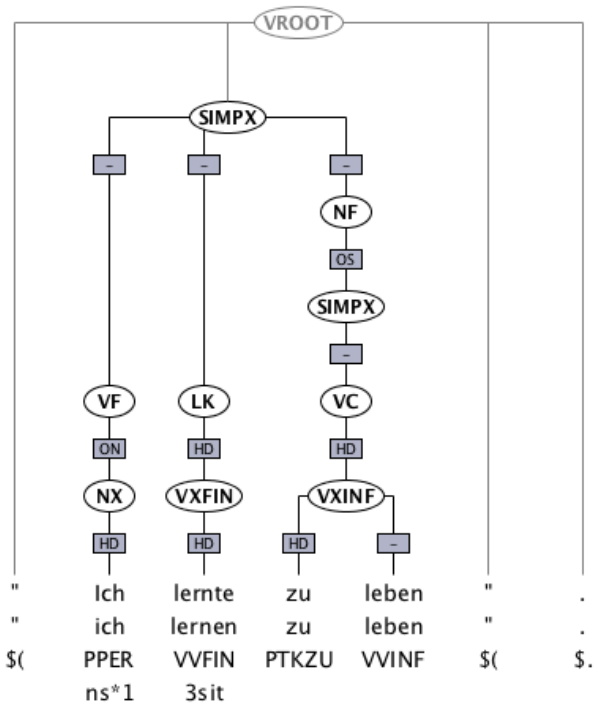

Figure 8.2: An example for frame label *AZ*.

German *Gleichsetzungsnominativ* or *Prädikatsnominativ*). The equivalence in TüBa-D/Z annotates such cases with the edge label *PRED*. An example is given in sentence (15) above.

DS_confidence *DS* frame labels represent subordinate clauses introduced with *dass* 'that'. They are mapped to the annotation in the TüBa-D/Z as follows: subordinate clauses with the edge label *OS* whose first token is *dass* (or *daß* in the old German ortography).

A lower confidence score is assigned to instances of direct speech (naively identified by subordinate clauses in double quotes or by subordninate clauses preceded by colons) and where the *OS* does not start with a *dass/daß*.

FSo_confidence The frame label *FSo* stands for interrogative sentences with the question particle *ob* 'whether'/'if'. They are identified in the treebank by subordinate clauses starting with token *ob*.

FSw_confidence Analogously, interrogative sentences with the interrogative pronouns starting with 'w' (e.g., *wer* 'who', *was* 'what', *wie* 'how', etc.) are identified in the treebank by subordinate clauses whose first token starts with character 'w'.

FS_confidence The identification of 'general' interrogative sentences (GermaNet frame label *FS*), i.e., interrogative sentences without the explicit demand for a certain interrogative pronoun, is less reliable compared to the other above-described frame labels: it simply checks the existence of a subordinate clause (*OS*).

PP_confidence Although the identification of obligatory prepositional phrases (frame label *PP*) appears straight-forward, i.e., edge label *OPP*, the assigned confidence score is lower than for the other frame labels. This is due to the wide and diverse range of possibilities to realize prepositional objects. It even happens that the obligatory frame label *PP* in GermaNet is realized as an optional prepositional object (*FOPP*) in TüBa-D/Z; but in that case, the confidence score is even lower.

Pp_confidence Optional prepositional phrases (indicated by a lowercase second letter in the GermaNet frame label *Pp*) are mapped to TüBa-D/Z's *FOPP* edge label. Again, it happens that concrete example sentences are annotated with *OPP* although the phrase in question is specified as optional on the GermaNet side. The confidence score is rather low for the same reason given for the previous feature.

NN_confidence Identifying GermaNet's *NN* frame labels, i.e., noun phrases in nominative case which represent the subject of the sentence, is straight-forward: those constituents annotated with the edge label *ON* in TüBa-D/Z. However, this frame label is the least meaningful, because nearly all active German sentences have subjects.

8.1.10 Other Features

Since the features described in this subsection do not exactly match any of the other feature categories, they are subsumed in this 'other' group. The three features are heterogeneous but applicable to all word classes. Note, however, that the second feature is available only for the manual TüBa-D/Z treebank annotations, while the last feature is available only in connection with the automatic preprocessing.

headline This boolean-valued feature encodes whether a target word is part of a headline or occurs in regular text. This information is interesting because headlines often contain incomplete or elliptical sentence structures. For example, for target words that occur as part of a headline the values of syntactical features might be misleading or at least divergent from those in regular sentences. Headlines are indicated appropriately in the manually annotated TüBa-D/Z. For the automatically preprocessed texts, the value of this feature is evaluated by a simple heuristic that checks whether the first sentence of the text at hand does not end with a punctuation. Technically, the heuristic checks whether the POS tag of the last token in the first sentence of the text does not start with a dollar symbol (STTS POS tags starting with dollar symbols indicate punctuation marks).

named_entity Although TüBa-D/Z's part-of-speech tag layer includes the tag *NE*, which marks named entity tokens on the word level, proper information on named entities is annotated on the phrase level (as described in Section 5.3.1). This boolean-valued feature specifies whether or not a target word is part of a named entity phrase.[1]

translation The English translation of the target word in its specific context – as described in Subsection 8.1.1 above – is employed as a string feature for target words of all word classes.[2] The value set for lemma *Frau*, for

[1] Since the used model of the Berkeley parser is not adapted to output such information on named entities, this feature is available for the manual treebank annotations only.

[2] Since the manually annotated TüBa-D/Z treebank does not contain translations, this feature is available only for automatically annotated texts.

example, contains *{mrs, woman, wife, madam, female, ms}*.

8.1.11 Number of Features

To get an impression of how many of the above-described features are actually created for a lemma, Table 8.3 lists the average numbers of features per lemma.

Table 8.3: Average numbers of features per lemma.

POS	**WebCAGe**	**TüBa-D/Z (manual)**	**TüBa-D/Z (automatic)**	**deWaC**
Adj.	159			46
Nouns	197	356	348	53
Verbs	102	271	253	111

For each word class, the table provides average numbers of features for each of the sense-annotated corpora (specified in the header column). The labels *WebCAGe* and *deWaC* unambiguously refer to the particular corpora with the same names, while the caption *TüBa-D/Z (manual)* refers to TüBa-D/Z with its manual linguistic treebank annotations and *TüBa-D/Z (automatic)* refers to TüBa-D/Z with automatic linguistic annotations (as described in Subsection 8.1.1). The reason why Table 8.3 does not include values for adjectives in the TüBa-D/Z is simply because the treebank contains only sense annotations for nouns and verbs.

Counting the features described in the previous subsections for each word class amounts to at most 23 different features for adjectives, at most 28 different features for nouns, and at most 59 different features for verbs[1] – excluding the 'context lemma' features, since these might result in an arbitrary number of features (see Subsection 8.1.3 above). This means that the differences between these amounts (i.e., a maximum of 23, 28, or 59) and the numbers provided in Table 8.3 are made up by 'context lemma' features, which is, in fact, the majority or features.

[1]Note that these amounts are the maximum possible numbers for the potential features, since only those features are considered for which at least two distinct values occur for a lemma in the sense-annotated data.

Table 8.3 shows that for lemmas in both TüBa-D/Z versions considerably more features are created than for the other two corpora; and in deWaC, lemmas have fewest features (especially for adjectives and nouns) compared to the corresponding numbers in the other corpora. These differences in the average numbers of features are mainly due to the number of 'context lemma' features, which make up the majority class of features. If this type of feature is plentiful, it means that the contexts of many sense annotations have several lemmas in common. By contrast, if the amount of context lemma features is sparse, it means that only few lemmas are common to at least three annotated contexts (the minimum amount of three is chosen in order to keep the number of lemmas manageable and to ensure a certain level of meaningfulness and generalizability for the created features – as described in Subsection 8.1.3 above). Of course, few lemmas that occur in at least three annotated contexts can also be caused by few annotations per lemma. But since this annotation frequency is similar in WebCAGe where there are three to four times as many features for adjectives and verbs than in deWaC, the lower number of created (context lemma) features for deWaC must have to do with very heterogeneous contexts. In the evaluation and discussion of the supervised WSD experiments in Section 8.3 below, it is shown that this fact (i.e, more heterogeneous contexts which result in a lower number of features) negatively affects the WSD performance when evaluated on deWaC.

After the 'context lemma' features, the features based on part-of-speech tags (see Subsection 8.1.4 above) are the second most common features. The reason for this is simple: with altogether nine 'part-of-speech' features, this feature group constitutes the largest group (ignoring 'constituent structure' and 'verbal frame' features, which are by definition restricted to verbs). Further, six of these features (those that rely on contextual information, i.e., 'pos_1|2|3_left' and 'pos_1|2|3_right') are generally available for almost all lemmas, since they are applicable to all word classes and the chance that at least two distinct values occur for a lemma in the sense-annotated data is high.

8.2 Supervised Machine Learning with Weka

In the WSD setup in this chapter the *Waikato Environment for Knowledge Analysis* (short: *Weka*)[1] is used – a machine learning tool suite developed at the University of Waikato in New Zealand [Hall et al., 2009]. It provides implementations for several machine learning tools, including tools and algorithms for data pre-processing, classification, clustering, feature selection, and visualization. Weka's usefulness for the task of WSD has already been demonstrated in many works, including Paliouras et al. [2000], Pedersen [2001, 2002], Lee and Ng [2002], Lee et al. [2004], Mohammad and Pedersen [2004], Steffen et al. [2004], Turney [2004], Joshi et al. [2006], Bas et al. [2008], Młodzki and Przepiórkowski [2011], Biemann [2012], Kopeć et al. [2012], and Maarouf et al. [2014].

The main reason for employing Weka in this chapter is to be able to apply and compare a wide range of supervised machine learning algorithms to German WSD. Even though for some algorithms the implementations in Weka might not necessarily be the best performing ones[2], the great advantage of Weka is the free availability of many heterogeneous algorithms, which easily allows the application of a large variety of supervised classification algorithms [Frank et al., 2005; Hall et al., 2009; Witten et al., 2011].[3] The subset of classifiers that is applied in this chapter is chosen (i) to cover several generally popular, yet distinct machine learning approaches (rule-based, instance-based, probabilistic, etc.) and (ii) to include those algorithms that have prevalently been applied to WSD on other languages and that have shown to perform well in related work (see Subsection 2.3.2). All classifiers are applied in their

[1] Weka (`http://www.cs.waikato.ac.nz/ml/weka/`) is implemented in Java and is freely available under the GNU General Public License (`http://www.gnu.org/licenses/gpl.html`). In the implementation, Weka's stable version 3.6.10 is used [Bouckaert et al., 2013].

[2] Sandra Kübler, personal communication.

[3] For the same reason given above for putting so much effort into the implementation of features, less effort is put into the refinement of the classification algorithms: that is, the impact of applied classification algorithms on the performance of WSD systems is significantly lower than the impact of the implemented features [Yarowsky and Florian, 2002]. Thus, for the purpose of this thesis, the application of many classifiers on a large variety of high-quality features is more important than the application of one or only a few classifiers with a high-quality implementation on only a small subset of moderately implemented features.

default parameter configurations[1] – if not otherwise stated.

The basic ideas (referring to the implementations in Weka) of the applied classification algorithms are summarized in the following subsections[2] – grouped into baselines (Subsection 8.2.1), classifiers based on decision rules (Subsection 8.2.2), instance-based classifiers (Subsection 8.2.3), probabilistic classifiers (Subsection 8.2.4), support vector machines (Subsection 8.2.5), and combined classifiers (Subsection 8.2.6) and a classifier for automatic feature selection (Subsection 8.2.7). More comparative details including the strengths and weaknesses of the particular approaches as well as pointers to corresponding related works that previously applied the approaches in question to the task of WSD are given in Subsection 2.3.2 of the related work chapter.

8.2.1 Baseline Classifiers

A baseline is a very simple, commonly used algorithm for solving the task in question (see Subsection 2.2.3). Since baselines are assumed to be outperformed by more elaborated algorithms, comparing the performance of a WSD system to the performance of a baseline allows estimating the impact and efficiency of a WSD system [Navigli, 2009]. For the experiments in this chapter, the following two baseline classifiers are applied:

ZeroR (short for *Zero Rules*) always predicts the majority class[3] [Witten et al., 2011]. This classifier does not use any features and is thus very simplistic. Due to its simplicity, *ZeroR* is often used as a baseline for supervised learning systems. In the WSD terminology, *ZeroR* represents the most frequent sense baseline (see Subsection 2.2.3).

OneR (short for *One Rule*) [Witten et al., 2011] uses one classification rule which incorporates one feature. On the training data, OneR generates

[1]The classifiers' performance can certainly be boosted by adjusting their parameter settings appropriately. However, experiments to identify optimal parameter settings for the task of word sense disambiguation are left to future work.

[2]Also see the references in these subsections to the original papers for detailed descriptions of the classifiers and Witten et al. [2011] for an overview of all approaches and classifiers.

[3]Technically, *ZeroR* predicts the mode for nominal classes and it predicts the mean value for numeric classes.

one classification rule for each feature where each feature value predicts the majority class for that value. From this set of rules, the classifier chooses the rule with the smallest error rate. This single rule is used for final predictions. Although it is also a simplistic classifier, OneR produces surprisingly accurate results compared to state-of-the-art algorithms [Holte, 1993].

8.2.2 Classifiers Based on Decision Rules

Classification algorithms based on decision rules use conditions to discriminate between the classes to be predicted. For the task at hand, the rules discriminate between the senses of the ambiguous target words. The conditions encode specific values of the features representing the target words' contexts. The three classifiers in this subsection differ from each other in the way how they create, represent, and apply these rules.

PART is a simple decision list classifier that recursively creates decision rules: it temporarily constructs a partial decision tree (according to *J48*, see below) for a set of training instances and creates a rule for the leaf node with the largest coverage; all instances covered by this rule are removed from the training set and the procedure is repeated for the smaller set of instances until there are no instances left. This algorithm does not store any trees and does not apply any global optimization. It is therefore fast and simple, yet has it been demonstrated to produce competitive results compared to other (rule-based) classifiers. [Frank and Witten, 1998]

DecisionTable represents rules in a tabular form: a set of conditions (for the WSD task, the features) is contrasted to a set of resulting values (i.e., word senses). For the WSD task, columns correspond to features and rows represent instances with specific values for these features and assigned word senses. For constructing a decision table, a subset of features and instances is selected from the training data with the help of a best-first search algorithm. During the classification process, a new instance is compared to all feature combinations stored in the decision

table, and, if there is at least one matching instance, the assigned word sense is the one which is stored with the majority of these matching instances. [Kohavi, 1995; Witten et al., 2011]

J48 is Weka's implementation of the popular C4.5 decision tree algorithm [Quinlan, 1993]. It induces decision trees recursively and in a top down procedure from training data: the most informative conditions (i.e., features) are put to the top of the tree and more general ones are added to the bottom – with the aim of achieving high classification performance with a minimal set of conditions. In order not to overfit the training data but be able to generalize to new instances, trees in J48 are pruned by default.[1] [Witten et al., 2011]

The classification of a new instance follows a path from the decision tree's root node to one of its leaf nodes – branching at each inner node according to the value of the corresponding feature. The finally assigned word sense is the one which is represented by the reached leaf node.

8.2.3 Instance-Based Classifiers (Lazy)

During the training phase, instance-based learners store all training instances in memory. Proper processing is postponed until classification, which is why instance-based classifiers are also referred to as lazy learners. For classification, new instances are compared to these stored instances and a similarity metric determines the k nearest neighbors, i.e., the k most similar instances, for a given test instance. The majority sense of these k nearest neighbors is assigned to a given test instance. [Màrquez et al., 2006; Witten et al., 2011]

The two relevant parameters of instance-based classifiers are: (i) the number k that defines how many nearest neighbors are considered for classification and (ii) the similarity metric used for measuring the similarity between instances.

IB1 sets k to 1 and is, thus, the simplest instance-based classifier [Aha et al., 1991]. It defines the closest training instance for a given test instance in

[1] The default confidence value for pruning is 25%.

terms of Euclidean distance, and predicts the class of this single nearest neighbor. In case of a tie, i.e., that the smallest distance occurs for more than one training instance, *IB1* takes the first instance found. *IB1* is one specific *IBk* classifier (see next item).

IBk is a k-nearest neighbor classifier which allows several parameter configurations, including the number of neighbors to use, the distance weighting method, and the algorithm for searching nearest neighbors. In its default configuration, *IBk* is equivalent to *IB1*.[1] [Witten et al., 2011]

8.2.4 Probabilistic Classifiers

The three classifiers described in this subsection are based on probabilistic computations.

NaiveBayes [Duda and Hart, 1973; John and Langley, 1995] is based on the Bayes theorem and naively assumes conditional independence of all features given a class.[2] During the training process, *NaiveBayes* estimates for individual features their probabilities to belong to specific classes. These probabilities are calculated on the basis of the frequencies with which the features occur for certain classes in the training data. For classification, *NaiveBayes* estimates for a new instance the conditional probabilities with which the instance belongs to a certain class. These probabilities are estimated by multiplying the individual probabilities for each feature to occur for a certain class. The classifier assigns the class (i.e., word sense), which has the highest overall probability (i.e., the product of the probabilities of the individual features to belong to a certain class) for the new instance. [Mitchell, 1997; Paliouras et al., 2000; Navigli, 2009; Witten et al., 2011]

A naive Bayes classifier is one specific Bayesian network classifier (see

[1] Due to this equivalence, WSD evaluation results in Section 8.3 are reported for only one of the two classifiers (namely, *IBk*).

[2] Albeit this strong assumption is practically seldom fulfilled, *NaiveBayes* performs well for the task of word sense disambiguation compared to other supervised machine learning methods.

next item) where all feature nodes have exactly one common parent node, which represents the class node [Witten et al., 2011].

BayesNet builds a Bayesian network by representing features as nodes and connecting these nodes in a directed acyclic graph [Bouckaert et al., 2013]. Each of the feature nodes defines a probability distribution that models the class probabilities given the possible feature values. The two relevant parameters for the *BayesNet* classifier are: (i) the search method to find possible networks and (ii) the evaluation method to qualify these networks. The default method to search for possible networks is *K2* [Cooper and Herskovits, 1991, 1992]. *K2* constructs a network by going through all feature nodes (assuming a given order) and adding edges to previously processed nodes if they maximize the score of the network. *SimpleEstimator* is set in Weka as the default network evaluation method, which estimates the conditional probability tables of a Bayesian network [Witten et al., 2011].

Logistic implements a multinomial logistic regression model, i.e., it implements a logistic regression variant which is able to handle multiple classes. Logistic regression is a probabilistic classifier which models the relationship between nominal classes (i.e., word senses) and a set of given features with a logistic regression function. By contrast to naive Bayes, these feature do not need to be statistically independent. [Witten et al., 2011]

Since the implementation of Weka's *Logistic* classifier is based on the work by le Cessie and van Houwelingen [1992], it uses a ridge estimator to improve the classification performance for large feature sets or for highly correlated features.

Note that multinomial logistic regression is equivalent to maximum entropy models [Klein and Manning, 2002; Yu et al., 2011; Wiriyathammabhum et al., 2012] – as introduced in Subsection 2.3.2.

8.2.5 Support Vector Machines

Support Vector Machines (SVMs) [Cortes and Vapnik, 1995; Vapnik, 1995] use hyperplanes to separate training instances into two classes. Therefore, training instances are mapped into the feature space by a function referred to as a *kernel*. The optimization goal of SVMs is to identify the hyperplane with the largest margin to the closest instances of both classes. In case the training instances are not separable by a hyperplane, a trade-off between a low training error and a large margin has to be made. This trade-off is controlled by the regularization parameter C. For classification, a test instance is mapped into the same feature space in which the hyperplane is located and classified according to the side of the hyperplane it belongs to. [Cortes and Vapnik, 1995; Witten et al., 2011]

SMO is Weka's implementation of the sequential minimal optimization algorithm for training a support vector machine classifier [Platt, 1998; Keerthi et al., 2001]. It uses pairwise classification (one-versus-one) for solving multi-class problems [Hastie and Tibshirani, 1998]. In its default configuration, Weka's *SMO* uses a linear model [Witten et al., 2011].

LibSVM is a library for support vector machines [Chang and Lin, 2011]. It is not part of the Weka tool suite, but an external library[1] that can easily be used in the Weka environment. *LibSVM* also implements a sequential minimal optimization algorithm for training support vector machines [Platt, 1998; Fan et al., 2005]. It uses a one-versus-one approach for multi-class classification [Knerr et al., 1990]. In its default configuration, *LibSVM* uses a Gaussian kernel [Chang and Lin, 2011].

8.2.6 Combination

Previous WSD studies [Florian et al., 2002; Florian and Yarowsky, 2002; Klein et al., 2002; Mihalcea et al., 2004] have shown that the performance of multiple

[1] Chih-Chung Chang and Chih-Jen Lin developed *LibSVM*, available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`.

classifier systems outperform single classifiers. In this chapter, the following two heterogeneous methods for combining single classifiers are used:

Vote provides a generic method for combining individual classifiers [Kittler et al., 1998; Kuncheva, 2004]. Its two relevant parameters allow selecting (i) the set of individual classifiers to be combined and (ii) the rule used for their combination. The default combination rule in Weka averages the probability estimates of the individual classifiers and selects the class with the highest average.[1]

In the current WSD setup, *Vote* combines four heterogeneous methods – one from each of the previous subsections: the decision tree classifier *J48*, the instance-based *IBk* classifier, the probabilistic *NaiveBayes* classifier, and the *SMO* support vector machine classifier.[2] Since these four classification algorithms are all based on different underlying theories, they are likely to produce diverse results and it is thus likely that they complement each other well in a combined classifier.

AdaBoostM1 is a method for boosting another base classifier: the base classifier is initially run on the unweighted training data; after each run, all instances in the training data are reweighted to give higher weights to misclassified instances and the base classifier is rerun on this reweighted data; this process of running and reweighting is repeated for a fixed number of iterations. In its default configuration, the number of iterations is set to 10 and the base classifier to be boosted is *DecisionStump*, a one-level decision tree classifier that is intended to be used with boosting.[3]

[1] In order to test the same combination rule used for the knowledge-based experiments in Chapter 7, *Vote* has also been applied with simple majority voting. However, the results in this chapter are reported with Weka's default combination rule, since it performed slightly better than the combination by simple majority voting.

[2] Note that this set of classifiers is selected for combination mainly due to their heterogeneous underlying classification theories, with only a minimal amount of comparison. In future work, more effort should be put into identifying the most optimal combination of classifiers.

[3] A decision stump classifier was also used in related WSD studies which applied AdaBoost to WSD [Lee and Ng, 2002; Martínez et al., 2002; Màrquez et al., 2006; Kopeć et al., 2012]. Initial experiments with boosting other classifiers such as *J48*, *IBk*, *NaiveBayes*, and *SMO* did not or only marginally improve the experimental results in this chapter over the single classifier alternatives.

[Freund and Schapire, 1996; Witten et al., 2011]

8.2.7 Automatic Feature Selection

Most machine learning classifiers are capable to determine the relevant features that are most suitable for their classification process. Nevertheless, many studies [Hall, 1999; Mihalcea, 2002a,b; Guyon and Elisseeff, 2003; Liu and Yu, 2005] have shown that the performance of most classifiers is negatively affected by irrelevant features and that a preselection of the most relevant features helps to improve the classifiers' performance. Feature preselection reduces the dimensionality of the data by removing irrelevant features and thereby optimizes the predictive accuracy of machine learning algorithms. [Hall, 1999; Manning et al., 2008; Witten et al., 2011]

AttributeSelectedClassifier has the goal of improving the performance of a base classifier. It can be regarded as a wrapper that performs automatic feature[1] selection on the training data before executing a specified base classifier on these reduced data sets. [Witten et al., 2011]

AttributeSelectedClassifier has three parameters: (i) the search method to find good subsets of features, (ii) the evaluation method to qualify these feature subsets, and (iii) the base classifier to execute on the feature-selected data sets. *BestFirst* is set in Weka as the default method to search subsets of features (see Witten et al. [2011] for details). The default feature evaluation method is *CfsSubsetEval* [Hall, 1999]. *CfsSubsetEval*[2] evaluates each feature's individual predictive ability and the redundancy among features in order to identify feature sets with low correlations among the features but high correlations with the class.

In order to investigate the impact of automatic feature selection in the current WSD setup (see Section 8.3.2 below), it is applied to all classifiers described above, i.e., all classifiers are individually used as base classifiers for *AttributeSelectedClassifier*.

[1] As explained in Footnote 1 on page 217 the terms *feature* and *attribute* are used synonymously in this thesis.

[2] The *cfs* in *CfsSubsetEval* stands for *correlation-based feature selection*.

8.3 Evaluating Supervised Machine Learning Applied to WSD

In order to evaluate the supervised machine learning setup for German word sense disambiguation, an extensive set of experiments using a wide range of machine learning features (see Section 8.1 above) as well as many different supervised classification algorithms (see Section 8.2 above) is performed. As described in the previous section, the implementation relies on the Weka machine learning tool suite.[1] The evaluation in this section considers all three sense-annotated corpora available for German: the semi-automatically constructed, web-harvested WebCAGe (see Section 6.3), the manually sense-annotated TüBa-D/Z treebank (Section 6.4), and the web-harvested and manually sense-annotated deWaC (Section 6.5).

In order to assure a minimum level of meaningfulness when experimenting with supervised machine learning methods, a critical mass of annotated material is required per lemma. Thus, only a certain subset of all available gold standard data, which fulfills certain criteria as discussed and specified in Section 6.1, is used for the evaluation in this chapter. Further, to allow a more realistic, meaningful, and accurate estimate of an algorithm's ability to generalize after several experiments with distinct classifiers, parameters, and features on the training data, testing is performed on a previously unseen portion of the annotated gold standard data. This procedure is in line with most related works on supervised disambiguation of a lexical sample (see Section 6.1 for pointers to corresponding previous works). In fact, since the availability of sense annotations is sparse, the tuning of the system is evaluated by 10-fold cross-validation (see Subsection 2.2.2) on all available training data. During

[1] In the implementation, version 3.6.10 of Weka's Java API is used. The main reason why the Java API is used rather than the Weka Experimenter is because the Experimenter does not support all functionality required in the experiments at hand: most importantly, it does not support the use of a separate test set. Note, however, that the used version of the API does not yet support the calculation of micro average values as described in Subsection 2.2.2. 'Not yet', because the developer version 3.7.x supports this functionality. The Weka API in its *stable* version 3.6.10 only allows calculating averages weighted by classes, but since this way of calculating averages is not common for the task of WSD, the calculation of average values is self-implemented.

this tuning process, the test set is held back and used only to obtain final evaluation results.[1]

Since the sets of word senses and thus the sets of classes to be predicted for the WSD task differs for each lemma, the supervised classification procedure in this chapter is performed separately for each lemma. The separate classification of each word lemma is sometimes also referred to as *word-experts* [Berleant, 1995]. This approach is in line with most related works on supervised disambiguation of a lexical sample (see Subsection 2.3.2 for pointers to corresponding previous works).

As for the previous chapter on knowledge-based WSD, performance of the algorithms is measured in terms of coverage, recall, precision, and F_1 – calculated by their standard formulas as described in Subsection 2.2.1. Results are reported as overall numbers, i.e., micro-averaged over all annotated instances.[2]

The following subsections focus on several evaluation aspects under consideration: including a comparison of several classification algorithms (Subsection 8.3.1), a detailed analysis of machine learning features (Subsections 8.3.2 and 8.3.3), and an investigation of the influence of syntactic structures on the disambiguation performance for verbs (Subsection 8.3.4).

8.3.1 Overview of WSD Results Using All Features

Table 8.4 includes WSD performance scores in terms of the F_1-measure[3] for 13 supervised classifiers (as described in Section 8.2) evaluated on the test sets of the available sense-annotated corpora (as described in Chapter 6) – taking into account all available features (as described in Section 8.1). The principle structure of the table prints the results for each classifier (specified on the left

[1]Note that this combination of 10-fold cross-validation for algorithm tuning with a held back test set for obtaining final results was also performed by previous studies, including Hoste et al. [2002a], Decadt et al. [2004], Escudero Bakx [2006], and Biemann [2012]. Also see Appendix D for a comparison between the WSD results obtained by cross-validation with the results obtained on a separate, unseen test set.

[2]See Subsection 2.2.2 for more information on micro- and macro-averages.

[3]Since the supervised WSD setup tries to assign a word sense for each instance, its coverage is 100% and, thus, the value of F_1 is equal to the values of recall and precision (see Subsection 2.2.1). For this reason Table 8.4 does not explicitly include precision and recall, but reports F_1-scores.

side of the table) one below the other. As specified in column *POS*, micro-averaged scores are given for the three word classes of adjectives, nouns, and verbs.

Table 8.4: WSD results for several supervised classifiers (F-score).

Classifier		**POS**	**WebCAGe**	**TüBa-D/Z (manual)**	**TüBa-D/Z (automatic)**	**deWaC**
Baselines	*ZeroR*	Adj.	80.49			60.00
		Nouns	79.96	80.54	80.54	60.20
		Verbs	76.92	*68.19*	*68.19*	48.17
	OneR	Adj.	82.93			*36.00*
		Nouns	*62.42*	*49.05*	*34.14*	*45.92*
		Verbs	*73.08*	81.19	74.37	*43.90*
Decision rules	*PART*	Adj.	87.80			64.00
		Nouns	86.22	87.92	88.25	60.20
		Verbs	86.54	85.68	77.84	53.05
	Decision-Table	Adj.	85.37			60.00
		Nouns	81.42	86.78	86.86	65.31
		Verbs	75.00	85.79	78.15	51.22
	J48	Adj.	87.80			64.00
		Nouns	88.52	89.32	89.21	65.31
		Verbs	86.54	86.25	78.58	55.49
Lazy	*IBk*	Adj.	82.93			64.00
		Nouns	83.72	84.14	85.61	67.35
		Verbs	88.46	79.46	73.59	55.49
Probabilistic	*Naive-Bayes*	Adj.	82.93			64.00
		Nouns	89.35	85.10	84.65	64.29
		Verbs	92.31	83.07	77.06	54.27
	Bayes-Net	Adj.	*78.05*			52.00
		Nouns	90.40	88.73	89.17	64.29
		Verbs	90.38	86.28	80.63	59.15
	Logistic	Adj.	85.37			56.00
		Nouns	89.77	86.45	86.82	60.20
		Verbs	92.31	77.77	74.76	49.39
SVM	*SMO*	Adj.	82.93			**68.00**
		Nouns	91.44	90.57	**90.86**	**68.37**
		Verbs	96.15	86.53	79.99	**60.37**
	Lib-SVM	Adj.	87.80			60.00
		Nouns	84.13	81.06	80.84	59.18
		Verbs	92.31	78.30	72.57	48.78
Combined	*Vote*	Adj.	**90.24**			64.00
		Nouns	**92.28**	**90.71**	90.38	67.35
		Verbs	**98.08**	**86.71**	**80.74**	59.76
	Ada-Boost-M1	Adj.	82.93			52.00
		Nouns	84.97	85.21	85.50	61.22
		Verbs	88.46	84.66	77.31	53.66

For each classifier, the table provides results for each of the sense-annotated

corpora (specified in the header column). The labels *WebCAGe* and *deWaC* unambiguously refer to the particular corpora with the same names, while the caption *TüBa-D/Z (manual)* refers to TüBa-D/Z with its manual linguistic treebank annotations (as described in Subsection 5.3.1) and *TüBa-D/Z (automatic)* refers to TüBa-D/Z with automatic linguistic annotations (as described in Sections 6.6 and 8.1.1). The reason why Table 8.4 does not include values for adjectives in the TüBa-D/Z is simply because the treebank contains only sense annotations for nouns and verbs. For each corpus, the highest result achieved for a particular word class is highlighted in boldface; the lowest values are italicized.

Comparison of Classifiers

Table 8.4 shows that the 13 classifiers produce heterogeneous performance – depending on the gold standard corpus and the word class. For instance, *IBk* performs better on verbs in WebCAGe than on nouns in WebCAGe, while it performs worse on verbs in TüBa-D/Z than on nouns in TüBa-D/Z; on the WebCAGe corpus, *DecisionTable* performs best for adjectives and worst on verbs, while *NaiveBayes* conversely performs best for verbs and worst for nouns in WebCAGe – to give but two examples. This behavior that several classifiers produce heterogeneous results for diverse data sets and word classes corroborates the finding by Yarowsky and Florian [2002] on their experiments with several supervised machine learning algorithms (cosine vector, decision lists, transformation-based learning, naive Bayes, BayesRatio, and feature-enhanced naive Bayes) on multiple datasets for several languages (English, Spanish, Swedish, and Basque). However, there are several recurring patterns in this apparent heterogeneity, which are discussed below (in the paragraph entitled *Comparison of Word Classes and Sense-Annotated Corpora* on page 265 below). In general, it is important to note that there are no extreme differences between the individual classifiers. Their average results are all in the same order of magnitude.

The most striking finding among the results presented in Table 8.4 is that the overall best performing classifiers are the single *SMO* support vector ma-

chine classifier and the combined *Vote* classifier. These two classifiers achieve consistently high performance, i.e., their results are among the highest results across all word classes and corpora and they both achieve highest overall results in at least four of the ten *POS–corpus* combinations. That is, *SMO* achieves overall the best results for adjectives, nouns, and verbs in deWaC and for nouns in the TüBa-D/Z with automatic linguistic annotations, and *Vote* achieves overall the best results for all word classes in WebCAGe, for nouns in the TüBa-D/Z with manual linguistic annotations and for verbs in both TüBa-D/Z versions.

Good performance was to be expected both for SVMs and combined algorithms, since both of them previously demonstrated high performance among supervised machine learning methods when applied to the task of WSD: for example, Lee and Ng [2002], who experimented with the four classifiers[1] of SVMs, naive Bayes, AdaBoost, and decision trees, obtained best WSD results with linear SVMs. Zavrel et al. [2000] also found SVMs overall to perform best when comparing several classification algorithms, including SVMs, naive Bayes, memory-based algorithms, decision trees, maximum entropy, rule induction, and neural networks. For Agirre and Martínez [2004], who experimented with decision lists, naive Bayes, vector space models, and SVMs and with combinations thereof, SVMs obtained – among the single classifiers – best results for Basque WSD and second best results for English WSD. In the experiments by Màrquez et al. [2006], SVMs also proved superior performance when compared to naive Bayes, k-nearest neighbor, decision lists, and AdaBoost. The explanation for a good performance of SVMs on the task of WSD is that SVMs are capable of maintaining complex decision boundaries without overfitting, which is especially important for WSD given sparse training data for each lemma [Zavrel et al., 2000]. The combination of single classifiers also often (e.g., by Florian et al. [2002] and Klein et al. [2002]) showed superior WSD performance compared to the single classifier alternatives; many of the best performing systems at SensEval-3 were combined classifiers [Mihalcea et al., 2004]. The explanation for a good performance of combined methods is that

[1] Lee and Ng [2002] also used the implementations provided by Weka.

combining heterogeneous classification algorithms that produce diverse results are likely to complement each other well and increase the WSD performance in a combined classifier [Jaeger et al., 2008].

Contrasting results are obtained for the second employed support vector machine classifier. The *LibSVM* classifier is one of the worst performing classifiers for both TüBa-D/Z versions and deWaC. Only for WebCAGe, *LibSVM* obtains competitive results. The real explanation for what looks like contradictory findings (i.e., one SVM classifier performs best, while the other SVM classifier performs worst) must have to do with the implementations – more specifically with the default kernel functions – applied in this chapter (see Subsection 8.2.5 above). While *SMO* defaults to a linear model [Witten et al., 2011], *LibSVM* uses by default a Gaussian kernel [Chang and Lin, 2011]. The observed behavior corroborates the finding by related studies (including Lee and Ng [2002] and Wiriyathammabhum et al. [2012]) that achieved superior WSD results by linear SVMs than by using non-linear kernels such as polynomial or Gaussian. The reason for this behavior is probably due to the large number of employed features when using the full set of features as described in Section 8.1. According to Hsu et al.'s [2003] practical guide on how to apply support vector machines, linear kernels are more suitable than non-linear kernels, if the number of features is large.[1]

Due to the skewed distribution of word senses, the most frequent sense baseline (i.e., *ZeroR*) is difficult to beat and WSD systems are usually – if at all – able to beat it only by a small margin [McCarthy, 2009]. The second baseline employed in the current setup (i.e., *OneR*) makes use of only one classification rule drawn from a single feature. Although it also is a very simplistic classifier, *OneR* is said to produce surprisingly accurate results compared to state-of-the-art algorithms [Holte, 1993]. However, for most *POS–corpus* combinations, *ZeroR* outperforms *OneR*. This was to be expected since it is unlikely that a single feature provides enough evidence so as to have a higher impact on the disambiguation than given by the skewed distribution of word senses. Apart from the low performance for *LibSVM*, whose results lie only minimally above

[1] This subject is revisited in the following Subsection 8.3.2 on automatic feature selection, where it is shown that *LibSVM* performs better with a reduced set of features.

the most frequent sense baseline, for most *POS–corpus* combinations the other classifiers perform significantly better than the two baseline classifiers. This outcome confirms the general viability of the proposed supervised machine learning approach for German word sense disambiguation.

Comparison of Word Classes and Sense-Annotated Corpora

Comparing the results for the three sense-annotated corpora in Table 8.4, the WSD results obtained for WebCAGe are the best, the results obtained for the TüBa-D/Z treebank second, while the results for deWaC are the worst. The most apparent reason for the better performance on WebCAGe is the lower polysemy of words. The other way around, the higher polysemy for words in TüBa-D/Z and deWaC make the sense disambiguation more difficult. That is, while the average polysemy of sense-annotated words (in the gold standards used for evaluating supervised WSD systems) is 2.2 in WebCAGe, the average polysemies of sense-annotated words in TüBa-D/Z and deWaC are 2.8 and 3.0, respectively (as documented in Chapter 6).

A comparison of the results for the three different word classes yields the following tendencies (see Table 8.4):

- When measured on TüBa-D/Z and deWaC, the WSD performance of supervised machine learning classifiers is better for nouns than for verbs. This finding corroborates the results reported by Kilgarriff and Rosenzweig [2000] and Pradhan et al. [2007] for English WSD.

- In accordance to the previous bullet point, the WSD performance for adjectives in deWaC is also better than for verbs.

- By contrast, for WebCAGe, most classifiers perform better for verbs than for adjectives and nouns.

Although the first two proportions correspond with the outcome of the knowledge-based WSD experiments in Chapter 7, the underlying reason for this behavior cannot be adopted. That is, semantic relatedness measures depend on GermaNet's hierarchies and since these hierarchies are much more

shallow for verbs and adjectives than for nouns, path-based and information-content-based measures are much more effective for nouns while more of an experimental nature when applied to verbs and adjectives [Pedersen et al., 2005, page 27].

Since the supervised machine learning experiments in this chapter do not rely on GermaNet's hierarchies, there must be another explanation for why the prediction of noun senses is more efficient than the disambiguation of verbs and adjectives in TüBa-D/Z and deWaC, while the disambiguation of verb senses is better for WebCAGe compared to the other two word classes. In fact, the WSD performance in the current setup depends on many conditions. For instance, it depends on (i) the extracted features from the context, which are clearer and more consistently distinct for particular word classes and corpora, (ii) the available training data, which again is more suitable and more efficient for particular word classes and corpora, and (iii) the specific polysemy, which is also a good indicator for the difficulty of particular lemmas.

The real explanation for the observed WSD performance is the complex interaction of many dimensions – in particular it is the interaction of the three aforementioned factors (i) to (iii). For example, the higher polysemy of verbs in deWaC compared to adjectives and nouns partly explains the worse performance of verbs in deWaC. The higher average frequency of annotated occurrences per lemma for nouns in TüBa-D/Z compared to verbs in TüBa-D/Z influences the better performance of nouns in that corpus.

The lower performance for deWaC must have to do with very heterogeneous contexts for the sense-annotated words in that corpus. This fact is partly reflected by the number of features. In deWaC, lemmas have on average considerably fewer features (especially for adjectives and nouns) compared to the corresponding numbers in the other corpora – as shown in Table 8.3 (in Subsection 8.1.11 above). Considerably fewer features are due to fewer context lemma features, which make up the majority class of features in the current setup. If this type of feature is sparse, it means that only few lemmas are common to at least three annotated contexts (see Subsection 8.1.3 for a description of the context lemma features). Of course, few lemmas that occur in at least three annotated contexts can also be caused by few annotations

per lemma. But since this frequency is similar in WebCAGe where there are three to four times as many features than in deWaC, the lower performance for deWaC must have to do with very heterogeneous contexts.

Manual versus Automatic Linguistic Annotations

The supervised WSD systems implemented in this chapter rely on various features. As described in Section 8.1 above, the features employ different kinds of linguistic information, such as information on surface forms of words, co-occurring lemmas, POS tags, morphology, and sentence structures. While this information is – for WebCAGe and deWaC – gained solely from automatic annotation tools, the sources for TüBa-D/Z are twofold: (i) manual linguistic annotations as described in Subsection 5.3.1 as well as (ii) automatic annotations as for WebCAGe and deWaC as described in Sections 6.6 and 8.1.1. There are two main reasons why the TüBa-D/Z treebank texts are automatically annotated with linguistic annotation, although the treebank already contains manual annotation for all linguistic phenomena in question. Firstly, to allow a direct comparison of the WSD results for all three corpora with the same compilation of linguistic details. Secondly, to study the influence of linguistic annotation quality (manual versus automatic) on the WSD results.

Column *TüBa-D/Z (manual)* in Table 8.4 shows the WSD results obtained by using the TüBa-D/Z with its manual linguistic treebank annotations and column *TüBa-D/Z (automatic)* shows the corresponding results obtained by using the automatic linguistic annotations. Since the word sense annotations are identical in both versions, the reported most frequent sense baseline, i.e., classifier *ZeroR*, is – by definition – identical.

With average improvements between 3.0 and 7.8 F_1-score points for the different classifiers (disregarding the baselines) compared to the results obtained by the automatic features, the manual features have a consistently positive impact on verbs. By contrast, the impact of manual features on nouns is much less noticeable ($< |1.5|$) and clearly not only positive. With average changes between –1.5 and +0.5 F_1-score points, manual features surprisingly have even a negative impact on the majority of seven (versus four) of the classifiers (again

disregarding the baselines).

The real explanation for what looks like contradictory behavior must have to do with the kind of information on which the corresponding features are based: that is, the most effective features for verbs in TüBa-D/Z are based on syntactic information such as 'constituent structure' features and 'verbal frame' features (see paragraph *Automatically Selected Features* on page 273). Syntactic structures are much more difficult to annotate – and, thus, more inaccurate to obtain by automatic tools – than tokens, lemmas, and part-of-speech tags, which are the basis for the most frequently used features for nouns. On the other side, the very effective 'translation' feature, which is especially helpful for nouns (again, see paragraph *Automatically Selected Features* on page 273), but which is available only for automatically annotated texts (see Subsection 8.1.10), significantly increases the performance for nouns in connection with the automatically gained features.

Comparison to Knowledge-Based Results

A comparison of the results from the knowledge-based WSD experiments in Chapter 7 (Section 7.3), which were based on semantic relatedness measures, with the WSD results obtained by supervised machine learning classifiers reported in Table 8.4 corroborate the finding of previous research that supervised machine learning systems perform much better than knowledge-based approaches to WSD [Kilgarriff and Rosenzweig, 2000; Mihalcea, 2006; McCarthy, 2009; Navigli, 2009].

While the knowledge-based sense disambiguation in Chapter 7 performed well for nouns, it was little suitable for disambiguating adjective and verb senses: the best performing knowledge-based methods achieved average F-scores between 53 and 67 for nouns, but only between 40 and 56 for adjectives and between 28 and 56 for verbs – depending on the gold standard corpus.

By contrast, the best performing supervised classifiers achieve average F-scores between 68 and 90 for adjectives, between 68 and 92 for nouns, and between 60 and 98 for verbs – depending on the corpus used for evaluation, but without any boosting such as feature selection. That is, the supervised machine

learning methods outperform the knowledge-based systems by a very wide margin: by about 28–34 F_1 points for adjectives, by about 13–36 for nouns, and even by about 32–52 for verbs – again, depending on the gold standard. It is, however, important to keep in mind that the supervised WSD approach is applicable to only a subset of word token annotations (see Section 6.1 for details).

There are two crucial parallels between the results of the knowledge-based and the supervised machine learning experiments. Firstly, combined methods perform better or at least as good as the strongest single methods. The obvious explanation for this behavior is that the individual algorithms produce diverse results that complement each other well in a combined approach. Secondly, among the three gold standard corpora considered, the results obtained for deWaC are the worst – both for the knowledge-based and the supervised systems. The two most apparent reasons for the bad performance on the deWaC corpus are the much higher polysemy of annotated words as well as more heterogeneous contexts compared to the other two corpora (see the paragraph entitled *Comparison of Word Classes and Sense-Annotated Corpora* on page 265 above for a detailed discussion), which both make the sense disambiguation more difficult.

8.3.2 WSD with Automatic Feature Selection

Machine learning classifiers are generally capable to determine the relevant features that are most suitable for the classification process. Nevertheless, many studies [Hall, 1999; Mihalcea, 2002a,b; Guyon and Elisseeff, 2003; Liu and Yu, 2005; Dinu and Kübler, 2007; Kübler and Zhekova, 2009] have shown that the performance of most classifiers is negatively affected by irrelevant features and that a preselection of the most relevant features helps to improve the classifiers' performance. Since feature preselection reduces the dimensionality of the data by removing irrelevant features and thereby optimizes the predictive accuracy of machine learning algorithms, it is popular to automatically preselect features before applying the proper classification algorithm. [Hall, 1999; Manning et al., 2008; Witten et al., 2011]

The purpose of this subsection is twofold: (i) to investigate the impact of automatic feature selection on the performance of supervised classifiers when applied to German WSD and (ii) to identify the most popular selected features. Feature selection is realized with Weka's correlation-based feature selection algorithm *CfsSubsetEval* (see the description of the *AttributeSelectedClassifier* in Subsection 8.2.7 above).

Comparison of Classifiers

Table 8.5 presents the WSD results with automatic feature selection in comparison to the WSD results without feature selection. The principle structure of the table follows the layout of Table 8.4 above, which presented the WSD results without feature selection: it includes WSD performance scores in terms of the F_1-measure for the same 13 supervised classifiers (as described in Section 8.2) evaluated on the available sense-annotated corpora[1] (as described in Chapter 6).

For an easier comparison, each entry in Table 8.5 reports the absolute F-score obtained with automatic feature selection as well as the relative difference[2] compared to the corresponding result without feature selection from Table 8.4 (i.e., values preceded by + or – symbols). For each corpus, the highest result achieved for a particular word class is highlighted in boldface; the lowest values are italicized. Particularly high increases or decreases in performance (with absolute values above 10) are analogously highlighted. The reason why feature selection does not influence the performance of *ZeroR* is by definition. That is, since the most frequent sense baseline relies solely on the distribution of senses rather than on any features, the results of the *ZeroR* classifier are identical with and without feature selection.

The results in Table 8.5 yield the tendency that automatic feature selection improves the average performance for several classifiers. It is, thus, generally a very simple approach to boost the performance of a WSD system and again

[1]Here, again, caption *TüBa-D/Z (manual)* refers to TüBa-D/Z with its manual linguistic treebank annotations and *TüBa-D/Z (automatic)* refers to TüBa-D/Z with automatic linguistic annotations.

[2]This value is printed only if the corresponding difference is unequal to zero.

Table 8.5: WSD results for several supervised classifiers with automatic feature selection (F-score; +/– difference compared to results without feature selection from Table 8.4).

Classifier		POS	WebCAGe		TüBa-D/Z (manual)		TüBa-D/Z (automatic)		deWaC	
Baselines	*ZeroR*	Adj.	*80.49*						60.00	
		Nouns	*79.96*		80.54		80.54		60.20	
		Verbs	76.92		*68.19*		*68.19*		48.17	
	OneR	Adj.	85.37	+2.4					*40.00*	+4.0
		Nouns	80.17	**+17.8**	*76.43*	**+27.4**	*71.37*	**+37.2**	*52.04*	+6.1
		Verbs	*73.08*		81.62	+0.4	75.72	+1.3	*46.34*	+2.4
Decision rules	*PART*	Adj.	**87.80**						56.00	–8.0
		Nouns	87.06	+0.8	88.55	+0.6	88.55	+0.3	**67.35**	+7.1
		Verbs	78.85	–7.7	86.25	+0.6	77.91	+0.1	50.00	–3.0
	Decision-Table	Adj.	82.93	–2.4					60.00	
		Nouns	84.76	+3.3	87.37	+0.6	87.30	+0.4	**67.35**	+2.0
		Verbs	75.00		86.00	+0.2	78.72	+0.6	48.78	–2.4
	J48	Adj.	**87.80**						60.00	–4.0
		Nouns	86.43	–2.1	88.51	–0.8	88.40	–0.8	66.33	+1.0
		Verbs	78.85	–7.7	86.89	+0.6	78.69	+0.1	**58.54**	+3.0
Lazy	*IBk*	Adj.	*80.49*	–2.4					64.00	
		Nouns	88.94	+5.2	89.28	+5.1	89.43	+3.8	66.33	–1.0
		Verbs	**94.23**	+5.8	85.86	+6.4	78.58	+5.0	56.10	+0.6
Probabilistic	*Naive-Bayes*	Adj.	85.37	+2.4					68.00	+4.0
		Nouns	89.14	–0.2	89.43	+4.3	89.50	+4.8	64.29	
		Verbs	92.31		87.56	+4.5	80.21	+3.1	57.93	+3.7
	Bayes-Net	Adj.	82.93	+4.9					68.00	**+16.0**
		Nouns	89.35	–1.1	89.21	+0.5	89.39	+0.2	63.27	–1.0
		Verbs	84.62	–5.8	87.56	+1.3	80.63		57.93	–1.2
	Logistic	Adj.	*80.49*	–4.9					52.00	–4.0
		Nouns	84.97	–4.8	85.28	–1.2	86.60	–0.2	61.22	+1.0
		Verbs	86.54	–5.8	81.37	+3.6	74.27	–0.5	51.83	+2.4
SVM	*SMO*	Adj.	82.93						68.00	
		Nouns	88.10	–3.3	**90.57**		**90.31**	–0.5	65.31	–3.1
		Verbs	90.38	–5.8	86.00	–0.5	79.89	–0.1	54.88	–5.5
	Lib-SVM	Adj.	*80.49*	–7.3					64.00	+4.0
		Nouns	86.43	+2.3	85.90	+4.8	86.01	+5.2	61.22	+2.0
		Verbs	86.54	–5.8	83.46	+5.2	77.06	+4.5	48.17	–0.6
Combined	*Vote*	Adj.	82.93	–7.3					**72.00**	+8.0
		Nouns	**89.56**	–2.7	89.98	–0.7	89.98	–0.4	65.31	–2.0
		Verbs	92.31	–5.8	**87.73**	+1.0	**80.77**	+0.0	**58.54**	–1.2
	Ada-Boost-M1	Adj.	85.37	+2.4					56.00	+4.0
		Nouns	86.01	+1.0	85.17	–0.0	85.87	+0.4	66.33	+5.1
		Verbs	86.54	–1.9	84.59	–0.1	76.92	–0.4	52.44	–1.2

confirms the general viability of the proposed supervised machine learning approach for German word sense disambiguation. Automatic feature selection particularly improves the overall performance of classifiers *OneR*, *IBk*, *NaiveBayes*, and *LibSVM*. The increase in performance for *IBk* with automatic feature selection was previously reported by Mihalcea [2002a,b]. One explanation for this increase is that *IBk* is particularly sensitive to irrelevant features because its decision considers only few nearest neighbors (in the current setup where k is set to 1, in fact only the single closest instance is considered) [Witten et al., 2011, page 308]. The observation for *NaiveBayes* corroborates, for example, the finding by Lee and Ng [2002] that this classifier performs better with feature selection. The improvement for *LibSVM* confirms the hypothesis made in Subsection 8.3.1 above that the application of support vector machines with non-linear kernels (*LibSVM* uses by default a Gaussian kernel) are more suitable if the number of features is restricted rather than arbitrarily large.

However, the observed influence of automatic feature selection on the WSD performance is in some cases heterogeneous. It very much depends on the classifier, the word class, and the corpus used for evaluation whether and how much automatic feature selection influences the disambiguation performance. The impact of the *OneR* classifier on the disambiguation of nouns has the most extreme improvement when comparing all classifiers in Table 8.5: +18 for WebCAGe, +27 for the manual TüBa-D/Z, +37 for the automatic TüBa-D/Z, and +6 for deWaC . Another extreme improvement of +16 is reported for *BayesNet* when evaluated on adjectives in deWaC. With absolute differences of at most 8 F_1-score points for all other results, automatic feature selection does much less noteworthy influence the performance of the other classifiers.

In fact, the performance for some classifiers decreases slightly – depending on the gold standard used for evaluation. For example, the performance of *J48*, *Logistic*, and *Vote* decreases when measured on WebCAGe, and the performance of *SMO* decreases for all corpora. The latter observation confirms the finding by Lee and Ng [2002] that linear support vector machines (*SMO* uses by default a linear model) perform best without feature selection.

Even if there are no large deviations (besides the *OneR* baseline) when comparing the results of the individual classifiers, it is important to notice

that *SMO* and *Vote* are still overall the best performing classifiers. Due to three main reasons, the following subsections report performance results for the *SMO* classifier only (and, where appropriate, also for the *ZeroR* baseline):

(i) If all numbers in the following tables and subsections were given for all available classifiers, the amount of reported performance results would explode without much information gain since there are generally no extreme discrepancies between the results of the individual classifiers (their performances are all in very similar ranges). That is, to allow more detailed discussions of the reported results and to avoid unnecessary overhead, the following experiments are reported for one classifier only.

(ii) Since *SMO* has proven – with and without feature selection – to be among the best performing classifiers, consistent and reliable behavior is expected without extreme deficiencies.

(iii) In the literature, support vector machines are – besides naive Bayes and instance-based classifiers – among the most frequently used classifiers for WSD and have shown superior performance compared to other machine learning classifiers for the task at hand (see Subsection 2.3.2 for more details and for pointers to corresponding related works).

Automatically Selected Features

An inspection of the frequencies with which certain features are automatically selected by the automatic feature selection algorithm allows identifying the most popular and most effective features [Hall, 1999]. For each sense-annotated corpus and word class, Table 8.6 lists the selected features[1] (as described in

[1]The labels in Table 8.6 correspond to the feature labels introduced in Section 8.1. Note that the 'context lemma' features are grouped according to the six types of context windows specified in Subsection 8.1.3, because regarding each lemma-specific feature individually would give a wrong impression: since the individual 'context lemma' features suffer from data sparseness (i.e., there is no single lemma – excluding stopwords – that occurs in multiple contexts of many different target word lemmas), they would be moved to the bottom of the list. For reasons of space, some of the labels are abbreviated: the relevant contexts for the corresponding 'context lemma' features are abbreviated as *SB* (referring to the boolean variant of the inner-sentential context), *6SB* (referring to the boolean variant of the 6-word, inner-sentential context window), and *50B* (boolean variant of the 50-word context window);

Section 8.1 above) in decreasing order of frequency. The percentages specify for how many lemmas the feature in question is automatically selected.

Table 8.6 conveys the following general tendencies: features based on contextual clues around the target word (i.e., all features that disregard information on the target word itself) are much more effective than features based on the target word itself (i.e., the four surface features, the six morphological features, and the single 'pos' feature[1]). This was to be expected. The interpretation of this behavior is that the information extracted from the context around the target word is more informative and predictive for the disambiguation process than information about the target word itself. It corroborates the previous finding by Kopeć et al. [2012] for Polish WSD where *thematic features* (which correspond to the features here referred to as 'context lemmas') were selected most frequently while *keyword features* (which are based on the target word itself) were selected most seldom from their set of heterogeneous features.

The pragmatic explanation why features from the 'context lemma' feature group are overall most frequently selected is that the majority of features are context lemmas (see Subsection 8.1.11). The fact that the boolean variants of these 'context lemma' features are selected more often than the numeric variants is not surprising given the overall sparse amount of training data. The individual 'context lemma' features suffer from data sparseness: even if a lemma occurred a certain amount of times overall in the training data contexts, it certainly very scarcely occurs multiple times in the same context. Thus, certain feature values with counts above 1 are usually assigned to single instances only, which makes the boolean variant more efficient.

The reason why features that encode part-of-speech tags are the second most frequently selected feature type is the same as for the 'context lemma' features: they constitute a larger amount of features than the other feature types (see Subsection 8.1.11). Those 'part-of-speech' features that encode POS

the corresponding feature types in the numeric versions are abbreviated as *SN*, *6SN*, and *50N*, respectively. The less alienated label abbreviations are *part_of_conj* (part_of_conjunction), *grammatical_func* (grammatical_function), and *verbs_ign_aux* (verbs_ignoring_aux).

[1]Here, the single 'pos' feature encoding the target word's part-of-speech tag (available for adjectives and verbs) is meant rather than all features in the POS feature group.

Table 8.6: Frequency of automatically selected features.

	WebCAGe	TüBa-D/Z (manual)	TüBa-D/Z (automatic)	deWaC
Adjectives	66.7% ctx_lemmas_50B 66.7% ctx_lemmas_SB 66.7% pos_1_right 33.3% ctx_lemmas_50N 33.3% headline 33.3% pos 33.3% pos_2_left 33.3% pos_2_right 33.3% pos_3_left 33.3% translation			75.0% pos_1_right 50.0% pos_2_left 25.0% ctx_lemmas_50B 25.0% pos_2_right 25.0% sentence_length 25.0% verbs 25.0% verbs_ign_aux 25.0% verbs_pos
Nouns	33.3% ctx_lemmas_6SB 33.3% ctx_lemmas_50B 33.3% ctx_lemmas_50N 33.3% ctx_lemmas_SB 33.3% ctx_lemmas_SN 33.3% pos_1_left 30.3% verbs 24.2% adposition 18.2% ctx_lemmas_6SN 18.2% morph_case 18.2% pos_2_left 18.2% sentence_type 18.2% translation 15.2% article 15.2% part_of_conj 15.2% sentence_length 12.1% adjective 12.1% grammatical_func 12.1% pos_3_right [13 more suppressed]	45.8% adposition 45.8% ctx_lemmas_6SB 45.8% ctx_lemmas_50B 45.8% ctx_lemmas_50N 45.8% ctx_lemmas_SB 45.8% verbs_ign_aux 41.7% pos_1_left 41.7% word_form 33.3% article 33.3% ctx_lemmas_6SN 33.3% pos_1_right 29.2% verbs 25.0% grammatical_func 25.0% morph_case 20.8% part_of_conj 20.8% morph_number 16.7% ctx_lemmas_SN 16.7% last_3_chars 12.5% named_entity 12.5% last_2_chars 12.5% sentence_type [10 more suppressed]	62.5% ctx_lemmas_6SB 62.5% ctx_lemmas_50B 62.5% ctx_lemmas_SB 62.5% translation 58.3% verbs 45.8% adposition 33.3% ctx_lemmas_6SN 33.3% ctx_lemmas_50N 33.3% word_form 29.2% pos_1_left 25.0% article 25.0% pos_1_right 20.8% ctx_lemmas_SN 20.8% morph_case 20.8% morph_number 16.7% sentence_type 12.5% part_of_conj 12.5% last_3_chars 12.5% nx_length 12.5% pos_3_left 12.5% verbs_ign_aux [9 more suppressed]	33.3% ctx_lemmas_6SB 33.3% ctx_lemmas_50B 33.3% verbs 33.3% word_form 26.7% pos_1_right 26.7% translation 20.0% article 20.0% ctx_lemmas_6SN 20.0% ctx_lemmas_50N 20.0% ctx_lemmas_SB 20.0% morph_case 20.0% nx_length 20.0% adposition 20.0% sentence_type 13.3% adjective 13.3% part_of_conj 13.3% pos_1_left 13.3% pos_2_right 13.3% pos_3_left 13.3% sentence_length [7 more suppressed]
Verbs	57.1% ctx_lemmas_6SB 57.1% ctx_lemmas_50B 57.1% ctx_lemmas_SB 57.1% pos_1_left 42.9% pos_2_right 42.9% sentence_length 28.6% AZ_confidence 28.6% pos_2_left 14.3% AI_confidence 14.3% DR_confidence 14.3% FS_confidence 14.3% ctx_lemmas_6SN 14.3% ctx_lemmas_50N 14.3% ctx_lemmas_SN 14.3% has_OS 14.3% has_OV 14.3% has_PRED 14.3% head 14.3% word_form	42.6% AR_confidence 35.3% AN_confidence 27.9% ctx_lemmas_6SB 27.9% ctx_lemmas_50B 27.9% ctx_lemmas_50N 27.9% ctx_lemmas_SB 27.9% ctx_lemmas_SN 27.9% has_OA 27.9% pos_1_right 26.5% has_OD 20.6% pos_1_left 20.6% pos_2_left 16.2% ctx_lemmas_6SN 16.2% has_FOPP 13.2% DN_confidence 13.2% has_OPP 13.2% auxiliary_verb 11.8% PP_confidence 11.8% has_PRED 11.8% word_form 10.3% morph_person 10.3% pos_2_right [24 more suppressed]	57.4% ctx_lemmas_6SB 57.4% ctx_lemmas_50B 57.4% ctx_lemmas_SB 57.4% translation 48.5% pos_1_right 30.9% ctx_lemmas_50N 30.9% ctx_lemmas_SN 30.9% pos_1_left 26.5% pos_2_left 23.5% auxiliary_verb 20.6% AN_confidence 16.2% DN_confidence 16.2% has_OA 16.2% last_3_chars 16.2% word_form 14.7% PP_confidence 11.8% has_PRED 11.8% Pp_confidence 11.8% pos_3_left 10.3% passive [29 more suppressed]	50.0% ctx_lemmas_6SB 50.0% ctx_lemmas_50B 50.0% ctx_lemmas_SB 50.0% ctx_lemmas_SN 50.0% pos_1_right 50.0% pos_2_right 33.3% translation 33.3% word_form 25.0% ctx_lemmas_6SN 25.0% ctx_lemmas_50N 25.0% has_OA 25.0% has_ON 25.0% auxiliary_verb 25.0% pos_1_left 25.0% pos_3_right 16.7% NN_confidence 16.7% has_OPP [15 more suppressed]

tags of the tokens immediately preceding or following the target words are more frequently selected than the other 'part-of-speech' features. This implies that the part-of-speech tags in the immediate context are more meaningful for disambiguating word senses than the more distant POS tags. The high frequency of selected 'part-of-speech' features is especially prevalent for adjectives, which means that POS tags surrounding target adjectives are indicative of particular word senses.

For nouns, the two features 'verbs' and 'adposition' (both from the 'context detail' feature group, see Subsection 8.1.6) are often selected. This means that (i) for target nouns in general the verb of the sentence is helpful for the disambiguation process and (ii) for target nouns that are syntactically part of a prepositional phrase the corresponding adpositions (i.e., preposition, postposition, or circumposition) provide meaningful clues specific to particular word senses.

The fact that features encoding verbal frames and constituent structures are often selected and, thus, very effective for verbs was to be expected. This corroborates previous studies on verb sense disambiguation, including Chen and Palmer [2005, 2009], Chen et al. [2007], and Dligach and Palmer [2008], who also found syntactic features useful for disambiguating verb senses. This topic is investigated in more detail in Subsection 8.3.4 below.

The 'translation' feature, which is not available for the manually annotated TüBa-D/Z, is among the most effective features for target nouns and verbs in the automatically processed TüBa-D/Z and in the other two corpora. This means that English translations of the target words in their specific contexts are highly predictive for different word senses. This observation relates in an interesting way to the questions whether word sense disambigation is a separate task or should better be viewed as part of a 'larger' task such as machine translation [Kilgarriff, 1997b; Resnik, 2006] and whether or not explicit word sense disambigation improves the performance of automatic machine translation systems [Cabezas and Resnik, 2005; Carpuat and Wu, 2005, 2007; Chan et al., 2007]. For machine translation, word sense disambiguation generally takes place – if not explicitly, then implicitly – and cannot be omitted altogether.

Albeit for reasons of space only features with frequencies of at least 10 percent are printed in the table, the numbers of how many features are suppressed indicates the overall amount and variety of selected features. In order to interpret the frequencies in Table 8.6, it is helpful to know for each word class and corpus the average numbers of selected features per lemma and the total numbers of lemmas. These statistics are contributed in Table 8.7: for each sense-annotated corpus and word class the number of selected features averaged over all corresponding lemmas is given in columns *Features–cfs*. For an easier comparison, columns *Features–all* replicate the average numbers of total features available from Table 8.3 above. Columns *Num. of lemmas* give the total number of lemmas available in the corresponding gold standards used for supervised WSD experiments (these numbers are taken from Tables 6.3, 6.5, and 6.7, see Chapter 6).

Table 8.7: Average numbers of features per lemma.

POS	**WebCAGe**			**TüBa-D/Z (manual)**			**TüBa-D/Z (automatic)**			**deWaC**		
	Features		Num. of	Features		Num. of	Features		Num. of	Features		Num. of
	cfs	all	lemmas	cfs	all	lemmas	cfs	all	lemmas	cfs	all	lemmas
Adj.	6.7	159	3							2.8	46	4
Nouns	8.8	197	33	15.8	356	24	13.9	348	24	5.6	53	15
Verbs	6.3	102	7	12.2	271	68	12.9	253	68	9.6	111	12

In general, the average numbers of selected features per lemma are between 3 and 16 – depending on the word class and gold standard corpus. These average numbers of selected features reflect the ratios of the overall available features: for lemmas in both TüBa-D/Z versions considerably more features are selected than for the other two corpora; and in deWaC, lemmas have fewest features selected (especially for adjectives and nouns) compared to the other corpora.

The discrete frequencies reported in Table 8.6 for adjectives can be explained with the numbers in Table 8.7: WebCAGe contains only three sense-annotated adjectives, which explains the values of 33.3% and 66.6%, while the values of 25%, 50%, and 75% are due to four sense-annotated adjectives in deWaC. The numbers in Table 8.7 also explain the apparently skewed amount

of features for verbs, where there are altogether only 19 selected features for WebCAGe, while clearly more in deWaC and the two TüBa-D/Z versions (see Table 8.6): this reflects the ratio of sense-annotated verbs in the different corpora, i.e., there are only 7 sense-annotated verbs in WebCAGe.

8.3.3 Profiling Feature Groups

While the previous subsection analyzed the impact of automatic feature selection, this subsection investigates the impact of individual feature groups on the performance of German WSD. Table 8.8 presents WSD performance scores for the *SMO* classifier[1] in terms of the F_1-measure when evaluated separately on groups of features extracted from the available sense-annotated corpora. These sets of features under investigation rely on different kinds of linguistic clues, such as information on surface forms of words, co-occurring lemmas, POS tags, morphology, and sentence structures (as described in Section 8.1). All values are presented separately for each word class and gold standard corpus. For each corpus, the highest results achieved for a particular word class are highlighted in boldface; the lowest values are italicized.

For some lemmas, where a feature group does not yield at least one feature with at least two distinct feature values in the annotated data, the WSD system is not able to attempt disambiguation. In this case, F_1 values are calculated only on the subset of lemmas for which the WSD system is able to attempt disambiguation. Percentages (in parenthesis) behind the F-score values indicate the amount of lemmas for which the WSD system attempts disambiguation, i.e., for how many percent of the lemmas at least one feature in the specific group exists for which at least two distinct feature values occur in the annotated data. The percentages are specified only if the WSD system is not able to attempt disambiguation for all lemmas. That is, for all omitted percentages, the value is implicitly 100%. The only exceptions are adjectives and nouns for the 'constituent structure' and 'verbal frames' features, where not even F_1-scores are provided. The reason is simple: these feature types

[1] See the paragraph entitled *Comparison of Classifiers* on pages 270ff. for an explanation on why performance in this subsection is reported for the *SMO* classifier only.

Table 8.8: Profiling feature groups (in terms of F-score for *SMO*).

Feature group	POS	WebCAGe	TüBa-D/Z (manual)	TüBa-D/Z (automatic)	deWaC
Surface	Adj.	82.93			56.00
	Nouns	82.62 (73%)	81.83	81.83	66.67 (67%)
	Verbs	84.62	70.52	70.52	49.39
Context lemmas	Adj.	87.80			*52.00*
	Nouns	84.97	*80.14*	*80.32*	60.20
	Verbs	84.62	72.39	*67.20*	50.00
POS	Adj.	82.93			**64.00**
	Nouns	**89.14**	**86.34**	84.69	*57.14*
	Verbs	**94.23**	74.34	73.81	50.00
Morpho-logical	Adj.	82.93			56.00
	Nouns	*79.54*	82.23	82.27	63.27
	Verbs	80.43 (86%)	70.10	69.42	50.61
Context details	Adj.	**92.68**			60.00
	Nouns	85.39	85.06	84.18	62.24
	Verbs	*73.08*	69.21	68.40	48.17
Sentence structure	Adj.	*80.49*			60.00
	Nouns	80.17	81.90	80.51	59.18
	Verbs	75.00	69.88	68.96	*47.74* (92%)
Constituent structure	Verbs	78.85	81.87	73.24	51.22
Verbal frames	Verbs	75.00	**86.96**	**76.81**	**51.83**
Other	Adj.	82.93			57.89 (75%)
	Nouns	82.17 (79%)	81.11 (92%)	**87.48** (92%)	**68.48**(93%)
	Verbs	*73.08*	*65.76* (72%)	75.30 (97%)	48.78

are not available for adjectives and nouns and, thus, the F-scores and the percentages would be 0.

The reason why the results for both TüBa-D/Z versions are the same for the 'surface' feature group is because the extraction of surface information (see Subsection 8.1.2) is equivalent in both versions since it is independent from any manual or automatic preprocessing such as part-of-speech tagging or parsing.

The results obtained with *SMO* for individual feature groups are generally – with the only exception of adjectives in WebCAGe – between 2 and 9 F_1-score points worse than the corresponding results obtained with *SMO* when using all available features. This finding corroborates the results by Lee and Ng [2002] for English WSD that there is no single best knowledge source and that the combination of several types of features leads to a better performance than the use of only one single knowledge source.

Most results reported in Table 8.8 are still competitive and often consistently better than the results obtained by the knowledge-based WSD approach in Chapter 7 (see Table 7.4 for a summary of the knowledge-based WSD results). This result is especially remarkable given that some of the feature groups contain only very few features for several word classes. For instance, although there are only three to four 'morphological' features applicable for a particular word class, when using only 'morphological' features the supervised WSD system outperforms the knowledge-based WSD system by about 8–34 F_1-score points – depending on the word class and gold standard corpus. However, it must be noted that the reason for the good performance despite few features is due to the behavior of a support vector machine classifier never to perform (much) worse than the performance of the majority class baseline (here, *ZeroR*). The explanation for this behavior has to do with the concept of SVMs: when the given data is imbalanced and not separable by a linear hyperplane, a support vector machine resembles the majority class baseline [Akbani et al., 2004; Ben-Hur and Weston, 2010]. This happens in the present example; when only very few features with very few distinct values occur, many of these instances are mapped to the same point in the feature space and are, thus, on the same side of the hyperplane. In this case, the behavior of *SMO* resembles a majority class classification (i.e., *ZeroR*), but the performance of *ZeroR* is often consistently better than the performance of the knowledge-based WSD system.

The relatively bad performance reported for 'context lemma' features in Table 8.8 when compared to the other feature types seems to contradict the finding from the previous subsection (see paragraph *Automatically Selected Features* on page 8.3.2) where 'context lemma' features were often selected by the automatic feature selection algorithm. The real explanation for what looks like contradictory findings has to do with the interaction of features from several feature groups: since the number of 'context lemma' features is large, the chance that they are able to complement other features is high. However, when employing only features from context lemmas, their performance is not competitive compared to the other feature types. One explanation for this behavior might be that although this feature type comprises many features –

and thus its coverage is very large – the individual features suffer from data sparseness: even if a lemma occurs a certain amount of times in the training contexts and seems to be a good sense indicator, it does not necessarily have to occur in the test data contexts at all, since the number of potentially occurring lemmas in the context is theoretically infinite.

When evaluated separately on groups of features, the overall best and most consistent performance is achieved for 'part-of-speech' features. The explanation is twofold: on the one hand, this type of feature has a very high coverage, because it is applicable to all word classes and evidently produces several distinct feature values. On the other hand, this feature type does not suffer from the data sparseness problem as the 'context lemma' features, because the range of attribute values is restricted to the 54 STTS part-of-speech tags. In short, the POS features represent a good balance between diversity and coverage, which makes them particularly attractive and effective for disambiguating word senses.

The fact that features from the 'other' group are available to only a subset of all lemmas is not surprising since there are at most two features available for which often no distinct values are recorded for the annotated instances. However, the good performance on those nouns in TüBa-D/Z and deWaC for which disambiguation is possible with these 'other' features is due to the 'translation' feature. This result corroborates the finding from the previous subsection where this feature turned out to be predictive since it was often selected by the automatic feature selection algorithm (see paragraph on *Automatically Selected Features* on page 8.3.2).

Finally, the features encoding verbal frames and constituent structures perform well – as expected [Chen and Palmer, 2005; Chen et al., 2007; Dligach and Palmer, 2008; Chen and Palmer, 2009]. 'Verbal frame' features achieve best results for TüBa-D/Z and deWaC compared to the results obtained for the other feature groups. When explicitly using a certain subset of features, a comparison of features obtained by the TüBa-D/Z with its manual linguistic annotations with the corresponding features obtained by automatic linguistic annotations is possible: it is not surprising that for syntactic features manually obtained annotations outperform their automatically obtained counterparts.

For the other feature types, the WSD performance is approximately the same with manually or automatically obtained features. The explanation for this behavior has to do with the annotation difficulty: syntactical structures are more difficult and inaccurate to obtain by automatic tools – compared to other linguistic annotations such as lemmas, part-of-speech tags, and morphology. Also note that verbs differ in the degree of correlation between word senses and verbal frames, ranging from total correlation to a complete lack of correlation, and that the set of sense-annotated verb lemmas in the TüBa-D/Z treebank has been selected to represent a mixture of such correlations (see Section 5.3.2 for more details on these correlations and on the selection of sense-annotated verbs). This implies that the set of verbs sense-annotated in the TüBa-D/Z includes a certain amount of verbs with a high degree of such correlation. This topic is investigated in more detail in the following subsection.

8.3.4 Profiling Verbs

This subsection investigates the influence of features based on syntactical structures – especially of 'constituent structure' and 'verbal frame' features – on the WSD performance for verbs. It first summarizes the underlying criteria from Subsection 5.3.2 on how the sense-annotated verbs were chosen, before it analyzes several aspects of the performance of a supervised WSD system on different verb classes (see pages 284ff.).

Different Correlations between Word Senses and Verbal Frames

Verbs differ in the degree of correlation between word senses and frames, ranging from total correlation to a complete lack of correlation. That is, while the syntactic structure in which a verb occurs is highly predictive of different word senses for some verbs, syntactic structures are not informative enough to distinguish word senses for other verbs. As explained in Subsection 5.3.2, the manually sense-annotated verbs in the TüBa-D/Z treebank are selected to represent a mixture of correlations among word senses and frames in order to provide a fine-grained spectrum of possible degrees of correlations between word senses and verbal frames. In short, the sense-annotated verbs are taken

from four distinct classes:[1]

Class 1 All verbs in this class have perfect correlation between senses and verbal frames; i.e., they have distinct frames for each of their word senses.

Class 2 This class contains verbs where at least one sense has a verbal frame distinct from the frames for the other senses.

Class 3 All verbs in this class lack any correlation between word senses and verbal frames; they share the same frames for all of their senses.

Class 4 This class comprises verbs that do not fall into any of the classes 1–3.

Table 8.9[2] shows the numbers of sense-annotated verbs in the TüBa-D/Z that are available in the gold standard used for supervised WSD experiments (see Subsection 6.4.3 for more details). Note that this table represents an updated version of Table 5.10, where the numbers were calculated on (i) old versions of GermaNet and TüBa-D/Z and (ii) for all available sense annotations rather than on the subset used for supervised machine learning experiments.

Table 8.9: Distribution of the four sense/frame correlation classes.

Correlation class	**1**	**2**	**3**	**4**	**Total**
Total verb lemmas	31	15	14	8	68
- in representative sample	8	8	14	8	38
- in *high correlation* sample	23	7	0	0	30
Average polysemy	2.13	2.60	2.93	4.00	2.62
Average frequency	82	175	168	128	126
Total annotated occurrences	2548	2623	2347	1022	8540

The total set of sense-annotated verbs in the TüBa-D/Z is divided into two samples:[3]

[1] See Section 5.3.2 for examples and more details on the correlation classes for verbs.

[2] Note that the *total* numbers in Table 8.9 correspond to the *verb* numbers in Table 6.5.

[3] Technically, the two samples are constructed by the following procedure: first, all verb lemmas that belong to correlation classes 1 and 2 are sorted in decreasing order of frequency and every second lemma from these two correlation classes is taken for the 'high correlation' sample. This means that for class 1 every second lemma is taken until the required number of 8 lemmas is reached and for class 2 every second lemma is taken until the end of the sorted list is reached (which results in the required amount of 7 lemmas). The remaining verb lemmas, i.e., those left from classes 1 and 2 and all lemmas from classes 3 and 4, are used for the 'representative' sample.

Representative sample One sample is representative and well-balanced according to the class distributions in GermaNet. It is not biased toward specific sense/frame correlation classes.

High correlation sample The second sample contains verbs with a high correlation between word senses and verbal frames (i.e., verbs from classes 1 and 2). The assumption is that an automatic WSD algorithm based on verbal frame information is expected to return good results for these verbs since the frames should allow distinguishing different senses reliably.

The amounts of lemmas per correlation class included in these lexical samples are listed in the second and third rows denoted as *in representative sample* and *in high correlation sample* in Table 8.9.

The motivation behind the distinction into two samples is to facilitate the evaluation of WSD algorithms that use verbal frame information for disambiguation. If a WSD algorithm fails to perform well on the 'high correlation' sample, this would provide evidence that such an algorithm has major drawbacks and will not work for the representative, well-balanced sample at all. Furthermore, a comparison on how machine learning algorithms perform on the two samples is highly interesting. That is, to test the deviation in performance and learn about how powerful such an algorithm can be for verbs that show a high correlation between word senses and verbal frames.

Analyzing the WSD Performance for Verbs

Table 8.10 presents the WSD performance in terms of F_1 evaluated on verbs in the TüBa-D/Z. For the above-described verb samples and correlation classes, the table includes detailed results obtained for different feature groups.

The principle structure of the table divides the results for the two TüBa-D/Z variants by two horizontal lines: it first prints the results obtained by applying *SMO* to the features extracted from the manual linguistic treebank annotations and then prints the results obtained by applying *SMO* to the features based on the automatic linguistic annotations. On the bottom, the table shows the performance of the *ZeroR* baseline for comparison.

Table 8.10: Profiling verb classes in TüBa-D/Z (in terms of F-score).

	Feature groups used	**all verbs**	**Represent. sample**	**High corr. sample**	**Correlation class**			
					1	**2**	**3**	**4**
SMO for manual features	all features	86.53	81.46	92.27	95.37	92.42	76.06	73.45
	constituent structures	81.87	79.32	84.87	88.60	83.58	77.61	70.50
	verbal frames	86.96	83.15	94.12	97.39	91.85	76.32	72.86
	const. struct.+ verbal frames	87.70	83.57	94.32	97.51	91.50	77.99	75.81
	all except const. struct.+ verbal frames	77.94	74.41	83.21	88.95	77.61	74.52	59.29
SMO for automatic features	all features	79.99	77.55	79.73	85.75	84.39	73.87	68.44
	constituent structures	73.24	73.19	75.31	83.25	67.85	74.90	58.41
	verbal frames	76.81	75.65	76.37	83.61	80.25	72.84	60.18
	const. struct.+ verbal frames	77.09	76.47	76.97	83.85	79.91	74.26	59.59
	all except const. struct.+ verbal frames	77.27	75.66	77.28	84.56	78.65	73.87	63.42
ZeroR baseline		68.19	70.91	69.89	80.40	60.16	72.72	48.08

As specified in column *Feature groups used*, evaluation scores for both TüBa-D/Z variants are provided for several feature groups: (i) using all available features, (ii) using the 'constituent structure' features, (iii) using the 'verbal frame' features, (iv) using both 'constituent structure' and 'verbal frame' features at the same time, and (v) using all available features except 'constituent structure' and 'verbal frame' features. The WSD performance for each of these feature groups is calculated for *all verbs*, for the *representative sample*, for the *high correlation sample*, and for the four sense/frame *correlation classes* – as recorded in the corresponding columns.[1] Note that even

[1]Note that some of the results for *all verbs* in Table 8.10 are equivalent to the corre-

though the most frequent sense baselines (i.e., *ZeroR*) for the individual correlation classes are very heterogeneous, the two samples are selected insofar that the baselines obtain comparable results: both have an F_1-score of around 70.

One of the most striking findings among the results of this evaluation is that – when employing manual treebank annotations – the performance on the 'high correlation' sample is significantly better than the performance on the 'representative' sample. Depending on the employed feature groups, the difference in performance is between 9 and 15 F_1-score points. Further, the performance on correlation class 1 is better than the performance on class 2, and the performance on both classes 1 and 2 are significantly better than the performance on correlation classes 3 and 4. This behavior was to be expected. It confirms that the correlation between word senses and verbal frames is a strong indicator for disambiguating between verb senses. Maximum F_1-scores of 94.3 for the 'high correlation' sample and 97.5 for correlation class 1 – obtained with 'constituent structure' and 'verbal frame' features – show how powerful a WSD algorithm can be for the subset of verbs that show a high correlation between word senses and verbal frames given that manual treebank information is available.

For the automatically annotated treebank annotations, however, the same difference in performance is much less pronounced. For instance, when using the automatically obtained 'verbal frame' features, the performance on the 'high correlation' sample is only 0.7 F_1-score point higher than the performance measured on the 'representative' sample. However, with F_1-scores of 83.6 and 80.3, the performance of correlation classes 1 and 2 is between 8–23 points higher than the performance of 72.8 and 60 obtained for correlation classes 3 and 4. The fact that the performance increase is much lower when evaluated on the automatically obtained features is not surprising. It is due to the same reason as given in Subsection 8.3.3 above that the quality of syntactic annotations plays an important role for the performance of a supervised WSD system that uses these structures as features: syntactic annotations are difficult

sponding values reported previously: the results for *all features* were already reported in Table 8.4 and the results for the 'constituent structure' features and 'verbal frame' features were already reported in Table 8.8.

to obtain and, thus, their quality is more inaccurate to obtain by automatic tools than by manual annotations.

A comparison of the results for the individual feature groups yields the tendency that 'verbal frame' features achieve superior results compared to 'constituent structure' features. This behavior is especially prevalent when using the manual treebank annotations. The main reason for this higher performance is due to the definitions of the correlation classes and verb samples, which are based on verbal frames rather than on constituent structures. Another explanation for the higher performance is probably that the 'verbal frame' features use linguistic clues that are more indicative of different word senses than the clues used by the 'constituent structure' features. For instance, the features 'has_OADJP' and 'has_OADVP', which are based on the corresponding constituents *OADJP* and *OADVP*, are much less relevant than the features 'AR_confidence' and 'DR_confidence', which take into account the availability of accusative and dative objects realized by reflexive personal pronouns (also see the feature descriptions in Subsections 8.1.8 and 8.1.9). The only exception is for correlation class 3: for those lemmas that share the same verbal frames for all of their senses in GermaNet, the performance of 'constituent structure' features is slightly better than the performance of 'verbal frame' features. This can only be due to the fact that the occurring constituents for some of these lemmas correlate with the corresponding senses – even though there is no correlation between the senses and the verbal frames encoded in GermaNet.

8.4 Conclusion and Future Work

This chapter has applied a wide range of supervised machine learning methods to the task of German word sense disambiguation. These supervised machine learning methods include classifiers based on decision rules, instance-based classifiers, probabilistic classifiers, support vector machines, and combined classifiers – all taken from the Weka machine learning tool suite. WSD experiments have been run on the three corpora WebCAGe, TüBa-D/Z, and

deWaC – for the three word classes of adjectives, nouns, and verbs.

The WSD evaluation in Section 8.3 has shown that a support vector machine classifier and a voting combination classifier have overall performed best among all applied classifiers. This finding corroborates previous studies, which found that these classifiers perform best for the task of WSD, including Zavrel et al. [2000], Lee and Ng [2002], Martínez et al. [2002], and Agirre and Martínez [2004] for support vector machines and Florian et al. [2002], Florian and Yarowsky [2002], Klein et al. [2002], and Mihalcea et al. [2004] for combined algorithms. In general, the supervised word sense disambiguation system has significantly outperformed the most frequent sense baseline, which is known to be often difficult to beat by WSD systems [Gale et al., 1992a; Navigli, 2009; Preiss et al., 2009].

The use of an automatic feature preselection algorithm, which removes irrelevant features and tries to optimize the predictive accuracy of machine learning algorithms, has improved the performance for many of the used WSD classifiers (see Subsection 8.3.2) – as previously shown by Mihalcea [2002a,b]. An inspection of several types of machine learning features has shown that the coverage of 'context lemma' features is high and that these features have complemented well with other features. By contrast, the performance of a WSD system that relies only on context lemmas has performed badly. When using a single feature type only, the best WSD results have been achieved by 'part-of-speech' features. However, overall best disambiguation performance has been achieved by using all available features. This finding corroborates the results by Lee and Ng [2002] for English WSD that there is no single best knowledge source and that the combination of several types of features leads to a better performance than the use of only one single knowledge source.

Since the TüBa-D/Z contains high-quality manual treebank annotations and all underlying texts have additionally been automatically annotated with the same kind of linguistic annotations, the comparison of the results obtained by these two annotation sources has allowed studying the influence of linguistic annotation quality (manual versus automatic) on the WSD performance. While the manual features have had a consistently positive impact on verbs, their impact on nouns has been much less noticeable and not only positive

compared to the results obtained by the automatic features. The reason for this behavior has to do with the relevant features and their quality: on the one hand, many effective features for verbs are based on syntactic annotations, which are difficult to obtain and, thus, their quality is more inaccurate to obtain by automatic tools than by manual annotations. On the other hand, the English translation of a target word in its context has been – especially for nouns – a very effective feature, but it has been available only for automatically annotated texts.

Verbs differ in the degree of correlation between word senses and verbal frames, ranging from total correlation to a complete lack of correlation. When employing manual treebank annotations, the WSD performance significantly increases when evaluated on verbs with a high correlation between senses and frames compared to the WSD performance on verbs with no such correlation. This behavior confirms that the correlation between word senses and verbal frames is a strong indicator for disambiguating between verb senses. For the automatically annotated treebank annotations, however, this increase in performance is much lower. The explanation for this behavior is again related to the quality of syntactic annotations, which are more accurate when manually annotated than automatically obtained.

In comparison to the previous chapter, the supervised machine learning methods have outperformed the knowledge-based systems by a very wide margin: by about 28–34 F_1 points for adjectives, by about 13–36 for nouns, and even by about 32–52 for verbs – depending on the gold standard used for evaluation. It is, however, important to keep in mind that the supervised WSD approach presupposes the availability of expensive training data and thus their coverage is restricted to those lemmas for which training data is available [Màrquez et al., 2006; Mihalcea, 2006; Navigli, 2009].

Chapter 9

Concluding Remarks

The overall objective of this thesis has been to overcome the bottleneck of German word sense disambiguation resources and to boost research on WSD for German. In order to achieve this goal, the work has started with the preparation of the necessary resources before actual word sense disambiguation experiments have been performed. That is, GermaNet has been aligned with Wiktionary in order to harvest Wiktionary's sense definitions (Chapter 4). Two sense-annotated corpora have been constructed – the automatically web-harvested WebCAGe and the manually sense-annotated TüBa-D/Z treebank (Chapter 5). On the basis of these corpora, word sense disambiguation experiments have been performed. These WSD experiments have utilized a wide range of knowledge-based methods (Chapter 7) as well as a wide range of supervised machine learning methods (Chapter 8).

9.1 Knowledge-Based Approaches versus Supervised Learning Approaches to WSD

A comparison of the results from the knowledge-based WSD experiments in Chapter 7 with the supervised machine learning WSD experiments in Chapter 8 corroborates the two findings of previous research (i) that knowledge-based systems have a larger coverage than supervised machine learning systems and (ii) that supervised learning systems perform much better than

knowledge-based approaches to WSD [Kilgarriff and Rosenzweig, 2000; Mihalcea, 2006; McCarthy, 2009; Navigli, 2009]. The reason for a larger coverage of the knowledge-based systems is that they do not need annotated training data. By contrast, the supervised ML systems have achieved lower coverage because they have relied on sense-annotated training data and have, thus, been applicable to only a restricted set of lemmas for which sense-annotated training material is available. On the other hand, for this restricted set of lemmas, the supervised systems have outperformed the knowledge-based systems.

While the knowledge-based sense disambiguation methods in Chapter 7 have performed well for nouns, they have been ill-suited for disambiguating adjective and verb senses. By contrast, the supervised classifiers in Chapter 8 have performed well for all word classes and far better than the knowledge-based algorithms for all word classes.

In short, the supervised machine learning methods have outperformed the knowledge-based systems by a very wide margin. It is, however, important to keep in mind that the supervised WSD approach is applicable to only a subset of word token annotations for which training data is available.

9.2 Comparison of Sense-Annotated Corpora

Comparing the results for the three sense-annotated corpora, the WSD results obtained for deWaC are the worst – both for the knowledge-based and for the supervised learning methods. The most apparent reason for the bad performance of knowledge-based WSD algorithms on the deWaC corpus is the much higher polysemy of annotated words, which makes sense disambiguation more difficult. Since the polysemy for words in deWaC's gold standard used for supervised experiments is comparable to the polysemy of words in the corresponding TüBa-D/Z gold standard, there must be another reason for the lower WSD performance when evaluating supervised methods on deWaC. One explanation for this lower performance are very heterogeneous contexts for the sense-annotated words in deWaC (see paragraph entitled as *Comparison of Word Classes and Sense-Annotated Corpora* on page 265 for a discussion).

While both knowledge-based and supervised WSD experiments perform worst when evaluated on deWaC, the results deviate on which corpus they perform best. That is, the knowledge-based WSD systems perform best for TüBa-D/Z while the supervised learning methods perform best for WebCAGe. The explanation is as follows: WebCAGe contains several annotations for which only a restricted context is available (i.e., the example sentences harvested from Wiktionary itself are single sentences only without a larger context), whereas all of the sense annotations in TüBa-D/Z and in deWaC have sufficiently large contexts. This restricted context is the reason why several knowledge-based methods (particularly path-based and information-content-based) obtain particular low coverages for WebCAGe. On the other side, the higher performance of supervised WSD systems when evaluated on WebCAGe can be explained with the polysemies of words in the gold standards used to evaluate supervised WSD experiments, which are lower for WebCAGe compared to the corresponding polysemies recorded for the other two corpora.

9.3 Future Work

In order to boost the performance of knowledge-based and supervised WSD systems, natural next steps would be to experiment with different parameter settings of the algorithms and to implement further state-of-the-art WSD algorithms that have not been included in the WSD experiments in Chapters 7 and 8 but that proofed to perform well in related WSD studies.

Further, a very simple way to achieve perfect coverage for the knowledge-based system and, at the same time, to improve the disambiguation results is to use a backoff strategy – such as choosing a sense at random for cases where the WSD algorithms are unable to assign any word sense – which has shown to work well (for example, in the experiments by Broscheit et al. [2010] and Miller et al. [2012]). However, even if the performance of the knowledge-based systems can be improved, results can be boosted only to a certain extent. Since the supervised learning systems have generally performed much better than the knowledge-based approaches to German WSD (also see Subsection 9.1),

it seems much more effective to refine and improve the supervised machine learning WSD methods. In particular, the coverage of the supervised system should be extended to those lemmas for which no sense-annotated training data is available.

In order to generally boost the performance and improve the impact of the supervised WSD system, natural next steps include:

- The WSD performance of the individual classifiers could be boosted by experimenting with different parameter settings of the algorithms. [Hoste et al., 2002c]

- Since combined classifiers usually outperform single WSD decision systems [Florian et al., 2002; Florian and Yarowsky, 2002; Klein et al., 2002; Mihalcea et al., 2004], in future work, more effort could be put into identifying the most optimal combination of classifiers.

- It might be worth to employ further classification algorithms (that are not part of the Weka tool suite) such as the memory-based algorithm implementations in TiMBL [Daelemans et al., 2010], which previously yielded good performance for English WSD [Hoste et al., 2002b; Dinu and Kübler, 2007; de Oliveira et al., 2011].

- Further, the application of unsupervised discrimination methods in combination with sense labeling seems promising for extending the disambiguation coverage to lemmas for which no sense-annotated training material is available. That is, to first discriminate between senses in an unsupervised manner and, then, to assign a sense label to each of these sense clusters – as proposed by Schütze [1998].

- For improving the performance of a supervised WSD system and for extending its coverage to lemmas for which no sense-annotated training data is available, algorithms for automatic acquisition of corpus examples could be applied – similarly to the experiments by Yarowsky [1995], Mihalcea [2004], Gonzalo and Verdejo [2006], and Kübler and Zhekova [2009].

- In addition to extending the sense-annotated data, available data could also be modified by *instance sampling* (sometimes also *active learning*). The general idea of instance sampling is to modify the set of available instances insofar that all class distributions are balanced – e.g., by removing a certain amount of instances that are annotated with the most frequent class from the training set [Lewis and Gale, 1994]. Sampling has shown to have a high impact particularly on imbalanced datasets, which usually cause over-fitting. It was previously applied to WSD by Fujii et al. [1998], Chen et al. [2006, 2013], and Zhu and Hovy [2007].

- Some studies, including de Oliveira et al. [2011] and Kawahara and Palmer [2014], have successfully trained and applied one classifier to predict word senses for several or all lemmas at once (rather than training one classifier per lemma). This approach could also help to extend the coverage of a supervised WSD system to those lemmas for which no sense-annotated training data is available.

- A combination of supervised machine learning with knowledge-based methods could integrate the strength of both approaches in order to achieve high coverage and good performance at the same time. That is, using knowledge-based WSD for those lemmas to which a supervised WSD system cannot be applied due to no available training data. This approach is comparable to what Steffen et al. [2004] call a *disjunctive combination* of two heterogeneous systems.

- The representation of words as vectors, e.g., in the form of generic word embeddings learned from large unannotated corpora, became very popular for many natural language processing tasks [Collobert et al., 2011]. It is particularly popular, because it is universally applicable to many NLP tasks and achieves high performance for many tasks while mainly relying on unannotated data. The learning of sense-specific word representations has recently opened up a new approach to word sense disambiguation. Several studies used this approach for disambiguating word senses, including Bordes et al. [2012], Chen et al. [2014], and Guo et al. [2014].

Part IV

Appendices

Appendix A

Comparison of GermaNet Releases

This appendix summarizes all GermaNet versions that have been used during the work on this dissertation. Information is provided on which GermaNet release serves as the basis for which task of this thesis and which release contains which newly contributed information.

The development of GermaNet is still in progress, and it is released on a yearly basis. That is, during the work on this dissertation, six official GermaNet releases have been published. Table 3.5 in Section 3.9 overviews coverage numbers for the four relevant versions (releases 6.0 through 9.0). Each task in this thesis relies on the most recent version of GermaNet that was available when the corresponding work commenced. Thus, several GermaNet releases serve as a basis for different tasks; and, vice versa, several manual and semi-automatic extensions flow into new GermaNet releases. The corresponding GermaNet versions are also specified in the thesis' chapters for the described tasks.

In all releases the GermaNet lexicographers add new entries for synsets and lexical units for all three word classes of adjectives, nouns, and verbs, including their relations.[1] The following list provides specifics on each of the GermaNet releases. It includes information on general, manual extensions (that are not part of this dissertation) as well as on (semi-)automatic extensions created in

[1] Although several GermaNet extensions are created semi-automatically, this purely manual, lexicographic work is not part of this dissertation. It rather forms the continuous process of GermaNet development.

the context of this thesis. The list also provides details on which GermaNet release forms the basis for which extension or experiment in this dissertation.

GermaNet release 6.0 as of April 2011 serves as the basis for mapping GermaNet with Wiktionary to harvest sense descriptions (see Chapter 4). Since this mapping builds the basis for the (semi-)automatic creation of the sense-annotated corpus WebCAGe (see Section 5.2), the corpus is initially sense-annotated with senses from GermaNet's release 6.0.[1]

Automatic processing of nouns with the goal of splitting nominal compounds is first performed on release 6.0 (see Section 3.6).

On the manual side, the conceptual meronymy/holonymy relation is further differentiated into the four subrelations of *component*, *member*, *substance*, and *portion* meronymy/holonymy – see Section 3.3 as well as Hinrichs et al. [2013]. At the same time, the denotations of GermaNet relations are adopted for consistency with the Princeton WordNet for English and for clearly indicating the direction of a relation. For instance, the former *'hyperonymy'/'hyponymy'* relation is now *'has hypernym'/'has hyponym'*, the former *'antonymy'* is now *'has antonym'*, *'pertonymy'* changed to *'has pertainym'*, etc.

GermaNet release 7.0 (published in May 2012) is the first release which contains sense descriptions from Wiktionary. GermaNet senses have been mapped semi-automatically to Wiktionary sense descriptions with the goal of extending lexical units in GermaNet with the respective descriptions from Wiktionary (this sense alignment is described in Chapter 4).

The first publicly released version of the web-harvested, (semi-)automatically sense-annotated corpus WebCAGe (see Section 5.2) refers to sense identifiers from GermaNet 7.0.

Although GermaNet's working copy has been stored in a relational database since release 5.2 (see Appendix B), a dump of the database is in GermaNet 7.0 for the first time included in the official release

[1] In Chapter 6, the sense-annotations in WebCAGe are updated to release 9.0.

package. This also facilitates the use of the GermaNet editing tool GernEdiT (described in Henrich and Hinrichs [2010a]), which was at the same time made publicly available.

Another software tool made available is an API to calculate semantic relatedness between senses in GermaNet. The implemented algorithms are used in the knowledge-based WSD experiments in Chapter 7.

The interlingual index (ILI), initially created in the context of the EuroWordNet project, represents a mapping between wordnets of different languages – see Section 3.5. The German part of the ILI, which allows the mapping of GermaNet senses to the corresponding entries in the Princeton WordNet, has been manually revised and extended and it has been integrated in GermaNet since release 7.0.

GermaNet release 8.0 was published in April 2013. This version serves as a basis for the manual sense annotation of the TüBa-D/Z treebank – as described in Section 5.3. Updates to GermaNet resulting from this sense annotation flow into GermaNet release 9.0.[1]

For the first time, GermaNet also includes information on split compounds. A subset of nominal compounds was split into their constituents (modifier and head) and labeled with linguistic information such as foreign words and named entities (see Section 3.6).

GermaNet release 9.0 as of April 2014 is the latest release available for the work on this dissertation. It includes manual updates resulting from sense-annotating the TüBa-D/Z treebank (described in Section 5.3).

All sense-annotated corpora, i.e., WebCAGe, TüBa-D/Z, as well as deWaC, used throughout the WSD experiments in Chapters 7 and 8 have been updated for this latest available version of GermaNet – see Chapter 6.

For this release, all nominal compounds have been identified, split into their constituent parts, and labeled with appropriate linguistic information (as described in Section 3.6).

[1] In Chapter 6, the sense annotations in the treebank are updated to GermaNet 9.0.

Appendix B

GermaNet's Database Format

For GermaNet's release 5.2 as of December 2009 the working development copy of the resource is converted from lexicographer files into a state-of-the-art relational database. The original database model (for GermaNet 5.2) is summarized in Henrich and Hinrichs [2010a]. This appendix presents the updated database structure of the most recent version of GermaNet available at the time of finishing the writing of this dissertation – which is release 9.0.

The model of the database follows the internal structure of GermaNet: there are tables to store synsets, lexical units, conceptual and lexical relations, etc. The model implies persistent database identifiers for all entries in GermaNet, including lexical units and synsets. The complete database structure for GermaNet is shown as an entity-relationship diagram in Figure B.1.[1] The succeeding list briefly explains each of the database tables and columns. Also see Chapter 3 for details on information encoded in the GermaNet resource.

[1]Note that there are further database tables not shown in Figure B.1 because they do not contain primary GermaNet data. These auxiliary tables are rather used internally by GermaNet's editing tool *GernEdiT* [Henrich and Hinrichs, 2010a], for example, to store the editing history.

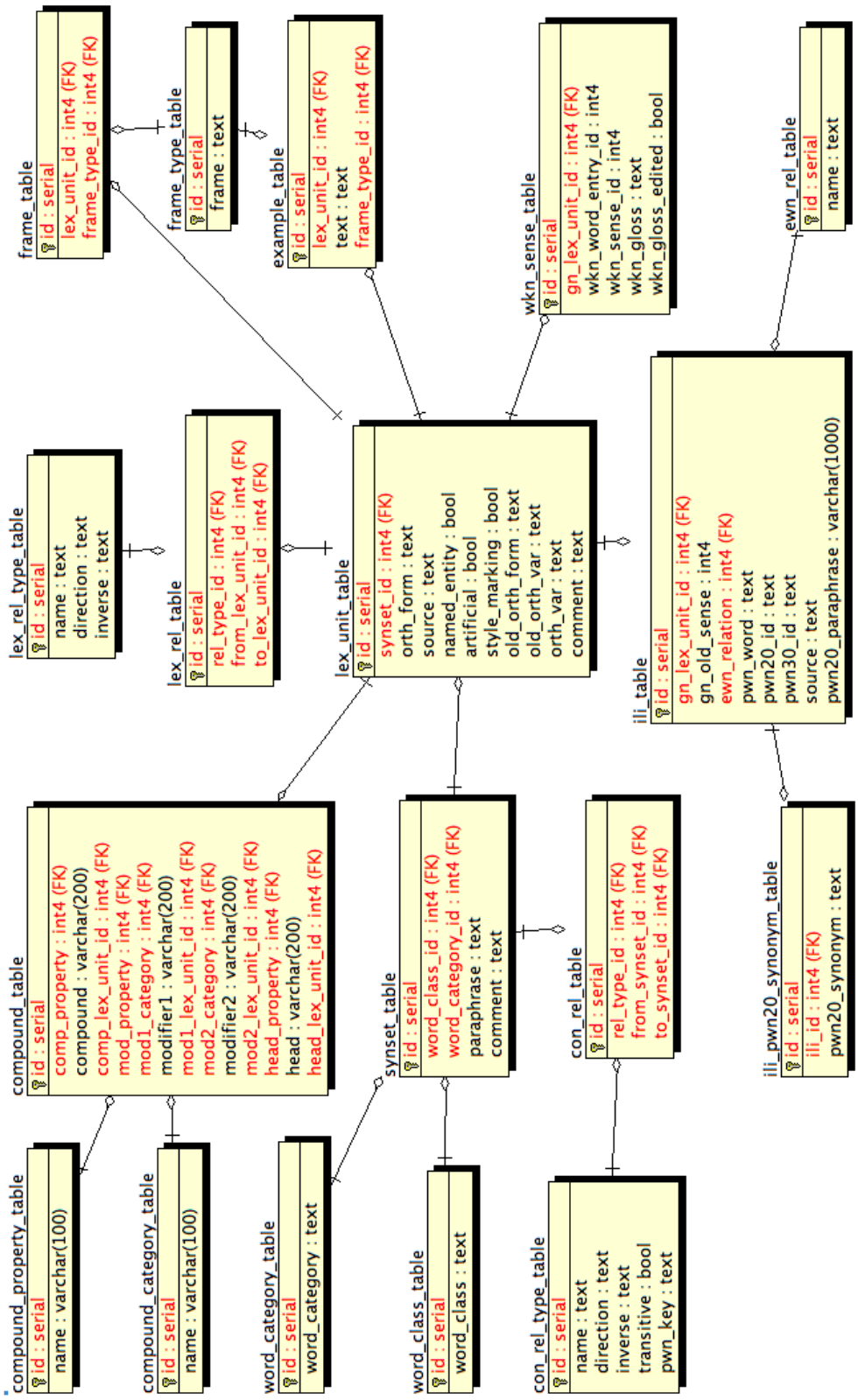

Figure B.1: Entity-relationship model of the GermaNet database.

- **synset_table:** Each entry in this table represents a synset with all its information. Synsets in GermaNet belong to a word class as well as to a word category. Optionally, definitions can be assigned to synsets.

  id: A unique identifier for this synset.

  word_category_id: Specifies the unique identifier for the word category (i.e., *adj*, *nomen*, or *verben*; see description of word_category_table below) of this synset.

  word_class_id: Specifies the unique identifier for the word class (i.e., semantic field; see description of word_class_table below) of this synset.

  paraphrase: An optional description or definition of this synset.

  comment: An optional comment for this synset; mainly for internal use during the lexicographic work.

- **word_class_table:** This database table stores all possible word classes[1] (also referred to as semantic fields). Originally, the division into semantic fields was for organizational purposes, i.e., to divide synsets into multiple files. Since GermaNet's conversion into a relational database, these semantic fields are not organizationally required anymore, but rather serve as a grouping of synsets into semantically related topics.

  id: The unique identifier of this word class.

  word_class: The semantic field itself, e.g. *Artefakt* 'artifact', *Bewegung* 'motion', *Geist* 'spirit', etc. Table 3.1 in Section 3.2 lists all 38 semantic fields available in GermaNet.

[1]Note that in GermaNet's terminology, the term *word class* is used to refer to semantic fields such as *Artefakt* 'artifact', *Bewegung* 'motion', etc., while the term *word category* is used to refer to the more common sense of 'word class' such as adjective, noun, and verb.

- **word_category_table:** Each synset is assigned a word category. This table stores the possible values for word categories.

  id: A unique identifier for this word category.

  word_category: Contains the value for the word class itself. Possible values are *adj* 'adjective', *nomen* 'noun', or *verben* 'verb'.

- **lex_unit_table:** Each entry in this table represents a lexical unit with all its information. A lexical unit can have – besides its obligatory main orthographic form – further optional variants, which characterize the differences between the old and the new German spellings (see Section 3.2).

  id: The unique identifier for this lexical unit.

  synset_id: Specifies the unique identifier for the synset to which this lexical unit belongs. This identifier can also be used to determine synonymous lexical units, i.e., by taking all lexical units with the same synset_id.

  orth_form: A lexical unit always encodes this main orthographic form, which represents the correct spelling of a word according to the rules of the recently adopted German spelling reform (*Neue Deutsche Rechtschreibung*, Rat für deutsche Rechtschreibung [2006]).

  orth_var: In the case of an alternative spelling that is permissible according to the new German spelling, a lexical unit can optionally have an orthographic variant. An example of this kind is the German noun *Delfin* 'dolphin'. Apart from the main form *Delfin*, there is an orthographic variant *Delphin*.

old_orth_form: If the orthography of a word has changed in the context of the spelling reform, the old orthographic form represents the main form from the old German spelling.

old_orth_var: This encodes an orthographic variant that was permissible prior to the spelling reform. It is encoded only if the variant is no longer valid in the new orthography.

named_entity: Specifies whether this lexical unit is a named entity or not.

artificial: Specifies whether this lexical unit is used to represent an artificial node in the graph.

style_marking: Specifies whether the style of this lexical unit is marked – i.e., whether this lexical units represents a stylistic variant.

comment: An optional comment for this lexical unit; mainly for internal use during the lexicographic work.

- **example_table:** Each entry in this table represents an example that belongs to a lexical unit. Each example can have an associated verbal frame that indicates a possible usage of the lexical unit for that particular frame.

  id: A unique identifier for the example.

  lex_unit_id: Refers to the unique identifier for the lexical unit to which this example belongs.

  text: The example sentence itself.

  frame_type_id: Specifies the frame type of this example (optional), see description of frame_type_table below.

- **frame_table:** Each entry in this table represents a syntactic frame that belongs to a lexical unit. Frames specify the verbal frames of lexical units – as described in Section 3.7.

  id: As before, a unique identifier for this frame entry.

  lex_unit_id: The unique identifier of the lexical unit, to which this frame belongs.

  frame_type_id: Specifies the frame type of this frame, see description of frame_type_table below.

- **frame_type_table:** This table contains all possible verbal frames (see Section 3.7).

  id: A unique identifier.

  frame: The frame type itself, e.g., *NN.AN* for subject plus accusative object, *NE* for expletive subject *es* 'it', or *NN.DN.Az* for subject and dative object plus an optional infinitive clause with *zu* 'to' – as described in Section 3.7.

- **lex_rel_table:** All lexical relations are stored in this table. A relation connects a source lexical unit (*from*) with a target lexical unit (*to*). Also see Section 3.3).

  id: An identifier, as before.

  rel_type_id: Specifies the type of lexical relation, see description of lex_rel_type_table below.

  from_lex_unit_id: The source lexical unit from which this lexical relation starts.

  to_lex_unit_id: The target lexical unit to which this lexical relation goes.

- **lex_rel_type_table:** Here, all types of lexical relations are stored, as described in Section 3.3.

  id: Unique identifier.

  name: The name of the lexical relation, i.e., *has antonym*, *has pertainym*, or *has participle*, see Section 3.3. Note that synonymy does not appear in this table, because the synonymy relation is established indirectly by grouping synonymous lexical units into synsets. That is, synonymy can be determined by taking all lexical units with the same synset_id.

  direction: Specifies whether this lexical relation is valid in one or both directions.

  inverse: If the relation is valid in both directions, this column contains the name of the inverse relation.

- **con_rel_table:** This table contains all conceptual relations (see Ssection 3.3). Analogously to the encoding of lexical relations, this table connects a source synset (*from*) with a target synset (*to*).

  id: Unique identifier.

  rel_type_id: Specifies the type of conceptual relation (see description of con_rel_type_table below).

  from_synset_id: The source synset from which this conceptual relation starts.

  to_synset_id: The target synset to which this conceptual relation goes.

- **con_rel_type_table:** Each entry in this table specifies one type of conceptual relation, as listed in Section 3.3.

  id: A unique identifier.

  name: The name of the conceptual relation, e.g., *has hypernym*, *has member meronym*, or *entails* (see Section 3.3).

  direction: Specifies whether this conceptual relation is valid in one or both directions, or whether the relation is valid in both directions, but with a different inverse relation name.

  inverse: Specifies the name for the conceptual relation in the inverse direction, if the relation is valid in both directions. For instance, if the relation name is *has hypernym* the inverse name is *has hyponym*.

  transitive: Specifies whether or not this conceptual relation is transitive.

- **ili_table:** This table contains the interlingual index (ILI) records which map lexical units from GermaNet to corresponding synsets in the Princeton WordNet; as explained in Section 3.5.

  id: A unique identifier for this ILI record.

  gn_lex_unit_id: The identifier for the lexical unit which is linked to WordNet.

  ewn_relation: Specifies the type of interlingual relation, e.g., *synonymy*, *near-synonymy*, *hypernymy*, etc. – see the description of ewn_rel_table.

  pwn_word: The corresponding English word from the WordNet synset belonging to pwn20_id, i.e., a translation of the GermaNet lexical unit gn_lex_unit_id.

pwn20_id: The synset offset identifier from WordNet 2.0 matching the GermaNet lexical unit.

pwn30_id: The NLP group of the Universitat Politècnica de Catalunya provide automatically created mappings for several WordNet versions.[1] This synset offset represents the WordNet 3.0 entry available in this mapping for the corresponding WordNet 2.0 synset (pwn20_id).

source: Captures whether the data originates from the EuroWordNet project or from the University of Tübingen.

pwn20_paraphrase: The definition from WordNet belonging to the synset identified by pwn20_id.

- **ewn_rel_table:** This table contains all possible interlingual relation types from EuroWordNet which are used to link GermaNet to Princeton WordNet (see Section 3.5).

  id: Unique identifier.

  name: The name of the interlingual relation between GermaNet lexical units and WordNet synsets, e.g., *synonymy*, *near-synonymy*, *hypernymy*, etc. – see Section 3.5.

[1] Available at `http://www.talp.upc.edu/index.php/technology/resources/multilingual-lexicons-and-machine-translation-resources/multilingual-lexicons/98-wordnet-mappings`.

- **ili_pwn20_synonym_table:** This table contains all synonyms from WordNet for a given ILI record. It thus represents alternative translations for a GermaNet lexical unit.

  id: A unique identifier.

  ili_id: The identifier of the ILI record from ili_table.

  pwn20_synonym: A WordNet synonym from the synset specified in the corresponding ILI record in ili_table identified by ili_id.

- **wkn_sense_table:** This table contains harvested Wiktionary descriptions assigned to GermaNet lexical units. The alignment between GermaNet and Wiktionary is detailed in Chapter 4.

  id: A unique identifier.

  gn_lex_unit_id: The identifier for the lexical unit which is linked to the Wiktionary sense represented by wkn_word_entry_id, wkn_sense_id, and wkn_gloss.

  wkn_word_entry_id: Identifier for a Wiktionary entry. This identifier was assigned by the API[1] used.

  wkn_sense_id: Identifier for the Wikntionary sense inside a Wiktionary entry; again, this identifier was taken from the API[2].

  wkn_gloss: The Wiktionary sense description harvested for a GermaNet lexical unit.

  wkn_gloss_edited: Indicates whether the sense description from Wiktionary has been taken as it was or slightly altered for GermaNet.

[1]The Java-based library JWKTL `http://www.ukp.tu-darmstadt.de/software/jwktl` was used to access all Wiktionary data.

[2]Ibid.

- **compound_table:** This table contains information about the constituents comprising nominal compounds, referring to a lexical unit. A compound consists of a modifier and a head, each of which can have a property. In addition, modifiers also have categories. For some compounds, two alternative modifiers exist. The encoding of compounds is described in Section 3.6.

  id: A unique identifier for this compound entry.

  comp_lex_unit_id: Identifier of the corresponding lexical unit in lex_unit_table for which this compound information is provided.

  modifier1: Orthographic form of the first modifier.

  mod1_property: Specifies a property of the first modifier (optional). See compound_property_table.

  mod1_category: Specifies the category of the first compound modifier (optional). See compound_category_table

  modifier2: Orthographic form of the second modifier, if it exists.

  mod2_property: Optionally specifies a property of the second modifier.

  mod2_category: Optionally specifies the category of the second compound modifier.

  head: Orthographic form of the compound's head.

  head_property: Optionally specifies a property of the compound's head.

- **compound_property_table:** This table contains properties that might be assigned for compound modifiers or heads. The complete list of properties, including explanations and examples, is available in Table 3.3 in Section 3.6. It includes entries such as *abbreviation*, *affixoid*, or *foreign word*.

  id: Unique identifier.

  name: Contains a property for a compound constituent, e.g., *abbreviation*, *affixoid*, or *foreign word* (also see Table 3.3 in Section 3.6).

- **compound_category_table:** This table contains categories such as *adjective*, *adverb*, *noun*, or *particle* assigned to compound modifiers. Table 3.2 in Section 3.6 provides a complete list of categories, including an example for each category.

  id: Unique identifier.

  name: Specifies a category of a compound modifier, e.g., *adjective*, *adverb*, *noun*, or *particle* (see Table 3.2 in Section 3.6).

Appendix C

Gold Standards Lemma Lists

This appendix contains lemma-specific details for the gold standard corpora described in Chapter 6. While knowledge-based word sense disambiguation experiments can be evaluated on all available annotations, the annotations employed for supervised WSD systems have to fulfill certain criteria – as described in Section 6.1. That is, for each of the three sense-annotated corpora deWaC, TüBa-D/Z, and WebCAGe, the entire set of annotations is used for evaluating knowledge-based WSD (Chapter 7) while only those annotations which fulfill the criteria specified in Section 6.1 are used for evaluating supervised WSD experiments (Chapter 8).

Altogether, this appendix includes five tables:

- Table C.1 lists all lemmas in WebCAGe 3.0's supervised gold standard.
- Table C.2 shows details for all lemmas annotated in TüBa-D/Z 9.1.
- Table C.3 provides details on the supervised gold standard subset of TüBa-D/Z 9.1.
- Table C.4 presents all GermaNet 9.0 sense-annotated lemmas available in the deWaC corpus.
- Table C.5 corresponds to the subset of sense-annotated lemmas in the deWaC corpus that is used for supervised WSD experiments.

While there are two tables for both the TüBa-D/Z and deWaC listing (i) all annotated lemmas and (ii) lemmas included in the supervised gold standard subset, for WebCAGe only the list of lemmas included in the supervised gold standard is presented, because it would be too much to list each of the 2 708 lemmas contained in WebCAGe.

Each of the five tables follows the same template. They first list all adjectives (if any), after the horizontal line all nouns, and finally all verbs (see first column *POS*). For each of the word classes, the lemmas are sorted in decreasing order of their number of annotated occurrences in the corresponding gold standard. The overall number of annotations per lemma is stated in the fourth columns, labeled with *Freq.* Besides these overall numbers of annotations per lemma, the supervised gold standard tables, i.e., Tables C.1, C.3, and C.5, show the numbers of annotations contained in the training sets (in columns *Training set freq.*) and in the test sets (columns *Test set freq.*).

While the third columns (labeled as *GN*) refer to the number of word senses the lemma has in GermaNet, i.e., the lemma's polysemy in general, the columns marked with $\#s$ list the numbers of senses for which at least one annotation exists in the gold standard. A superscript $^+$ indicates the existence of at least one token for which no GermaNet sense is annotated. These cases occur, for example, for idiomatic expressions or figurative meanings where it is not obvious from the context which sense to chose. Due to the stratified approach for sampling all annotated data into training and test sets (see Section 6.1), the count of occurring senses in the annotations (columns $\#s$) is preserved for training and test sets.

The ratio between the number of overall annotations and the number of occurring senses is given in columns $F/\#s$.[1] It describes the average number of annotated occurrences per word sense in the overall dataset. The average annotations per sense in the training and test sets are proportionally smaller. The division is supposed to be two thirds and one third of the overall average of occurrences per word sense.[2]

[1]This ratio is especially important for supervised WSD experiments, and thus included for the supervised gold standards.

[2]The calculation of the average numbers of annotated occurrences per sense for the train-

The columns labeled with *Distr. (%)* list the distributions of all occurring word senses as percentages. For example, the most frequent sense of the noun *Frau* in the TüBa-D/Z occurs in 82 percent of all annotations for this lemma (see Table C.2). The other two senses occur in 10 and 8 percent. Due to rounding, the percentages provided in the tables do not necessarily add up to exactly 100. The first number in the *Distr.* columns is comparable to the most frequent sense baseline, i.e., when the WSD system always assigns the sense which has most occurrences in the annotations.

The counts of percentages (in columns *Distr.*) should theoretically match the numbers of occurring senses (given in columns *#s*, also counting the occurrence of *no sense* from GermaNet indicated as superscript $^{+}$). For those lemmas where there are fewer percentage numbers listed than senses annotated, 0 percentages due to rounding down infrequent senses were omitted. For example, there are only 5 percentages listed in Table C.2 for the noun *Land*, although there are 8[1] senses for which at least 1 annotation exists in the gold standard. That is, whenever there are fewer *Distr.* entries than occurring senses, it can be assumed that the senses that are not listed occur very seldomly.

Table C.1 provides details for each of the 43 lemmas (3 adjectives, 33 nouns, and 7 verbs) remaining in WebCAGe 3.0's supervised gold standard, which is described in Subsection 6.3.2.

Table C.1: 66 lemmas in WebCAGe's supervised gold standard.

POS	Lemma	GN	All supervised data				Training set freq.	Test set freq.
			Freq.	#s	F/s	Distr. (%)		
Adj.	preußisch	2	83	2	42	96,4	56	27
	beredt	3	28	2	14	57,43	19	9
	schamlos	2	17	2	9	53,47	12	5
Nouns	Mut	3	184	2	92	91,9	123	61
	San Marino	2	146	2	73	84,16	98	48
	Kongo	2	140	2	70	97,3	94	46
	Option	2	129	2	65	98,2	86	43
	Dank	2	69	2	35	54,46	46	23

Continued on next page

ing and test sets are straight-forward – by respectively dividing the *training set frequency* or the *test set frequency* by the number of annotated senses *#s*.

[1] 7 senses from GermaNet plus the *no sense* annotation.

Table C.1: *Continued from previous page.*

POS	Lemma	GN	All supervised data				Training set freq.	Test set freq.
			Freq.	#s	F/s	Distr. (%)		
Nouns (continued)	Steuer	3	68	2	34	81,19	46	22
	Besetzung	3	62	3	21	89,6,5	42	20
	Bogen	5	61	5	12	48,21,16,10,5	41	20
	Gemeinderat	2	59	2	30	95,5	40	19
	Atrium	3	51	3	17	55,27,18	34	17
	Beichte	2	44	2	22	91,9	30	14
	Eichel	2	39	2	20	87,13	26	13
	Wende	4	39	2	20	64,36	26	13
	Export	2	29	2	15	69,31	20	9
	Harz	2	26	2	13	85,15	18	8
	Pfote	2	26	2	13	58,42	18	8
	Aspiration	3	24	2	12	88,13	16	8
	Holocaust	2	23	2	12	65,35	16	7
	Hydraulik	2	23	2	12	83,17	16	7
	Masche	2	20	2	10	70,30	14	6
	Explosion	2	19	2	10	74,26	13	6
	Muff	2	18	2	9	50,50	12	6
	Disziplin	4	17	3	6	53,24,24	12	5
	Übermut	2	17	2	9	59,41	12	5
	Koks	2	16	2	8	69,31	11	5
	Schöpfer	3	16	2	8	56,44	11	5
	Strang	2	16	2	8	63,38	11	5
	Aufgebot	3	15	2	8	53,47	10	5
	Berufung	5	15	3	5	47,27,27	10	5
	Geflügel	2	15	2	8	80,20	10	5
	Kluft	3	15	3	5	33,33,33	10	5
	Vatikan	3	15	2	8	80,20	10	5
	Verfall	4	15	2	8	60,40	10	5
Verbs	vergessen	2	45	2	23	91,9	30	15
	abschieben	2	25	2	13	84,16	17	8
	verarbeiten	3	24	3	8	54,33,13	16	8
	verdauen	2	19	2	10	63,37	13	6
	wundern	2	17	2	9	65,35	12	5
	begehren	2	16	2	8	56,44	11	5
	präparieren	2	16	2	8	81,19	11	5

Table C.2 lists the entire set of 30 nouns and 79 verbs annotated in TüBa-D/Z 9.1 with lemma-specific details – see Subsection 6.4.2.

Table C.2: 109 lemmas annotated in TüBa-D/Z 9.1.

POS	Lemma	GN	Overall annotations		
			Freq.	#s	Distr. (%)
Nouns	Frau	3	1699	3	82,10,8
	Mann	3	1114	3	90,8,2
	Land	7	1112	7^+	72,21,3,2,1
	Partei	3	811	3	97,3
	Haus	5	789	5	81,9,8,2
	Grund	5	460	5	87,10,2,1
	Stunde	4	426	4	86,9,3,2
	Stimme	4	289	4	52,38,10
	Mal	2	284	1	100
	Kopf	6	269	4^+	92,5,1,1
	Band	6	159	5	71,13,7,6,4
	Tor	4	137	4	40,39,20,1
	Fuß	3	129	3	95,4,1
	Höhe	4	126	4	64,22,8,6
	Freundin	3	122	2	65,35
	Anschlag	5	99	3^+	95,2,2,1
	Spur	5	94	5^+	63,22,9,3,2,1
	Bein	3	91	1^+	99,1
	Runde	6	83	6	43,24,18,8,5,1
	Karte	4	76	4	50,38,9,3
	Sender	5	76	4	71,14,9,5
	Stuhl	3	60	3	88,10,2
	Ausschuß	2	50	1	100
	Bestimmung	6	48	4	71,10,10,8
	Gewinn	3	48	3	75,17,8
	Überraschung	3	42	3	76,24,2
	Teilnahme	3	37	1	100
	Kette	4	25	4	48,40,8,4
	Abfall	4	24	2	96,4
	Abgabe	5	24	3	71,25,4
Verbs	heißen	4	801	4	44,30,26
	gelten	5	502	5	53,35,7,4,1
	setzen	14	404	9^+	26,23,16,11,8,8,3,3,1
	erhalten	4	399	4^+	61,21,19
	sitzen	7	345	6^+	66,11,10,8,3,2,1
	fragen	2	344	2	78,22
	aussehen	2	231	2	83,18
	reden	3	227	3^+	83,8,6,3

Continued on next page

Table C.2: *Continued from previous page.*

POS	Lemma	GN	Overall annotations		
			Freq.	#s	Distr. (%)
Verbs (continued)	sterben	2	220	2	96,4
	ankündigen	2	211	2	98,2
	unterstützen	2	188	2	95,7
	bedeuten	3	187	3	96,3,1
	verkaufen	5	186	4^+	81,11,6,1,1
	verurteilen	2	180	2^+	85,14,1
	leisten	3	176	3	64,23,15
	bauen	3	167	3^+	83,8,8,1
	verschwinden	2	159	2	53,47
	gründen	4	148	4	91,6,1,1
	reichen	4	146	4	60,29,10,1
	geschehen	2	145	2^+	97,2,1
	herrschen	2	128	2	95,6
	präsentieren	2	127	2	70,31
	informieren	2	125	2	86,14
	freuen	2	121	2	83,17
	verdienen	2	115	2	71,29
	demonstrieren	3	111	3	59,26,14
	holen	5	110	5^+	36,28,24,7,4,1
	aufrufen	2	105	2	99,1
	verfolgen	6	105	6	30,28,12,11,10,9
	weitergehen	2	101	2	93,7
	besitzen	2	99	2	67,33
	versichern	3	99	3	89,6,5
	vorliegen	2	96	2	81,19
	enthalten	2	94	2	91,9
	liefern	4	93	4^+	41,29,18,10,3
	erweisen	3	89	3	89,7,4
	existieren	2	88	2	99,1
	drängen	3	84	3	43,32,25
	behandeln	4	82	4	55,24,21,2
	begrüßen	2	79	2	71,29
	beschränken	2	78	2	65,35
	betragen	2	73	1	100
	beraten	3	70	3	60,39,3
	merken	2	62	2	89,11
	entziehen	3	61	2	52,48
	widmen	2	60	2	50,50
	empfehlen	3	58	3	90,9,2
	gestalten	2	57	2	93,7
	bekennen	2	54	2	61,39
	wundern	2	54	2	74,26

Continued on next page

Table C.2: *Continued from previous page.*

POS	Lemma	GN	Overall annotations		
			Freq.	#s	Distr. (%)
Verbs (continued)	auffallen	2	51	2	61,39
	engagieren	2	51	2	78,22
	raten	2	46	2	96,4
	rücken	2	45	2	76,24
	bedenken	3	43	2	72,28
	versammeln	2	42	2	55,45
	vollziehen	2	42	2	60,40
	erweitern	2	41	2	90,10
	zugehen	6	41	4^+	76,12,7,2,2
	gestehen	2	40	2	50,50
	berufen	2	39	2	59,41
	klappen	3	39	2	97,3
	kündigen	2	39	2	77,26
	trauen	4	39	4	46,33,15,13
	stehlen	2	36	1^+	97,3
	verstoßen	2	36	2	97,3
	zurückgeben	2	36	2	75,25
	ärgern	2	36	2	53,47
	befassen	2	35	2^+	89,9,3
	einschränken	2	35	2	94,6
	identifizieren	3	34	3	53,29,18
	beschweren	3	33	2	91,9
	vorschreiben	2	31	1	100
	nützen	2	29	2	97,3
	kleben	2	28	2	61,39
	verdoppeln	2	26	2	54,46
	fressen	3	25	2	64,36
	wiedergeben	3	24	2	79,21
	verlesen	2	21	1	100

Table C.3 gives details of the supervised gold standard subset of TüBa-D/Z 9.1, as explained in Subsection 6.4.3. This subset contains 24 nouns and 68 verbs, i.e., 92 lemmas altogether.

Table C.3: 92 lemmas in TüBa-D/Z's supervised gold standard.

POS	Lemma	GN	All supervised data				Training set freq.	Test set freq.
			Freq.	#s	F/s	Distr. (%)		
Nouns	Frau	3	1699	3	566	82,10,8	1133	566
	Mann	3	1114	3	371	90,8,2	743	371
	Land	7	1111	6^+	159	72,21,3,2,1	741	370
	Partei	3	811	3	270	97,3	541	270
	Haus	5	788	4	197	81,9,8,2	526	262
	Grund	5	458	4	115	87,10,2,1	306	152
	Stunde	4	426	4	107	86,9,3,2	284	142
	Stimme	4	288	3	96	52,38,10	192	96
	Kopf	6	268	3^+	67	93,5,1,1	179	89
	Band	6	159	5	32	71,13,7,6,4	106	53
	Tor	4	136	3	45	40,40,20	91	45
	Fuß	3	128	2	64	96,4	86	42
	Höhe	4	126	4	32	64,22,8,6	84	42
	Freundin	3	122	2	61	65,35	82	40
	Spur	5	91	4	23	65,23,9,3	61	30
	Runde	6	82	5	16	44,24,18,9,5	55	27
	Sender	5	76	4	19	71,14,9,5	51	25
	Karte	4	74	3	25	51,39,9	50	24
	Stuhl	3	59	2	30	90,10	40	19
	Bestimmung	6	48	4	12	71,10,10,8	32	16
	Gewinn	3	48	3	16	75,17,8	32	16
	Überraschung	3	41	2	21	76,24	28	13
	Abgabe	5	23	2	12	74,26	16	7
	Kette	4	22	2	11	55,45	15	7
Verbs	heißen	4	799	3	266	44,30,26	533	266
	gelten	5	502	5	100	53,35,7,4,1	335	167
	setzen	14	403	8^+	45	27,23,16,11,8,8, 3,3,1	269	134
	erhalten	4	397	3	132	60,21,19	265	132
	sitzen	7	345	6^+	49	66,11,10,8,3,2,1	230	115
	fragen	2	344	2	172	78,22	230	114
	aussehen	2	231	2	116	82,18	154	77
	reden	3	227	3^+	57	83,8,6,3	152	75
	sterben	2	220	2	110	96,4	147	73
	ankündigen	2	211	2	106	98,2	141	70
	unterstützen	2	188	2	94	93,7	126	62
	bedeuten	3	185	2	93	97,3	124	61

Continued on next page

Table C.3: *Continued from previous page.*

POS	Lemma	GN	All supervised data				Training set freq.	Test set freq.
			Freq.	#s	F/s	Distr. (%)		
Verbs (continued)	verkaufen	5	184	3	61	82,11,7	123	61
	verurteilen	2	179	2	90	85,15	120	59
	leisten	3	176	3	59	64,23,13	118	58
	bauen	3	165	3	55	84,8,8	110	55
	verschwinden	2	159	2	80	53,47	106	53
	reichen	4	145	3	48	61,30,10	97	48
	gründen	4	144	2	72	94,6	96	48
	geschehen	2	143	2	72	98,2	96	47
	herrschen	2	128	2	64	95,5	86	42
	präsentieren	2	127	2	64	70,30	85	42
	informieren	2	125	2	63	86,14	84	41
	freuen	2	121	2	61	83,17	81	40
	verdienen	2	115	2	58	71,29	77	38
	demonstrieren	3	111	3	37	59,26,14	74	37
	holen	5	109	4^+	22	37,28,24,7,4	73	36
	verfolgen	6	105	6	18	30,28,12,11,10,9	70	35
	weitergehen	2	101	2	51	93,7	68	33
	besitzen	2	99	2	50	67,33	66	33
	versichern	3	99	3	33	89,6,5	66	33
	vorliegen	2	96	2	48	81,19	64	32
	enthalten	2	94	2	47	91,9	63	31
	liefern	4	93	4^+	19	41,28,18,10,3	62	31
	erweisen	3	89	3	30	89,7,4	60	29
	drängen	3	84	3	28	43,32,25	56	28
	behandeln	4	80	3	27	56,23,21	54	26
	begrüßen	2	79	2	40	71,29	53	26
	beschränken	2	78	2	39	65,35	52	26
	beraten	3	68	2	34	60,40	46	22
	merken	2	62	2	31	89,11	42	20
	entziehen	3	61	2	31	52,48	41	20
	widmen	2	60	2	30	50,50	40	20
	empfehlen	3	57	2	29	91,9	38	19
	gestalten	2	57	2	29	93,7	38	19
	bekennen	2	54	2	27	61,39	36	18
	wundern	2	54	2	27	74,26	36	18
	auffallen	2	51	2	26	61,39	34	17
	engagieren	2	51	2	26	78,22	34	17
	rücken	2	45	2	23	76,24	30	15
	bedenken	3	43	2	22	72,28	29	14
	versammeln	2	42	2	21	55,45	28	14
	vollziehen	2	42	2	21	60,40	28	14
	erweitern	2	41	2	21	90,10	28	13

Continued on next page

Table C.3: *Continued from previous page.*

POS	Lemma	GN	All supervised data				Training set freq.	Test set freq.
			Freq.	#s	F/s	Distr. (%)		
Verbs (continued)	gestehen	2	40	2	20	50,50	27	13
	berufen	2	39	2	20	59,41	26	13
	kündigen	2	39	2	20	74,26	26	13
	trauen	4	39	4	10	44,28,15,13	26	13
	zugehen	6	39	3	13	79,13,8	26	13
	zurückgeben	2	36	2	18	75,25	24	12
	ärgern	2	36	2	18	53,47	24	12
	befassen	2	34	2	17	91,9	23	11
	identifizieren	3	34	3	11	53,29,18	23	11
	beschweren	3	33	2	17	91,9	22	11
	kleben	2	28	2	14	61,39	19	9
	verdoppeln	2	26	2	13	54,46	18	8
	fressen	3	25	2	13	64,36	17	8
	wiedergeben	3	24	2	12	79,21	16	8

Table C.4 presents all GermaNet 9.0 sense annotations available in the deWaC corpus. As outlined in Subsection 6.5.3, these annotations comprise 37 lemmas (of which 4 are adjectives, 18 are nouns, and 15 are verbs).

Table C.4: 37 lemmas annotated in deWaC.

POS	Lemma	GN	Overall annotations		
			Freq.	#s	Distr. (%)
Adj.	frei	9	30	8	37,30,10,7,7,3,3,3
	fein	6	20	3	70,25,5
	kühl	2	20	2	80,20
	natürlich	5	20	3	45,35,20
Nouns	Schaltung	5	30	4	53,40,3,3
	Zug	11	30	5	43,37,13,3,3
	Nutzen	3	25	2	60,40
	Anfall	2	20	2	75,25
	Arbeit	4	20	2	60,40
	Aufschlag	4	20	2	55,45
	Autorität	2	20	2	55,45
	Bar	3	20	1	100
	Blende	3	20	2^+	55,25,20
	Einrichtung	4	20	3	75,15,10
	Halt	4	20	4	45,25,25,5
	Kanal	5	20	4^+	30,25,20,15,10
	Natur	3	20	2	70,30
	Ohnmacht	2	20	2	65,35
	Post	3	20	2	80,20
	Sinn	3	20	2	90,10
	Spiel	6	20	5^+	55,15,10,10,5,5
	Stuhl	3	20	3	80,15,5
Verbs	halten	26	127	19^+	35,9,9,9,6,6,5,4,2,2,2,2,2,2,1,1,1,1,1,1
	spielen	15	75	8^+	41,21,21,5,4,3,1,1,1
	sehen	11	60	8	23,18,17,13,12,12,3,2
	treiben	11	53	8	34,25,15,13,8,2,2,2
	fahren	6	35	4	46,46,6,3
	aufschlagen	8	30	6^+	37,30,17,7,3,3,3
	finden	4	30	4	57,27,13,3
	leben	4	30	3	67,30,3
	passen	5	30	4	80,13,3,3
	tragen	11	30	7	40,33,10,7,3,3,3
	verlassen	3	30	3	80,13,7
	arbeiten	4	20	4	85,5,5,5
	benutzen	3	20	2	90,10
	freimachen	4	20	3	90,5,5
	greifen	3	18	2^+	61,22,17

Table C.5 lists the subset of 31 sense-annotated lemmas (4 adjectives, 15 nouns, and 12 verbs) in the deWaC corpus that is used for supervised WSD experiments. See Subsection 6.5.4 for a description.

Table C.5: 31 lemmas in deWaC's supervised gold standard.

POS	Lemma	GN	All supervised data				Training set freq.	Test set freq.
			Freq.	#s	F/s	Distr. (%)		
Adj.	frei	9	23	3	8	48,39,13	16	7
	kühl	2	20	2	10	80,20	14	6
	natürlich	5	20	3	7	45,35,20	14	6
	fein	6	19	2	10	74,26	13	6
Nouns	Schaltung	5	28	2	14	57,43	19	9
	Zug	11	28	3	9	46,39,14	19	9
	Nutzen	3	25	2	13	60,40	17	8
	Anfall	2	20	2	10	75,25	14	6
	Arbeit	4	20	2	10	60,40	14	6
	Aufschlag	4	20	2	10	55,45	14	6
	Autorität	2	20	2	10	55,45	14	6
	Blende	3	20	2^+	7	55,25,20	14	6
	Natur	3	20	2	10	70,30	14	6
	Ohnmacht	2	20	2	10	65,35	14	6
	Post	3	20	2	10	80,20	14	6
	Halt	4	19	3	6	47,26,26	13	6
	Stuhl	3	19	2	10	84,16	13	6
	Einrichtung	4	18	2	9	83,17	12	6
	Kanal	5	18	4	5	33,28,22,17	12	6
Verbs	halten	26	115	11	10	38,10,10,10,7,6,5,4,3,3,3	77	38
	spielen	15	70	4^+	14	44,23,23,6,4	47	23
	sehen	11	57	6	10	25,19,18,14,12,12	38	19
	treiben	11	50	5	10	36,26,16,14,8	34	16
	fahren	6	32	2	16	50,50	22	10
	finden	4	29	3	10	59,28,14	20	9
	leben	4	29	2	15	69,31	20	9
	passen	5	28	2	14	86,14	19	9
	verlassen	3	28	2	14	86,14	19	9
	aufschlagen	8	25	2^+	8	44,36,20	17	8
	tragen	11	25	3	8	48,40,12	17	8
	greifen	3	18	2^+	6	61,22,17	12	6

Appendix D

WSD Results on Test Set versus Results by Cross-Validation

The two main approaches to evaluate the performance of supervised algorithms are by cross-validation or on a separate, unseen test set (see Subsection 2.2.2 for details, including advantages and disadvantages of the two procedures). Since the availability of sense annotations is sparse, the tuning of the supervised machine learning WSD systems in Chapter 8 is evaluated by 10-fold cross-validation on all available training data. During this tuning process, the test set is held back and used only to obtain final evaluation results. In order to realistically, meaningfully, and accurately estimate the algorithms' ability to generalize after several experiments, the WSD experiments reported in Chapter 8 is evaluated on separate test sets for each of the available sense-annotated corpora.

In order to judge the impact of the two evaluation procedures and to analyze how much the evaluation results differ for the two procedures, Table D.1 compares the results obtained on separate test sets (marked with *Test set* in column *Eval. type*) with the results obtained by 10-fold cross-validation on all available training data (marked with *10-CV*). Analogously to Table 8.4 in Subsection 8.3.1, Table D.1 includes WSD performance scores in terms of the F_1-measure separately for the available sense-annotated corpora and the

supervised classifiers.[1]

In general, the performance results reported for the separate test sets are minimally higher than those obtained by cross-validation. However, there are only small differences – mostly below 2.0 F_1-score points. This confirms the viability of the proposed evaluation procedure for the task at hand.

Table D.1: Comparison of WSD results evaluated by cross-validation compared to results obtained on separate test sets (F-score).

Classifier		**Eval. type**	**WebCAGe**	**TüBa-D/Z (manual)**	**TüBa-D/Z (automatic)**	**deWaC**
Baselines	*ZeroR*	10-CV	76.37	74.26	74.26	52.76
		Test set	79.72 ↷ +3.3	74.25 ↷ ≈	74.25 ↷ ≈	53.31 ↷ +0.6
	OneR	10-CV	65.43	64.41	53.95	40.42
		Test set	64.86 ↷ –0.6	65.42 ↷ +1.0	54.64 ↷ +0.7	43.90 ↷ +3.5
Decision rules	*PART*	10-CV	80.66	86.29	81.62	56.66
		Test set	86.36 ↷ +5.7	86.78 ↷ +0.5	82.95 ↷ +1.3	56.45 ↷ –0.2
	Decision-Table	10-CV	80.57	85.57	81.06	55.84
		Test set	81.12 ↷ +0.6	86.28 ↷ +0.7	82.42 ↷ +1.4	56.79 ↷ +0.9
	J48	10-CV	81.50	87.26	82.24	58.28
		Test set	88.29 ↷ +6.8	87.75 ↷ +0.5	83.79 ↷ +1.6	59.58 ↷ +1.3
Lazy	*IBk*	10-CV	80.91	80.04	78.42	56.82
		Test set	84.09 ↷ +3.2	81.76 ↷ +1.7	79.49 ↷ +1.1	60.28 ↷ +3.5
Probabilistic	*Naive-Bayes*	10-CV	85.53	82.74	80.25	57.63
		Test set	89.16 ↷ +3.6	84.06 ↷ +1.3	80.79 ↷ +0.5	58.54 ↷ +0.9
	Bayes-Net	10-CV	84.52	86.65	83.90	60.23
		Test set	89.51 ↷ +5.0	87.48 ↷ +0.8	84.82 ↷ +0.9	60.28 ↷ +0.1
	Logistic	10-CV	86.21	81.28	79.02	56.98
		Test set	89.69 ↷ +3.5	82.03 ↷ +0.8	80.68 ↷ +1.7	53.66 ↷ –3.3
SVM	*SMO*	10-CV	87.22	87.68	84.92	62.18
		Test set	91.26 ↷ +4.0	88.51 ↷ +0.8	85.32 ↷ +0.4	63.76 ↷ +1.6
	LibSVM	10-CV	81.83	79.17	76.49	53.08
		Test set	85.14 ↷ +3.3	79.65 ↷ +0.5	76.63 ↷ +0.1	53.31 ↷ +0.2
Combined	*Vote*	10-CV	86.21	87.35	83.96	60.55
		Test set	92.66 ↷ +6.5	88.67 ↷ +1.3	85.47 ↷ +1.5	62.72 ↷ +2.2
	AdaBoostM1	10-CV	84.02	84.31	80.72	54.22
		Test set	85.14 ↷ +1.1	84.93 ↷ +0.6	81.33 ↷ +0.6	56.10 ↷ +1.9

[1]The WSD results reported for separate test sets in Table D.1 correspond to the results reported in Table 8.4. The reason why the numbers in the two tables are not identical is because Table 8.4 reports separate numbers for adjectives, nouns, and verbs, while Table D.1 presents average numbers over all word classes.

References

Eneko Agirre and Phil Edmonds, editors. *Word Sense Disambiguation: Algorithms and Applications*. Springer, 2006a. 5, 11, 20, 29, 30, 32

Eneko Agirre and Philip Edmonds. Introduction. In Eneko Agirre and Philip Edmonds, editors, *Word Sense Disambiguation: Algorithms and Applications*, volume 33 of *Text, Speech and Language Technology*, chapter 1, pages 1–28. Springer Netherlands, Dordrecht, The Netherlands, 2006b. ISBN 978-1-4020-4808-1. doi: 10.1007/978-1-4020-4809-8_1. URL `http://dx.doi.org/10.1007/978-1-4020-4809-8_1`. 20

Eneko Agirre and Oier Lopez De Lacalle. Clustering WordNet Word Senses. In *Proceedings of the Conference on Recent Advances on Natural Language*, RANLP'03, pages 121–130, Borovetz, Bulgaria, 2003. 10

Eneko Agirre and Oier Lopez de Lacalle. Publicly Available Topic Signatures for all WordNet Nominal Senses. In *Proceedings of the 4th International Conference on Languages Resources and Evaluations*, LREC '04, pages 1123–1126, 2004. 32, 109, 110, 119, 120

Eneko Agirre and David Martínez. The Basque Country University system: English and Basque tasks. In Rada Mihalcea and Phil Edmonds, editors, *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 44–48, Barcelona, Spain, July 2004. Association for Computational Linguistics. 42, 43, 47, 49, 263, 288

Eneko Agirre and Aitor Soroa. Personalizing PageRank for Word Sense Disambiguation. In *Proceedings of the 12th Conference of the European Chapter*

of the Association for Computational Linguistics, EACL '09, pages 33–41, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. 53, 96, 197, 214

Eneko Agirre and Mark Stevenson. Knowledge Sources for WSD. In Eneko Agirre and Philip Edmonds, editors, *Word Sense Disambiguation: Algorithms and Applications*, volume 33 of *Text, Speech and Language Technology*, chapter 8, pages 217–252. Springer Netherlands, Dordrecht, The Netherlands, 2006. ISBN 978-1-4020-4808-1. doi: 10.1007/978-1-4020-4809-8_8. URL `http://dx.doi.org/10.1007/978-1-4020-4809-8_8`. 21

Eneko Agirre, Lluís Màrquez, and Richard Wicentowski, editors. *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, Stroudsburg, PA, USA, June 2007. Association for Computational Linguistics. URL `http://aclweb.org/anthology/S07-1001`. 11, 26, 28, 29, 147

Eneko Agirre, Johan Bos, Mona Diab, Suresh Manandhar, Yuval Marton, and Deniz Yuret, editors. **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, Montréal, Canada, June 2012. Association for Computational Linguistics. URL `http://aclweb.org/anthology/S12-1001`. 28

David W. Aha, Dennis Kibler, and Marc K. Albert. Instance-Based Learning Algorithms. *Machine Learning*, 6(1):37–66, 1991. ISSN 0885-6125. doi: 10.1007/BF00153759. URL `http://dx.doi.org/10.1007/BF00153759`. 253

Rehan Akbani, Stephen Kwek, and Nathalie Japkowicz. Applying Support Vector Machines to Imbalanced Datasets. In Jean-François Boulicaut, Floriana Esposito, Fosca Giannotti, and Dino Pedreschi, editors, *Machine Learning: ECML 2004*, volume 3201 of *Lecture Notes in Computer Science*, pages 39–50. Springer Berlin Heidelberg, 2004. ISBN 978-3-540-23105-9.

doi: 10.1007/978-3-540-30115-8_7. URL http://dx.doi.org/10.1007/978-3-540-30115-8_7. 280

Sue Atkins. Tools for Computer-aided Corpus Lexicography: the Hector Project. *Acta Linguistica Hungarica*, 41:5–72, 1993. 101, 103

R. Harald Baayen, Richard Piepenbrock, and Léon Gulikers. The CELEX Lexical Database (CD-ROM), 1995. 69, 240

Satanjeev Banerjee and Ted Pedersen. An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet. In *Proceedings of the Third International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing '02, pages 136–145, London, UK, 2002. Springer-Verlag. ISBN 3-540-43219-1. 34, 35, 36

Satanjeev Banerjee and Ted Pedersen. Extended gloss overlaps as a measure of semantic relatedness. In *Proceedings of the 18th international joint conference on Artificial intelligence*, IJCAI'03, pages 805–810, San Francisco, CA, USA, 2003. Morgan Kaufmann Publishers Inc. 33, 34, 35, 36, 37, 86, 180

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226, 2009. 107, 160, 162, 195

Dominik Bas, Bartosz Broda, and Maciej Piasecki. Towards Word Sense Disambiguation of Polish. In *Proceedings of the International Multiconference on Computer Science and Information Technology, IMCSIT 2008, Wisla, Poland, 20-22 October 2008*, pages 73–78, 2008. doi: 10.1109/IMCSIT.2008.4747220. URL http://dx.doi.org/10.1109/IMCSIT.2008.4747220. 41, 42, 43, 45, 47, 250

Pierpaolo Basile, Marco de Gemmis, Anna Lisa Gentile, Pasquale Lops, and Giovanni Semeraro. UNIBA: JIGSAW algorithm for Word Sense Disambiguation. In *Proceedings of the Fourth International Workshop on Se-*

mantic Evaluations, SemEval-2007, pages 398–401, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL http://aclweb.org/anthology/S07-1088. 35, 38

Pierpaolo Basile, Annalina Caputo, and Giovanni Semeraro. An Enhanced Lesk Word Sense Disambiguation Algorithm through a Distributional Semantic Model. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1591–1600, Dublin, Ireland, August 2014. Dublin City University and Association for Computational Linguistics. URL http://www.aclweb.org/anthology/C14-1151. 35

Laurie Bauer. *English Word-Formation*. Cambridge Textbooks in Linguistics. Cambridge University Press, 1983. ISBN 9780521284929. 68

Thomas Bayes. An Essay towards Solving a Problem in the Doctrine of Chances. By the Late Rev. Mr. Bayes, communicated by Mr. Price, in a letter to John Canton, M. A. F. R. S. *Philosophical Transactions (1683-1775)*, 53:370–418, 1763. doi: 10.1098/rstl.1763.0053. URL http://dx.doi.org/10.1098/rstl.1763.0053. 47

Asa Ben-Hur and Jason Weston. A User's Guide to Support Vector Machines. In Oliviero Carugo and Frank Eisenhaber, editors, *Data Mining Techniques for the Life Sciences*, volume 609 of *Methods in Molecular Biology*, chapter 13, pages 223–239. Humana Press, 2010. ISBN 978-1-60327-240-7. doi: 10.1007/978-1-60327-241-4_13. URL http://dx.doi.org/10.1007/978-1-60327-241-4_13. 280

Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics*, 22(1):39–71, March 1996. ISSN 0891-2017. 47

Daniel Berleant. Engineering "word experts" for word disambiguation. *Natural Language Engineering*, 1:339–362, 12 1995. ISSN 1469-8110. doi: 10.1017/S1351324900000255. URL http://journals.cambridge.org/article_S1351324900000255. 39, 260

Chris Biemann. Word Sense Induction and Disambiguation. In *Structure Discovery in Natural Language*, Theory and Applications of Natural Language Processing, chapter 7, pages 145–155. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-25922-7. doi: 10.1007/978-3-642-25923-4_7. URL http://dx.doi.org/10.1007/978-3-642-25923-4_7. 250, 260

Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. Joint Learning of Words and Meaning Representations for Open-Text Semantic Parsing. In Neil D. Lawrence and Mark Girolami, editors, *AISTATS*, volume 22 of *JMLR Proceedings*, pages 127–135. JMLR.org, 2012. URL http://jmlr.csail.mit.edu/proceedings/papers/v22/bordes12.html. 295

Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A Training Algorithm for Optimal Margin Classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 144–152, New York, NY, USA, 1992. ACM. ISBN 0-89791-497-X. doi: 10.1145/130385.130401. URL http://doi.acm.org/10.1145/130385.130401. 49

Remco R. Bouckaert, Eibe Frank, Mark Hall, Richard Kirkby, Peter Reutemann, Alex Seewald, and David Scuse. WEKA Manual for Version 3-6-10. Technical report, University of Waikato, July 2013. 250, 255

Sabine Brants, Stefanie Dipper, Peter Eisenberg, Silvia Hansen-Schirra, Esther König, Wolfgang Lezius, Christian Rohrer, George Smith, and Hans Uszkoreit. TIGER: Linguistic Interpretation of a German Corpus. *Research on Language and Computation*, 2(4):597–620, 2004. ISSN 1570-7075. URL http://dx.doi.org/10.1007/s11168-004-7431-3. 164, 222

Samuel Broscheit, Anette Frank, Dominic Jehle, Simone Paolo Ponzetto, Danny Rehl, Anja Summa, Klaus Suttner, and Saskia Vola. Rapid bootstrapping of Word Sense Disambiguation resources for German. In *Proceedings of the 10. Konferenz zur Verarbeitung Natürlicher Sprache*, pages 19–27, Saarbrücken, Germany, 2010. 13, 16, 51, 53, 107, 108, 126, 141, 145, 160, 162, 192, 195, 196, 197, 202, 206, 212, 214, 293

Rebecca Bruce and Janyce Wiebe. Word-Sense Disambiguation Using Decomposable Models. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, ACL'94, pages 139–146, Las Cruces, New Mexico, USA, June 1994. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P94-1020`. 105

Rebecca Bruce and Janyce Wiebe. Word sense distinguishability and inter-coder agreement. In *Proceedings of the 3rd Conference on Empirical Methods in Natural Language Processing*, EMNLP'98, pages 53–60, Granada, Spain, June 1998. Association for Computational Linguistics SIGDAT. 105

Alexander Budanitsky and Graeme Hirst. Evaluating WordNet-based Measures of Lexical Semantic Relatedness. *Compututational Linguistics*, 32: 13–47, 2006. ISSN 0891-2017. 172, 173

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó, and Manfred Pinkal. *Using FrameNet for the semantic analysis of German: Annotation, representation, and automation*, pages 209–244. Mouton, 2009. 51

Clara Cabezas and Philip Resnik. Using WSD Techniques for Lexical Selection in Statistical Machine Translation. Technical Report LAMP-TR-124,CS-TR-4736,UMIACS-TR-2005-42, University of Maryland, College Park, July 2005. 276

Clara Cabezas, Philip Resnik, and Jessica Stevens. Supervised Sense Tagging using Support Vector Machines. In *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 59–62, Toulouse, France, July 2001. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/S01-1014`. 49

Marine Carpuat and Dekai Wu. Word Sense Disambiguation vs. Statistical Machine Translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, ACL'05, pages 387–394, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

doi: 10.3115/1219840.1219888. URL http://www.aclweb.org/anthology/P05-1048. 276

Marine Carpuat and Dekai Wu. Improving Statistical Machine Translation Using Word Sense Disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 61–72, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D07-1007. 276

Yee Seng Chan, Hwee Tou Ng, and David Chiang. Word Sense Disambiguation Improves Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 33–40, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P07-1005. 276

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3): 27:1–27:27, April 2011. ISSN 2157-6904. doi: 10.1145/1961189.1961199. URL http://doi.acm.org/10.1145/1961189.1961199. 256, 264

Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. *Semi-Supervised Learning*. Adaptive computation and machine learning. MIT Press, 2006. ISBN 9780262033589. 32

Eugene Charniak. A Maximum-entropy-inspired Parser. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*, NAACL 2000, pages 132–139, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics. 223

Jinying Chen. *TOWARDS HIGH-PERFORMANCE WORD SENSE DISAMBIGUATION BY COMBINING RICH LINGUISTIC KNOWLEDGE AND MACHINE LEARNING APPROACHES*. PhD thesis, University of Pennsylvania, 2006. 48

Jinying Chen and Martha Palmer. Towards Robust High Performance Word Sense Disambiguation of English Verbs Using Rich Linguistic Features. In

Robert Dale, Kam-Fai Wong, Jian Su, and Oi Yee Kwong, editors, *Natural Language Processing – IJCNLP 2005*, volume 3651 of *Lecture Notes in Computer Science*, pages 933–944. Springer Berlin Heidelberg, 2005. ISBN 978-3-540-29172-5. doi: 10.1007/11562214_81. URL http://dx.doi.org/10.1007/11562214_81. 276, 281

Jinying Chen and Martha Palmer. Improving English Verb Sense Disambiguation Performance with Linguistically Motivated Features and Clear Sense Distinction Boundaries. *Language Resources and Evaluation*, 43(2):181–208, 2009. ISSN 1574-020X. 120, 237, 276, 281

Jinying Chen, Andrew Schein, Lyle Ungar, and Martha Palmer. An Empirical Study of the Behavior of Active Learning for Word Sense Disambiguation. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 120–127, New York City, USA, June 2006. Association for Computational Linguistics. URL http://aclweb.org/anthology/N06-1016. 295

Jinying Chen, Dmitriy Dligach, and Martha Palmer. Towards Large-scale High-Performance English Verb Sense Disambiguation by Using Linguistically Motivated Features. *2012 IEEE Sixth International Conference on Semantic Computing*, 0:378–388, 2007. URL http://doi.ieeecomputersociety.org/10.1109/ICSC.2007.69. 276, 281

Xinxiong Chen, Zhiyuan Liu, and Maosong Sun. A Unified Model for Word Sense Representation and Disambiguation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1025–1035. Association for Computational Linguistics, October 2014. URL http://aclweb.org/anthology/D14-1110. 295

Yukun Chen, Hongxin Cao, Qiaozhu Mei, Kai Zheng, and Hua Xu. Applying active learning to supervised word sense disambiguation in MEDLINE. *Journal of the American Medical Informatics Association (JAMIA)*, 20(5):1001–1006, 2013. doi: doi:10.1136/amiajnl-2012-001244. 295

Timothy Chklovski and Rada Mihalcea. Building a Sense Tagged Corpus with Open Mind Word Expert. In *Proceedings of the ACL-02 Workshop on Word Sense Disambiguation: Recent Successes and Future Directions*, pages 116–122. Association for Computational Linguistics, July 2002. URL http://www.aclweb.org/anthology/W02-0817. 104, 106

Jacob Cohen. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46, Apr 1960. 138

Michael Collins. *Head-driven Statistical Models for Natural Language Parsing*. PhD thesis, University of Pennsylvania, 1999. 223

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537, November 2011. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1953048.2078186. 295

Gregory F. Cooper and Edward Herskovits. A Bayesian Method for Constructing Bayesian Belief Networks from Databases. In *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, UAI'91, pages 86–94, San Francisco, CA, USA, 1991. Morgan Kaufmann Publishers Inc. ISBN 1-55860-203-8. URL http://dl.acm.org/citation.cfm?id=2100662.2100674. 255

Gregory F. Cooper and Edward Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4): 309–347, 1992. ISSN 0885-6125. doi: 10.1007/BF00994110. URL http://dx.doi.org/10.1007/BF00994110. 255

Corinna Cortes and Vladimir Vapnik. Support-Vector Networks. *Machine Learning*, 20(3):273–297, September 1995. ISSN 0885-6125. doi: 10.1023/A:1022627411411. URL http://dx.doi.org/10.1023/A:1022627411411. 48, 256

Thomas Cover and Peter Hart. Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory*, 13(1):21–27, September 1967. ISSN

0018-9448. doi: 10.1109/TIT.1967.1053964. URL http://dx.doi.org/10.1109/TIT.1967.1053964. 45

Stephen Crane. *The Red Badge of Courage*. D. Appleton & Company, New York, USA, 1895. 104

D. Alan Cruse. *Lexical Semantics*. Cambridge Textbooks in Linguistics. Cambridge University Press, 1986. ISBN 9780521276436. 7

D. Alan Cruse. *A glossary of semantics and pragmatics*. Glossaries in Linguistics Series. Edinburgh University Press, 2006. ISBN 9780748621118. 5

Walter Daelemans and Véronique Hoste. Evaluation of Machine Learning Methods for Natural Language Processing Tasks. In *Proceedings of the Third International Conference on Language Resources and Evaluation*, LREC'02, pages 755–760, Las Palmas, Canary Islands - Spain, May 2002. European Language Resources Association (ELRA). URL http://aclweb.org/anthology/L02-1094. 45, 46

Walter Daelemans, Antal Van Den Bosch, and Jakub Zavrel. Forgetting Exceptions is Harmful in Language Learning. *Machine Learning*, 34(1-3): 11–41, Feb 1999. ISSN 0885-6125. doi: 10.1023/A:1007585615670. URL http://dx.doi.org/10.1023/A:1007585615670. 45, 46

Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. TiMBL: Tilburg Memory Based Learner, version 6.3, Reference Guide. Ilk technical report – ilk 10-01, Induction of Linguistic Knowledge, Tilburg University and CLiPS, University of Antwerp, 2010. 294

Hoa Trang Dang and Martha Palmer. Combining Contextual Features for Word Sense Disambiguation. In *Proceedings of the ACL-02 Workshop on Word Sense Disambiguation: Recent Successes and Future Directions*, pages 88–94, Stroudsburg, PA, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1118675.1118688. URL http://www.aclweb.org/anthology/W02-0813. 48

Rodrigo de Oliveira, Lucas Hausmann, and Desislava Zhekova. Is Three the Optimal Context Window for Memory-Based Word Sense Disambiguation? In Irina Temnikova, Ivelina Nikolova, and Natalia Konstantinova, editors, *RANLP Student Research Workshop*, pages 91–96. RANLP 2011 Organising Committee, 2011. URL `http://www.aclweb.org/anthology/R11-2014`. 26, 39, 147, 148, 294, 295

Bart Decadt, Véronique Hoste, Walter Daelemans, and Antal Van den Bosch. GAMBL, Genetic Algorithm Optimization of Memory-Based WSD. In Rada Mihalcea and Phil Edmonds, editors, *Proceedings of the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text (Senseval-3)*, pages 108–112, 2004. 45, 260

Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959. 96

Georgiana Dinu and Sandra Kübler. Sometimes less is more: Romanian word sense disambiguation revisited. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, RANLP 2007, Borovets, Bulgaria, 2007. 26, 39, 147, 269, 294

Dmitriy Dligach and Martha Palmer. Improving Verb Sense Disambiguation with Automatically Retrieved Semantic Knowledge. In *Proceedings of Second IEEE International Conference on Semantic Computing (ICSC)*, pages 182–189, Santa Clara, CA, 2008. IEEE Computer Society. URL `http://doi.ieeecomputersociety.org/10.1109/ICSC.2008.48`. 237, 276, 281

Erich Drach. *Grundgedanken der deutschen Satzlehre*. Verlag M. Diesterweg, Frankfurt am Main, 1937. 125

Kai-Bo Duan and S. Sathiya Keerthi. Which Is the Best Multiclass SVM Method? An Empirical Study. In Nikunj C. Oza, Robi Polikar, Josef Kittler, and Fabio Roli, editors, *Multiple Classifier Systems*, volume 3541 of *Lecture Notes in Computer Science*, pages 278–285. Springer Berlin Heidelberg, 2005. ISBN 978-3-540-26306-7. doi: 10.1007/11494683_28. URL `http://dx.doi.org/10.1007/11494683_28`. 49

Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis*. John Willey & Sons, New Yotk, 1973. 46, 254

Philip Edmonds and Scott Cotton. SENSEVAL-2: Overview. In *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 1–5, Toulouse, France, July 2001. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/S01-1001`. 23, 26, 104, 147

Peter Eisenberg. *Das Wort – Grundriss der deutschen Grammatik*. Verlag J. B. Melzer, Stuttgart/Weimar, Germany, 3rd edition, 2006. 66, 67, 84

Katrin Erk. Frame assignment as word sense disambiguation. In *Proceedings of IWCS-6*, volume 6, Tilburg, The Netherlands, 2005. 51

Katrin Erk and Carlo Strapparava, editors. *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, Stroudsburg, PA, USA, July 2010. Association for Computational Linguistics. URL `http://aclweb.org/anthology/S10-1001`. 28

Gerard Escudero, Lluís Màrquez, and German Rigau. Boosting Applied Toe Word Sense Disambiguation. In *Proceedings of the 11th European Conference on Machine Learning*, ECML '00, pages 129–141, London, UK, UK, 2000a. Springer-Verlag. ISBN 3-540-67602-3. URL `http://dl.acm.org/citation.cfm?id=645327.649539`. 50

Gerard Escudero, Lluís Màrquez, and German Rigau. Naive Bayes and Exemplar-based Approaches to Word Sense Disambiguation Revisited. In Werner Horn, editor, *ECAI*, pages 421–425. IOS Press, 2000b. 45, 47

Gerard Escudero Bakx. *Machine Learning Techniques for Word Sense Disambiguation*. PhD thesis, Universitat Politècnica de Catalunya, Barcelona, Catalunya, 2006. 148, 260

Rong-En Fan, Pai-Hsuen Chen, and Chih-Jen Lin. Working Set Selection Using Second Order Information for Training Support Vector Machines. *Journal of*

Machine Learning Research, 6:1889–1918, December 2005. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1046920.1194907. 256

Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, 1998a. 6, 20, 55

Christiane Fellbaum. A Semantic Network of English Verbs. In Christiane Fellbaum, editor, *WordNet: An Electronic Lexical Database*, chapter 3, pages 69–104. MIT Press, Cambridge, MA, 1998b. 64

Christiane Fellbaum, Martha Palmer, Hoa Trang Dang, Lauren Delfs, and Susanne Wolf. Manual and Automatic Semantic Annotation with WordNet. In *Proceedings of the NAACL Workshop on WordNet and Other Lexical Resources Applications Customizations*, pages 3–10, 2001. 10, 40, 103, 104, 120, 121, 134, 136, 139, 140, 237

Samuel Fernando and Mark Stevenson. Mapping WordNet synsets to Wikipedia articles. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation*, LREC'12, pages 590–596, Istanbul, Turkey, 2012. European Language Resources Association (ELRA). 95

Radu Florian and David Yarowsky. Modeling Consensus: Classifier Combination for Word Sense Disambiguation. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - Volume 10*, EMNLP '02, pages 25–32, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. 38, 49, 186, 256, 288, 294

Radu Florian, Silviu Cucerzan, Charles Schafer, and David Yarowsky. Combining Classifiers for Word Sense Disambiguation. *Natural Language Engineering*, 8(4):327–341, December 2002. ISSN 1351-3249. doi: 10.1017/S1351324902002978. URL http://dx.doi.org/10.1017/S1351324902002978. 38, 40, 49, 50, 186, 256, 263, 288, 294

W. Nelson Francis and Henry Kučera. *Frequency Analysis of English Usage: Lexicon and Grammar*. Houghton Mifflin, Boston, MA, 1982. 104

Eibe Frank and Ian H. Witten. Generating Accurate Rule Sets Without Global Optimization. In Jude W. Shavlik, editor, *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 144–151, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc. ISBN 1-55860-556-8. URL `http://dl.acm.org/citation.cfm?id=645527.657305`. 252

Eibe Frank, Mark A. Hall, Geoffrey Holmes, Richard Kirkby, Bernhard Pfahringer, Ian H. Witten, and Leonhard Trigg. WEKA - A Machine Learning Workbench for Data Mining. In Oded Maimon and Lior Rokach, editors, *The Data Mining and Knowledge Discovery Handbook*, pages 1305–1314. Springer, 2005. 250

Yoav Freund and Robert E. Schapire. Experiments with a New Boosting Algorithm. In Lorenza Saitta, editor, *Proceedings of the Thirteenth International Conference on Machine Learning*, ICML 1996, pages 148–156. Morgan Kaufmann, 1996. ISBN 1-55860-419-7. URL `http://www.biostat.wisc.edu/~kbroman/teaching/statgen/2004/refs/freund.pdf`. 50, 258

Atsushi Fujii, Takenobu Tokunaga, Kentaro Inui, and Hozumi Tanaka. Selective Sampling for Example-based Word Sense Disambiguation. *Computational Linguistics, Volume 24, Number 4, December 1998*, 24(4):573–597, 1998. URL `http://aclweb.org/anthology/J98-4002`. 295

William Gale, Kenneth Ward Church, and David Yarowsky. Estimating Upper and Lower Bounds on the Performance of Word-Sense Disambiguation Programs. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 249–256, Newark, Delaware, USA, June 1992a. Association for Computational Linguistics. doi: 10.3115/981967.981999. URL `http://www.aclweb.org/anthology/P92-1032`. 27, 28, 288

William A. Gale, Kenneth W. Church, and David Yarowsky. One sense per discourse. In *Proceedings of the workshop on Speech and Natural Language*,

HLT '91, pages 233–237, Stroudsburg, PA, USA, 1992b. Association for Computational Linguistics. ISBN 1-55860-272-0. 114

Anna Gastel, Sabrina Schulze, Yannick Versley, and Erhard Hinrichs. Annotation of Explicit and Implicit Discourse Relations in the TüBa-D/Z Treebank. In Hanna Hedeland, Thomas Schmidt, and Kai Wörner, editors, *Multilingual Resources and Multilingual Applications – Proceedings of the Conference of the German Society for Computational Linguistics and Language Technology. Working Papers in Multilingualism*, number 96 in B, pages 99–104, Hamburg, 2011. 136

Dirk Geeraerts. *Theories of Lexical Semantics*. Oxford Linguistics. Oxford University Press, Oxford, 2010. ISBN 0198700318. 5, 7

Cliff Goddard and Anna Wierzbicka. *Words and Meanings: Lexical Semantics Across Domains, Languages, and Cultures*. Oxford University Press, 2014. 5

Hugo Gonçalo Oliveira. *Onto.PT: Towards the Automatic Construction of a Lexical Ontology for Portuguese*. PhD thesis, University of Coimbra, 2013. 97

Hugo Gonçalo Oliveira and Paulo Gomes. On the Automatic Enrichment of a Portuguese Wordnet with Dictionary Definitions. In *Advances in Artificial Intelligence, Local Proceedings of the 16th Portuguese Conference on Artificial Intelligence*, EPIA 2013, pages 486–497, Angra do Heroísmo, Azores, Portugal, 2013. APPIA. 97

Julio Gonzalo and Felisa Verdejo. Automatic Acquisition of Lexical Information and Examples. In Eneko Agirre and Philip Edmonds, editors, *Word Sense Disambiguation: Algorithms and Applications*, volume 33 of *Text, Speech and Language Technology*, chapter 9, pages 253–274. Springer Netherlands, Dordrecht, The Netherlands, 2006. ISBN 978-1-4020-4808-1. doi: 10.1007/978-1-4020-4809-8_9. URL `http://dx.doi.org/10.1007/978-1-4020-4809-8_9`. 294

Ward H. Goodenough. Componential Analysis and the Study of Meaning. *Language*, 32:195–216, 1956. 6

Algirdas Julien Greimas. *Sémantique structurale: recherche de méthode*. Larousse, Paris, 1966. ISBN 2-03-070314-1. 6

Léon Gulikers, Richard Piepenbrock, and Gilbert Rattink. *German Linguistic Guide*. 1995. 69

Jiang Guo, Wanxiang Che, Haifeng Wang, and Ting Liu. Learning Sense-specific Word Embeddings By Exploiting Bilingual Resources. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 497–507, Dublin, Ireland, August 2014. Dublin City University and Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/C14-1048`. 295

Iryna Gurevych. Using the Structure of a Conceptual Network in Computing Semantic Relatedness. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing*, IJCNLP'2005, pages 767–778, Jeju Island, Republic of Korea, 2005. Springer Berlin Heidelberg. 82

Isabelle Guyon and André Elisseeff. An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research*, 3:1157–1182, March 2003. ISSN 1532-4435. URL `http://www.jmlr.org/papers/v3/guyon03a.html`. 258, 269

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, 11(1):10–18, 2009. 16, 148, 216, 250

Mark A. Hall. *Correlation-based Feature Subset Selection for Machine Learning*. PhD thesis, University of Waikato, Hamilton, New Zealand, 1999. 258, 269, 273

Birgit Hamp and Helmut Feldweg. GermaNet – a Lexical-Semantic Net for German. In *Proceedings of ACL workshop Automatic Information Extraction*

and Building of Lexical Semantic Resources for NLP Applications, Madrid, 1997. 3, 7, 55, 57, 58

Patrick Hanks. Do Word Meanings Exist? *Computers and the Humanities*, 34 (1-2):205–215, 2000. ISSN 0010-4817. doi: 10.1023/A:1002471322828. URL http://dx.doi.org/10.1023/A%3A1002471322828. 8

Sanda M. Harabagiu, George A. Miller, and Dan I. Moldovan. WordNet 2 – a morphologically and semantically enhanced resource. In *Proceedings of SIGLEX 1999*, 1999. 104

Zellig S. Harris. *Methods in Structural Linguistics*. University of Chicago Press, Chicago, 1955. 7

Zellig S. Harris. Distributional structure. *Word*, 10(23):146–162, 1956. 7

Trevor Hastie and Robert Tibshirani. Classification by pairwise coupling. *The Annals of Statistics*, 26(2):451–471, 04 1998. doi: 10.1214/aos/1028144844. URL http://dx.doi.org/10.1214/aos/1028144844. 49, 256

Verena Henrich and Erhard Hinrichs. GernEdiT – The GermaNet Editing Tool. In Nicoletta Calzolari, Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the 7th International Conference on Language Resources and Evaluation*, LREC '10, pages 2228–2235, Valletta, Malta, May 2010a. European Language Resources Association (ELRA). ISBN 2-9517408-6-7. 57, 73, 301, 303

Verena Henrich and Erhard Hinrichs. Standardizing Wordnets in the ISO Standard LMF: Wordnet-LMF for GermaNet. In *Proceedings of the 23rd International Conference on Computational Linguistics*, Coling 2010, pages 456–464, Beijing, China, August 2010b. Coling 2010 Organizing Committee. URL http://www.aclweb.org/anthology/C10-1052. 72

Verena Henrich and Erhard Hinrichs. Determining Immediate Constituents of Compounds in GermaNet. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 420–426,

Hissar, Bulgaria, September 2011. RANLP 2011 Organising Committee. URL `http://aclweb.org/anthology/R11-1058`. 57, 66, 182

Verena Henrich and Erhard Hinrichs. A Comparative Evaluation of Word Sense Disambiguation Algorithms for German. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation*, LREC'12, pages 576–583, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA). ISBN 978-2-9517408-7-7. 51, 53, 54, 171, 185, 187, 192, 193, 194, 206, 214

Verena Henrich and Erhard Hinrichs. Extending the TüBa-D/Z Treebank with GermaNet Sense Annotation. In Iryna Gurevych, Chris Biemann, and Torsten Zesch, editors, *Language Processing and Knowledge in the Web*, volume 8105 of *Lecture Notes in Computer Science*, pages 89–96. Springer Berlin Heidelberg, 2013. ISBN 978-3-642-40721-5. 101, 108

Verena Henrich and Erhard Hinrichs. Consistency of Manual Sense Annotation and Integration into the TüBa-D/Z Treebank. In Verena Henrich, Erhard Hinrichs, Daniël de Kok, Petya Osenova, and Adam Przepiórkowski, editors, *Proceedings of the 13th International Workshop on Treebanks and Linguistic Theories*, TLT13, pages 62–74, Tübingen, Germany, December 2014. 101, 108

Verena Henrich, Timo Reuter, and Hrafn Loftsson. CombiTagger: A System for Developing Combined Taggers. In H. Chad Lane and Hans W. Guesgen, editors, *Proceedings of Florida Artificial Intelligence Research Society Conference*, FLAIRS'09, pages 254–259. AAAI Press, 2009. ISBN 978-1-57735-419-2. 186

Verena Henrich, Erhard Hinrichs, and Tatiana Vodolazova. Semi-Automatic Extension of GermaNet with Sense Definitions from Wiktionary. In *Proceedings of the 5th Language & Technology Conference: Human Language*

Technologies as a Challenge for Computer Science and Linguistics, LTC '11, pages 126–130, Poznań, Poland, 2011. 79, 95, 97

Verena Henrich, Erhard Hinrichs, and Klaus Suttner. Automatically Linking GermaNet to Wikipedia for Harvesting Corpus Examples for GermaNet Senses. *Journal for Language Technology and Computational Linguistics (JLCL)*, 27(1):1–19, 2012a. 95, 111, 120

Verena Henrich, Erhard Hinrichs, and Tatiana Vodolazova. WebCAGe – A Web-Harvested Corpus Annotated with GermaNet Senses. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 387–396, Avignon, France, 2012b. 101, 110

Verena Henrich, Erhard Hinrichs, and Tatiana Vodolazova. An Automatic Method for Creating a Sense-Annotated Corpus Harvested from the Web. *International Journal of Computational Linguistics and Applications (IJCLA)*, 3(2):35–50, 2012c. 111, 120

Verena Henrich, Erhard Hinrichs, and Reinhild Barkey. Aligning Word Senses in GermaNet and the DWDS Dictionary of the German Language. In Heili Orav, Christiane Fellbaum, and Piek Vossen, editors, *Proceedings of the Seventh Global Wordnet Conference*, GWC '14, pages 63–70, Tartu, Estonia, January 2014a. URL `http://www.aclweb.org/anthology/W14-0109`. 95

Verena Henrich, Erhard Hinrichs, and Tatiana Vodolazova. Aligning germanet senses with wiktionary sense definitions. In Zygmunt Vetulani and Joseph Mariani, editors, *Human Language Technology Challenges for Computer Science and Linguistics*, Lecture Notes in Computer Science, pages 329–342. Springer International Publishing, 2014b. ISBN 978-3-319-08957-7. doi: 10.1007/978-3-319-08958-4_27. URL `http://dx.doi.org/10.1007/978-3-319-08958-4_27`. 80

Simon Heinrich Adolf Herling. Über die Topik der deutschen Sprache. In *Abhandlungen des frankfurterischen Gelehrtenvereins für deutsche Sprache*, pages 296–362, 394, Frankfurt am Main, 1821. Drittes Stück. 125

Erhard Hinrichs, Verena Henrich, and Reinhild Barkey. Using Part-Whole Relations for Automatic Deduction of Compound-Internal Relations in GermaNet. *Language Resources and Evaluation*, 47(3):839–858, 2013. ISSN 1574-020X. doi: 10.1007/s10579-012-9207-y. URL `http://dx.doi.org/10.1007/s10579-012-9207-y`. 61, 300

Graeme Hirst and David St-Onge. Lexical Chains as Representations of Context for the Detection and Correction of Malapropisms. In Christiane Fellbaum, editor, *WordNet: An Electronic Lexical Database*, chapter 13, pages 305–332. MIT Press, Cambridge, MA, 1998. 37, 175, 176

Robert C. Holte. Very Simple Classification Rules Perform Well on Most Commonly Used Datasets. *Machine Learning*, 11(1):63–90, April 1993. ISSN 0885-6125. doi: 10.1023/A:1022631118932. 252, 264

Véronique Hoste, Walter Daelemans, Iris Hendrickx, and Antal van den Bosch. Dutch Word Sense Disambiguation: Optimizing the Localness of Context. In *Proceedings of the ACL-02 Workshop on Word Sense Disambiguation: Recent Successes and Future Directions - Volume 8*, WSD '02, pages 61–66, Stroudsburg, PA, USA, 2002a. Association for Computational Linguistics. doi: 10.3115/1118675.1118684. URL `http://dx.doi.org/10.3115/1118675.1118684`. 26, 39, 147, 260

Véronique Hoste, Walter Daelemans, Iris Hendrickx, and Antal van den Bosch. Evaluating the results of a memory-based word-expert approach to unrestricted word sense disambiguation. In *Proceedings of the ACL-02 Workshop on Word Sense Disambiguation: Recent Successes and Future Directions*, pages 95–081. Association for Computational Linguistics, July 2002b. doi: 10.3115/1118675.1118689. URL `http://www.aclweb.org/anthology/W02-1014`. 45, 294

Véronique Hoste, Iris Hendrickx, Walter Daelemans, and Antal Van Den Bosch. Parameter Optimization for Machine-learning of Word Sense Disambiguation. *Natural Language Engineering*, 8(4):311–325, December

2002c. ISSN 1351-3249. doi: 10.1017/S1351324902003005. URL http://dx.doi.org/10.1017/S1351324902003005. 41, 50, 294

Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. A practical guide to support vector classification. Technical report, Department of Computer Science, National Taiwan University, 2003. URL http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf. 264

Tilmann N. Höhle. Der Begriff 'Mittelfeld'. Anmerkungen über die Theorie der topologischen Felder. In *Akten des VII. Internationalen Germanisten-Kongresses Göttingen 1985*, volume 3, pages 329–340. Niemeyer, Tübingen, 1986. 125

Nancy Ide and Jean Véronis. Introduction to the Special Issue on Word Sense Disambiguation: The State of the Art. *Compututational Linguistics*, 24 (1):2–40, March 1998. ISSN 0891-2017. URL http://www.aclweb.org/anthology/J98-1001. 11, 22, 28, 29, 32

Nancy Ide and Yorick Wilks. Making Sense About Sense. In Eneko Agirre and Philip Edmonds, editors, *Word Sense Disambiguation: Algorithms and Applications*, volume 33 of *Text, Speech and Language Technology*, chapter 3, pages 47–73. Springer Netherlands, Dordrecht, The Netherlands, 2006. ISBN 978-1-4020-4808-1. doi: 10.1007/978-1-4020-4809-8_3. URL http://dx.doi.org/10.1007/978-1-4020-4809-8_3. 9, 10, 20

Nancy Ide, Collin Baker, Christiane Fellbaum, and Rebecca Passonneau. The Manually Annotated Sub-Corpus: A Community Resource For and By the People. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, Uppsala, Sweden, 2010. 105

H. Tolga Ilhan, Sepandar D. Kamvar, Dan Klein, Christopher D. Manning, and Kristina Toutanova. Combining Heterogeneous Classifiers for Word-sense Disambiguation. In *The Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems*, SENSEVAL '01, pages 87–90, Stroudsburg, PA, USA, 2001. Association for Computational Lin-

guistics. URL http://dl.acm.org/citation.cfm?id=2387364.2387385. 50

Stefan Jaeger, Huanfeng Ma, and David Doermann. Combining Classifiers with Informational Confidence. In Simone Marinai and Hiromichi Fujisawa, editors, *Machine Learning in Document Analysis and Recognition*, volume 90 of *Studies in Computational Intelligence*, pages 163–191. Springer Berlin Heidelberg, 2008. ISBN 978-3-540-76279-9. doi: 10.1007/978-3-540-76280-5_7. URL http://dx.doi.org/10.1007/978-3-540-76280-5_7. 264

Jay J. Jiang and David W. Conrath. Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. In *Proceedings of the International Conference on Research in Computational Linguistics*, ROCLING X, pages 19–33, Taiwan, 1997. URL http://www.cse.iitb.ac.in/~cs626-449/Papers/WordSimilarity/4.pdf. 37, 39, 178, 179

George H. John and Pat Langley. Estimating Continuous Distributions in Bayesian Classifiers. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, UAI'95, pages 338–345, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. ISBN 1-55860-385-9. 254

Mahesh Joshi, Serguei V. S. Pakhomov, Ted Pedersen, and Christopher G. Chute. A Comparative Study of Supervised Learning as Applied to Acronym Expansion in Clinical Reports. In *AMIA 2006, American Medical Informatics Association Annual Symposium*, Washington, DC, USA, November 2006. 42, 44, 47, 49, 250

Jerrold J. Katz and Jerry A. Fodor. The Structure of a Semantic Theory. *Language*, 39:170–210, 1963. 6

Daisuke Kawahara and Martha Palmer. Single Classifier Approach for Verb Sense Disambiguation based on Generalized Features. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4210–4213,

Reykjavik, Iceland, may 2014. European Language Resources Association (ELRA). ISBN 978-2-9517408-8-4. 39, 295

S. Sathiya Keerthi, Shirish K. Shevade, Chiranjib Bhattacharyya, and Karuturi R. Krishna Murthy. Improvements to Platt's SMO Algorithm for SVM Classifier Design. *Neural Comput.*, 13(3):637–649, March 2001. ISSN 0899-7667. doi: 10.1162/089976601300014493. URL `http://dx.doi.org/10.1162/089976601300014493`. 256

Adam Kilgarriff. "I Don't Believe in Word Senses". *Computers and the Humanities*, 31(2):91–113, 1997a. ISSN 0010-4817. doi: 10.1023/A:1000583911091. URL `http://dx.doi.org/10.1023/A:1000583911091`. 7, 8, 9

Adam Kilgarriff. What is word sense disambiguation good for? *Computing Research Repository (CoRR)*, cmp-lg/9712008, 1997b. URL `http://arxiv.org/abs/cmp-lg/9712008`. 11, 276

Adam Kilgarriff. Sample the Lexicon. Technical report, University of Brighton, Brighton, UK, March 1997c. 102

Adam Kilgarriff. Gold standard datasets for evaluating word sense disambiguation programs. *Computer Speech & Language*, 12(4):453–472, 1998a. 22, 103, 126, 134, 136

Adam Kilgarriff. SENSEVAL: An Exercise in Evaluating Word Sense Disambiguation Programs. In Archibald Michiels André Moulin Siegfried Theissen Thierry Fontenelle, Philippe Hiligsmann, editor, *Proceedings of the 8th EURALEX International Congress*, pages 167–174, Liège, Belgium, aug 1998b. Euralex. ISBN 2-87233-091-7. 103

Adam Kilgarriff. English Lexical Sample Task Description. In *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 17–20, Toulouse, France, July 2001. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/S01-1004`. 28, 34, 148

Adam Kilgarriff. Word Senses. In Eneko Agirre and Philip Edmonds, editors, *Word Sense Disambiguation: Algorithms and Applications*, volume 33 of *Text, Speech and Language Technology*, chapter 1, pages 29–46. Springer Netherlands, Dordrecht, The Netherlands, 2006. ISBN 978-1-4020-4808-1. doi: 10.1007/978-1-4020-4809-8_2. URL `http://dx.doi.org/10.1007/978-1-4020-4809-8_2`. 7

Adam Kilgarriff and Martha Palmer. Introduction to the Special Issue on SENSEVAL. *Computers and the Humanities*, 34(1-2):1–13, 2000. doi: 10.1023/A:1002619001915. URL `http://dx.doi.org/10.1023/A:1002619001915`. 11, 28, 29, 50

Adam Kilgarriff and Joseph Rosenzweig. Framework and Results for English SENSEVAL. *Computers and the Humanities*, 34(1-2):15–48, 2000. 22, 26, 27, 33, 34, 35, 103, 136, 147, 185, 186, 214, 265, 268, 292

Josef Kittler, Mohamad Hatef, Robert P. W. Duin, and Jiri Matas. On Combining Classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, March 1998. ISSN 0162-8828. doi: 10.1109/34.667881. URL `http://dx.doi.org/10.1109/34.667881`. 257

Dan Klein and Christopher D. Manning. Conditional Structure versus Conditional Estimation in NLP Models. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 9–16. Association for Computational Linguistics, July 2002. doi: 10.3115/1118693.1118695. URL `http://www.aclweb.org/anthology/W02-1002`. 47, 255

Dan Klein and Christopher D. Manning. Accurate Unlexicalized Parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 423–430, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. 223

Dan Klein, Kristina Toutanova, H. Tolga Ilhan, Sepandar D. Kamvar, and Christopher D. Manning. Combining Heterogeneous Classifiers for Word Sense Disambiguation. In *Proceedings of the ACL-02 Workshop on Word Sense Disambiguation: Recent Successes and Future Directions*, pages 74–80.

Association for Computational Linguistics, July 2002. doi: 10.3115/1118675.1118686. URL http://www.aclweb.org/anthology/W02-0811. 38, 49, 50, 186, 256, 263, 288, 294

Stefan Knerr, Léon Personnaz, and Gérard Dreyfus. Single-layer learning revisited: a stepwise procedure for building and training a neural network. In Françoise Fogelman Soulié and Jeanny Hérault, editors, *Neurocomputing*, volume 68 of *NATO ASI Series*, pages 41–50. Springer Berlin Heidelberg, 1990. ISBN 978-3-642-76155-3. doi: 10.1007/978-3-642-76153-9_5. URL http://dx.doi.org/10.1007/978-3-642-76153-9_5. 256

Ron Kohavi. The Power of Decision Tables. In Nada Lavrac and Stefan Wrobel, editors, *Machine Learning: ECML-95*, volume 912 of *Lecture Notes in Computer Science*, pages 174–189. Springer Berlin Heidelberg, 1995. ISBN 978-3-540-59286-0. doi: 10.1007/3-540-59286-5_57. URL http://dx.doi.org/10.1007/3-540-59286-5_57. 43, 253

Ron Kohavi and Foster Provost. Glossary of Terms. *Machine Learning*, 30 (2-3):271–274, February 1998. ISSN 0885-6125. 217

Mateusz Kopeć, Rafał Młodzki, and Adam Przepiórkowski. Word Sense Disambiguation in the National Corpus of Polish. *Prace Filologiczne*, LXIII:155–165, 2012. URL http://www.ceeol.com/aspx/issuedetails.aspx?issueid=5849eadf-a657-40af-99be-5aeea82393f1&articleId=dc412489-60fc-4e05-bcd5-49896effc366. 40, 41, 42, 43, 44, 45, 47, 50, 227, 231, 233, 250, 257, 274

Sandra Kübler and Desislava Zhekova. Semi-Supervised Learning for Word Sense Disambiguation: Quality vs. Quantity. In *Proceedings of the International Conference RANLP-2009*, pages 197–202, Borovets, Bulgaria, September 2009. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/R09-1037. 40, 45, 269, 294

Ludmila I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. John Wiley and Sons, Inc., Hoboken, New Jersey, 2004. ISBN 978-0-471-21078-8. 257

Claudia Kunze and Lothar Lemnitzer. GermaNet - representation, visualization, application. In *Proceedings of the Third International Conference on Language Resources and Evaluation*, LREC'02, Las Palmas, Canary Islands - Spain, May 2002. European Language Resources Association (ELRA). URL `http://www.lrec-conf.org/proceedings/lrec2002/pdf/73.pdf`. 72

Oi Yee Kwong. Aligning WordNet with Additional Lexical Resources. In *Proceedings of the COLING-ACL'98 Workshop on 'Usage of WordNet in Natural Language Processing Systems'*, pages 73–79, Montreal, QC, Canada, 1998. 95

Oi Yee Kwong. *New Perspectives on Computational and Cognitive Strategies for Word Sense Disambiguation.* Springer Briefs in Electrical and Computer Engineering. Springer Publishing Company, Incorporated, 2012. ISBN 9781461413202. doi: 10.1007/978-1-4614-1320-2. URL `http://dx.doi.org/10.1007/978-1-4614-1320-2`. 5, 7, 8, 19, 23

Abolfazl Lamjiri, Osama El Demerdash, and Leila Kosseim. Simple features for statistical Word Sense Disambiguation. In Rada Mihalcea and Phil Edmonds, editors, *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 133–136, Barcelona, Spain, July 2004. Association for Computational Linguistics. 47, 48

Shari Landes, Claudia Leacock, and Randee I. Tengi. Building semantic concordances. In Christiane Fellbaum, editor, *WordNet: An Electronic Lexical Database*, chapter 8, pages 199–216. MIT Press, Cambridge, MA, 1998. 104, 121

Stefan Langer. Zur Morphologie und Semantik von Nominalkomposita. In Bernhard Schröder, Winfried Lenders, Wolfgang Hess, and Thomas Portele, editors, *Computer, Linguistik und Phonetik zwischen Sprache und Sprechen: Tagungsband der 4. Konferenz zur Verarbeitung natürlicher Sprache*, KONVENS'98, pages 83–96, Frankfurt a.M., Berlin, Bern, New York, Paris, Wien, 1998. Peter Lang. ISBN 3-631-33844-9. 66

Mirella Lapata and Frank Keller. An Information Retrieval Approach to Sense Ranking. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 348–355, Rochester, New York, 2007. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/N/N07/N07-1044`. 53, 196

Cuong Anh Le and Akira Shimazu. High WSD Accuracy Using Naive Bayesian Classifier with Rich Features. In *Proceedings of the 18th Pacific Asia Conference on Language, Information and Computation*, PACLIC'04, pages 105–114, Tokyo, Japan, December 2004. Logico-Linguistic Society of Japan. URL `http://aclweb.org/anthology/Y04-1011`. 41, 42, 47

Saskia le Cessie and Johannes C. van Houwelingen. Ridge Estimators in Logistic Regression. *Applied Statistics*, 41(1):191–201, 1992. 255

Claudia Leacock and Martin Chodorow. Combining Local Context and WordNet Similarity for Word Sense Identification. In Christiane Fellbaum, editor, *WordNet: An Electronic Lexical Database*, chapter 11, pages 265–283. MIT Press, Cambridge, MA, 1998. 37, 175

Claudia Leacock, Geoffrey Towell, and Ellen Voorhees. Corpus-Based Statistical Sense Resolution. In *Proceedings of the ARPA Workshop on Human Language Technology*, pages 260–265, March 1993. 105

Claudia Leacock, George A. Miller, and Martin Chodorow. Using Corpus Statistics and WordNet Relations for Sense Identification. *Computational Linguistics*, 24(1):147–165, mar 1998. ISSN 0891-2017. 32, 105, 109, 110, 119, 120

Yoong Keok Lee and Hwee Tou Ng. An Empirical Evaluation of Knowledge Sources and Learning Algorithms for Word Sense Disambiguation. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*, EMNLP '02, pages 41–48, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1118693.1118699. URL `http://dx.doi.org/10.3115/1118693.1118699`.

26, 40, 41, 42, 44, 47, 49, 50, 147, 148, 218, 227, 231, 250, 257, 263, 264, 272, 279, 288

Yoong Keok Lee, Hwee Tou Ng, and Tee Kiah Chia. Supervised Word Sense Disambiguation with Support Vector Machines and Multiple Knowledge Sources. In *Proceedings of SENSEVAL-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 137–140, 2004. 26, 49, 250

Els Lefever and Véronique Hoste. SemEval-2010 Task 3: Cross-Lingual Word Sense Disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 15–20, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/S10-1003`. 28, 29

Els Lefever and Véronique Hoste. SemEval-2013 Task 10: Cross-lingual Word Sense Disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 158–166, Atlanta, Georgia, USA, June 2013. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/S13-2029`. 28, 29

Michael Lesk. Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. In *Proceedings of the 5th annual international conference on Systems documentation*, SIGDOC '86, pages 24–26, New York, NY, USA, 1986. ACM. ISBN 0-89791-224-1. 33, 34, 35, 36, 38, 39, 52, 54, 82, 180

David D. Lewis and William A. Gale. A Sequential Algorithm for Training Text Classifiers. In Bruce W. Croft and C.J. van Rijsbergen, editors, *SIGIR '94*, pages 3–12. Springer London, 1994. ISBN 978-3-540-19889-5. doi: 10.1007/978-1-4471-2099-5_1. URL `http://dx.doi.org/10.1007/978-1-4471-2099-5_1`. 295

Wolfgang Lezius. *Ein Suchwerkzeug für syntaktisch annotierte Textkorpora*. PhD thesis, IMS, University of Stuttgart, December 2002. Arbeitspapiere

des Instituts für Maschinelle Sprachverarbeitung (AIMS), volume 8, number 4. 122

Dekang Lin. An Information-Theoretic Definition of Similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 296–304, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc. ISBN 1-55860-556-8. 38, 39, 179, 180

Leonhard Lipka. Prototype Semantics or Feature Semantics: An Alternative? In Wolfgang Lörscher and Rainer Schulze, editors, *Perspectives on Language in Performance. Studies in Linguistics, Literary Criticism, and Language Teaching and Learning. To Honour Werner Huellen on the Occasion of his Sixtieth Birthday.*, volume 317 of *Tuebinger Beitraege zur Linguistik. 317*, pages 282–298. Narr, Tübingen, 1987. ISBN 3-87808-377-7. URL `http://nbn-resolving.de/urn/resolver.pl?urn=nbn:de:bvb:19-epub-5098-5`. 6

Ken Litkowski. Senseval-3 task: Word Sense Disambiguation of WordNet glosses. In Rada Mihalcea and Phil Edmonds, editors, *Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 13–16, Barcelona, Spain, July 2004. Association for Computational Linguistics. 104

Kenneth C. Litkowski. Towards a Meaning-Full Comparison of Lexical Resources. In *Proceedings of the ACL Special Interest Group on the Lexicon Workshop on Standardizing Lexical Resources*, pages 30–37, College Park, MD, USA, 1999. 95

Huan Liu and Lei Yu. Toward Integrating Feature Selection Algorithms for Classification and Clustering. *IEEE Transactions on Knowledge and Data Engineering*, 17(4):491–502, April 2005. ISSN 1041-4347. doi: 10.1109/TKDE.2005.66. URL `http://dx.doi.org/10.1109/TKDE.2005.66`. 258, 269

John Lyons. *Structural Semantics*. Oxford: Blackwell, 1963. 7

John Lyons. *Introduction to Theoretical Linguistics*, volume 510 of *Language Cam.* Cambridge University Press, 1968. ISBN 9780521095105. 7

Ismail El Maarouf, Jane Bradbury, Vít Baisa, and Patrick Hanks. Disambiguating Verbs by Collocation: Corpus Lexicography meets Natural Language Processing. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1001–1006, Reykjavik, Iceland, may 2014. European Language Resources Association (ELRA). ISBN 978-2-9517408-8-4. 26, 49, 250

Suresh Manandhar and Deniz Yuret, editors. *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, Atlanta, Georgia, USA, 2013. Association for Computational Linguistics. URL `http://aclweb.org/anthology/S13-2000`. 28

Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing.* MIT Press, Cambridge, MA, USA, 1999. ISBN 0-262-13360-1. 47

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval.* Cambridge University Press, New York, NY, USA, 2008. ISBN 0521865719, 9780521865715. 258, 269

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of English: the penn treebank. *Computational Linguistics*, 19(2):313–330, Jun 1993. ISSN 0891-2017. 103

David Martínez, Eneko Agirre, and Lluís Màrquez. Syntactic Features for High Precision Word Sense Disambiguation. In *Proceedings of the 19th International Conference on Computational Linguistics*, COLING 2002, pages 626–632, 2002. URL `http://aclweb.org/anthology/C02-1112`. 40, 43, 50, 227, 231, 257, 288

Michael Matuschek and Iryna Gurevych. Dijkstra-WSA: A Graph-Based Approach to Word Sense Alignment. *Transactions of the Association for Computational Linguistics (TACL)*, 1:151–164, May 2013. 96, 98

Diana McCarthy. Word Sense Disambiguation: An Overview. *Language and Linguistics Compass*, 3(2):537–558, 2009. ISSN 1749-818X. doi: 10.1111/j.1749-818X.2009.00131.x. URL `http://dx.doi.org/10.1111/j.1749-818X.2009.00131.x`. 27, 30, 33, 40, 217, 264, 268, 292

Diana McCarthy, Rob Koeling, Julie Weeds, and John Carroll. Finding Predominant Word Senses in Untagged Text. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 279–286, Barcelona, Spain, July 2004. doi: 10.3115/1218955.1218991. URL `http://www.aclweb.org/anthology/P04-1036`. 53, 196

I. Dan Melamed and Philip Resnik. Tagger Evaluation Given Hierarchical Tag Sets. *Computers and the Humanities*, 34(1-2):79–84, 2000. doi: 10.1023/A:1002402902356. URL `http://dx.doi.org/10.1023/A:1002402902356`. 28

Christian M. Meyer and Iryna Gurevych. What Psycholinguists Know About Chemistry: Aligning Wiktionary and WordNet for Increased Domain Coverage. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, IJCNLP '11, pages 883–892, 2011. 81, 86, 95, 96, 111, 120

Rada Mihalcea. Instance Based Learning with Automatic Feature Selection Applied to Word Sense Disambiguation. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, pages 1–7, Stroudsburg, PA, USA, 2002a. Association for Computational Linguistics. doi: 10.3115/1072228.1072267. URL `http://dx.doi.org/10.3115/1072228.1072267`. 41, 42, 45, 258, 269, 272, 288

Rada Mihalcea. Word Sense Disambiguation with Pattern Learning and Automatic Feature Selection. *Natural Language Engineering*, 8(4):343–358, December 2002b. ISSN 1351-3249. doi: 10.1017/S1351324902002991. URL

http://dx.doi.org/10.1017/S1351324902002991. 40, 41, 42, 45, 227, 231, 258, 269, 272, 288

Rada Mihalcea. Co-training and Self-training for Word Sense Disambiguation. In Hwee Tou Ng and Ellen Riloff, editors, *HLT-NAACL 2004 Workshop: Eighth Conference on Computational Natural Language Learning (CoNLL-2004)*, pages 33–40, Boston, Massachusetts, USA, May 2004. Association for Computational Linguistics. 32, 294

Rada Mihalcea. Knowledge-Based Methods for WSD. In Eneko Agirre and Philip Edmonds, editors, *Word Sense Disambiguation: Algorithms and Applications*, volume 33 of *Text, Speech and Language Technology*, chapter 5, pages 107–132. Springer Netherlands, Dordrecht, The Netherlands, 2006. ISBN 978-1-4020-4808-1. doi: 10.1007/978-1-4020-4809-8_5. URL http://dx.doi.org/10.1007/978-1-4020-4809-8_5. 27, 33, 34, 39, 268, 289, 292

Rada Mihalcea. Using Wikipedia for Automatic Word Sense Disambiguation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, pages 196–203, Rochester, NY, USA, 2007. 105

Rada Mihalcea and Phil Edmonds, editors. *Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL http://aclweb.org/anthology/W04-0811. 11, 26, 28, 29, 147

Rada Mihalcea and Dan I. Moldovan. An automatic method for generating sense tagged corpora. In *Proceedings of the American Association for Artificial Intelligence*, AAAI '99, pages 461–466, Menlo Park, CA, USA, 1999. American Association for Artificial Intelligence. 32, 109, 110, 119, 120

Rada Mihalcea and Dan I. Moldovan. eXtended WordNet: progress report. In *Proceedings of NAACL Workshop on WordNet and Other Lexical Resources*, pages 95–100, 2001. 104

Rada Mihalcea, Timothy Chklovski, and Adam Kilgarriff. The Senseval-3 English lexical sample task. In Rada Mihalcea and Phil Edmonds, editors, *Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 25–28, Barcelona, Spain, July 2004. Association for Computational Linguistics. 10, 28, 49, 103, 104, 126, 148, 256, 263, 288, 294

George A. Miller. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11):39–41, November 1995. ISSN 0001-0782. doi: 10.1145/219717.219748. URL `http://doi.acm.org/10.1145/219717.219748`. 6, 7, 20, 55, 77

George A. Miller. Nouns in WordNet. In Christiane Fellbaum, editor, *WordNet: An Electronic Lexical Database*, chapter 1, pages 23–46. MIT Press, Cambridge, MA, 1998a. 64

George A. Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. A Semantic Concordance. In *Proceedings of the 3 DARPA Workshop on Human Language Technology*, 1993. 104, 121, 136

George A. Miller, Martin Chodorow, Shari Landes, Claudia Leacock, and Robert G. Thomas. Using a Semantic Concordance for Sense Identification. In *Proceedings of the Workshop on Human Language Technology*, HLT '94, pages 240–243, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics. ISBN 1-55860-357-3. doi: 10.3115/1075812.1075866. URL `http://dx.doi.org/10.3115/1075812.1075866`. 27

Katherine J. Miller. Modifiers in WordNet. In Christiane Fellbaum, editor, *WordNet: An Electronic Lexical Database*, chapter 2, pages 47–67. MIT Press, Cambridge, MA, 1998b. 57

Tristan Miller, Chris Biemann, Torsten Zesch, and Iryna Gurevych. Using Distributional Similarity for Lexical Expansion in Knowledge-based Word Sense Disambiguation. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, pages 1781–1796, December 2012. 34, 35, 214, 293

Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997. ISBN 0070428077, 9780070428072. 42, 43, 45, 46, 47, 254

Saif Mohammad and Ted Pedersen. Combining Lexical and Syntactic Features for Supervised Word Sense Disambiguation. In Hwee Tou Ng and Ellen Riloff, editors, *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 25–32, Boston, Massachusetts, USA, May 6 - May 7 2004. Association for Computational Linguistics. URL `http://aclweb.org/anthology/W04-2404`. 26, 44, 147, 250

Raymond J. Mooney. Comparative Experiments on Disambiguating Word Senses: An Illustration of the Role of Bias in Machine Learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-96)*, pages 82–91, Philadelphia, PA, 1996. URL `http://www.cs.utexas.edu/users/ai-lab/?mooney:emnlp96`. 42, 44, 45, 47

Gregory L. Murphy. *The Big Book of Concepts*. MIT Press, Massachusetts Institute of Technology, first mit press paperback edition edition, 2004. ISBN 9780262632997. 6

Lluís Màrquez, Gerard Escudero, David Martínez, and German Rigau. Supervised Corpus-Based Methods for WSD. In Eneko Agirre and Philip Edmonds, editors, *Word Sense Disambiguation: Algorithms and Applications*, volume 33 of *Text, Speech and Language Technology*, chapter 7, pages 167–216. Springer Netherlands, Dordrecht, The Netherlands, 2006. ISBN 978-1-4020-4808-1. doi: 10.1007/978-1-4020-4809-8_7. URL `http://dx.doi.org/10.1007/978-1-4020-4809-8_7`. 39, 42, 43, 44, 45, 47, 49, 50, 146, 148, 217, 253, 257, 263, 289

Frank Henrik Müller. Markup Manual for the Tübingen Partially Parsed Corpus of Written German (TüPP-D/Z). Technical report, Department of Linguistics, University of Tübingen, Germany, 2004. 177

Rafał Młodzki and Adam Przepiórkowski. The WSD Development Environment. In Zygmunt Vetulani, editor, *Human Language Technology. Challenges for Computer Science and Linguistics*, volume 6562 of *Lecture Notes in Computer Science*, pages 224–233. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-20094-6. doi: 10.1007/978-3-642-20095-3_21. URL `http://dx.doi.org/10.1007/978-3-642-20095-3_21`. 250

Preslav Nakov and Torsten Zesch, editors. *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, Dublin, Ireland, August 2014. Association for Computational Linguistics and Dublin City University. URL `http://aclweb.org/anthology/S14-2000`. 28

Karin Naumann. *Manual for the annotation of in-document referential relations*. Seminar für Sprachwissenschaft, Abt. Computerlinguistik, Universität Tübingen, Tübingen, Germany, 2007. 126, 242

Roberto Navigli. Meaningful Clustering of Senses Helps Boost Word Sense Disambiguation Performance. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL'44, pages 105–112, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics. 95

Roberto Navigli. Word Sense Disambiguation: a Survey. *ACM Computing Surveys*, 41(2):1–69, 2009. 5, 11, 19, 20, 22, 23, 24, 27, 29, 30, 31, 32, 33, 39, 43, 44, 45, 46, 47, 146, 169, 251, 254, 268, 288, 289, 292

Roberto Navigli, Kenneth C. Litkowski, and Orin Hargraves. SemEval-2007 task 07: coarse-grained English all-words task. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, SemEval'07, pages 30–35, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics. 10, 28, 35, 104

Roberto Navigli, David Jurgens, and Daniele Vannella. SemEval-2013 Task 12: Multilingual Word Sense Disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*,

pages 222–231, Atlanta, Georgia, USA, June 2013. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/S13-2040. 28, 29

Hwee Tou Ng. Getting Serious about Word Sense Disambiguation. In Marc Light, editor, *Proceedings of the Workshop Tagging Text with Lexical Semantics: Why, What, and How?*, Washington, D.C., USA, 1997. Association for Computational Linguistics. URL http://aclweb.org/anthology/W97-0201. 45

Hwee Tou Ng and Hian Beng Lee. Integrating Multiple Knowledge Sources to Disambiguate Word Sense: An Exemplar-Based Approach. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, ACL'96, pages 40–47, Santa Cruz, California, USA, June 1996. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P96-1006. 39, 45, 105, 233

Elisabeth Niemann and Iryna Gurevych. The people's web meets linguistic knowledge: automatic sense alignment of Wikipedia and WordNet. In *Proceedings of the 9th International Conference on Computational Semantics*, IWCS '11, pages 205–214, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. 86, 95

Ian Niles and Adam Pease. Linking Lexicons and Ontologies: Mapping WordNet to the Suggested Upper Merged Ontology. In *Proceedings of the IEEE International Conference on Information and Knowledge Engineering*, IKE'03, pages 412–416, Las Vegas, Nevada, 2003. 95

Georgios Paliouras, Vangelis Karkaletsis, Ion Androutsopoulos, and Constantine D. Spyropoulos. Learning Rules for Large-Vocabulary Word Sense Disambiguation: A Comparison of Various Classifiers. In Dimitris N. Christodoulakis, editor, *Natural Language Processing — NLP 2000*, volume 1835 of *Lecture Notes in Computer Science*, pages 383–394. Springer Berlin Heidelberg, 2000. ISBN 978-3-540-67605-8. doi: 10.1007/3-540-45154-4_35.

URL http://dx.doi.org/10.1007/3-540-45154-4_35. 43, 44, 45, 46, 47, 250, 254

Martha Palmer. Consistent Criteria for Sense Distinctions. *Computers and the Humanities*, 34(1-2):217–222, 2000. ISSN 0010-4817. 10, 104

Martha Palmer and Nianwen Xue. Linguistic Annotation. In Alexander Clark, Chris Fox, and Shalom Lappin, editors, *Handbook of Computational Linguistics and Natural Language Processing*. Wiley-Blackwell, 2010. 102

Martha Palmer, Hoa Trang Dang, and Joseph Rosenzweig. Sense Tagging the Penn Tree Bank. In *Proceedings of the Second Language Resources and Evaluation Conference*, LREC'00, Athens, Greece, 2000. 105

Martha Palmer, Christiane Fellbaum, Scott Cotton, Lauren Delfs, and Hoa Trang Dang. English Tasks: All-Words and Verb Lexical Sample. In *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 21–24, Toulouse, France, July 2001. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/S01-1005. 28, 102, 103, 121, 126, 136, 160

Martha Palmer, Hwee Tou Ng, and Hoa Trang Dang. Evaluation of WSD Systems. In Eneko Agirre and Philip Edmonds, editors, *Word Sense Disambiguation: Algorithms and Applications*, volume 33 of *Text, Speech and Language Technology*, chapter 4, pages 75–106. Springer Netherlands, Dordrecht, The Netherlands, 2006. ISBN 978-1-4020-4808-1. doi: 10.1007/978-1-4020-4809-8_4. URL http://dx.doi.org/10.1007/978-1-4020-4809-8_4. 10, 22, 23, 24, 25, 106, 140, 141, 147, 148

Martha Palmer, Hoa Trang Dang, and Christiane Fellbaum. Making fine-grained and coarse-grained sense distinctions, both manually and automatically. *Natural Language Engineering*, 13(2):137–163, 2007. 10, 104

Aarón Pancardo-Rodríguez, Manuel Montes-y Gómez, Luis Villaseñor-Pineda, and Paolo Rosso. A Mapping Between Classifiers and Training Conditions

for WSD. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing*, volume 3406 of *Lecture Notes in Computer Science*, pages 246–249. Springer Berlin Heidelberg, 2005. ISBN 978-3-540-24523-0. doi: 10.1007/978-3-540-30586-6_27. URL http://dx.doi.org/10.1007/978-3-540-30586-6_27. 43, 45, 47, 49

Patrick Pantel. Inducing ontological co-occurrence vectors. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 125–132, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics. doi: 10.3115/1219840.1219856. URL http://dx.doi.org/10.3115/1219840.1219856. 7

Rebecca Passonneau, Collin Baker, Christiane Fellbaum, and Nancy Ide. The MASC Word Sense Sentence Corpus. In *Proceedings of the Eighth Language Resources and Evaluation Conference*, Istanbul, Turkey, 2012. 105, 127, 134, 136

Siddharth Patwardhan, Satanjeev Banerjee, and Ted Pedersen. Using Measures of Semantic Relatedness for Word Sense Disambiguation. In *Proceedings of the 4th International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing'03, pages 241–257, Berlin, Heidelberg, 2003. Springer-Verlag. ISBN 3-540-00532-3. 26, 33, 37, 38, 173, 177, 214

Karl Pearson. Mathematical Contributions to the Theory of Evolution. III. Regression, Heredity, and Panmixia. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 187:253–318, 1896. doi: 10.1098/rsta.1896.0007. URL http://rsta.royalsocietypublishing.org/content/187/253.short. 139

Ted Pedersen. A Simple Approach to Building Ensembles of Naive Bayesian Classifiers for Word Sense Disambiguation. In Janyce Wiebe, editor, *Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 63–69, Seattle, Washington, USA, April/May 2000. Association for Computational Linguistics. URL http://aclweb.org/anthology/A00-2009. 47

Ted Pedersen. A Decision Tree of Bigrams is an Accurate Predictor of Word Sense. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics on Language Technologies*, NAACL '01, pages 1–8, Stroudsburg, PA, USA, 2001. Association for Computational Linguistics. doi: 10.3115/1073336.1073347. URL `http://dx.doi.org/10.3115/1073336.1073347`. 42, 43, 44, 47, 250

Ted Pedersen. Evaluating the Effectiveness of Ensembles of Decision Trees in Disambiguating SENSEVAL Lexical Samples. In *Proceedings of the ACL-02 Workshop on Word Sense Disambiguation: Recent Successes and Future Directions - Volume 8*, WSD '02, pages 81–87, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1118675.1118687. URL `http://dx.doi.org/10.3115/1118675.1118687`. 44, 250

Ted Pedersen. Unsupervised Corpus-Based Methods for WSD. In Eneko Agirre and Philip Edmonds, editors, *Word Sense Disambiguation: Algorithms and Applications*, volume 33 of *Text, Speech and Language Technology*, chapter 6, pages 133–166. Springer Netherlands, Dordrecht, The Netherlands, 2006. ISBN 978-1-4020-4808-1. doi: 10.1007/978-1-4020-4809-8_6. URL `http://dx.doi.org/10.1007/978-1-4020-4809-8_6`. 31

Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. WordNet::Similarity: Measuring the Relatedness of Concepts. In *Demonstration Papers at HLT-NAACL 2004*, HLT-NAACL–Demonstrations '04, pages 38–41, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics. 172, 179

Ted Pedersen, Satanjeev Banerjee, and Siddharth Patwardhan. Maximizing Semantic Relatedness to Perform Word Sense Disambiguation. Research Report UMSI 2005/25, University of Minnesota Supercomputing Institute, 2005. 13, 33, 37, 38, 172, 173, 174, 176, 184, 191, 192, 194, 200, 203, 209, 211, 212, 213, 266

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. Learning Accurate, Compact, and Interpretable Tree Annotation. In *Proceedings of the*

21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, pages 433–440, Sydney, Australia, July 2006. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P/P06/P06-1055`. 222

Oliver Plaehn. *Annotate Bedienungsanleitung*. Universität des Saarlandes, Sonderforschungsbereich 378, Projekt C3, Saarbrücken, Germany, 1998. 122

John C. Platt. Fast Training of Support Vector Machines Using Sequential Minimal Optimization. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 185–208. MIT Press, Cambridge, MA, USA, January 1998. ISBN 0-262-19416-3. URL `http://research.microsoft.com/apps/pubs/default.aspx?id=68391`. 256

Robi Polikar. Ensemble Based Systems in Decision Making. *IEEE Circuits and Systems Magazine*, 6(3):21–45, 2006. 186

Simone Paolo Ponzetto and Roberto Navigli. Large-scale taxonomy mapping for restructuring and integrating Wikipedia. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, IJCAI '09, pages 2083–2088, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc. 95

Simone Paolo Ponzetto and Roberto Navigli. Knowledge-rich Word Sense Disambiguation rivaling supervised systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1522–1531, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. 34, 35, 36, 95

Martin Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980. 84, 182, 184, 226

Bernard Pottier. Vers une sémantique moderne. *Travaux de Linguistique et de Littérature*, 2:107–137, 1964. 6

Bernard Pottier. La définition sémantique dans les dictionnaires. *Travaux de Linguistique et de Littérature*, 3:44–39, 1965. 6

Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. SemEval-2007 Task-17: English Lexical Sample, SRL and All Words. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 87–92, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/S/S07/S07-1016`. 104, 265

Judita Preiss and David Yarowsky, editors. *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, Toulouse, France, July 2001. Association for Computational Linguistics. URL `http://aclweb.org/anthology/S01-1001`. 11, 28, 29

Judita Preiss, Jon Dehdari, Josh King, and Dennis Mehay. Refining the most frequent sense baseline. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, SEW-2009, pages 10–18, Boulder, Colorado, June 2009. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/W09-2403`. 27, 288

Paul Procter, editor. *Longman Dictionary of Contemporary English*. Longman Group, Harlow, UK, 1978. 101, 105

J. Ross Quinlan. Learning Efficient Classification Procedures and Their Application to Chess End Games. In Ryszard S. Michalski, Jaime G. Carbonell, and Tom M. Mitchell, editors, *Machine Learning*, Symbolic Computation, pages 463–482. Springer Berlin Heidelberg, 1983. ISBN 978-3-662-12407-9. doi: 10.1007/978-3-662-12405-5_15. 221

J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993. ISBN 1-55860-238-0. 43, 44, 253

Diana Raileanu, Paul Buitelaar, Spela Vintar, and Jörg Bay. Evaluation Corpora for Sense Disambiguation in the Medical Domain. In *Proceedings of the*

3rd International Conference on Language Resources and Evaluation, LREC '02, 2002. 107, 108, 126, 135, 138, 141, 161

Ganesh Ramakrishnan, B. Prithviraj, and Pushpak Bhattacharya. A gloss-centered algorithm for disambiguation. In Rada Mihalcea and Phil Edmonds, editors, *Proceedings of Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 217–221, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL `http://aclweb.org/anthology/W04-0853`. 35

Rat für deutsche Rechtschreibung, editor. *Deutsche Rechtschreibung – Regeln und Wörterverzeichnis: Amtliche Regelung*. Gunter Narr Verlag Tübingen, 2006. ISBN 978-3-8233-6270-8. 59, 60, 306

Payam Refaeilzadeh, Lei Tang, and Huan Liu. Cross-Validation. In Ling Liu and M. Tamer Özsu, editors, *Encyclopedia of Database Systems*, pages 532–538. Springer US, 2009. ISBN 978-0-387-39940-9. doi: 10.1007/978-0-387-39940-9_565. URL `http://dx.doi.org/10.1007/978-0-387-39940-9_565`. 25, 26, 147

Ines Rehbein, Josef Ruppenhofer, and Jonas Sunde. MaJo – a toolkit for supervised word sense disambiguation and active learning. In *Proceedings of the Eigth International Workshop on Treebanks and Linguistic Theories : Catholic University of the Sacred Heart, Milan 4-5 December 2009*, Milan, 2009. 51

Philip Resnik. Using Information Content to Evaluate Semantic Similarity in a Taxonomy. In *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 1*, pages 448–453, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. ISBN 1-55860-363-8, 978-1-558-60363-9. 37, 38, 173, 177, 178

Philip Resnik. WSD in NLP Applications. In Eneko Agirre and Philip Edmonds, editors, *Word Sense Disambiguation: Algorithms and Applications*, volume 33 of *Text, Speech and Language Technology*, chapter 11, pages 299–337. Springer Netherlands, Dordrecht, The Netherlands, 2006.

ISBN 978-1-4020-4808-1. doi: 10.1007/978-1-4020-4809-8_11. URL http://dx.doi.org/10.1007/978-1-4020-4809-8_11. 11, 276

Philip Resnik and David Yarowsky. A Perspective on Word Sense Disambiguation Methods and Their Evaluation. In *Proceedings of the SIGLEX Workshop "Tagging Text with Lexical Semantics: Why, What, and How?"*, pages 79–86, Washington, DC, April 1997. 28, 151

Philip Resnik and David Yarowsky. Distinguishing systems and distinguishing senses: new evaluation methods for Word Sense Disambiguation. *Natural Language Engineering*, 5(02):113–133, 6 1999. ISSN 1469-8110. URL http://journals.cambridge.org/article_S1351324999002211. 28, 151

Ronald L. Rivest. Learning Decision Lists. *Machine Learning*, 2(3):229–246, November 1987. ISSN 0885-6125. doi: 10.1023/A:1022607331053. URL http://dx.doi.org/10.1023/A:1022607331053. 42, 44

Eleanor Rosch. Cognitive representations of semantic categories. *Journal of Experimental Psychology: General*, 104(3):192–233, 1975. doi: 10.1037/0096-3445.104.3.192. URL http://dx.doi.org/10.1037/0096-3445.104.3.192. 6

Eleanor Rosch. Principles of Categorization. In Eleanor Rosch and Barbara B. Lloyd, editors, *Cognition and Categorization*, pages 27–48. John Wiley & Sons Inc, 1978. ISBN 0470263776. 6

Herbert Rubenstein and John B. Goodenough. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633, October 1965. ISSN 0001-0782. doi: 10.1145/365628.365657. URL http://doi.acm.org/10.1145/365628.365657. 7

Maria Ruiz-Casado, Enrique Alfonseca, and Pablo Castells. Automatic Assignment of Wikipedia Encyclopedic Entries to WordNet Synsets. In *Proceedings of the Atlantic Web Intelligence Conference, AWIC-2005. Volume 3528 of Lecture Notes in Computer Science*, pages 380–386. Springer Verlag, 2005. 95, 96

Magnus Sahlgren. The Distributional Hypothesis. *Rivista di Linguistica (Italian Journal of Linguistics)*, 20(1):33–53, 2008. ISSN 1120-2726. 7

Jahn-Takeshi Saito, Joachim Wagner, Graham Katz, Philip Reuter, Michael Burke, and Sabine Reinhard. Evaluation of GermanNet: Problems Using GermaNet for Automatic Word Sense Disambiguation. In *Proceedings of the Workshop on "Wordnet Structures and Standardization, and how these affect Wordnet Applications and Evaluation" at LREC 2002*, pages 14–19, Las Palmas, Grand Canaria, 2002. 12, 51, 106, 107, 134

Celina Santamaría, Julio Gonzalo, and Felisa Verdejo. Automatic association of web directories with word senses. *Computational Linguistics*, 29(3):485–502, September 2003. ISSN 0891-2017. 110

Anne Schiller, Simone Teufel, Christine Stöckert, and Christine Thielen. Guidelines für das Tagging deutscher Textcorpora mit STTS. Technical report, Universities of Stuttgart and Tübingen, 1999. 123, 227

Helmut Schmid. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *Proceedings of International Conference on New Methods in Language Processing*, Manchester, UK, 1994. 84, 116, 160, 165, 221

Helmut Schmid and Florian Laws. Estimation of Conditional Probabilities With Decision Trees and an Application to Fine-Grained POS Tagging. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 777–784, Manchester, UK, August 2008. Coling 2008 Organizing Committee. URL `http://www.aclweb.org/anthology/C08-1098`. 222

Hinrich Schütze. Dimensions of meaning. In *Proceedings of the 1992 ACM/IEEE Conference on Supercomputing*, Supercomputing '92, pages 787–796, Los Alamitos, CA, USA, 1992. IEEE Computer Society Press. ISBN 0-8186-2630-5. URL `http://dl.acm.org/citation.cfm?id=147877.148132`. 7

Hinrich Schütze. Automatic Word Sense Discrimination. *Computational Linguistics*, 24(1):97–123, March 1998. ISSN 0891-2017. URL `http://dl.acm.org/citation.cfm?id=972719.972724`. 7, 31, 294

Hinrich Schütze and Jan Pedersen. Information Retrieval based on Word Senses. In *Proceedings of the 4th Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, Las Vegas, USA, 1995. 7

Rion Snow, Sushant Prakash, Daniel Jurafsky, and Andrew Y. Ng. Learning to Merge Word Senses. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1005–1014, Prague, Czech Republic, June 2007. Association for Computational Linguistics. 10

Benjamin Snyder and Martha Palmer. The English all-words task. In Rada Mihalcea and Phil Edmonds, editors, *Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 41–43, Barcelona, Spain, July 2004. Association for Computational Linguistics. 103

Diana Steffen, Bogdan Sacaleanu, and Paul Buitelaar. Domain Specific Sense Disambiguation with Unsupervised Methods. *LDV-Forum – Anwendungen des deutschen Wortnetzes in Theorie und Praxis. Beiträge des GermaNet-Workshops Tübingen, Oktober 2003*, 19(1/2):93–101, 2004. ISSN 0175-1336. 51, 52, 53, 250, 295

Armando Suárez and Manuel Palomar. A Maximum Entropy-based Word Sense Disambiguation System. In *Proceedings of the 19th International Conference on Computational Linguistics*, COLING '02, pages 1–7, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1072228.1072343. URL `http://aclweb.org/anthology/C02-1115`. 48

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-654-7. 95

Armando Suárez Cueto. *Resolución de la ambigüedad semántica de las palabras mediante modelos de probabilidad de máxima entropía.* PhD thesis, Universidad de Alicante, Departamento de Lenguajes y Sistemas Informáticos, Alicante, Spain, June 2004. URL `http://hdl.handle.net/10045/4070`. 48

Heike Telljohann, Erhard W. Hinrichs, and Sandra Kübler. The Tüba-D/Z Treebank: Annotating German with a Context-Free Backbone. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, LREC'04, pages 2229–2232. European Language Resources Association, 2004. URL `http://www.lrec-conf.org/proceedings/lrec2004/pdf/135.pdf`. 4, 120, 156

Heike Telljohann, Erhard W. Hinrichs, Sandra Kübler, Heike Zinsmeister, and Kathrin Beck. Stylebook for the Tübingen Treebank of Written German (TüBa-D/Z). Technical report, Department of General and Computational Linguistics, University of Tübingen, Germany, 2012. 4, 120, 126, 156, 232, 237, 239

Antonio Toral, Óscar Ferrández, Eneko Agirre, and Rafael Muñoz. A Study on Linking Wikipedia Categories to WordNet Synsets using Text Similarity. In *Proceedings of the International Conference RANLP-2009*, RANLP '09, pages 449–454, Borovets, Bulgaria, September 2009. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/R09-1080`. 95

Sulema Torres and Alexander Gelbukh. Comparing Similarity Measures for Original WSD Lesk Algorithm. *Advances in Computer Science and Applications. Special issue of Research in Computing Science*, (43):155–166, 2009. ISSN 1870-4069. 26, 34, 38

Stephen Tratz, Antonio Sanfilippo, Michelle Gregory, Alan Chappell, Christian Posse, and Paul Whitney. PNNL: A Supervised Maximum Entropy Approach to Word Sense Disambiguation. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 264–267, Prague, Czech Republic, June 2007. Association for Computational

Linguistics. URL http://www.aclweb.org/anthology/S/S07/S07-1057. 48

Peter D. Turney. Word Sense Disambiguation by Web Mining for Word Co-occurrence Probabilities. In Rada Mihalcea and Phil Edmonds, editors, *Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 239–242, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL http://aclweb.org/anthology/W04-0858. 50, 250

Tylman Ule. Markup Manual for the Tübingen Partially Parsed Corpus of Written German (TüPP-D/Z). Technical report, Department of Linguistics, University of Tübingen, Germany, 2004. 177

Hans van Halteren, Jakub Zavrel, and Walter Daelemans. Improving Accuracy in Word Class Tagging through the Combination of Machine Learning Systems. *Computational Linguistics*, 27:199–229, 2001. ISSN 0891-2017. 186

Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995. ISBN 978-0-387-98780-4. 48, 256

Florentina Vasilescu, Philippe Langlais, and Guy Lapalme. Evaluating Variants of the Lesk Approach for Disambiguating Words. In *Proceedings of the 4th International Conference On Language Resources And Evaluation*, LREC'04, pages 633–636, Lisbon, Portugal, 2004. 34, 184

Jorn Veenstra, Antal van den Bosch, Sabine Buchholz, Walter Daelemans, and Jakub Zavrel. Memory-Based Word Sense Disambiguation. *Computers and the Humanities*, 34(1-2):171–177, 2000. ISSN 0010-4817. doi: 10.1023/A:1002459020102. URL http://dx.doi.org/10.1023/A%3A1002459020102. 39, 45

Jean Véronis. A study of polysemy judgments and inter-annotator agreement. In *Proceedings of SENSEVAL-1*, Herstmonceux Castle, England, 1998. 135, 138, 141, 152

Andrew J. Viterbi. Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm. *IEEE Transactions on Information Theory*, IT-13(2):260–269, April 1967. 221

Piek Vossen, editor. *EuroWordNet: A Multilingual Database with Lexical Semantic Networks*. Kluwer Academic Publishers, Norwell, MA, USA, 1998. ISBN 0-7923-5295-5. 65

Piek Vossen. EuroWordNet General Document. Eurowordnet project le2-4003; le4-8328 report, University of Amsterdam, 2002. 65

Sholom M. Weiss and Casimir A. Kulikowski. *Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1991. ISBN 1-55860-065-5. 26, 217

Dominic Widdows, Stanley Peters, Scott Cederberg, Chiu-Ki Chan, Diana Steffen, and Paul Buitelaar. Unsupervised monolingual and bilingual word-sense disambiguation of medical documents using UMLS. In *Proceedings of the ACL 2003 workshop on Natural language processing in biomedicine - Volume 13*, BioMed '03, pages 9–16, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. 51, 52, 53, 107

Janyce Wiebe, Julie Maples, Lei Duan, and Rebecca Bruce. Experience in WordNet sense tagging in the Wall Street Journal. In *Proceedings of the ANLP-97 Workshop, Tagging Text with Lexical Semantics: Why, What, and How?*, pages 8–11, Washington, D.C., April 1997. Association for Computational Linguistics SIGLEX. 105

Peratham Wiriyathammabhum, Boonserm Kijsirikul, Hiroya Takamura, and Manabu Okumura. Applying Deep Belief Networks to Word Sense Disambiguation. *Computing Research Repository (CoRR)*, 2012. URL `http://arxiv.org/abs/1207.0396`. 26, 42, 45, 47, 48, 49, 255, 264

Ian H. Witten, Eibe Frank, and Mark A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, Amsterdam, 3 edition,

2011. ISBN 978-0-12-374856-0. 25, 44, 48, 147, 152, 217, 250, 251, 253, 254, 255, 256, 258, 264, 269, 272

Dekai Wu, Weifeng Su, and Marine Carpuat. A Kernel PCA Method for Superior Word Sense Disambiguation. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 637–644, Barcelona, Spain, July 2004. doi: 10.3115/1218955.1219036. URL `http://www.aclweb.org/anthology/P04-1081`. 47, 48, 49

Zhibiao Wu and Martha Palmer. Verb Semantics and Lexical Selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, ACL '94, pages 133–138, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics. 37, 175

David Yarowsky. DECISION LISTS FOR LEXICAL AMBIGUITY RESOLUTION: Application to Accent Restoration in Spanish and French. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 88–95, Las Cruces, New Mexico, USA, June 1994. Association for Computational Linguistics. doi: 10.3115/981732.981745. URL `http://www.aclweb.org/anthology/P94-1013`. 42

David Yarowsky. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, ACL '95, pages 189–196, Stroudsburg, PA, USA, 1995. Association for Computational Linguistics. 32, 42, 52, 108, 119, 294

David Yarowsky. Hierarchical Decision Lists for Word Sense Disambiguation. *Computers and the Humanities*, 34(1-2):179–186, 2000. ISSN 0010-4817. doi: 10.1023/A:1002674829964. URL `http://dx.doi.org/10.1023/A%3A1002674829964`. 42

David Yarowsky and Radu Florian. Evaluating Sense Disambiguation Across Diverse Parameter Spaces. *Natural Language Engineering*, 8(4):293–310, December 2002. ISSN 1351-3249. doi: 10.1017/S135132490200298X. URL

http://dx.doi.org/10.1017/S135132490200298X. 41, 42, 47, 217, 250, 262

Hsiang-Fu Yu, Fang-Lan Huang, and Chih-Jen Lin. Dual coordinate descent methods for logistic regression and maximum entropy models. *Machine Learning*, 85(1-2):41–75, 2011. ISSN 0885-6125. doi: 10.1007/s10994-010-5221-8. URL http://dx.doi.org/10.1007/s10994-010-5221-8. 255

Jakub Zavrel, Sven Degroeve, Anne Kool, Walter Daelemans, and Kristiina Jokinen. Diverse Classifiers for NLP Disambiguation Tasks – Comparisons, Optimization, Combination, and Evolution. In Kristiina Jokinen, Dirk Heylen, and Anton Nijholt, editors, *Proceedings of the CELE-Twente Workshops on Natural Language Technology "Learning to Behave"*, TWLT 18, pages 201–221, Enschede, 2000. PARLEVINK, PARLEVINK. URL http://www.cnts.ua.ac.be/papers/2000/zdk00.pdf. 42, 44, 45, 47, 48, 49, 263, 288

Torsten Zesch. *Study of Semantic Relatedness of Words Using Collaboratively Constructed Semantic Resources*. PhD thesis, TU Darmstadt, Darmstadt, Germany, Februar 2010a. URL http://tuprints.ulb.tu-darmstadt.de/2041/. 173

Torsten Zesch. What's the Difference? – Comparing Expert-Built and Collaboratively-Built Lexical Semantic Resources. FLaReNet Forum 2010, Barcelona, Spain, 2010b. 81

Torsten Zesch, Christof Müller, and Iryna Gurevych. Extracting Lexical Semantic Knowledge from Wikipedia and Wiktionary. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, LREC'08, Marrakech, Morocco, May 2008. 81, 113

Jingbo Zhu and Eduard Hovy. Active Learning for Word Sense Disambiguation with Methods for Addressing the Class Imbalance Problem. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*,

pages 783–790, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL http://aclweb.org/anthology/D07-1082. 295

Xiaojin Zhu and Andrew B. Goldberg. *Introduction to Semi-Supervised Learning*. Synthesis lectures on artificial intelligence and machine learning. Morgan & Claypool, 2009. ISBN 9781598295474. 32