Induction, Semantic Validation and Evaluation of a Derivational Morphology Lexicon for German

Dissertation

zur Erlangung der Doktorwürde
der Neuphilologischen Fakultät
der Ruprecht-Karls-Universität Heidelberg

vorgelegt von
Britta Dorothee Zeller

Supervisor and first reviewer:	Prof. Dr. Sebastian Padó Institut für Maschinelle Sprachverarbeitung Universität Stuttgart
Secondary reviewer:	Prof. Dr. Anette Frank Institut für Computerlinguistik Ruprecht-Karls-Universität Heidelberg
Date of submission:	5 May 2015
Date of defense:	11 December 2015

Abstract

This thesis is about computational morphology for German derivation. Derivation is a word formation process that creates new words from existing ones, where the base and the derived word share the same stem. Mostly, derivation is conducted by means of relatively regular affixation rules, as in *to bake*$_V$ – *bakery*$_N$. In German, derivation is highly productive, thus leading to a high language variability which can be employed to express similar facts in different ways, as derivationally related words are often also semantically related (or transparent). However, linguistic variance is a challenge for computational applications, particularly in semantic processing: It makes it more difficult to automatically grasp the meaning of texts and to match similar information onto each other. Thus, computational systems require linguistic knowledge.

We develop methods to induce and represent derivational knowledge, and to apply it in language processing. The main outcome of our study is DERIVBASE, a German derivational lexicon. It groups derivationally related words (words that are derived from the same stem) into derivational families. To achieve high quality and high coverage, we induce DERIVBASE by combining rule-based and data-driven methods: We implement linguistic derivation rules to define derivational processes, and feed lemmas extracted from a German corpus into the rules to derive new lemmas. All words that are connected – directly or indirectly – by such rules are considered a derivational family.

As mentioned above, a derivational relationship often implies semantic relationship, but this is not always the case. Semantic drifts can cause semantically unrelated (opaque) derivational relations, such as *to depart*$_V$ – *department*$_N$. Capturing the difference between transparent and opaque relations is important from a linguistic as well as a practical point of view. Thus, we conduct a semantic refinement of DERIVBASE, i.e., we determine which lemma pairs are derivationally *and* semantically related, and which are not. We establish a second, semantically validated version of our lexicon, where families are sub-clustered according to semantic coherence, using supervised machine learning methods: We learn a binary classifier based on features that arise from structural information about the derivation rules, and from distributional information about the semantic relatedness of lemmas. Accordingly, the derivational families are subdivided into semantically coherent clusters.

To demonstrate the utility of the two lexicon versions, we evaluate them on three extrinsic – and in the broadest sense, semantic – tasks. The underlying assumption for applying DERIVBASE to semantic tasks is that derivational relatedness is a reasonable approximation to semantic relatedness, since derivation is often semantically transparent.

Our three experiments are the following: 1., we incorporate DERIVBASE into distributional semantic models to overcome sparsity problems and to improve the prediction quality of the underlying model. We test this method, which we call derivational smooth-

ing, for semantic similarity prediction, and for synonym choice. 2., we employ DErivBase to model a psycholinguistic experiment that examines priming effects of transparent and opaque derivations to draw conclusions about the mental lexical representation in German. Derivational information is again incorporated into a distributional model, but this time, it introduces a kind of morphological generalisation. 3., in order to solve the task of Recognising Textual Entailment, we integrate DErivBase into a matching-based entailment system by means of a query expansion. Assuming that derivational relationships between two texts suggest them to be entailing rather than non-entailing, this expansion increases the chance of a lexical overlap, which should improve the system's entailment predictions.

The incorporation of DErivBase indeed improves the performance of the underlying systems in each task, however, it is differently suitable in different settings. In experiment 1., the semantically validated lexicon yields improvements over the purely morphological lexicon, and the more coarse-grained similarity prediction profits more from DErivBase than the synonym choice. In experiment 2., purely morphological information clearly outperforms the other lexicon version, as the latter cannot model opaque derivations. On the entailment task in experiment 3., DErivBase has only minor impact, because textual entailment is hard to solve by addressing only one linguistic phenomenon.

In sum, our findings show that the induction of a high-quality, high-coverage derivational lexicon is beneficial for very different applications in computational linguistics. It might be worthwhile to further investigate the semantic aspects of derivation to better understand its impact on language and thus, on language processing.

Acknowledgements

First and foremost, I sincerely thank my supervisor Sebastian Padó for the best support I could have imagined throughout the work on this thesis: always inspiring, objective, constructive and encouraging – and: always. It was a great pleasure to work with you. Furthermore, I thank Jan Šnajder so much for bringing the topic of this thesis from Zagreb to Heidelberg, for the continuous and great cooperation and support until the very end, and for our friendship. Also, my special thanks to Tae-Gil Noh, my colleague and friend who taught me many best practices in daily working life as a computer engineer, and helped me not to freak out. — Without you all, this thesis would not be the same.

In addition, I am grateful for having had the opportunity to work in the EC project EXCITEMENT (FP7-ICT 287923) and for having received a scholarship by the German Research Foundation in the SFB 732 – not only for the funding, but also for the resulting collaboration with great colleagues who gave me inspiration and useful input. In this context, I thank Jason Utt for so much support in various experiments, and our student helpers, Jan Pawellek, Jonas Placzek and in particular Julia Kreutzer, for doing annotations, implementations and analyses in the EC project.

Meinen Eltern, meiner Tante, Eva und Heike: Danke, dass Ihr an mich geglaubt und mich unterstützt habt.

Lastly, I wholeheartedly thank my beloved Sascha for last minute tikz help, hours of illuminating discussions, and all the mental support over the years. — Without you, I would not be the same.

Contents

List of Figures

List of Tables

Part I

Introduction and Background

1 Introduction

Morphological processing, or the processing of words, is a fundamental step for many tasks in Natural Language Processing (NLP). It precedes other linguistic analyses such as part of speech (POS) tagging or parsing (Trost, 2005). Morphological analysers are one of the first NLP tools developed for any language (Koskenniemi, 1983, Sproat, 1992), and are also applied in applications where little other linguistic analysis is performed, such as linguistic annotation of corpora or terminology acquisition (Daille et al., 2002).

There are three main types of morphological processes: 1., *Inflection* modifies word forms according to the grammatical context (Bickel and Nichols, 2001); 2., *composition* combines multiple words into new lexical items; 3., *derivation* constructs new words from individual existing words, where the basis and the derived word share the same stem (Lieber and Štekauer, 2014). The computational treatment of morphology is often restricted to normalisation, such as *stemming* (covering inflection and derivation heuristically; e.g., (Porter, 1980)), or – which is used more often – *lemmatisation* (covering inflection only). In other words, most work on computational morphology has focused on inflection (Trost, 2005). Derivation, in contrast, has received less attention in the computational linguistics literature.

Nonetheless, **derivation** is a word formation process that exists in many natural languages and gives rise to a large and varied vocabulary, e.g., by means of the derivation of agent nouns from verbs, like *to run*$_V$ – *runner*$_N$. Often, derivation is conducted through affixation; some of these affixation processes can affect the part of speech of the base word (Trost, 2005), as just shown. Also, the words by which a language is enriched through derivation, have often a meaning that is similar to that of the original base word (ibid.). The resulting variety is valuable to express similar circumstances with different utterances, as in Example (1.1), where "laughing" and "laughter" share a semantic core:

(1.1) (a) He could not stop *laughing* about it.

(b) There was no end to his *laughter*.

An important reason for the relative lack of attention to derivational morphology in computational linguistics is that, still, English is the most widely researched language, but morphologically relatively simple. Its morphological processes are less versatile as well as less complex than in many other languages: Composition is usually not marked morphologically (*zoo gate*), and a frequent derivational operation is conversion (or zero derivation), where the input and output terms are identical surface forms (*fish*$_N$ – *to fish*$_V$). Also, there is a clear dominance of suffixational derivation (*to treat*$_V$ – *treatment*$_N$), and a relative absence of derivational stem changes. For these reasons,

simple stemming algorithms provide a cheap and sufficiently accurate approximation to English derivation.

In contrast, German is derivationally very complex: Besides suffixation and conversion, which occur in many languages, German derivation also comprises prefixation, circumfixation and stem-changing processes, and uses most of them extensively. For instance, while English employs particles, adverbs or phrases for certain linguistic expressions (e.g., aspects of place, time and manner (Smolka et al., 2014, p33)), German uses prefixes for such cases, e.g., *schlafen*$_V$ – *ausschlafen*$_V$ (*to sleep*$_V$ – *to sleep*$_V$ *in*$_{RP}$).

If the versatility of German derivation is not appropriately considered, many NLP tasks are arguably harder to solve. The most obvious consequence of **morphological richness** of a language is the increased lexicon size, or word inventory. This is notably a problem when two texts are matched against each other, which is a paradigm that NLP tasks (e.g., Information Retrieval or Recognising Textual Entailment) often instantiate: Many systems build upon the hypothesis that high lexical overlap between two texts indicates that they refer to the same issue. Thus, a high overlap between, e.g., a query and a document is taken as evidence for the document to be relevant to the query (Berger et al., 2000). Obviously, unrecognised overlaps due to linguistic variation – e.g., by means of derivation –, lead to false negatives. Thus, the systems need to take into account linguistic variance. As an example, the query *print fails* and a relevant document containing the word *printer* can only be mapped onto each other if it is known that the words *print* and *printer* are derivationally related. Also for distributional semantic models, languages with a large word inventory are problematic: The more lexical entries a language has, the more face such models the problem of sparse representations for each of these entries. For instance, the English verb *to catch* has two semantically very similar realisations in German (*fangen* and *einfangen*), which leads to a sparser distributional representation for each of them. In sum, it is desirable to capture the versatility of derivationally rich languages, and to use this knowledge to improve computational linguistic processes.

In this thesis, we address this goal: We develop methods for inducing, representing, and utilising derivational information, and apply them to German. The core of our study is **DErivBase, a lexicon of derivational families for German** that we compile. Such a **derivational lexicon** groups words that are morphologically derived from the same stem (i.e., **derivational families**). For instance, the lemmas *friend*$_N$, *friendly*$_A$, *friendless*$_A$, *to befriend*$_V$ and *friendship*$_N$ are members of the same derivational family. Conceptually, such a family-oriented structure of a derivational lexicon is motivated by the fact that it complies with a line of research in (cognitive) linguistics that argues that the mental lexicon is organised according to morphological families (Bybee, 1985, 1988, Langacker, 1987, Nagy et al., 1989). To our knowledge, there are only two designated derivational lexicons that are structured in this fashion, and represent the notion of derivational relatedness by means of families: an unnamed and not publicly available lexicon for French (Gaussier, 1999), and CatVar (Habash and Dorr, 2003) for English.[1] Gaussier's lexicon was employed for Question Answering, and CatVar has proven to be useful in various semantic tasks, e.g., in Textual Entailment, or for the improvement

[1] Additionally, a derivational lexicon for Croatian has been developed based on our work (Šnajder, 2014).

of text fluency in language generation (cf. Section 3.1.3). A particular advantage of a derivational lexicon is that it provides indications of **relatedness across parts of speech** (cf. Example (1.1)), whereas many traditional resources such as ontologies mostly contain relationships within the same part of speech, and recent dynamic approaches such as distributional semantic models rather retrieve collocations or topical similarity across parts of speech.

We build DERIVBASE by means of a **rule-based approach**, using traditional grammar books as our source of linguistic knowledge. First, we define the set of derivations that should be covered by the lexicon. Then, we implement these derivations in a rule-based morphology modelling framework (Šnajder and Dalbelo Bašić, 2010) which is able to deal with all admissible derivational processes of the German language, i.e., all affixation operations, conversion, and stem changes. We combine the structural rule information with **evidence from a corpus**, i.e., we extract nouns, verbs, and adjectives from a large German web corpus, and use them as input for the derivation rules to derive new words. In order prevent the rules from overgenerating, the generated words are filtered, again on the corpus material. However, false positives can still occur whenever the output of a derivation rule matches an existing, but derivationally unrelated word, such as $corn_N$ – $corner_N$. Finally, all lemmas that are transitively connected by derivation rules are considered a derivational family. Lemmatisation of the corpus and a separate inflectional component in the employed framework enable us to concentrate on derivation only, and to largely exclude inflectional processes. With this combination of rule- and corpus-based methods, we induce a resource of high precision as well as high coverage that provides information about the derivational relatedness of German words in the form of derivational families. Additionally, the rule-based induction combined with the transitive connections results in an **internal structure** of the families, namely in the form of derivation rule paths that relate any two lemmas of a family to one another.

As mentioned above, derivational processes often preserve a base word's core meaning and propagate it to the derived word, i.e., many derivationally related words are also semantically related. This fact is often exploited for the application of derivational lexicons: Derivational relatedness is assumed to be a reasonable approximation of semantic relatedness. However, this analogy is not always given. Derivational processes can cause **semantic drifts**, so that the basis and the derived word do not share any common semantics. For example, the prefixation in the word pair $hören_V$ – $aufhören_V$ (*to* $hear_V$ – *to* $stop_V$) leads to a completely different meaning, i.e., it is a semantically **opaque** derivation. From a linguistic point of view, a computational resource in which opaque lemma pairs are tantamount to semantically **transparent** (i.e., related) pairs, such as $hören_V$ – $zuhören_V$ (*to* $hear_V$ – *to* $listen_V$), is undesirable.

For this reason, we go a step beyond a purely derivational lexicon, and develop strategies for the **semantic validation** of derivational families, i.e., methods to achieve semantically coherent sub-families. Note that we do not want to specify the kind of semantic relationship which occurs between derivationally related words, e.g., agentive or diminutive. Instead, we aim at determining whether the words are semantically related *at all* (i.e., the distinction between transparent and opaque derivations). To do so,

we employ supervised machine learning techniques. We define derivational as well as distributional features for pairs of derivationally related lemmas, and learn a model that separates derivationally *and* semantically related words (transparency) from those that are *only* derivationally, but not semantically related (opacity). As a side benefit, this classification improves the purity of DERIVBASE in terms of derivational relatedness, because it sorts out words which have no derivational relationship at all to the derivational family they have been assigned to. Finally, we use the pairwise decisions of the semantic validation classifier to build a **semantically clustered version of DErivBase**. That is, the families are sub-divided into semantically coherent groups of derivationally related lemmas. We expect that the semantic specification of derivational relations that we achieve through this second step makes our lexicon more suitable for applications in lexical-semantic tasks.

Our two-step procedure to acquire a semantically validated derivational lexicon includes three contrasting methods commonly used in computational linguistics: While the induction of the purely morphologically motivated lexicon relies on hand-written **rule-based information** and **corpus evidence**, the semantic validation additionally brings into play **supervised classification methods**. In this way, we combine linguistic insights with statistical modelling of language, leading to a theoretically grounded, but quantitatively viable resource.

We consider the methodological challenge of evaluating our approach intrinsically, creating a benchmark dataset to specifically assess the quality as well as the coverage of DERIVBASE. In order to additionally test the usability of our lexicon in real applications, we evaluate our achievements for both the purely derivational version and its semantic refinement in three fairly different **extrinsic experiments**. For these experiments, we make the general assumption that derivationally related words are more often semantically related than unrelated, and a derivational lexicon thus provides reasonable **approximations to semantic relatedness**, even without semantic validation. We present the following evaluations: 1., we employ DERIVBASE to smooth distributional semantic models. In doing so, we want to overcome sparsity problems, but also improve the prediction quality of the underlying distributional model on two German standard lexical-semantic tasks. 2., again using DERIVBASE and a distributional semantic space, we computationally model the results of a priming experiment from a psycholinguistic study (Smolka et al., 2014). It examines the priming effects of transparent and opaque derivations in order to investigate whether the lexical representation in German is organised based on morphemes. 3., we use DERIVBASE to solve the task of Recognising Textual Entailment (RTE), i.e., an actual NLP task: We expand a matching-based entailment system with data from our lexicon, thus increasing the chance of a lexical overlap in Text/Hypothesis pairs that contain derivationally related lemmas and – as we assume – are likely to be entailing.

1.1 Structure of this Thesis

This thesis is structured as follows: The remainder of this Part (Part I) puts our work into context with respect to linguistics as well as computational linguistics. Part II describes the construction of DErivBase, and the incorporation of semantic aspects by means of semantic validation. Part III builds upon the achievements of Part II, and illustrates the evaluation of our derivational lexicon in three different – in the broadest sense – semantic experiments. Finally, Part IV concludes the thesis.

As to Part I, we establish in Chapter 2 the terminology used in this thesis in the context of morphology, discuss linguistic fundamentals, and give a first impression of what a derivational lexicon is.

We discuss related work from computational linguistics in Chapter 3. There are two main areas of previous studies which are important for our work: Computational morphology (Section 3.1), which is related to the general topic of this thesis of building a German derivational lexicon, and distributional semantics (Section 3.2) which we employ for the semantic refinement of our derivational lexicon and which constitutes an important element in two of our extrinsic evaluations.

In Chapters 4 and 5, which constitute Part II, we explain the two successive steps of the construction of DErivBase. First, we describe the rule-and corpus-based induction of the purely morphological version of DErivBase, which concentrates on gathering derivationally related words into derivational families (Chapter 4). Second, we investigate on how to take into account semantic aspects into the lexicon, i.e., how to decide for derivationally related pairs of lemmas whether they are also semantically related, and how to propagate this information to entire families. We present our realisation of this semantic validation of DErivBase, using machine learning methods, in Chapter 5. Both Chapters include intrinsic evaluations for the respective lexicon version.

The applicability of DErivBase to extrinsic tasks, i.e., the smoothing of distributional semantic models, the modelling of morphological priming, and the expansion of an entailment system in Rte, is presented in Chapters 6 to 8 in Part III. We show the results of each experiment for both the purely morphological and the semantically refined version of our lexicon to make the impact of the two variants maximally transparent.

This thesis concludes with Chapter 9 (Part IV), summarising the contributions we made and giving an outlook on future directions how to build upon the presented work.

In the progress of work on this thesis, many additional settings and experiments have been investigated that yielded not particularly insightful results and/or no improvements. Whenever we still found it useful, we shortly report such additional experiments at the end of the respective Section or Chapter, aiming at informing the reader about less promising directions in our research area.

1.2 Bibliographic Note

Some of the work presented in this thesis has previously appeared in publications. The general induction of DErivBase (Chapter 4) was published in Zeller et al. (2013).

My three main contributions in this work were a pre-study of German derivation, the implementation of the derivation rules, and the analysis of the rules as well as the resulting lexicon (incl. error analysis). The framework which we use for our approach existed previously (Šnajder and Dalbelo Bašić, 2010); functional changes in the framework were carried out by my co-authors. The ground work for the semantic validation of a derivational lexicon (Chapter 5) was disseminated in Zeller et al. (2014). I analysed the derivational and distributional differences between the two classes to be separated by the semantic validation, developed most classifier features, and implemented substantial parts of the experimental setup of the machine learning classification. Finally, I analysed the results. The compilation of the distributional spaces and their application to the datasets was largely done by my co-authors.

Concerning the evaluations and applications of DErivBase, we published an experimental study about derivational smoothing (Section 6.1) in Padó et al. (2013b). I mainly participated in the general discussion of conceptual questions and the development of our method, and conducted results analyses, while my co-authors delivered the fundamental ideas and carried out the implementation. The psycholinguistic experiments described in Chapter 7 have been published in Padó et al. (2015), where I prepared as well as analysed the data, and compared our results to the related experimental study.

The DErivBase lexicon is publicly available (under the CreativeCommons license CC BY-SA 3.0) in various formats and versions at `http://www.ims.uni-stuttgart.de/permalink/56cc6c89-c421-11e4-a5e6-000e0c3db68b.html`.

1.3 Notation

In this thesis, example lemmas and morphemes are denoted in *italics*, often indicated with the respective part of speech. Pairs of lemmas (incl. the corresponding part of speech) which share a derivational relationship are denoted as $lemma1_{pos1}$ – $lemma2_{pos2}$.

2 Linguistic Foundations

In this Chapter, we introduce fundamental linguistic concepts about morphological derivation, clarify the terminology we use and assumptions we make, and make preliminary considerations about the task of computationally building a derivational lexicon. Our terminology and definitions are based on that of Glück (2010), unless otherwise specified.

Since we build a derivational lexicon for German, we will focus on this language. German is an interesting object of study regarding morphology because it is a synthetic language, meaning that the syntactic role of a word is often marked morphologically (Glück, 2010, p666). Consequently, new words are often formed on the morphological level, i.e., by derivation (Greenberg, 1963, p17). Thus, although German is morphologically not as rich as, e.g., Slavic languages like Croatian or Russian (Xanthos et al., 2011, p465), it offers a far broader range of derivational variety than a morphologically relatively simple language like English (Derwing, 1990, Štekauer and Lieber, 2005), and is thus interesting to investigate.

Section 2.1 gives a brief introduction to the area of morphology and word formation in general, and of derivation in particular. Section 2.2 discusses the role and realisation of derivation in German. Section 2.3 addresses computational representation possibilities of derivational relatedness. Finally, Section 2.4 resumes the main findings and our focus in this work.

2.1 Morphology, Word Formation, and Derivation

In this Section, we start by introducing the field of morphology, then its subdomain word formation and, finally, derivation, one specific morphological word formation process.

2.1.1 Morphology

Morphology is concerned with the form and the internal structure of words; it builds upon *morphemes*, which are the smallest meaningful unit of a language (Krovetz (1993), Glück (2010, p441f)). Basically, morphology subsumes two linguistic processes: Lexical word formation, and inflection (Bußmann, 2002, p450). In German, word formation is mainly realised via composition and derivation (cf. Sections 2.1.2, 2.1.3). Inflection is not concerned with building new lexemes, but with grammatically motivated modifications of existing lexemes. That is, inflection differs from lexical word formation in that it is paradigmatic: There are structures (so-called inflectional paradigms) which group all lexemes that frame inflectional word forms in the same manner (Glück, 2010, p202).[1]

[1] Domenig and ten Hacken (1992) define inflectional paradigms as a (maximal) group of word forms with an element in common, and at least one element being different that is associated with grammatical

In contrast, word formation operates on a case by case basis (ten Hacken, 2009), which means that it cannot be generalised or predicted as simply as inflectional processes. We will consider the most important German word formation processes in Section 2.2.

Verb infinitives and participles are situated in the grey area between inflection and word formation: They "form" a word that can be further inflected using the underlying verb stem (e.g., *give* → *give_ø* for infinitives, *confuse* → *confused* for participles). Additionally, participles can be used as adjectives (*He was confused*). In German, the infinitive is marked with three specific infinitive suffixes: *-en*, *-eln*, and *-ern*.[2] Only two irregular verbs are marked with a different suffix, *-n*: *sein* (*to be*) and *tun* (*to do*). Mostly, linguists consider infinitives and participles as inflectional (Bußmann, 2002, p304). We adopt this point of view, however, we consider the adjective variant of participles as (derivational) word formation.

Excursus: Root vs. Stem. In the literature, the differentiation between root and stem as an element of word formation is often unclear.

The term *root* is used differently in diachronic and in synchronic contexts, as well as in synchronic word formation and synchronic inflection contexts (Glück, 2010, p776). Definitions in word formation agree in that the root is morphologically no more decomposable, and usually constitutes a core meaning which is passed on in derivation and other word formation processes. Thus, the definitions of the root and the base morpheme are often used interchangeably (Bußmann, 2002, p267, p758).

Glück reports that the term *stem*, in turn, was originally used for inflectional morphology, but has meanwhile become equally common in word formation. According to him, the main difference to the root is that a stem can be a morphologically complex entity, e.g., the result of a derivation like *readable* (Glück, 2010, p666).

This property made the use of the term "stem" more established in word formation than that of "root": Allowing for complex base entities, it is more practical for word formation procedures. For example, building a word group like *to read*$_V$, *readable*$_A$, *unreadable*$_A$, is more straightforward if the outcome of a derivation can be used as input for another one (*to read* → ***read****able* → *un****readable***). Moreover, deriving new words from complex base lemmas seems linguistically more plausible: A speaker who uses the word *unreadable*$_A$ will rather think of a contrast to *readable*$_A$ than of a relationship to *to read*$_V$.

Since we are interested in derivation, a word formation process, we adopt this definition, and employ the term *stem* as well as its concept for referring to a derivation's basis (for a definition of "base", cf. Section 2.1.3). □

2.1.2 Word Formation

Focusing on word formation, we are concerned with the formation of new lexemes in order to extend the vocabulary of a language. As argued above, inflection is not considered here,

roles.

[2]Some linguists regard *-eln* and *-ern* infinitive suffixes as derivational, e.g., Fleischer and Barz (2007).

since it does not form new lexemes, but only grammatical variants of a lexeme. There is a considerable number of different word formation processes, many of which exist only for a couple of languages, or have rather minor impact because they are not frequently used. Examples for such peripheral word formations in German are clipping (the short form *Kripo* for *Kriminalpolizei*, "criminal investigation department"), and blending (e.g., the conflation of *Kur*, "cure", and *Urlaub*, "vacation", into *Kurlaub*) (Bußmann (2002, p373, p751f), Glück (2010, p770f)). In German, the three synchronically most important word formation processes are composition, conversion, and derivation (Glück, 2010, p770). In composition, two or more free morphemes or words are combined to a new lexeme (Glück, 2010, p348). Examples are $schnelllebig_A$ (*fast moving*), $Wasserflasche_N$ (*water bottle*), or $kurzschneiden_V$ (*to crop close*). Conversion, also called "zero derivation" is considered in the grey area between being a proper word formation process, and a subtype of derivation (Glück, 2010, p5). In conversion, no explicit word formation elements are involved; however, the input and output word of the process differ in their part of speech, e.g., $laufen_V$ – $Laufen_N$ (*to* run_V – run_N). In the following, we consider conversion a subtype of derivation. Finally, derivation is the process of creating new words from existing ones, where the base and the derived word share the same stem. It can be distinguished from inflection by a couple of criteria (Glück, 2010, p5): 1., derivation is used for word formation, not for word *form* formation, 2., derivation can cause a change in meaning, 3., derivation can cause a change of the part of speech, and 4., derivation has restricted applicability. Similarly, derivation is distinguishable from composition by the fact that derivation combines a free morpheme and a functional affix. Such affixes are no free morphemes (*bound* morphemes), and cannot constitute the basis of a word formation process. Note, however, that this distinction is not always straightforward. We will come back to this problem in Section 2.2.

2.1.3 Derivation

As mentioned above, derivation is a morphological word formation process: Via *derivational processes*, we derive a new word from an existing word with the same stem, typically by involving bound morphemes. Derivational processes provide two levels of interest: The *string transformation* level, and the *meaning level*. First, we consider the string transformations, then, we turn to the semantic aspect.

Basic Concepts of Derivational Word Formation. The starting point of a derivational process is called *base lemma*, while its output is called *derived lemma*, or *derivative*. Pairs of base and derived lemma are connected by a *derivational relation*, or in other words, they are derivationally related. Most (but by far not all) derivations are conducted by *affixation* processes, i.e., the attachment of prefixes, suffixes, and their combination (circumfixes) to the base lemma (Glück, 2010, p16). A crucial property of derivational processes is that they are able to change the part of speech of the base lemma, e.g., $kochen_V$ – $Koch_N$ (*to* $cook_V$ – $cook_N$). Note that derivation is in most cases quite regular, which is why a derivational process is also called a *derivation rule*. However, it can also be irregular and "semiregular". Semi-regularity describes cases of recurrent

irregularity due to various reasons (e.g., lexeme-based, orthographical, or phonological reasons; these phenomena have been discussed in the literature, e.g., in Jackendoff (1975), Plank (1981), Riehemann (1994), Gor (2010)). One such semi-regularity in German is the occasional conflation or deletion of stem-final vowels for phonological reasons, as for the *-schaft* suffixation in $Erbe_N$ – $Erbschaft_N$ ($inheritance_N$ – $inheritance_N$). In contrast, a completely irregular conversion is $fahren_V$ – $Fahrt_N$ ($to\ drive_V$ – $trip_N$), the regular derivative of which would be $*Fahr_N$. As indicated in Section 2.1.1, a derivation's derivative can be transitively re-used as base, i.e., a base lemma can be the product of another derivational rule, and thus be morphologically complex. An example is $ableiten_V$ – $Ableitung_N$ ($to\ derive_V$ – $derivation_N$), where the base verb is a prefixation derivative of $leiten_V$ ($to\ lead_V$). Base lemmas are not necessarily words, but can be a so-called combining form (German: "Konfix"): a bound morpheme with a lexical meaning, like *bio-*, or *geo-* (Glück, 2010, p138, p350). An exemplary derivation for a combining form as base is $biotisch_A$ ($biotic_A$).

The Semantics of Derivations. Derivation forms new words and consequently, new expressions of meaning. This paragraph discusses to what extent derivational processes imply a meaning change, and whether derivation can be considered meaning-preserving.

Generally, lexical semantics often assumes a dichotomy for word relatedness: *Semantic transparency* and *semantic opacity*. The meaning of a semantically transparent utterance can be comprehended form the meaning of its parts (Partee, 1995), while for semantically opaque utterances, a *semantic drift*[3] takes place, so that the meaning cannot be derived from the parts.

Although the notion of semantic transparency and opacity arises from psycholinguistics (e.g., Libben et al. (2003)), it is operationalised in various fields of computational semantics.[4] On the word level, for instance, compounds are classified into transparent (*carwash*) and opaque cases (*brainwash*) (Schulte im Walde et al., 2013). Similarly, Villada Moirón and Tiedemann (2006) distinguish on the phrase-level between semantically transparent (literal: *to kick the ball*) and opaque (idiomatic: *to kick the bucket*) utterances.

Also on the morphological, and in particular on the derivational level, semantic transparency ($friend_N$ – $friendly_A$) and opacity ($sign_N$ – $signal_N$) are widely discussed. Some authors claim that every derivational process embodies a semantic drift. For example, Bybee (1985) argues, amongst others, with the influence of word class changes, and with the fact that derivations change the syntactic, semantic, and stylistic role of a word, as in the agentive formation in Example (2.1) – taken from Booij (2000, p360):

(2.1) (a) He who *reads* this book
(b) The *reader* of this book

[3] By "drift", we do not exclusively refer to diachronic drifts, but also include synchronic phenomena such as polysemy or semantic changes at the lexical level (Vanhove, 2008).

[4] Obviously, such a dichotomy is oversimplifying, as the semantics of language cannot be expressed by a binary distinction, but rather on a graded scale (Bybee, 1985, Gonnerman and Anderson, 2001). However, similar categorisations are frequently employed to apply computational models to natural language, e.g., for part of speech tagging, or the construction of word taxonomies.

Bybee claims that the degree of semantic relatedness between base lemma and derivative is often difficult to establish. Also, Booij (2000, p364) notes the semantic opacity of derivation, i.e., that the meaning of a derivative is often "not purely a compositional function of the meaning of its morphological constituents" and therefore not easy to capture. This makes it hard to clearly semantically define derivational processes.

Nevertheless, there are attempts to do so, e.g., by Pala and Hlaváčková (2007), Pala (2008) who examine the meaning of individual affixation processes in Czech. Partial studies also exist for German prefixation (Springorum et al., 2012, 2013), however, it seems hard to assign semantic functions to all German derivation processes. For example, it is easy to declare that the *-chen* suffix has a diminutive meaning, as in $Stadt_N$ – $Städtchen_N$ ($town_N$ – *small* $town_N$). But the semantics of the *be-* prefix, as in $treffen_V$ – $betreffen_V$ (*to* hit_V – *to* $concern_V$), is more subtle as well as incoherent (and even idiosyncratic) across applications. Thus, its meaning is not easily predictable.[5]

Due to these difficulties in determining precise semantics for derivational processes, computational linguists often follow a more relaxed definition of semantic relatedness and do not regard each derivation as meaning-changing. For instance, Jacquemin (2010) employs derivational information to distinguish synonymous and non-synonymous words. Representing a relatively theory-neutral view on derivation, he agrees that there is a "sense challenge inherent to the derivation phenomenon" (Jacquemin, 2010, p4), but judges changes that are still semantically largely transparent, as in Example (2.1) above, meaning-preserving. According to him, many derivationally related words are semantically related, and only striking discrepancies count as unrelated; e.g., the semantically opaque lemma pair *to* $treat_V$ – *to* $retreat_V$, where the base lemma expresses the act of dealing with or thinking about something in a particular way, while the derivative describes moving or going away from a place or situation.[6] We adopt Jacquemin's definition, and consider only clear semantic opacity as meaning-changing (e.g., the prefixation $tragen_V$ – $auftragen_V$ (*to* $carry_V$ – *to* $apply_V$)). Moreover, we make the following fundamental assumption:

Derivational Coherence Assumption (DCA): Derivational processes produce derived words that are semantically related to their base words. Phrased in binary terms, we assume that the majority of derivational processes yield transparent rather than opaque word pairs – despite the well-known irregularity of derivational processes.

That is, we expect derivational relationships to be sufficiently regular on the semantic level to legitimate their application in semantic processing. Here, we define *semantic relatedness* as a generalisation of the term "semantic similarity": In psycholinguistics, semantic similarity is described as the relationship between two words that are assigned a high degree of meaning relatedness by native speakers (Miller and Charles, 1991). Semantic relatedness, in turn, covers a broader range of (thematic) meaning relatedness, such as synonymy (*couch* – *sofa*), antonymy (*cut* – *uncut*), or strong associativity (*pen* –

[5] In the literature, the meaning of *be-* is considered particularly unpredictable, e.g., by Booij (2000, p365) for the corresponding Dutch prefix. Also, Pala (2008) admits that prefixation is still challenging.

[6] Definitions taken from `http://www.merriam-webster.com`.

paper) (Budanitsky and Hirst, 2006). As mentioned in Section 2.1.2, one particularity of derivational relatedness is that it crosses parts of speech boundaries. However, semantic similarity as defined in the tradition of psycholinguistics is restricted to words of the same word class. Although it is assumed that "morphological relations may play an important role in judgements of the semantic similarity of words from different syntactic categories" (Miller and Charles, 1991, p9), we are not aware of studies addressing this question. Thus, we expand our definition of semantic relatedness accordingly: Except for the aspect of syntactic substitutability, we assume semantic relatedness to exist also for derivationally related words of different word classes. In fact, German conversion is a good example for semantically related derivations across parts of speech; Example (2.2) illustrates two sentences that are, from a semantic point of view, largely synonymous (nonetheless, they exhibit grammatical changes as well as stylistic and pragmatic differences; cf. Welke (2011), Storrer (2006), Mentrup (1988), Schippan (1967)).

(2.2) (a) *Er hat schnell gelernt zu sprechen.*
He has quickly learned to speak.
He quickly learned to speak.

(b) *Das Sprechen hat er schnell gelernt.*
The speaking has he quickly learned.
He quickly learned to speak.

A prominent approach to measure semantic relatedness is to use distributional semantic models (cf. Section 3.2). In this thesis, we will investigate whether a distributional approach is also applicable to measure the semantic relatedness of derivationally related words (Chapters 5, 6, 7). Furthermore, we will concretise in Chapter 3.1.3 how the abstract definition of the DCA is generally operationalised in practice.

2.2 Derivation in German

This Section focusses on the realisation of derivational processes in German. In order to systematically examine derivation, it is necessary to establish a classification of the processes involved. We present proposals from the literature to classify derivation, and describe in detail the particularities of the five German derivation operations. Note that we concentrate on definitions that are relevant for our goal of building a computational derivational lexicon.

2.2.1 Classification of Derivation

According to Glück (2010, p5), there are at least three ways to categorise derivational processes: 1., by the part of speech of the derivative, 2., by the part of speech of the base word, and 3., by the semantic function of the derivative. While the third aspect is difficult to grasp (cf. Section 2.1.3), a classification according to 1. and 2. is completely and consistently feasible. We shortly present both classifications for German in the following.

Additionally, we discuss a classification according to the derivational operation conducted (e.g., suffixation). We find this an important view on the topic, especially because we plan to implement German derivation rules for lexicon induction.

Classification by the Part of Speech of the Derivative. One way to categorise German derivation is the outcome of each derivation. Derivation is a very productive word formation process, and is applicable to many parts of speech. The most important is the derivation of content words, i.e., nouns, verbs, adjectives, and adverbs. The corresponding classifications of the derivational processes are called *nominalisation*, *verbalisation*, *adjectivisation*, and *adverbialisation*. Thereof, nominalisation is the most frequent one in German, as well as across languages.[7] In contrast, derivations of adverbs that are not usable as adverbial adjectives, e.g., $gleich_A$ – $gleichfalls_{ADV}$ ($equal_A$ – $equally_{ADV}$), are rather unproductive in German (Donalies, 2005, Fleischer and Barz, 2007). By "productive", we mean processes that produce words which (have) become part of a language's word inventory; we do not consider spontaneous ad-hoc derivations, like $Verhängnis_N$ – $verhängniswärts_{ADV}$ ($fate_N$ – $fatewards_{ADV}$) (Donalies, 2005, p126), as productive.

Classification by the Part of Speech of the Base Word. Similar to derivatives, most base words of German derivation are content words. They serve for so-called *denominal*, *deverbal*, or *deadjectival* derivations (Glück, 2010, p5). Additionally, it is possible to derive new words from named entities (incl. proper nouns) as base lemma, e.g., $Kafka_{NE}$ – $kafkaesk_A$ ($Kafka_{NE}$ – $Kafkaesque_A$), or $Heidelberg_{NE}$ – $Heidelberger_N$ ($Heidelberg_{NE}$ – *inhabitant of* $Heidelberg_N$) (Donalies, 2005, p113).

Apart from these frequently used parts of speech as base lemmas, new words can be derived from adverbs, prepositions, or pronouns, as shown in Examples (2.3 a–c). However, these derivations are again unproductive (Donalies, 2005, p126).

(2.3) (a) $hinab_{ADV}$ – $hinabwärts_{ADV}$ ($down_{ADV}$ – $downwards_{ADV}$)
(b) auf_{PREP} – $aufwärts_{ADV}$ (up_{PREP} – $upwards_{ADV}$)
(c) $mein_{PPOSS}$ – $meinige_{PPOSS}$ (my_{PPOSS} – $mine_{PPOSS}$)

Classification by the Derivational Operation. From a procedural point of view, derivational processes can be categorised by the conducted string transformation, i.e., by the *derivational operation* (Lieber and Štekauer, 2014) that creates a derivational link between two lemmas. In German, there are five such operations:

- Suffixation: placing an affix at the end of the stem
  $Leser_N$ – $Leserschaft_N$ ($reader_N$ – $readership_N$)
- Prefixation: placing an affix at the beginning of the stem
  $heben_V$ – $beheben_V$ (*to* $raise_V$ – *to* $remedy_V$)

[7] Hall (2000, p538) argues that from a sample of 50 languages, 70% use nominalisation.

- Circumfixation: combination of suffixation and prefixation
  $reden_V$ – $Gerede_N$ (*to* $talk_V$ – $gossip_N$)
- Conversion, or zero derivation: part of speech change without affixation or stem alternation
  $laufen_V$ – $Lauf_N$ (*to* run_V – run_N)
- Stem change, or implicit derivation: changes to the stem's vowel with ablaut or umlaut
  $liegen_V$ – $legen_V$ (*to* lie_V – *to* lay_V)

Note that the application of these operations is not mutually exclusive. Instead, they can be combined in complex derivation rules, for instance, by combining an umlaut stem change ($u \rightarrow \ddot{u}$) and a suffixation (*-lich*): $Stunde_N$ – $stündlich_A$ ($hour_N$ – $hourly_A$). In this way, German derivation gets more variable, but also more complicated. In the following, we briefly sketch the main characteristics of each of these five operations.

The first three types taken together are called *explicit derivations*, because they denote the derivation explicitly with an affix (Donalies, 2005, p98). Their affixes can either originate from the Germanic language, e.g., $ölen_V$ – $Ölung_N$ (*to* oil_V – $oiling_N$), or from Latin or Greek, e.g., $generieren_V$ – $Generator_N$ (*to* $generate_V$ – $generator_N$). The base lemma for foreign affixes is often a foreign word as well, but this is not necessarily the case, e.g., $Haft_N$ – $inhaftieren_V$ ($arrest_N$ – *to* $arrest_V$).

Derivational affixes may not be confused with the attachment of the three German verb-denoting infinitive suffixes mentioned in Section 2.1.1. These verb suffixes are not derivational, but inflectional, which is why they are also appended or removed in conversion, as in the example above.

Suffixation, circumfixation, conversion and stem changes can cause a part of speech change between the base and the derived lemma; they are therefore called *transpositions* (Glück, 2010, p722).[8]

German suffixation operations for noun-involving derivations can be restricted to a specific gender; for the base lemma, for the derivative, or for both. For example, the noun-noun suffixation with *-ei* is only applicable to masculine base lemmas, and always produces feminine derivatives: $Bäcker_{Nm}$ – $Bäckerei_{Nf}$ ($baker_N$ – $bakery_N$).

Prefixation causes, according to Glück (2010, p525), a semantic specification of the underlying base lemma, but the effect of this specification can be very different, because the meaning of prefixes is very versatile. Additionally, they are often *meaning-changing*

[8] We found that the inverse of this observation is only partly correct: Prefixation is not necessarily excluded from being a transposition, although this assumption holds for most cases, e.g., $Erfolg_N$ – $Misserfolg_N$ ($success_N$ – $failure_N$). As a counter example, consider the derivation $Anspruch_N$ – $beanspruchen_V$ ($claim_N$ – *to* $claim_V$), where a part of speech change takes place, although the derivational operation is a prefixation. There is no German verb **anspruchen* that could close the gap between the two mentioned lemmas and satisfy the assumption that prefixes are no transpositions. One could merely insert an intermediate nominalisation step, i.e., $Anspruch_N$ – $Beanspruchen_N$ – $beanspruchen_V$ ($claim_N$ – $claiming_N$ – *to* $claim_V$). However, we believe that this is not how humans naturally perceive and produce language, because this nominalisation is far more abstract than the action verb *beanspruchen*. Instead, we claim that also prefixations can cause part of speech changes.

(cf. Section 2.1.3). This can be a general meaning change, as the $heben_V$ – $beheben_V$ example above illustrates, or a polarity change, like gut_A – $ungut_A$ ($good_A$ – bad_A). Note also that there is a grey area concerning the distinction of prefixation derivation and prepositional composition, which is defined differently by different linguists. For example, the prefix *bei-*, as in $mischen_V$ – $beimischen_V$ (*to* mix_V – *to* $admix_V$), is considered compositional in Donalies (2005), but derivational in the system described by Domenig and ten Hacken (1992).[9] We mostly rely on the classification of Domenig and ten Hacken (1992).

Conversions involve a particularity concerning the base word: While in general, the basis of a German derivational process is a word's stem (Donalies, 2005, p47f), verb conversions can be applied to stems as well as lemmas (i.e., infinitives) (Glück (2010, p366f), Donalies (2005, p125)). This difference becomes clear by considering the two nominalisation derivatives of the base verb $besuchen_V$ (*to* $visit_V$): $Besuchen_N$ ($visiting_N$) is derived from the verb lemma, while $Besuch_N$ ($visit_N$) is derived from the verb stem. Both derivation types are highly productive.

Stem change derivation sometimes requires changes additional to the vowel alternation, e.g., the duplication of the stem-final consonant in $greifen_V$ – $Griff_N$ (*to* $grasp_V$ – $grasp_N$). This particularity arises from derivations based on the stem in preterite or perfect tense (in the above example: $griff_{Vpret}$) (Donalies, 2005, p135), and makes this already complex derivation operation even more difficult.

Of the five types, suffixation is the most frequent one[10], but also prefixation and conversion are highly productive in German. Therefore, computational systems about German derivational morphology inevitably must regard these derivational operations. In contrast, neither circumfixation, nor stem changes are synchronically productive any more (Glück (2010, p17, p281), Donalies (2005, p109, p136)).[11] Such unproductive operations constitute a danger for the automatic induction of derivational relations: Considering them might lead to overgeneration and spurious derivational links. For instance, the *i–a* stem vowel change is valid in $klingen_V$ – $Klang_N$ (*to* $sound_V$ – $sound_N$), but not valid for $verdichten_V$ – $Verdacht_N$ (*to* $condense_V$ – $suspicion_N$). Through the combinability of the five operations, this problem becomes even more crucial. On the other hand, ignoring these derivations would restrict the coverage of the system, especially, because these derivation operations often constitute a single link between two subgroups of derivationally related lemmas. This phenomenon is exemplified in Figure 2.1: If the ablaut-involving link between *geben* (*to give*) and *Gabe* (*gift*) is ignored, the group of related words is split into two. Incorporating or ignoring these derivational operations is therefore a trade-off decision between precision and recall.

[9] This system is briefly described Section 3.1.2.
[10] This holds for German (Glück, 2010, p687) as well as for many other languages (Hall, 2000, p539).
[11] Again, we do not accept spontaneous derivations as "productive".

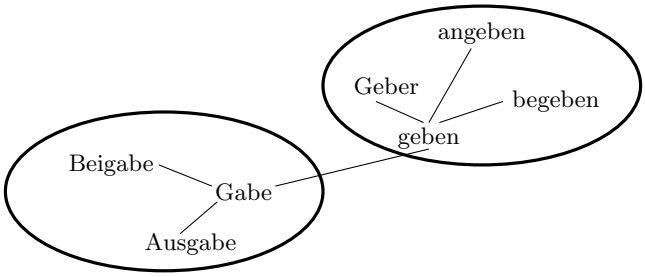


Figure 2.1: Single link by ablaut derivation in a group of derivationally related words

2.2.2 Characteristics of the Involved Word Classes

Finally, we briefly consider the four most prominent parts of speech involved in German derivation (nouns, verbs, adjectives, and adverbs), and their respective particularities. We do this as a pre-study for our induction of a derivational lexicon, where we will apply derivational processes (i.e., rules) to lemmas extracted from a German web corpus. Web corpora are known to be prone to, e.g., spelling errors, which makes standard processing like lemmatisation and part of speech tagging less reliable. Subsequently, the application of derivational rules might cause wrong derivations. In order avoid that the rules are applied to incorrectly lemmatised or tagged words, we want to establish heuristic strategies to correctly detect and distinguish the different parts of speech, and their lemmas. Our goal is to develop a pragmatic, yet linguistically motivated intuition for how to implement our derivational model.

German nouns are easy to distinguish, because they are characterised by capitalisation, as shown in the following simple conversions: *stauen*$_V$ – *Stau*$_N$ (*to jam*$_V$ – *jam*$_N$), *essen*$_V$ – *Essen*$_N$ (*to eat*$_V$ – *food*$_N$). Infinitive verbs (i.e., verb lemmas) bear the three typical suffixes mentioned above (*-en*, *-eln*, and *-ern*), which makes them well recognisable.[12] An example of a derived verb lemma is *fest*$_A$ – *festigen*$_V$ (*tight*$_A$ – *to tighten*$_V$), where *-ig* is the derivational suffix. Adjectives are orthographically distinguishable from nouns and verbs by the absence of capitalisation and verb suffixes, respectively. An adjectivisation example is *Tag*$_N$ – *täglich*$_A$ (*day*$_N$ – *daily*$_A$). Most productive adverb derivations, like *schwer*$_A$ – *schwerlich*$_{ADV}$ (*hard*$_A$ – *hardly*$_{ADV}$) are so-called "adverbial adjectives", which makes them identical to (attributive) adjectives on the surface level. Thus, they do not need to be handled separately.

These rough distinctions between the most important word classes for German derivation can help to select only those lemma-part of speech combinations that are of interest for the respective derivational process.

[12] Note that some adjectives also end with *-en* and *-ern*, e.g., *golden*$_A$ (*golden*$_A$) or *hölzern*$_A$ (*wooden*$_A$). That is, the presence of the verb suffixes does not guarantee that the word is a verb, but the *absence* of these suffixes guarantees that the word is *not* a verb.

2.3 Computational Representation of Derivation

This Section concerns practical aspects of derivational morphology, i.e., its representation in computational resources. Before we describe the structure of such resources, we outline their motivation and the status quo.

Representation of Derivation: Status Quo. In linguistics, derivational morphology has been extensively investigated. There are grammar books specifically dedicated to derivation. For example, for German, Hoeppner (1980), Augst (1975) and Fleischer and Barz (2007) provide lists, explanations, and analyses of derivational processes: Hoeppner (1980) describes many cases of semi-regularities in noun and adjective suffixation, e.g., the optional expansion of the suffix *-los* to *-slos* in $Einfall_N$ – $einfallslos_A$ ($idea_N$ – $unimaginative_A$). However, no resulting lexicon is available. In contrast, Augst (1975) supplies a lexicon about German word formation processes, i.e., about admissible morpheme combinations. Derivation is a clear focus of his work, though, not the only morphological phenomenon discussed. Knowledge about the underlying morphological processes are not given, but this lexicon serves as a comprehensive reference list for affixes. Fleischer and Barz (2007) give explanations about all derivational processes described in Section 2.2.1 and list many derivation rules, but leave semi-regularity largely unspecified.

Such textbooks provide structured and comprehensive material about German derivation. Thus, they could serve as references to implement models that exclusively describe derivational processes and the derivational relatedness of words, while leaving other morphological phenomena unconsidered. However, although such information would be valuable for many applications (cf. Section 3.1.3), no efforts in this direction have yet been made for German: There are – to our knowledge – neither conventional, nor electronic lexicons which systematically and exclusively provide derivational relations between words.

As to conventional lexicons, this lack can be explained by the fact that derivational relations are sometimes intransparent, i.e., there is a semantic drift between base and derived word (cf. Section 2.1.3). Such relations are not in the focus of dictionaries; instead, only semantically coherent information is typically provided. As an example, consider some entries in the German dictionary Duden (Klosa et al., 2001, 1099f) around the word $Mord_N$ (murder), which is a lemma of a group of derivationally related words: The entry of $Mörderin_N$ (female murderer) refers to $Mörder_N$ (murderer) in its definition *"w. Form zu ↑Mörder"*. Similarly, the entry of $Mörder_N$ refers to the verb $morden_V$ (to murder), and also the entry of $morden_V$ uses its base lemma $Mord_N$ in the definition. However, opaque derivatives like $mordsmäßig_A$ (enormous) are not linked to any of these other lemmas.

Also in computational linguistics, there is still little focus on derivation. Of course, derivation is handled, e.g., by morphological analysers (we will return to such analysers for German in Section 3.1.2), but these typically cover the entire range of morphological processes mentioned in Section 2.1 rather than concentrating on derivation. Apart from the issue of intransparent derivations just mentioned, the neglection of derivational

knowledge might be due to fact that most NLP research was, and is, concerned with English. As mentioned at the beginning of this Chapter, English is a derivationally less challenging language than, e.g., Slavic languages or German, because many derivations are simply conversions. Thus, the need of explicit derivational knowledge as a standalone resource is less urgent for English than for morphologically rich languages.

Nonetheless, there are a few computational studies that provide suggestions how to build and respresent derivational information (cf. Section 3.1.2). We call the resource which is the outcome of such approaches a *derivational lexicon*.

Derivational Lexicons. A *derivational lexicon* provides the information which words of a language are derivationally related. It is constructed by means of some (unspecified) knowledge about derivation, by which a language's lemmas can be partitioned into derivationally related groups. The employed knowledge sources can, e.g., consist of rules describing derivational processes, information from other (manually or automatically created) lexical resources or tools, probability scores for string similarity scores, or a mixture of these. Chapter 3 gives an overview of methods to compile derivational lexicons.

As indicated above, a derivational lexicon is not a traditional lexical resource, however, recent years have seen a growing body of computational work on derivation. Derivational lexicons emerged in computational formats through the need of more specific morphology resources, e.g., explicit derivational information, for computational applications (cf. Section 3.1.3). Note that we define a derivational lexicon as a resource which provides exclusively derivational information; that is, all other morphological processes, like inflection or composition, are excluded.

Basically, derivational relatedness can be represented pairwise, i.e., by listing all derivationally related word pairs of a language's word inventory. However, it is obviously more practical and compact to group all derivationally related words at once. We call such groups *derivational families*.

Derivational Families. A *derivational family* describes a group of lemmas which share a common stem, and which are related by linguistically motivated derivation processes. That is, the members of a derivational family are – more or less directly – derived from the same base lemma. One (incomplete) example family in English is:

$$sleep_V \quad sleepy_A \quad sleepless_A \quad sleep_N \quad sleeper_N \quad sleepiness_N \quad sleeplessness_N$$

Note that a derivational family contains neither compositionally related words (e.g., *sleeping bag*), nor products of other word formation processes. By definition, all lemmas are derived from a common base lemma (*sleep*) by one or several more or less complex derivational rules. For instance, the noun *sleeper* is directly derived from the verb *to sleep* by a nominalising suffixation (*-er*). In contrast, the noun *sleeplessness* requires two suffixation steps, *-less* and *-ness*, where the intermediate product is also a valid lemma.

The term "derivational family" has been previously used by other authors, for instance, by Gaussier (1999) and Jacquemin (2010). Similar expressions which allude to a family concept, like "morphological family", are also frequent both in the psycholinguistics

(Booij, 2005, Nagy et al., 1989) and the computational linguistics literature (Daille et al., 2002, Milin et al., 2009, Walther and Nicolas, 2011). Habash and Dorr (2003) use an exceptional terminology, naming their resource a "categorial variation database", and the derivational families "clusters". We find that this technical denomination does not reflect the linguistic fundamentals which account for the relations within one such cluster. Therefore, we adopt the terminology "derivational family".

Formally, the set of lemmas in a derivational family can be considered an equivalence class consisting of derivationally related words; that is, each lemma pair in a derivational family has a – direct or indirect – derivational connection. A formal definition of derivational families is given in Section 4.1.4.

In this thesis, we will construct a derivational lexicon consisting of derivational families. In parallel to previous studies (cf. Section 3.1.2), our requirements for the construction of a derivational family are relatively liberal: We accept all morphologically admissible derivations, regardless of whether they lead – synchronically or asynchronically – to semantically opaque or transparent relations. For example, we accept synchronically unproductive stem changes like *reiten*$_V$ – *Ritt*$_N$ (*to ride*$_V$ – *ride*$_N$), and a subsequent agentivisation like *Ritt*$_N$ – *Ritter*$_N$ (*ride*$_N$ – *knight*$_N$), and consider all involved lemmas being part of the same family, although *reiten*$_V$ and *Ritter*$_N$ are semantically opaque. Thus, our generous definition leads to a maximal coverage of derivational processes (if required, one can subsequently refine the definition, e.g., by only retaining transparently related words in a derivational family). Although this decision limits the semantic coherence of derivational families, we regard, in accordance with our DCA (cf. Section 2.1.3), most derivationally related words semantically related, and thus, derivational lexicons as semantic resources. This position has also been taken in the literature (cf. Section 3.1.3).

Derivational-semantic Families. As just discussed, derivational relationship does not always imply semantic relationship. Even if some members of a derivational family are semantically related, this shared meaning might not be stable across all pairings of members within this family (Jacquemin, 2010). If semantic transparency information about the lemma pairs was available, one could define a semantics-aware representation of derivational families that goes beyond purely derivational relatedness. Although, to our knowledge, such a representation was not discussed before in the literature, it can be realised straightforwardly by splitting the families into sub-clusters according to semantic coherence. Polysemous words which participate with different meanings in different sub-clusters, might be handled specifically. Either, they could be assigned the sub-cluster of their predominant sense, or they could be duplicated. For example, the (incomplete) derivational family (DF) in Example (2.4) can be split into two sub-clusters, both containing the polysemous lemmas *pauken*$_V$ and *Pauker*$_N$:

(2.4) DF *pauken*$_V$, *Pauker*$_N$, *einpauken*$_V$, *Pauke*$_N$
(to cram/play the kettledrum, crammer/drummer, to cram into, kettledrum)

Cluster 1 *pauken*$_V$, *Pauker*$_N$, *einpauken*$_V$
(to cram, crammer, to cram into)

Cluster 2 *pauken*$_V$, *Pauker*$_N$, *Pauke*$_N$
(to play the kettledrum, drummer, kettledrum)

Note that the processing of polysemy becomes highly complex if the derivational families refer to the token level rather than to the type level: In this case, word sense disambiguation is necessary. For each instance, its proper sense and thus, the proper derivational-semantic family must be determined depending on the context. For example, the lemma *Pauker*$_N$ might be an instance of the drummer sense if the context is about a concert, and an instance of the crammer sense if the context is about school. Such an approach might help to correctly determine the derivational family of a token and thus lead to an improved application of a derivational lexicon. However, it is significantly more complex than a lemma-based approach, in which individual instances are ignored, and only the type level is taken in to account, i.e., in which derivational-semantic families a lemma *potentially* can participate. For this reason, all approaches to construct derivational lexicons we are aware of are lemma-based (cf. Section 3.1.2). In this thesis, we will also pursue the lemma-based approach, and investigate how type-level derivational families can be employed.

Excursus: Family Concepts in Cognitive Linguistics. To support the notion of derivational families, we want to mention – without going into detail – that the concept of morphological families is also established in a line of research in cognitive linguistics: Usage-based models assume that the acquisition of linguistic knowledge takes place bottom-up, i.e., through active perception and usage of word instances rather than through rule-based processes. One representative of these models is the "network model" of Bybee (1985, 1988) (similar approaches are, e.g., Langacker (1987), Nagy et al. (1989)). It provides a cross-linguistic explanation for the storage and processing of morphologically complex words. As the name suggests, this model considers the mental lexicon as a network. Words are nodes, and edges are lexical connections between words that take into account phonological and semantic features. If words are phonologically *and* semantically related, a morphological relation between them exists, which is considered the strongest relation in the model. Words that are strongly morphologically related to their base word can even be mapped into this base node, e.g., *cats* and its phonologically and semantically highly related base *cat*. Remarkably, no abstractions are used to represent a word node (e.g., roots or stems), but only actual words. Apart from semantic and phonological aspects, the strength of relatedness between two words also depends on the frequency and formation regularity of the derived/inflected word.

Being based on these lexical connections, the network model implements a structure that connects morphologically related words, i.e., it constructs morphological families. Through the notion of phonological as well as semantic aspects, both purely morphological and morphological-semantic families, as outlined above, are reflected. Since Bybee found evidence for her model in 50 languages, we believe that such a structure is a good base for the representation of derivational relatedness between words in a derivational lexicon. However, contrary to the basic idea of usage-based models, we will address the construction of the derivational families by means of a rule-based approach: While the

bottom-up approach is reasonable from a cognitive point of view, a rule-based method is more suitable for the computational construction of a derivational lexicon. Furthermore, the information about derivational processes that arises from such rules might be helpful for subsequent processing of the derivational lexicon. □

2.4 Summary

Derivation is a morphological word formation process which builds new words from existing ones, where the base and the derived word share the same stem. Derivational processes can be described by linguistic rules that capture a large part of the admissible derivations of most languages. In German, derivation is very productive, especially for the generation of new content words, i.e., verbs, nouns (including named entities), and adjectives. These three word classes are well distinguishable from each other in their surface realisation. We omit adverb derivation, while adverbial adjectives are covered. Further, we consider participles as derived adjectives.

There are five derivation operations for German, generating Germanic as well as foreign words: suffixation, prefixation, circumfixation, conversion, and stem changes, of which mainly prefixation can cause a substantial meaning change between the base and the derived lemma (opacity). Nonetheless, many derivational processes produce semantically transparent words, leading to our Derivational Coherence Assumption which considers derivationally related words generally to be semantically related. Circumfixation and stem change neither occur in many languages, nor are they synchronically productive in German, and they easily overgenerate. Nevertheless, they are necessary to ensure maximal coverage of derivational relations.

Computational resources that exclusively address derivation have become popular only recently. They conflate derivationally related words into so-called derivational families. A derivational family can be considered an equivalence class of lemmas with derivational relations. A resource that provides a set of derivational families is called a derivational lexicon, and can – according to the DCA – be understood as a semantic resource. We will build upon this view for the application of our derivational lexicon (Part III).

3 Related Work

The first part of this Chapter deals with the computational point of view on derivational morphology. We illustrate the research in this area with particular regard to the induction and applicability of derivational morphology resources. In the second part, we examine work related to the field of distributional semantics, which we employ both to semantically filter DErivBase, and to evaluate its applicability in computational linguistics.

3.1 Computational Morphology

Computational morphology (Sproat, 1992) is a longstanding and substantial field in NLP that deals with the processing of word forms and aims at a description of the internal structure of words, and their relation to morphologically similar words. A word is specified by identifying its morpheme boundaries (morphological segmentation) and using morphological features to describe each morpheme, e.g., `[pl]` for the plural-s in *mountains*. Computational morphology captures the regularity of these features as well as exceptions from these rules in word formation. Typically, there are two components in a computational morphology system: A lexicon that describes a set of base words and admissible variations (i.e., the morphological features), and a set of rules that defines admissible combinations. There are (at least) three interesting aspects in computational morphology: how morphological information is *acquired*, how it is *represented* as a computational resource, and how it can be *applied* in NLP.

Morphological analyses of any type (i.e., inflectional, derivational, compositional) are widely used in NLP, either in terms of preprocessing, or in order to provide additional information. Daille et al. (2002) gives an overview of how versatile inflectional and derivational resources (such as CELEX (Baayen et al., 1996)) and algorithms (such as stemming (Porter, 1980)) can be used for tasks like corpus annotation, lexicon building, or terminology acquisition. Clearly, the most widely used type of morphological processing is stemming and lemmatisation, i.e., the conflation of word form variants (Krovetz, 1993), focusing on inflectional issues, as it is far more frequent than derivation. In this work, we concentrate on derivational morphology.

The pioneer of computational morphology is said to be Kimmo Koskenniemi, although various approaches in the context of computational morphology existed before, e.g., the famous Lovins and Porter stemming algorithms (Lovins, 1968, Porter, 1980), or a system based on morphophonemic rewrite rules combined with a dictionary (Kay and Martins, 1970). However, most of these approaches lack in linguistic motivation, generalisation for language independence, or simply coverage of linguistic phenomena; also, they implement only morphological analysis, but not generation (Karttunen and Beesley, 2001).

Koskenniemi (1983) introduced the fundamental theory of a two-level morphology and an accompanying system for morphological analysis as well as generation. As indicated above, a two-level morphological system comprises two components: A lexicon that lists the lexical items of a language (typically morphemes; i.e., word stems as well as affixes), and a set of rules that describes admissible morphological processes on the lexicon's items. A processing component (mostly a finite-state automaton) uses the lexicon and the rules in order to either analyse a word form to access its base word, or generate word forms for a given base word. In this process, the entries in the lexicon and the interaction of the rules constrain the surface realisation of a lexical string, leading to highly accurate analyses or generations, respectively. Two-level morphology is applicable to all word formation types mentioned in Chapter 2 (derivation, composition, inflection). For a brief summary of the history of two-level morphology, see Karttunen and Beesley (2001).

Clearly, Koskenniemi's procedure requires human knowledge about the morphological processes of each processed language. One main reason why the two-level morphology still made the breakthrough is that it can be efficiently realised with the technique of finite-state transducers (FSTs), where the underlying engine is usable interchangeably for any language. There are many subsequent works in this direction, guided to a large degree by Lauri Karttunen and colleagues. Karttunen and Beesley (2005) give an overview of the effort on finite-state morphology and its results for various languages.

Today, there are two main directions in computational morphology: *knowledge-poor*, and *knowledge-based* approaches, the former of which recently gained particular traction. Knowledge-poor methods infer morphological information at the level of word forms from large text collections, using machine learning techniques rather than linguistic knowledge. As a consequence, linguistic aspects are often not captured, e.g., knowledge-poor methods cannot distinguish between inflection and derivation. This contrasts with knowledge-based methods which build on human knowledge in form of linguistic rules (in the tradition of Koskenniemi) or other manually compiled information sources.

In the remainder of this Section, we summarise work related to the computational treatment of morphology particularly for derivation. These studies can comprise merely a derivation *acquisition algorithm*, or also a resulting *resource*. Additionally, one can distinguish between approaches that have the goal to tackle a particular *task*, and those whose benefits have not been attested on applications. We structure the Section according to these three concepts: Approaches that mainly propose a specific algorithm for the acquisition of derivational knowledge (Section 3.1.1), those whose focus is to establish an actual resource (Section 3.1.2), and those that are used particularly for an NLP application (Section 3.1.3). Of course, boundaries are fluid, and we indicate whenever an approach largely covers several of these aspects. We conclude with a discussion in Section 3.1.4.

3.1.1 Algorithms to Acquire Derivational Morphology

Although a clear separation of knowledge-poor and knowledge-based approaches is artificial (again, boundaries are fluid), we make this distinction in the following in order to emphasise the importance of linguistic knowledge to build a derivational lexicon. In

fact, many knowledge-poor studies claim to be unsupervised, which would mean that text corpora serve as their only linguistic evidence. However, they often take into account at least little additional information, which is why we prefer the term "knowledge-poor".

3.1.1.1 Knowledge-poor Approaches

Most knowledge-poor approaches mainly provide algorithms to acquire morphological information, but neither build resources, nor concentrate on specific application scenarios. An excellent overview of the state-of-the-art is given by Hammarström and Borin (2011). In their survey, they distinguish four classes of knowledge-poor methods, the most important two of which for derivational issues are the following:[1]

Boarder and Frequency. These approaches consider frequent occurrences of letter sequences to determine probable morpheme boundaries. According to Hammarström and Borin (2011), this class of approaches, inaugurated by Harris (1955), is the most frequently pursued one among the knowledge-poor approaches. As an example, Goldsmith (2001) induces sets of stems and suffixes, and their combination possibilities (so-called signatures) from corpora. He uses the information-theoretic measure of Minimum Description Length (Rissanen, 1989), which strives to express both the model and the described data as compact as possible, and refines this measure's result with a couple of heuristics.

Group and Abstract. The second most common approach first groups morphologically related words, mostly according to a simple measure like string edit distance, and then tries to abstract common patterns from these groups of words in order to find indicators for morpheme boundaries. For instance, Baroni et al. (2002) first group orthographically similar words of a corpus, and then determine their actual relatedness via Mutual Information. Schone and Jurafsky (2000) employ semantic similarity calculated with Latent Semantic Analysis (Deerwester et al., 1990) as grouping measure. They learn inflectional analyses for English word pairs, being capable to correctly resolve morphological disambiguations (e.g., *caring* should be assigned the stem *care* rather than *car*). In a subsequent study (Schone and Jurafsky, 2001), the authors increase performance by adding, e.g., orthographic measures for common affixes and syntactic information.

Overall, derivational morphology is not the focus of knowledge-poor approaches. This might be due to the fact that inflectional variation is much more frequent than derivation, and can therefore be better exploited to achieve high precision results. Note, however, that all inflection-treating procedures might also retrieve derivational relations, because knowledge-poor models have difficulties in distinguishing morphological processes (Moon et al., 2009, p669). That is, approaches without any linguistic information about morphological processes are virtually incapable of concentrating on one process.

[1] Another interesting technique are the recent non-parametric Bayesian models, e.g., the cross-lingual approach of Snyder and Barzilay (2008) to acquire morphological segmentation.

Hammarström and Borin (2011) mention two knowledge-poor "group and abstract" approaches for derivation: Gaussier (1999) constructs derivational families and extracts derivation rules for French and English.[2] The approach, which only considers suffix operations, is based on an inflectional lexicon, i.e., a list of base lemmas and their possible inflections. In this way, inflectional processes are excluded from the learning, and purely derivational suffixes are obtained. Gaussier acquires the families in two steps: First, the similarity of two words is measured by means of their longest common initial string (a length of 5 is deemed a good threshold). This string is the potential base word, while the two remaining, differing strings are two potential suffixes of this word. In order to discard segmentations and, thus, suffixes that are no actual morphemes of the language, each suffix pair is required to occur at least twice for all word pairs built from the inflectional lexicon. In a second step, these word pairs are conflated into families by means of hierarchical agglomerative clustering, where the number of occurrences of suffix pairs serves as similarity measure. The author suggests to employ this procedure to facilitate lexicographic annotations, and for Information Retrieval.

Jacquemin (1997) provides a distributionally motivated algorithm to collect morphological links that can, to some extent, be specified for derivation. It requires a corpus and a list of multi-word collocations.[3] As in Gaussier (1999), word pairs are truncated to their longest common initial string, which indicates a potential morphological relationship. False positives such as *genetic* and *generate* are excluded by requiring these trunks to co-occur with the same trunk of another word in the corpus. In this way, collocations like *genetic expression* and *gene expressed* are retrieved, using the collocation list as seeds. Repeatedly occurring suffixes (e.g., *-tic*) are considered morphemes, and derivational families are clustered using this notion of relatedness. Although this approach also covers inflection and composition, one can somewhat control the retrieved patterns via the seed collocations to achieve mainly derivations (which is, on the other hand, a drawback, as manual invention is necessary). Unfortunately, again only suffix operations are considered.

Generally, knowledge-poor approaches provide detailed algorithms, but do not establish resources; after all, some implementations are available, such as the Linguistica system (Goldsmith, 2001), or Morfessor (Creutz and Lagus, 2007). Also, Gaussier (1999) provided his algorithm to other researchers (Jacquemin, 2010). Most knowledge-poor algorithms are not designed for specific applications, but as an alternative to manually created morphological processing systems, i.e., mostly analysers.

A main drawback of knowledge-poor algorithms is that the results do not necessarily correspond to linguistic intuition. For example, Goldsmith (2001) reports that it is difficult to identify allomorphs with his approach, i.e., related stems such as *suppli/supply* for the lemmas *supplier/to supply* cannot be mapped onto each other.

[2] Since we aim at building a derivational lexicon, we only discuss the construction of derivational families.

[3] Zweigenbaum and Grabar (1999) realise a comparable, yet more trivial idea, as it is based on structured data from a thesaurus rather than on a raw corpus.

3.1.1.2 Knowledge-based Approaches

Knowledge-based approaches rely on structured, manually created linguistic information, either by means of rule implementations or in the form of input resources. Since the rule-based approaches were briefly illustrated in the beginning of Section 3.1, we now concentrate on a handful of representative and preferably recent methods that employ other knowledge bases with derivational information. As with knowledge-poor approaches, many knowledge-based methods only suggest algorithms to collect the relevant data, but do not provide actual results. Regarding their application, some approaches presented here are used to expand other resources, such as machine-readable dictionaries or WordNet. However, this Section focuses on the acquisition rather than the application.

Knowledge-based systems mostly rely on morphological analysers or existing linguistic knowledge bases. There is quite some work for languages other than English (especially French), which face the problem of recognising derivational relationships more often.

Knowledge from Morphological Analysers. To acquire purely derivational knowledge, morphological analysers are frequently used to process some (mostly inflectional) issues in advance. For instance, Walther and Nicolas (2011) expand a French and a Spanish lexicon with derivational links (pre- and suffixation) between existing entries, and with derivatives as new entries. To conduct these expansions, they induce derivational families from big corpora: They use an unsupervised morphological analyser (Nicolas et al., 2010) to stem the data, and an inflectional lexicon plus a number of heuristic filters to retain only derivational affixation. The quality of the derivational links is good, while the newly added lemmas are often erroneous, mostly due to English words and spelling errors in the corpora. The quality of the results highly depends on the size of the corpora given to the morphological analyser and the heuristic filtering. In a very similar manner, Baranes and Sagot (2014) extract derivational rules and lemma pairs from inflectional lexicons for various languages, including German (the German inflectional lexicon is the semi-automatically induced DeLex (Sagot, 2014)). Contrary to Walther and Nicolas (2011), they use inflected forms rather than stems as starting point to capture words that are derived by stem changes, e.g., the verb preterite derivation in $zwingen_V$ – $Zwang_N$ ($force_V$ – $force_N$) (cf. Section 2.2.1). A bootstrapping-alike procedure ensures that inflectional relations are removed, and that also circumfixation (as a combination of pre- and suffixation) is covered. In a fairly different approach, Sulaiman et al. (2011) build a derivational lexicon for Malay using a morphological analyser that outputs ambiguous analyses. The difficulty for Malay, a language with almost exclusively derivational morphology, lies in the fact that derivational affixes are combined to complex words by nesting and reduplication, where the order of the affixes is crucial. The authors tackle this problem with Expectation Maximisation (Dempster et al., 1977).

Knowledge from Other Lexical Resources. Another group of knowledge-based approaches relies on detailed resources. For instance, Hathout and Namer (2014) build Démonette, a derivational lexicon, enriched with semantic information about the respective derivational processes, using three underlying resources: DériF (Namer, 2009),

a morphological analyser, Morphonette (Hathout, 2011), a database of morphological relations between words, and a list of words extracted from an inflectional lexicon. While DériF provides semantic interpretations for directly derivationally related lemmas, Morphonette indicates direct (e.g., *care – careless*) as well as indirect (e.g., *care – carelessness*) derivational relations between lemmas. The authors combine and expand these resources with manually defined patterns, so that semantic information is available for all pairs of members of the same derivational family (i.e., directly as well as indirectly related words). The current version of the lexicon comprises 31,000 lemmas connected by a selection of seven suffixation processes, so that it is rather a proof of concept than a resource that can actually be deployed in NLP; accordingly, we are not aware of any applications. In another approach, Piasecki et al. (2012) identify derivational relations with a semi-supervised method based on the Polish WordNet (Derwojedowa et al., 2007): Starting with seed examples of derivationally related lemma pairs in WordNet, they build derivational transducers to determine derivation rules and to bootstrap more pairs. The transducers implement prefixation, suffixation and infixation, using information explicitly added by linguists. After validating the produced pairs with a morphological analyser (Piasecki and Radziszewski, 2008), the pairs deemed correct are added as new derivational edges into WordNet. Major problems occur for complex derivational relations, and due to faulty behaviour of the morphological analyser, e.g., for named entities.

3.1.2 Approaches to Build Derivational Resources

As we have shown in Section 3.1.1, there is a decent number algorithms that explicitly deal with derivational morphology, however, none of these studies provides the output of their approach as an actual standalone derivational lexicon, i.e., a machine-readable linguistic resource that is available to the community. Thus, there is a discrepancy between the interest in derivation and the number of derivational resources.

Nonetheless, some derivational resources are publicly available. This Section concerns studies that induce such resources – ideally exclusively, but also in combination with other linguistic phenomena. All resources we are aware of are clearly knowledge-based, i.e., they build partly on existing tools, hand-written derivation rules and/or structured data. As argued in Sections 2.1.1 and 2.1.3, derivation is relatively regular, albeit far from being easily predictable, so that manual intervention is necessary in order to produce sufficiently high quality. Although the processes used to induce the resources presented here include both manual and automatic steps, most studies describe the automatic steps only superficially, so that details about algorithms or parameters are unfortunately unknown. We first discuss our perspective on derivational resources, which is lexicon-oriented, and how it is interrelated with the traditional understanding of computational morphology as word processing. Then, we present approaches to construct derivational resources for several languages, and finally go into detail for German, because this is our language of interest. We focus on resources that are not constructed fully manually and thus offer a reasonable coverage.

Excursus: Lexicon-oriented vs. Word Structure-oriented Perspective. The traditional central task in computational morphology is to analyse (or generate) morphologically complex words, i.e., to deal with morphemes and *internal word structure* of individual word instances, typically using a two-level architecture (an FST and a morpheme-based lexicon; cf. the beginning of Section 3.1). There is a large number of morphological analysis and generation tools, some of which for German are the following:

There is the classification-based Morphix tool (Finkler and Neumann, 1988) which analyses inflection as well as composition, or SMOR (Schmid et al., 2004), which is based on a finite-state transducer and segments German words on the inflectional, derivational, and compositional level. SMOR uses IMSLEX,[4] a morphological lexicon that we will present below, and is one of the central German morphological analysers. It has also been used as the basis for other analysers like Morphisto (Zielinski and Simon, 2008), or Zmorge (Sennrich and Kunz, 2014). Word Manager (Domenig and ten Hacken, 1992) is another analyser that covers the same morphological processes as SMOR. It is handcrafted, but includes, e.g., an automatic out-of-dictionary component for unknown words. It can be queried online on `http://canoo.net`, however, its lexicon is not available standalone. Finally, the TAGH analyser (Geyken and Hanneforth, 2005) is conceptualised slightly differently than SMOR in that it works with weighted FSTs and thus returns only the most plausible analysis (rather than all) for structurally ambiguous words; for instance, it selects the first segmentation of the following two for *menschenfreundlich*$_A$ (*philanthrophic*$_A$):

(A (N (N Mensch) (link en) (N Freund)) (sfx lich))
(A (N Mensch) (link en) (A (N Freund) (sfx lich)))

Lexicon-based systems like SMOR achieve high-quality results. Thus, they are used in virtually any application that requires morphological preprocessing, e.g., parsing (Bohnet et al., 2013), or coreference resolution (Broscheit et al., 2010): Here, the *morphological analysis of words* is important to correctly resolve morpho-syntactic dependencies.

Our perspective, however, is different from this traditional word-structure perspective: We consider one specific morphological process (derivation) from a *lexicon-oriented* point of view. That is, we do not aim at analysing the internal structure of word instances, but the *connectedness of lexemes* through derivation. This goal is more general than that of morphological analysis in two respects: 1., we address words at the level of lexemes, i.e., we generalise over individual word instances;[5] 2., we investigate on which lexemes derivational morphology induces morphological families, i.e., the connections across lexemes rather than within instances of one lexeme. Such interconnections can be better represented in lexicons (i.e., actual lexical resources) than by computing them on the fly with a processing tool. To some extent, the lexicons integrated in morphological analysers provide such interconnections (e.g., the IMSLEX; Fitschen (2004)) – however, without focus on derivation. Approaches to establish lexical resources that exclusively address derivation have been pursued for English, as we will show shortly.

[4] Cf. `https://code.google.com/p/cistern/wiki/SMOR`; last accessed: Mar. 2015

[5] As mentioned in Section 2.3, this avoids matters of word sense disambiguation.

Besides, we would like to note that our lexicon-based perspective implies that inflectional aspects are excluded, as we concentrate on lexemes (typically operationalised by the respective lemmas), and on derivation. Thus, the methods used to build a derivational lexicon must be able to separate inflection from derivation. This, in turn, requires knowledge about the *internal word structure* – e.g., in order to recognise that *higher* is not a derived lemma, but a comparative form –, which brings us back to morphological analysers. That is, lexicons and analysers complement each other: The automatic induction of a high-quality derivational lexicon profits from morphological analysis. At the same time, high-quality morphological analysers mostly rely on lexicon information. However, in this thesis, we will not further examine this mutual influence.

In sum, all approaches to establish a derivational lexicon require morphologically preprocessed data (typically lemmas or stems) in order to exclude inflection. This preprocessing can be based either on morphological analysers, manual efforts, or other, existing lexicons. The following literature review will show that all three cases apply. □

Resources for Languages Other than German. One natural approach to collect derivational information is to exploit existing lexical ontology structures, and integrate morphological knowledge as an additional information layer. WordNet (Miller et al., 1990), the English ontology most frequently used in NLP, was enriched by exactly such information (Miller and Fellbaum, 2003). So-called "morpho-semantic links" between derivationally related lemmas establish links across parts of speech, which is exceptional in WordNet, as most relations connect lemmas of the same part of speech. Note that the term "morpho-semantic link" refers to purely derivational information. The addition of these links was based on an assumption similar to our DCA (cf. Section 2.1.3), i.e., that most derived words are semantically similar to their base words. However, it turned out that there is no clear mapping between certain derivational affixes and the corresponding senses, e.g., due to polysemy (Fellbaum et al., 2009) (cf. the discussion of word sense disambiguation issues for token-based derivational lexicons in Section 2.3). Nonetheless, the morpho-semantic links are used for NLP applications, e.g., deep text understanding (Clark et al., 2008), and a number of proposals exists to correspondingly extend wordnets in other languages (Bilgin et al., 2004, Pala and Hlaváčková, 2007, Bosch et al., 2008). Also for the German wordnet, there recently emerged initial studies about how to establish derivational links (Hoppermann and Hinrichs, 2014).

A lexicon that exclusively supplies derivational knowledge is CatVar (Habash and Dorr, 2003), the "Categorial Variation Database of English": It groups derivationally related verbs, nouns, adjectives and adverbs into families,[6] and was compiled semi-automatically on the basis of a range of high-quality lexical-semantic resources, such as NOMLEX (Macleod et al., 1998) and WordNet, which provide information about derivational relatedness. Derivational families are obtained by a clustering method based on the Porter stemmer (Porter, 1980) that serves as morphological analysis step. Unfortunately, the authors do not provide details on the algorithmic realisation of this

[6] While incorporating adverbs is sensible for English, German adverbs are not informative; cf. Section 2.2.1.

clustering procedure. CatVar corresponds to the style of derivational lexicon as we intend, containing, families such as:

$$ask_V \ asker_N \ asking_N \ asking_A$$

Note that multiple words of the same part of speech can be listed in one family. The above family lists two nouns: an event noun (*asking*) and an agentive noun (*asker*). However, CatVar does not consider prefixation and stem changes, which is why, e.g., the adjective *unasked* is missing in the above family, and ends up in a singleton cluster.[7] CatVar is purely morphologically motivated, i.e., semantically unrelated word pairs like $object_N$ – $objective_N$ are stored in the same family. Since its build process requires various language-specific, high-coverage and and elaborate resources, but such resources do not exist for many languages, it is not straightforward to adopt this approach to construct similar lexicons for other languages. Note that even if similar resources were available, their coverage, content, and quality might differ from those used in CatVar. Thus, it would be hard to compare two derivational lexicons compiled in this way. Also, CatVar does not provide information about the structure within the families, i.e., which word pairs are directly or indirectly derivationally related. For instance, $external_A$ is more closely related to $externalise_V$ than to $exteriorisation_N$, but all three lemmas are simply grouped without any distance measure. Nonetheless, this lexicon has found application in different areas of English NLP, as we will show in Section 3.1.3.

Another knowledge-intensive approach was pursued by Viegas et al. (1996), who build Spanlex, a Spanish lexical-semantic lexicon. Instead of relying on many resources, they manually implement linguistically very detailed derivation rules ("morpho-semantic lexical rules"), into which knowledge from merely one ontology is incorporated. These rules encode not only standard prefixal and suffixal derivation patterns, but also indicate, for each derivation, semantic properties of the derivative, e.g., *Event* for *explotar* → *explosión (to explode* → *explosion)*. These properties, gained from the ontology, are used to define a fairly fine-grained level of about 100 morpho-semantic lexical rules. Since the derivation procedure overgenerates, the produced words are double-checked against a lexicon and a corpus. Eventually, all 35,000 acquired entries are manually assessed. However, this lexicon is, to our knowledge, not available.

Resources for German Derivational Morphology. To our knowledge, there is no dedicated derivational lexicon available for German. Nonetheless, a couple of resources provide derivational information among others, and derivational lexicons might be inducible from these resources with some effort.

For instance, CELEX (Baayen et al., 1996) is a manually annotated lexical database for German, English and Dutch with a variety of linguistic annotation layers, including information about a word's morphology in terms of inflectional, compositional and derivational structure. The derivational structure covers both affixation and stem changing processes (e.g., $Sänger_N$ – $singen_V$ ($singer_N$ – *to* $sing_V$)), and could be exploited to build

[7] The absence of prefixation and stem changes is due to the build process of CatVar: The Porter stemmer used to cluster the derivational families is merely designed for suffix stripping.

derivational families; for example, the morphological decomposition of *Schlaflosigkeit* (*insomnia*) connects this lemma with the lemmas *schlaflos* (*sleepless*) and *schlafen* (*to sleep*), which could thus be grouped into one family. However, the notion of derivational families is not explicitly expressed in CELEX, so that additional procedures would be needed to correctly and exhaustively collect all derivationally related lemmas. To our knowledge, CELEX has not been used for such an approach before. Generally, this database is not built for a specific application, but is broadly used, e.g., in psycholinguistics for the norming of materials (as in Sonnenstuhl et al. (1999), Clahsen et al. (2003)), or in Information Retrieval (Monz and de Rijke, 2002).

As mentioned in the above excursus, morphological analysers typically rely on morpheme lexicons. These lexicons – if available as a standalone resource – could generally serve as a source of information to induce a derivational lexicon. IMSLEX (Fitschen, 2004) is one such lexicon, and provides (among others) morphological information for inflection, derivation, and composition. Although its major purpose is the application to morphological analysis of German (it is used in SMOR; see above), it is available as a separate resource. In contrast to CELEX, IMSLEX exclusively exists for German. It builds upon various manually created morphological and syntactic tools and resources, such as the morphological analyser DMOR (Schiller, 1996) (the predecessor of SMOR (Schmid et al., 2004)), or the lexicon structure of DeKo (Schmid et al., 2001), from which information of different types and granularity levels are gathered, merged, and enriched with additional information. These underlying resources have heterogeneous structures, so that IMSLEX contains both full-form derivatives and morphemes (stems and about 260 affixes) that can be combined to derivatives. That is, a thorough analysis of the data basis would be required before IMSLEX can be used for the induction of a derivational lexicon.

3.1.3 Derivational Morphology Applied in Natural Language Processing

At the beginning of Section 3.1, we mentioned possible applications of morphological information. This Section gives concrete examples specifically for derivational information. Most applications operationalise the notion of a derivational lexicon in parallel with our Derivational Coherence Assumption (cf. Section 2.1.3): Derivational families are assumed to frequently capture not only morphological, but also semantic relatedness and are thus employed as semantic resources. Clearly, this assumption is simplifying, even for semantically transparent derivations – both in terms of syntax (e.g., the verb in *I read a book* is not simply substitutable by the noun *reader*) and in terms of semantics (even the semantically highly similar words *read* and *reader* have slightly different meanings). In practice, however, it is reasonable since 1., many derivations in fact are transparent, and 2., the subtle semantic drifts of transparent derivations are often not essential. That is, *read* can be transferred to *reader*, if also other parts of the corresponding sentence change, e.g., to the phrase *the reader of the book* (Jacquemin, 2010). Since many applications deal with language variation, one can indeed take advantage of the semantic similarity of many derivational relations, making them a legitimate semantic information source. However, we note that the quality of the results might depend on the complexity and transparency of derivation in the language of interest. Thus, semantic restrictions and

refinements might be required, such as Jacquemin (2010) does for French (see below).

In the following, we describe three language-independent usages of derivational information to improve distributional modelling, and a variety of end-to-end applications.

Derivational Morphology Used for Distributional Modelling. On the basis of the DCA, lexical semantics is a common application area for derivational information, e.g., by integrating it into distributional semantic models (cf. Section 3.2). In Section 3.1.1.1, we have introduced how morphological information can be distributionally *acquired* (Jacquemin, 1997, Schone and Jurafsky, 2000, Baroni et al., 2002). Now, we briefly present three approaches with the exact opposite goal: They *employ* derivational morphology to improve and refine distributional similarity predictions, particularly for rare or unknown words that are often poorly represented and pose problems in morphologically rich languages. Note that here, derivation is not necessarily used to solve a specific task, but to improve the representation of word semantics.

Lazaridou et al. (2013) adapted the idea of compositional distributional semantics, where larger text passages such as phrases are represented by vector combinations using algebraic functions (Mitchell and Lapata, 2010), to the level of morphemes: They developed a model in which a derivationally complex word such as *rebuild* is represented by two separate vectors: one for the derivational affix (*re-*) and one for the stem (*-build*), respectively, rather than by one single vector for the whole word. The derivational decomposition is based on the CELEX database. The intuition of the authors is that affix morphemes change a stem's meaning and thus should be represented individually. The two vectors are combined using various distributional composition approaches, and their quality is compared to that of the representation of the full word in a non-compositional distributional space. Experiments show that the morpheme composition leads to more sensible distributional representations and better similarity predictions than a conventional full-word model. Notably, the model assigns reasonable relatedness scores even across parts of speech. However, it is still restricted to primitive derivations consisting of a stem and merely one affix.

Technically fairly different, but with the same basic idea, Luong et al. (2013) use recursive neural networks to derive representations of morphologically complex words. Again, each morpheme (i.e., affixes and stems) is considered an individual unit. Due to the recursion approach, this model is capable to handle multi-step derivations such as *in-oper-able*. For the derivational decomposition, the unsupervised morphological analyser Morfessor (Creutz and Lagus, 2007) is employed, followed by some postprocessing steps to increase precision. The authors demonstrate improvements with their approach on a number of word similarity tasks, including datasets with very rare words. As typical for knowledge-poor approaches, this model does not distinguish between derivation and inflection. Also, it is specialised on relatedness within rather than across parts of speech.

Finally, Botha and Blunsom (2014) employ a probabilistic log-bilinear language model rather than neural networks or traditional distributional vector spaces. Probabilistic models allow for preserving all possible segmentations (and thus, interpretations) of a multi-step derivation, and lead to more informed estimates. This approach is motivated

by the authors' goal to improve a machine translation system, which typically requires probabilistic information. Again, derivational decomposition is obtained from Morfessor, and derivation is not exclusively considered, although it is in the focus of this work.

Derivational Morphology for End-to-end NLP Applications. We are aware of two derivational lexicons that have been applied to various end-to-end tasks based on assumptions similar to our DCA. One of these lexicons is CatVar (cf. Section 3.1.2). With its release, a solid English resource became available that has been used, among others, for the induction and expansion of other lexical resources. For example, CatVar is employed by Green et al. (2004) to expand semantic roles resources: Their aim is to automatically induce a new frame-semantic resource called SemFrame, which extends the well-known FrameNet (Baker et al., 1998). By combining information from CatVar and WordNet, it is possible to link words in the FrameNet structure. For instance, one SemFrame frame is called ORNAMENTATION, while the corresponding FrameNet frame is called ADORNING. By knowing that $adorn_V$ and $ornament_V$ are related by a WordNet synset, and by knowing that $adorn_V$ and $adorning_V$ and $ornament_V$ $ornamentation_N$ belong to the same derivational families in CatVar, respectively, it is possible to link the ORNAMENTATION and the ADORNING frame. As a result, the lexical unit inventory for both resources can be expanded.

Also, CatVar has been used as (one of several) components to resolve NLP tasks like Textual Entailment, Language Generation, or Machine Translation. Unfortunately, none of the applications reports the impact of the derivational lexicon separately (i.e., ignoring other information sources), so that it remains unclear to what extent the respective tasks benefit from incorporating derivational information.

Derivational information can be used to recognise paraphrases and Textual Entailment (TE) relationships between two texts (Androutsopoulos and Malakasiotis, 2010, p147). Thus, Szpektor and Dagan (2008) use CatVar to improve their recognising Textual Entailment (RTE) system. TE is a task in which systems have to assess whether a human reading of a Text T infers that a Hypothesis H is most likely true (Dagan et al., 2005). CatVar is used to build new entailment rules, i.e., rules which determine whether one expression can be inferred from another one. Example (3.1) shows that a noun modifier which acts as a predicate, can be replaced by the derivationally related verb without changing the meaning of the original phrase:

(3.1) the $running_A$ X $\leftrightarrow$ X $runs_V$

Comparable approaches which integrate derivational information for RTE are, e.g., Shnarch et al. (2011), Berant et al. (2012).

Another application a derivational lexicon can contribute to is language generation. Thadani and McKeown (2011) employ CatVar's derivational families to increase the fluency of automatically generated text. More specifically, they deal with the task of sentence intersection generation, where two or more sentences are conflated into one single sentence which contains the information expressed in all input sentences, and nothing more. Derivational information serves for replacing words by their morphological variants,

if these improve the sentence's fluency. Example (3.2) shows two derivational variants of the same circumstance, which are supposed to obtain different fluency judgements:

(3.2) (a) Ferrero is mainly a candy *producer*$_N$.
(b) Ferrero mainly *produces*$_V$ candies.

As a last application of CatVar reported here, its creators themselves use the lexicon for word alignments in Machine Translation (Ayan et al., 2004). They build a framework to combine two different alignment algorithms: GIZA++ (Och and Ney, 2003), a standard word aligner, and DUSTer (Dorr et al., 2002), a linguistically informed partial aligner. The alignment of DUSTer works with dependency trees of the source language, extended with semantic information (i.e., semantic verb classes), and rules of derivational variability gained from CatVar, as in Example (3.3):

(3.3) x *fears*$_V$ y → x has *fear*$_N$ of y

In such variability rules, the connection (or "alignment") of semantically corresponding words is ensured by explicit indexes, e.g., *fears*$_1$ and *fear*$_1$. Finally, GIZA++ and DUSTer are combined to find the best common alignment. This combination can, to a certain degree, be understood as a query expansion for GIZA++. This approach leads to better alignments for cases of categorial variation between source and target sentence, as in the Spanish/English sentence pair in Example (3.4):

(3.4) *(ES)* *Ella tendrá* ***miedo*** *de sus enemigos .*
She will have **fear** of her enemies .
(EN) She will **fear** her enemies .

For French, Jacquemin (2010) uses derivational knowledge for Question Answering (QA). Following Gaussier (1999), he uses a – publicly unavailable – derivational lexicon for sentence rephrasing, which requires the lexicon to be semantically coherent.[8] Since the families contain incorrect information (in morphological as well as semantic respects), they are semantically filtered for each test instance using an elaborate French dictionary (Dubois and Dubois-Charlier, 1997), and word sense disambiguation. Unfortunately, the impact of the derivational lexicon on the QA task is small, possibly due to low coverage for some parts of speech – despite the usage of a comprehensive dictionary. Thus, this approach crucially depends on high-coverage, high-quality data, which prohibits the transfer to other languages. However, the author legitimately argues that it is hard to improve extrinsic tasks with one specific resource.

3.1.4 Discussion

We have presented various state-of-the-art algorithms, resources and applications to incorporate and consider derivational information in computational morphology. For our purpose of building a standalone derivational lexicon for German, we see the following major restrictions:

[8] Besides Démonette, this is one of the few studies that take semantic aspects of derivation into account.

Knowledge-poor Methods.

- A lack in *linguistic information* leads to problems in differentiating derivation and inflection, which results in low precision specifically for derivation, as it is less frequent than inflection.
- Most approaches are either very permissive, leading to *many false positives*, or very restrictive, so that some *linguistic phenomena remain uncovered*. A prototypical example is circumfixation, which occurs in many disparate language families (cf. Hall (2000, p540f)), but is, of all knowledge-poor methods presented above, only approached by Schone and Jurafsky (2001). Similarly, stem changes are problematic without linguistic knowledge. Consider the following German verb to noun derivations, $reißen_V$ – $Riss_N$ ($to\ rip_V$ – rip_N), and $schreiten_V$ – $Schritt_N$ ($to\ pace_V$ – $pace_N$). These derivations are conceptually similar: Both convert "ei" to "i", and duplicate the last letter (German "ss" is sometimes commuted to "ß"). However, there are only few cases of such derivations (cf. Section 2.2), which might hinder a knowledge-poor method from retrieving sufficient material to capture these regularities, and achieve reliable corpus statistics.

We find these restrictions in terms of quality, coverage and interpretability fairly severe, so that we decide against a largely unsupervised induction for our approach.

Knowledge-based Methods.

- Knowledge-based approaches tend to involve a *considerable machinery of linguistic resources*, processing tools, or both. The – typically manual – creation of such resources and tools is often costly and not easily transferable to other languages.
- As there is no derivational lexicon available for German, derivational information needs to be extracted from more general *resources with a different focus* (e.g., CELEX, IMSLEX). This, in turn, might cause problems in separating derivation from other morphological processes, and gathering all admissible relations.
- These workaround resources are either *not freely available* (CELEX), or have *limited coverage* by today's standards (e.g., CELEX covers about 50,000 lemmas, and IMSLEX about 60,000 lemmas, respectively, including many compounds that are not derivationally related to any other lemma).
- We believe that the *internal structure of families* that is described by derivational rules that relate two lemmas to one another is linguistically valuable. Such a relatedness indicator is not available in previous lexicons such as CatVar.

In this thesis, we employ a knowledge-based approach to induce a German derivational lexicon that will address these issues (cf. Chapter 4). We expect this lexicon to be similarly widely applicable as the English CatVar.

3.2 Distributional Semantics

As indicated in Chapters 1 and 2, there are relationships between derivation and word semantics; accordingly, our goal is to build a derivational lexicon that is not only morphologically, but also semantically motivated, i.e., that accounts for semantic drifts through derivation. Note that we refrain from trying to model the explicit meanings of derivational processes; studies like Pala (2008) show that a full (symbolic) classification of the semantic variety of derivation is almost intractable (cf. Section 2.2.1). While there are recent approaches that aim at predicting the meaning of derivation with distributional methods (Kisselew et al., 2015, Padó et al., 2015, Marelli and Baroni, 2015), we intend to model semantic relatedness on a simpler level, i.e., to reveal which derivational relationships also imply (transparent) semantic relationship, and which do not. To quantify semantic drifts by means of derivation, we also employ – as the studies just mentioned – *distributional semantics*, a specific method in computational semantics. Since we additionally base various experiments on distributional semantics to evaluate the applicability of DErivBase, we dedicate this method a separate literature review.

This Section explains the fundamental assumptions in distributional semantics and its key characteristics that need to be considered. We also note in which respects distributional semantics is to be preferred over ontologies.

Rationale of Distributional Semantics. Distributional semantics is a data-driven, statistical approach to capture the semantics of words, phrases or other linguistic units (Turney and Pantel, 2010, Erk, 2012). It builds on the *distributional hypothesis* (Harris, 1954, Firth, 1957), according to which words that occur in similar linguistic contexts tend to have similar meanings. In practice, this abstract notion of similarity between two target words is operationalised by constructing context vectors from large text corpora, and using these as approximations of the words' meanings. These context vectors are typically acquired in an unsupervised manner, e.g., by counting word co-occurrence frequencies within a word window of a predefined size, and are collected in large matrices. Based on these representations, the semantic similarity of word pairs is measured by the similarity of their vectors. Such vector spaces are called, e.g., vector space models (VSM), or distributional semantic models (DSM); we will employ the abbreviation DSM. DSMs have been applied successfully to many NLP problems, e.g., synonym detection (Landauer and Dumais, 1997), semantic priming (Lowe and McDonald, 2000), semantic clustering (Schulte im Walde, 2006), or word sense disambiguation (Agirre and Edmonds, 2006). For an insightful study on the interpretation of DSMs, cf. Sahlgren (2006).

Note that the vector representation in DSMs conflates all contexts and thus, all senses of the target word, which complicates their distinction; one approach to still differentiate the senses is the use of second-order contexts (Schütze, 1998).

Parametrisation of DSMs. A major advantage of DSMs is their automatic and straightforward construction: For their simplest conception, it is sufficient to have a big text corpus at hand. On the downside, this means that DSMs are strongly underspecified,

i.e., many parameters need to be determined about how construct and evaluate the information in the model. According to Lowe (2001), the four main parameters are 1., the set of basis elements for the matrix dimensions, 2., the form of the co-occurrence frequency mapping between target items and their contexts, 3., the measure to assess the vector similarity of two targets, and 4., a function by which the DSM can be transformed into another, e.g., lower-dimensional smoothed space. A fifth important aspect is the linguistic preprocessing (Bullinaria and Levy, 2007, 2012). In the following, we focus on *word-context matrices*, where the target items in the rows correspond to words, and the basis elements in the columns (or *dimensions*) are the words' contexts (as opposed to, e.g., term-document matrices, where rows correspond to words and columns to documents in which these words are looked up).

To briefly exemplify common instantiations of Lowe's parameters on word-context DSMs: The corpus data is often preprocessed, i.e., tokenised, stop word-filtered and lemmatised. Established context types (1.) are plain lemmas or lemmas and their syntactic relation to the target word; often, only the n most frequent content words in the underlying corpus are considered. Co-occurrence (2.) can be calculated as simple frequency counts within a specified word window, or by measures such as Mutual Information. The most common similarity measure (3.) used to compare two vectors is the cosine similarity, but there are also other popular measures as well as cosine variants, e.g., rank-based cosine similarity (Jones et al., 2006, Hare et al., 2009). The DSM can be used in form of a plain matrix or smoothed (4.), e.g., by dimension reduction using Singular Value Decomposition (SVD).[9]

Data Sparsity. Apart from the parameters mentioned above, the choice of the underlying corpus is essential for the quality of a DSM. The key rule is: The more data, and the cleaner it is, the better, because the representation of a word's meaning enhances as more corpus data is available. Conversely, this means that small amounts of text lead to bad representations. Thus, *data sparsity* is a real problem for DSMs. Words that rarely occur in the underlying corpus are not well represented, leading to strong biases towards the existing contexts. This poses difficulties particularly for languages that are highly productive in terms of, e.g., derivation or composition, as is the case in German. Some types of distributional spaces are more prone to sparsity than others; we will go into detail on that shortly.

In the following, we briefly describe bag of words and syntax-based models, two important word-context DSM types, and their respective strengths and weaknesses. Note that in this thesis, we concentrate on traditional "count"-based models (Baroni et al., 2014) rather than the more recent "predictive" models which learn low-dimensional distributed representations to predict contexts (Mikolov et al., 2013); we do so, as it is questionable whether this new trend is to be preferred over the established count-based approach (Levy and Goldberg, 2014).

[9] Such a mathematical smoothing has two advantages: It can improve the conceptual representation of the model, and it shrinks large spaces with many zero elements and hence makes them easier operable. However, the smoothing also complicates the interpretability of the vector space.

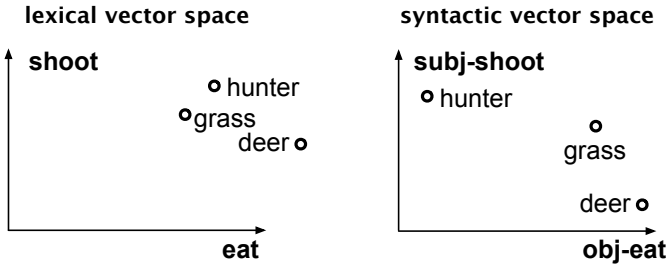


Figure 3.1: Information content in bag of words (left) and syntax-based spaces (right)

Bag of Words (BOW) Models. BOW models (Salton et al., 1975) represent target words by means of context words co-occurring within a surface window around the target word. These lexical models are simple, robust, can be built from any tokenised corpus, and typically achieve very high coverage on many datasets (close to 100%). Previous studies show that the optimal size of the context window depends on the respective task (Peirsman et al., 2008). More specifically, small windows are to be preferred for retrieving paradigmatic relations (i.e., substitutable terms), large windows for retrieving syntagmatic relations (i.e., topic-related terms). Languages with free word order might behave differently to this rule of thumb (Sahlgren, 2006), and lead to generally less exact representations. Applying dimensionality reduction methods like SVD to BOW models helps to generalise over the surface word choice (Bullinaria and Levy, 2012).

Syntax-based Models. These models (Grefenstette, 1992, Lin, 1998a, Padó and Lapata, 2007, Baroni and Lenci, 2010) are based on contexts of lexico-syntactic patterns. Typically, such patterns are word-link-word triples in the form of dependency links, extracted from parsed corpora. This elaborate notion of context makes DSMs better applicable to languages with free word order, and to a wide range of tasks. For instance, the Distributional Memory (DM) framework captures, among others, structure-dependent semantic phenomena such as predicate-argument plausibility (Baroni and Lenci, 2010). At the same time, however, syntax-based spaces are much sparser than BOW models, with a lower coverage (often 50–70%), which makes the modelling of rare targets problematic. Also, their construction usually requires a large, well-parsed corpus, which has limited a large-scale construction of syntax-based models to only a few languages (Baroni and Lenci, 2010, Padó and Utt, 2012, Šnajder et al., 2013).

Comparison of Word- and Syntax-based Models. The conceptual difference of lexical and syntax-based semantic spaces is illustrated in Figure 3.1. The contexts of the BOW space are two lemmas (*eat*, *shoot*); since they occur roughly equally often in the context windows of the three target words (*hunter*, *grass*, *deer*), these targets are fairly similar. In contrast, the contexts of the syntax-based space include the syntactic relationship

of the context word to the respective targets, leading to a linguistically more nuanced representation of the target words (i.e., they are more dissimilar).

Thus, a syntactic model incorporates a richer, structured notion of context than a word-based model: It represents not only words, but also word pairs and syntactic argument relations and can thus be used to model many linguistic problems (Baroni and Lenci, 2010). Also, it contains more accurate information about semantic relations than a BOW model (Padó and Lapata, 2007). Although we are not aware of a dedicated study to this question, it is commonly assumed that syntax-based DSMs are useful particularly for languages with free word order. At the same time, however, they are much more prone to sparsity problems, as syntactic co-occurrences are spread out over more dimensions. For rare words, the vectors can become so sparse that there is no overlap in contexts with any other word. As a consequence, syntax-based spaces have reliability and coverage problems, while word-based models are more robust and provide similarity estimates for many word pairs. Thus, the two model types offer an inherent trade-off.

In sum, the linguistically more grounded representation in syntax-based distributional models is a promising alternative to BOW models, particularly for the German language with its relatively free word order, provided that sufficient semantic information in the form of corpora – or other sources – are available.

Excursus: Distributional Semantics vs. Ontologies. The classical approach to represent semantic relatedness are ontologies such as WordNet (Miller et al., 1990) or its German counterpart (Hamp and Feldweg, 1997). Ontologies typically consist of a hierarchical graph structure that reflects relations between (different senses of) words, e.g., synonymy, meronymy or hyponomy, but also less typical relations such as morpho-semantic links that indicate derivational relatedness (cf. Section 3.1.2).

To measure the semantic similarity of two words, wordnets are usually employed by measuring the distance between the words in the graph, taking – among others – the path length or depth into account (Budanitsky and Hirst, 2006). However, such ontology-based approaches come with several drawbacks. For instance, wordnets are typically (semi-)manually constructed, which is costly, cannot be transferred to other languages, and leads to a static knowledge representation and coverage lacks, which is particularly undesirable for our purpose of building a high-coverage derivational lexicon.[10] More importantly though, the structure of an ontology restricts the linguistic generalisations that a model building upon it can realise. In other words, if word similarity arises from a specific context which is not represented in the ontology structure, it can not be measured adequately; this is the case, e.g., for so-called "ad hoc" categories (Barsalou, 1983).

Since ontologies cannot exhaustively capture all possible contexts, a dynamic, data-driven method such as distributional semantics, where very diverse contexts are considered, is a sensible alternative. This holds particularly for the semantics of derivation, since we expect derivationally related words to exhibit rather specific similarity patterns. □

[10] Nonetheless, there are approaches to expand ontologies dynamically, e.g., the bootstrapping approach of Piasecki et al. (2012) presented in Section 3.1.1.2, or the usage of lexico-syntactic patterns applied to dictionaries or big corpora (Hearst, 1998).

In this thesis, we employ distributional semantics in two respects: On the one hand, we use a DSM as an information source to introduce knowledge about the semantic relatedness of derivationally related lemmas into DERIVBASE (Chapter 5). On the other hand, we employ DSMs for the evaluation of our lexicon: We examine to what extent derivational information can improve the performance of a state-of-the-art distributional model on standard tasks in lexical semantics (Chapter 6) and in psycholinguistics (Chapter 7).

Part II

Modelling Derivational Knowledge for German

4 DErivBase: Inducing a Derivational Morphology Lexicon for German

In this Chapter, we describe the process of inducing DErivBase, a derivational knowledge resource for German. We aim at building a lexicon with a structure as described in Section 2.3: Information about derivational relatedness is provided in the form of *derivational families* which consist of related lemmas. Note that, at this point, our motivation is to build a purely morphological lexicon, meaning that we aim at grouping derivationally related words, no matter how similar or different their semantics might be. The classification of words according to their derivational relatedness is very reliable, as derivation is clearly defined. In contrast, the definition of semantic relatedness is somewhat fuzzier. Of course, semantic distinctions within one family can constitute an additional step to improve the (semantic) quality of the lexicon (cf. Chapter 5). Nonetheless, a purely morphological lexicon is already an interesting and useful linguistic resource: As shown in Chapter 3, similar lexicons have been successfully employed by taking advantage of the Derivational Coherence Assumption (cf. Section 2.1.3), i.e., that most derivational relationships are semantically transparent.

Our goal is to build a derivational lexicon that addresses the issues mentioned in Section 3.1.4: We want to induce derivational families with maximal coverage as well as high quality, in order to provide a reliable – and freely available – lexicon that is applicable to a broad range of tasks (cf. Section 3.1.3). As discussed, there is often a lack of precision as well as linguistic interpretability when knowledge-poor methods are employed, while many knowledge-based approaches require a considerable amount of knowledge sources and might lack in coverage. Since we target both precision and an approach with few resources, but also want to maintain comprehensibility, we choose a rule-based method combined with corpus evidence, in which manual intervention is kept relatively low: Following the work of Šnajder and Dalbelo Bašić (2010), we define the derivational processes by means of derivational rules in an intuitive and easy-to-use rule-based framework. The derivational rules induce a partition of the language's lemmas into derivational families, where the lemmas are gained from a big German web corpus. Thus, our method only requires a comprehensive set of lemmas, and knowledge about admissible derivational processes, which can be gathered, for instance, from linguistics textbooks. With our model, we induce derivational families of high precision, resulting in a reliable, but at the same time high-coverage lexical resource for German.

We begin with a short overview of our method. Figure 4.1 shows the overall procedure to induce the derivational families in DErivBase. On the one hand, we extract German lemmas from SdeWaC (Faaß and Eckart, 2013), a large web corpus. On the other hand, we implement German derivation rules in a language-independent, rule-based

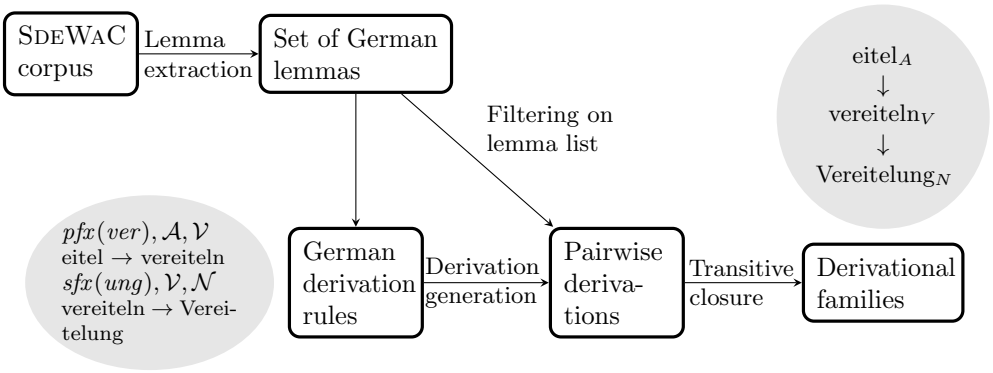


Figure 4.1: Induction of derivational families

framework by means of transformation functions. Then, the rules are applied to the extracted lemmas, where these lemmas act as base words. The bubble at the lower left shows two such rules and their exemplary application to German: The adjective *eitel* (vain) is derived to *vereiteln* (to block) by the displayed prefixation rule. Note that this rule also determines the part of speech of the base and the derived lemma; in this example, the base lemma is must be an adjective, and the derivative a verb. A problem for such rule applications is overgeneration: The second, valid German derivation rule displayed in this bubble could be used to derive an incorrect word such as **Liebung* from the verb *lieben* (to love). To avoid such overgenerations, we accept as derivatives only words which occur in the lemma list extracted from SdeWaC. After this step, we have information about pairs of derivationally related lemmas. Implementing derivation rules for all derivationally related lemma pairs of a language would lead to an overly big amount of rules, which would make maintenance as well as human assessments difficult. Thus, we decided to implement the minimal set of derivation rules, i.e., we omitted rules which can be replaced by combinations of other rules: Multi-step derivations like the derivation $eitel_A$ – $Vereitelung_N$ ($vain_A$ – $blocking_N$) are not implemented directly in the rule set, but can be reproduced by successively applying the two rules shown in the Figure. We do this composition by computing the transitive closure of the rule applications, i.e., by clustering all lemmas which are – directly or indirectly – related by the application of rules, as shown in the bubble on the right-hand side. These clusters then constitute our derivational families in DErivBase.

From a linguistic point of view, the question arises whether derivational relations are indeed transitive, i.e., whether every word pair that is connected by a sequence of derivation rules, actually instantiates a derivational relationship. For instance, the word pair $Stechen_N$ – $Stachel_N$ ($twinge_N$ – $spike_N$) is derived from the same base verb, *stechen* (to prick). But it is somewhat unclear whether such a shared internal word structure suffices to establish a derivational relation. To the best of our knowledge, there is no consensus about this question in the linguistics literature. In this work, our aim is to

automatically generate a high-coverage, but linguistically plausible computational lexicon in a human-understandable way (e.g., by a minimal set of derivation rules). To reach this goal, it is a simple, yet effective assumption that derivation is transitive, and that derivational relations can be approximated with the transitive closure.

The structure of this Chapter is as follows. First, we describe the derivation framework of Šnajder and Dalbelo Bašić (2010) which we employ (Section 4.1), and how we adapt and apply it to specifically induce the German lexicon DERIVBASE (Section 4.2). Sections 4.3 and 4.4 present our setup for an intrinsic evaluation, and the results, respectively. Finally, we summarise our findings and sketch further directions and challenges.

4.1 The HOFM Framework

In this Section, we describe HOFM, a rule-based model for morphology by Šnajder and Dalbelo Bašić (2010). After a short introduction to rule-based derivation models in general and a classification of HOFM into this field (Section 4.1.1), we focus on the components of this framework. We explain its procedures to define derivation rules (Section 4.1.2), their instantiation (Section 4.1.3) and, building on these rules, the induction of derivational families (Section 4.1.4). As we employ HOFM for German derivation, we illustrate the functionalities with German examples.

4.1.1 HOFM, a Rule-based Derivation Model

The purpose of a derivational model is to define a *derivation grammar*, i.e., a set of transformations along with a specification of their application that correspond to valid derivational word formation rules. Rule-based frameworks offer convenient representations for derivational morphology because they reflect the generally regular structure of derivation and have interpretable representations. For that reason, rules have frequently been employed in the literature to describe derivational processes (cf. Jackendoff (1975) and Aronoff (1976), but there are also theories outside the framework of generative grammar; cf. Naumann and Vogel (2000) for a short, or Štekauer and Lieber (2005) for an extensive overview). Also, rules can be explicitly tuned for high precision or recall: The implementer can decide how many derivations to cover and thus influence the coverage and precision of the rule set. For example, if precision is prioritised over recall, one can implement only highly regular rules which are expected to be correctly applied in most (if not all) cases of applicability.

The choice of the rule-based framework is in principle arbitrary, as long as it can comprehensively and conveniently express the derivational phenomena of a language. Therefore, it must fulfil two properties: First, the framework must be formally expressive enough to represent the derivational operations. For German, we need to formalise affixation and stem changes (cf. Section 2.2.1). In fact, both can be characterised with regular languages. Thus, the traditional approach in computational morphology to use finite-state transducers (FSTs), which can be regarded as more elaborate finite-state automata, would suffice. Morphologies for languages with similar challenges are commonly

implemented with two-level formalism rules (Karttunen and Beesley, 1992) or XFST replacement rules (Beesley and Karttunen, 2003). Second, the formalism must allow for a compact and simple, human-readable representation of the derivation rules. Compactness and simplicity facilitate the maintenance of derivation grammars in two respects: The rule sets are smaller and generally easier to handle, and a human-readable rule description eases the implementation, correction, and interpretation of the rules.

In this work, we adopt the language-independent morphology modelling framework proposed by Šnajder and Dalbelo Bašić (2010). More specifically, we employ HOFM[1], a Haskell implementation of this framework, because it fulfils both required properties: The expressiveness of the formalism is equivalent to that of the replacement rules commonly used in finite state frameworks, so that it can represent all admissible word formation rules of the German language. Also, it uses human-readable rule descriptions which we find easier to read than the traditional FST rules. In fact, human readability is a central feature of the framework: HOFM uses an Embedded Domain-Specific Language (EDSL) within the host language Haskell (Mernik et al., 2005), i.e., it embeds syntax specifically designed to implement morphology grammars. In this EDSL, derivation rules resemble the descriptions of derivational processes in traditional grammar books, which allows for a convenient modelling and easy understanding of derivational morphology.

The framework covers both inflection and derivation, but makes a clear distinction between these two different morphological aspects by providing them in two separate modelling components. We concentrate on the derivational component (cf. Section 4.1.2); since we will work on the basis of lemmas, i.e., uninflected words, inflectional processes can be largely ignored (details follow in Section 4.2.1). For detailed studies about inflection and derivation in HOFM, cf. Šnajder and Dalbelo Bašić (2008) and Šnajder and Dalbelo Bašić (2010), respectively. In a third component, the so-called transformation module, language-specific properties are defined which hold for derivation as well as for inflection. For instance, this component can determine crucial phonological regularities in the language of interest, such as the umlaut stem change in German (cf. Section 2.2.1), or the palatalisation alternation in Croatian (Šnajder et al., 2008).

The modular structure of HOFM makes the framework flexible and thus, applicable to many synthetic languages.

4.1.2 The Derivational Component of HOFM

Let us now examine HOFM's derivational component in more detail. Its building blocks are *derivational rules*, and *transformation functions*. A derivational rule d describes the word formation process, i.e., the derivation of a *derivative* from a *base* word. It is defined as a triple:

$$d = (t, \mathcal{P}_1, \mathcal{P}_2) \tag{4.1}$$

where t is the transformation function that maps a base word's stem (or lemma) into the derived word's stem (or lemma). $\mathcal{P}_1$ and $\mathcal{P}_2$ are the sets of *inflectional patterns* of the

[1]Higher-Order Functional Morphology, `http://takelab.fer.hr/data/hofm/`; last accessed: Nov. 2014

base word and the derived word, respectively, which specify the morphological properties of the lexical material that is the rule's input and output, e.g., parts of speech or other inflectional characteristics.

Derivational rules in HOFM can be defined both *on the basis of stems and of lemmas* for base and derived word, respectively. That is, the input and output of a derivation rule can be stems as well as lemmas. HOFM offers all four combination possibilities to define derivation rules: stem-to-stem, stem-to-lemma, lemma-to-stem, and lemma-to-lemma derivation (Šnajder and Dalbelo Bašić, 2010, p110). For example, the stem-to-lemma combination is implemented as follows: The inflectional component of HOFM first reduces a given base lemma to its stem (based on the lemma's inflectional pattern, $\mathcal{P}_1$), then, the derivational component derives a lemma using this stem as input; thus, the resulting derivative is a lemma. Of the four combinations, one can choose whichever best represents the language of interest. Obviously, this decision can turn out differently for different languages, and even within one language, using various combinations can make sense. Note that in this context, the term "stem" is defined as we did in Section 2.1.1: We consider lemmas with derivational affixes, e.g., $Aufmerksamkeit_N$ (attention), as stems. This view contrasts with stemming approaches, which would cut off the suffix *-keit*. For the sake of simplicity, the following definitions of derivation rules concentrate on lemmas as input and output; nonetheless, all declarations hold similarly for stems.

The actual derivational operations are incorporated in the form of the transformation functions, t. To apply a transformation function (i.e., to perform a derivation), we pass the input string to be transformed and its inflectional pattern to this transformation function: Given a pair of lemma and inflectional pattern (l, p) as input, a single derivational rule $d = (t, \mathcal{P}_1, \mathcal{P}_2)$ generates a set of possible derivations $L_d(l, p) = \{(l_1, p_1), \ldots, (l_n, p_n)\}$, where $p \in \mathcal{P}_1$ and $p_i \in \mathcal{P}_2$ for all i. Given a set of derivational rules $\mathcal{D}$, we define a binary derivation relation $\rightarrow_{\mathcal{D}}$ between two lemma-pattern pairs that holds if the second pair can be derived from the first one as:

$$\begin{aligned} &(l_1, p_1) \rightarrow_{\mathcal{D}} (l_2, p_2) \\ \text{iff} \quad &\exists d \in \mathcal{D} : (l_2, p_2) \in L_d(l_1, p_1) \end{aligned} \tag{4.2}$$

Derivations can exhibit slight irregularities in the surface realisation (cf. Section 2.1.3), e.g., due to phonological reasons, as in the following derivations: $Sommer_N$ – $sommerlich_A$ ($summer_N$ – $summery_A$), $Sprache_N$ – $sprachlich_A$ ($language_N$ – $linguistically_A$). In the second derivation, the noun-final *-e* is removed, but the derivation rule is in principle the same. In order to utilise merely one rule for both cases, the transformation function t is defined as $t : S \rightarrow \wp(S)$, i.e., as a mapping of a set of strings, S, including stems and affixes, to a set of strings, $\wp(S)$, which represent possible transformations. As a result, this set of transformed strings (possible derivations) can contain zero, one or many elements. Typically, it contains one string. It is empty if the rule is not applicable (e.g., because a string replacement cannot be conducted), and contains several strings whenever the rule is ambiguous, as in the example above.

At the lowest level, t is defined in terms of atomic string replacement operations, i.e., the replacement of prefixes, suffixes, and infixes (possibly with an empty string).

The framework uses the notion of higher-order functions – functions that take other transformations as arguments and return new transformations as results – to succinctly define common derivational operations such as prefixation, suffixation, and stem change (infix changes are the technical view on the derivational operation of stem changes). Basically, for each of these derivational operations, one higher-order function is used. More complex word-formation rules, such as those combining prefixation and suffixation to circumfixation, can be obtained straightforwardly by functional composition.

Transformation functions in HOFM that are applicable to the respective input – i.e., that result in at least one derivative – are *invertible* (for implementation and formal details, cf. Šnajder et al. (2008), Šnajder and Dalbelo Bašić (2008)). As a result, their inverse function can be similarly applied. This property is convenient for the processing of an arbitrary input word, which might already be the product of another derivational process. In such cases, the inverted derivation rule provides information about the lemma's base word. For example, the inverse of a suffixation function can be used to remove the respective suffix from a string, e.g., $Bäckerei_N$ – $Bäcker_N$ ($bakery_N$ – $baker_N$). This property holds similarly for combined transformation functions in a complex derivation rule.

The transformation functions constitute the core of the derivation rules, as they implement the derivation operations. Thus, we will consider them in more detail.

The Transformation Functions. Basically, all string transformations in HOFM are conducted by three primitive transformation functions:

$rpfx(s1, s2)$**:** replace prefix $s1$ by prefix $s2$

$rsfx(s1, s2)$**:** replace suffix $s1$ by suffix $s2$

$rifx(s1, s2)$**:** replace infix $s1$ by infix $s2$

These $r*fx$ functions are functionally complete, i.e., they would suffice to define all derivational processes that can be described with the HOFM framework. However, the purpose of an EDSL is to ease the handling of a domain-specific programming task. For that reason, HOFM contains a set of pre-defined "convenience" functions which make the rule syntax easily understandable, and facilitate the implementation of derivation grammars. Additional functions for language-specific needs can be comfortably implemented by the user.

Table 4.1 outlines the EDSL syntax for some of these higher-order functions, including the primitive *rifx* function. The Table shows the built-in standard, language-independent functions in the HOFM framework. Note that capitalisation for German nouns is a process defined outside derivation; we will return to this point in Section 4.2.1.

Atomic functions are combined by so-called combinators (which are, technically, higher-order functions as well). Three primitive combinators are implemented in HOFM:

t_1 & t_2**:** sequential application; concatenation of transformation functions t_1 and t_2

t_1 .|. t_2**:** logical "or"; t_1 or t_2 is applied (application of t_1 *and* t_2 is also possible)

Function	Description	Example of application
$rifx(s1, s2)$	replace the infix $s1$ by $s2$	$rifx(i, a)(klingen) = \{Klang\}$ *to sound* → *sound*
nul	do nothing	$nul(grau) = \{Grau\}$ *gray* → *gray*
$sfx(s)$	append the suffix s	$sfx(heit)(frech) = \{Frechheit\}$ *insolent* → *insolence*
$pfx(s)$	prepend the prefix s	$pfx(ver)(fallen) = \{verfallen\}$ *to fall* → *to expire*
$dsfx(s)$	delete the suffix s (always combined with other functions)	$(sfx(in)\,\&\,dsfx(e))(Kunde) = \{Kundin\}$ *client* → *female client*

Table 4.1: Exemplary language-independent transformation functions in HOFM, and their application

Function	Description	Example of application
$try(t)$	perform transformation t, if possible, else do nothing: t .||. *nul*	$(sfx(sam)\,\&\,try(dsfx(e)))(Ehre) = \{ehrsam\}$ *honor* → *honorably*
$opt(t)$	perform transformation t optionally: t .|. *nul*	$(sfx(haft)\,\&\,opt(sfx(en)))(Traum) =$ $\{traumhaft, traumenhaft\}$ *dream* → *dreamlike*

Table 4.2: Built-in combinators for transformation functions in HOFM, and their application

t_1 .||. t_2**:** logical "xor"; t_2 is only applied if t_1 fails (either t_1 or t_2 is applied)

Functional composition by concatenation (&) is executed from right to left. For instance, in the rule using *dsfx* in Table 4.1, the *dsfx* operator is applied before the *sfx* attachment. For the "or" operator (.|.), the order is arbitrary, while "xor" (.||.) is executed from left to right by definition. Again, HOFM provides two "convenience" variants for combinators, *try* and *opt*. They can be used as shortcuts for "xor" and "or", if the alternative transformation t_2 is *nul*. In other words, the input string remains unchanged if t_1 is not applicable. Both *try* and *opt* are shown in Table 4.2, and we will further explain them in Section 4.1.3. In the following, we will treat the three combinators (&, .|. and .||.) like functions.

4.1.3 Instantiation of the Derivation Rules

The transformation functions combined with the inflectional patterns, $\mathcal{P}_1$ and $\mathcal{P}_2$, instantiate a derivation rule in HOFM. In this Section, we illustrate with a couple of examples,

how practical instantiations look like in terms of t, $\mathcal{P}_1$, and $\mathcal{P}_2$.

The previous Section gave a comprehensive description of how t can be concretised. As to the inflectional patterns, we instantiate them for now in a fairly simple way: We describe $\mathcal{P}_1$ and $\mathcal{P}_2$ by the part of speech of the respective input and output word.

For example, a straightforward prefixation rule for the derivation $Preis_N$ – $Aufpreis_N$ ($charge_N$ – $surcharge_N$) is:

$$\big(pfx(auf), \mathcal{N}, \mathcal{N}\big) \tag{4.3}$$

where *pfx* is a transformation function for prefixation as defined in Table 4.1, while the two part of speech sets for base and derived lemma are set to nouns.

Infix replacement, which is used to model stem alternation, can be realised as in the following rule for, e.g., $setzen_V$ – $sitzen_V$ ($to\ set_V$ – $to\ sit_V$):

$$\big(rifx(e, i), \mathcal{V}, \mathcal{V}\big) \tag{4.4}$$

where *rifx* is the function for stem alternation (note that the stem is *setz*, which is why the second, verb-infinitive e is not replaced), and $\mathcal{V}$ is the part of speech set for verbs, used for both base and derived word.

To account for grammar ambiguities and semi-regularities as briefly described in Section 4.1.2, the higher-order functions *try* and *opt* are used to model conditional transformation and optionality, respectively. They are useful when a derivation rule processes two distinct base lemmas in a slightly different, but still regular way. For example, one can establish one single adjectivisation rule for $Aktion_N$ – $aktiv_A$ ($action_N$ – $active_A$) as well as $Instinkt_N$ – $instinktiv_A$ ($instinct_N$ – $instinctive_A$), although the former lemma pair involves the elision of the suffix string *-on*, while the latter does not. The corresponding rule is defined in (4.5):

$$\big(sfx(iv)\ \&\ try(dsfx(ion)), \mathcal{N}, \mathcal{A}\big) \tag{4.5}$$

where *sfx* and *dfsx* are functions for suffixation and suffix deletion, respectively, as defined in Table 4.1, and the part of speech of the base word and derivative must be noun and adjective, respectively. The argument of *try* (i.e., the *-ion* suffix deletion) is executed whenever possible (i.e., whenever a noun ends with *-ion*); otherwise, the input string remains unchanged. The framework executes the &-concatenated transformation functions in the following order: First, it tries to conduct the deletion, then it conducts the suffixation.

In contrast, the optionality function *opt* both executes and ignores the transformation denoted by its argument, which typically leads to multiple (i.e., ambiguous) derivations. *opt* is useful when the application of a derivation rule sometimes requires an additional, regular change. A respective example is shown in Table 4.2: The *-en* suffixation is once applied and once not applied, leading to two derivatives: the correct adjective *traumhaft*, and the incorrect **traumenhaft*. The correctness of the two variants is swapped when the rule is applied to, e.g., $Held_N$ – $*heldhaft_A/heldenhaft_A$ ($hero_N$ – $heroic_A$): Now, the

-en suffixation is required to derive a correct German lemma. That is, although the *opt* transformation is ambiguous, not all produced derivations need to be valid lemmas. With an appropriate word list at hand, incorrect derivations like **traumenhaft* and **heldhaft*, can be filtered out easily. Sections 4.1.4 and 4.2.3 describe how we avoid such overgenerations.

4.1.4 Induction of Derivational Families

Recall that our goal is to induce derivational families, that is, classes of derivationally related words. In accordance with the procedure proposed by Šnajder and Dalbelo Bašić (2010), we define a derivational family, $\mathcal{DF}$, on the basis of derivational rules as follows.

We are given a set of derivational rules $\mathcal{D}$ which defines binary derivation relations $\rightarrow_{\mathcal{D}}$ between two lemma-pattern pairs (l_1, p_1), (l_2, p_2) (cf. Section 4.1.2). Let $\mathcal{L}$ denote the set of lemma-pattern pairs of a language. Then, we define the set of *derivational families defined by $\mathcal{D}$ on $\mathcal{L}$* as the equivalence classes of the transitive, symmetric, and reflexive closure $\mathrm{cl}(\rightarrow_{\mathcal{D}})$ of $\rightarrow_{\mathcal{D}}$ over $\mathcal{L}$:

$$\{\mathcal{DF} \mid \mathcal{DF} = \{(l_2, p_2) \in \mathcal{L} \mid (l_1, p_1)\, \mathrm{cl}(\rightarrow_{\mathcal{D}})\, (l_2, p_2)\};\ (l_1, p_1) \in \mathcal{L}\} \tag{4.6}$$

That is, all lemma-pattern pairs which are connected by a sequence of derivation rules are grouped into one derivational family.[2] Inducing the derivational families by means of the transitive closure has a convenient side effect: It is not necessary to implement derivation rules for indirect, "multi-step" derivational relations. As an example, consider the two rules d_{12}, d_{23} which link the lemma-pattern pairs (l_1, p_1), (l_2, p_2), and (l_2, p_2), (l_3, p_3), respectively. The transitive closure over their applications leads to the following sequence: $(l_1, p_1) \rightarrow_{d_{12}} (l_2, p_2) \rightarrow_{d_{23}} (l_3, p_3)$, and therefore to a derivational family containing all three lemma-pattern pairs: $\mathcal{DF} = \{(l_1, p_1), (l_2, p_2), (l_3, p_3)\}$. As can be seen, the transitivity makes a third rule, d_{13}, to link (l_1, p_1), (l_3, p_3) redundant. In other words, a *minimal set* of derivation rules, which in sum describes the transitive closure, is sufficient to connect all lemmas of a derivational family. We call such sequences of rule applications a *derivation rule path.*

Note that in addition to the quality of the rules, the properties of $\mathcal{L}$ play a central role in the quality of the induced families. Concerning its size, a high coverage of the lemma-pattern pairs in the language of interest is important because the transitivity of $\rightarrow_{\mathcal{D}}$ ranges only over lemmas in $\mathcal{L}$, so low coverage of $\mathcal{L}$ may result in fragmented derivational families. However, $\mathcal{L}$ should also not contain erroneous lemma-pattern pairs but highly reliable information, which means that the way how $\mathcal{L}$ is compiled, is crucial: The derivation rules only define *admissible* derivations, which need not be morphologically valid. Therefore, they routinely overgenerate, which reduces the precision of the resulting families (cf. Section 2.2.1): Overgeneration can lead to spuriously merged derivational

[2] Note that building equivalence classes is not the only possibility to construct a derivational family. For instance, one could define families with an upper bound for the amount of sequentially applied derivation rules. However, we employ the simple strategy based on transitivity as well as symmetry and reflexivity for the reasons mentioned in the beginning of this Chapter.

families, notably effected by derivations of short (i.e., morphologically simple) base words. For instance, an incorrect German lemma such as *Mar_N might conflate the derivational families around $Märchen_N$ (fairy tale) and $Marine_N$ (navy) by means of matching derivation rules, although these two families are not morphologically related. This observation is in line with that of other approaches about derivational morphology, e.g., Walther and Nicolas (2011, p5) or Jacquemin (1997), where longer common string sequences are preferred for clustering. To counteract this effect, $\mathcal{L}$ can be used to explicitly filter out derivations that are not attested in the data (cf. the illustration of such a filtering in Figure 4.1). In that way, potential errors can be reduced considerably.

4.2 Building the Lexicon DErivBase

This Section explains how DERIVBASE is compiled concretely for German. We begin with some design decisions with respect to our utilisation of HOFM, i.e., by specifying the incorporated derivational processes, the consideration of inflectional aspects, and the treatment of noun capitalisation (Section 4.2.1). Then, we show how we employ and expand the HOFM transformation functions explained in Section 4.1.2 to specifically implement German derivations. In Section 4.2.3, we describe the origin and preprocessing of the lemma-pattern pairs that we use to induce the derivational families, while Section 4.2.4 illustrates the development cycle for our rules, and shows some statistics about the resulting rule set. Finally, in Section 4.2.5, we indicate some key figures about the resulting lexicon DERIVBASE.

4.2.1 Design Decisions for a German Derivational Morphology

The derivational model proposed by Šnajder and Dalbelo Bašić (2010) is generally language-independent. This means a high degree of flexibility and freedom, but also that some customisation is necessary. For that reason, we need to examine in which way the framework can be meaningfully applied to the particularities of our language of interest, and what we want to cover in our approach. This Section analyses the spectrum of derivational processes we take into account, and explains how we incorporate inflectional information and the German-specific noun capitalisation.

Selection of Included Derivational Processes. As described in Chapter 2, German is a morphologically complex language. In order to properly implement its derivational processes in our rule-based model, we need to consider the particularities of these processes, and decide which phenomena we want to cover. For details about the linguistic foundations of these decisions, please refer to Chapter 2.

To achieve high coverage, we decided to include all existing derivational operations in German (cf. Section 2.2.1), that is: suffixation, prefixation, conversion, circumfixation, and stem changes. As to the considered word classes for derivatives, we concentrate on the three most prominent parts of speech produced by derivation, namely nouns (*to understand* → *the understanding*), verbs (*the shelf* → *to shelve*), and adjectives (*the*

size → *sizable*). Similarly, we restrict the base lemmas to these three word classes. We ignore adverb derivation for the reasons mentioned in Sections 2.2.1 and 2.2.2: German distinguishes between adverbs and adverbial adjectives (Schiller et al., 1999). The former, e.g., $bald_{ADV}$ ($soon_{ADV}$), constitute a rather unproductive class for derivation, and is therefore of no interest, while the latter, e.g., $schnell_{ADJD}$ ($quickly_{ADV}$), are structurally identical to attributive adjectives, which are already included in the adjective derivation.

As can be seen, our choice of covered German derivational operations involves affixation as well as stem transformation. This is a crucial technical aspect to be taken into account by implementing both string manipulation types. Fortunately, as shown in Section 4.1.2, HOFM fulfils both requirements, so that we can implement all admissible German derivation operations. In this way, we offer a uniform and comprehensive approach to derivation. In contrast, many other studies about derivational morphology ignore some affixation processes or the stem transformation (cf. Chapter 3).

To obtain linguistic definitions for the derivations we chose, we relied on traditional grammar books and lexicons, i.e., Hoeppner (1980), Augst (1975) and Fleischer and Barz (2007) (cf. Section 2.3). We use their descriptions of the derivational processes in order to linguistically justify our decisions during the rule implementation. Additionally, we looked up derivational information in two established online information sources: grammis[3], the grammar information system of the Institute of German Language, which essentially contains the derivational information provided in Donalies (2005), and canoo[4], an implementation of the Word Manager system mentioned in Section 3.1.2.

Integration of Morphological and Semantic Information. As shown in Section 4.1.2, a derivation rule needs to be given the inflectional patterns for the input and output word, $\mathcal{P}_1$ and $\mathcal{P}_2$. These patterns trigger the actions (if any) of the inflectional component. For that reason, we need to determine with which information we instantiate the inflectional patterns.

For German, our study assumes that the inflectional patterns are combinations of parts of speech for the base and the derived word, including infinitive suffix information for verbs. Additionally, we found that purely inflectional information, as proposed by the HOFM framework, does not fully reflect the dimension of German derivation, but that the noun gender is also important: It provides crucial semantic information which helps increasing the precision of many noun-involving derivation rules (Section 4.2.2 will show such cases). For that reason, we use a total of seven inflectional patterns:

$$\mathcal{N}_f,\ \mathcal{N}_m,\ \mathcal{N}_n,\ \mathcal{V}_{en},\ \mathcal{V}_{eln},\ \mathcal{V}_{ern},\ \mathcal{A} \tag{4.7}$$

where the noun subscripts define the required noun gender, and the verb subscripts define the required verb infinitive suffix, respectively, to match a given input or output lemma-pattern pair against the rule. This degree of granularity is sufficient to adequately represent the German content words we are interested in for derivation. We will elaborate

[3] `http://hypermedia.ids-mannheim.de/index.html`
[4] `http://www.canoo.net`

that in the following.

As we apply HOFM to a list of German *lemmas* (cf. Section 4.2.3), i.e., uninflected words, inflection is generally not an issue. However, as mentioned in Section 4.1.2, the derivational rules can be defined on the basis of lemmas as well as stems in four different combinations. Thus, we need to determine with which of these we implement our German derivation rules.

In fact, most German derivational processes across all parts of speech apply to a word's stem (Donalies, 2005, p47f). Due to this linguistic motivation, we generally favour stem-based derivation rules. More specifically, we mainly choose the *stem-to-lemma combination* for input and output word, respectively: An input lemma is reduced to its stem by the inflectional component. Then, the derivational component applies a rule and derives a new lemma from this stem, which can be compared with the lemmas in $\mathcal{L}$. For nouns and adjectives, lemma and stem are essentially identical (according to our definition in Section 2.1.1), and no actual processing through the inflectional component is necessary: The inflectional component simply passes the untransformed lemma to the derivational component and declares it to be a stem. In contrast, most deverbal derivations use the verb stem as base, which is not identical to the lemma and needs to be generated accordingly: $rufen_V$ – Ruf_N (*to* $shout_V$ – $shout_N$). Thus, the question how to transform our lemmas to stems boils down to the question how the inflectional verb infinitives are processed.

In order to correctly capture the verb infinitives, we have defined the inflectional patterns for verbs as just described: Each verb infinitive suffix is represented by a separate inflectional pattern, $\mathcal{V}_{en}$, $\mathcal{V}_{eln}$, $\mathcal{V}_{ern}$. We implemented simple procedures in the inflectional component to appropriately detach each infinitive suffix. Then, the produced stems are handed over to the derivational component which applies the derivation rules. In sum, the elision of verb infinitive suffixes is the only part in our German derivational grammar where inflection comes into play. This may be different for other, morphologically more complex languages, such as Croatian (Šnajder, 2014), or approaches where inflected words serve as input.

As an example, consider the stem-to-lemma rule (4.8) for derivations such as $rechnen_V$ – $Rechnung_N$ (*to* $calculate_V$ – $calculation_N$):

$$\big(sfx(ung), \mathcal{V}, \mathcal{N}\big) \tag{4.8}$$

The inflectional suffix *-en* is removed by the inflectional component, and it needs not be handled in the derivation rule. This effect is also shown in the *rifx* example in Table 4.1.

Whenever it is linguistically plausible, we also utilise *stem-to-stem* and *lemma-to-lemma* derivation rules: For the derivation of verbs, we use stems for both the base and the derived word. The transformation from/to a lemma to match our lemma-pattern pairs in $\mathcal{L}$ is again conducted by the inflectional component. For instance, the rule in (4.9) used to derive $aktiv_A$ – $aktivieren_V$ ($active_V$ – *to* $activate_V$) shows that the inflectional suffix of the derived verb is handled outside the rule:

$$\big(sfx(ier), \mathcal{A}, \mathcal{V}\big) \tag{4.9}$$

Only for conversions from infinitive verbs as bases (cf. Section 2.2.1), we apply lemma-to-lemma rules, e.g., in $reden_V$ – $Reden_N$ ($to\ talk_V$ – $talking_N$):

$$(nul, \mathcal{V}, \mathcal{N}) \tag{4.10}$$

Treatment of German Noun Capitalisation. Finally, we need to consider the German idiosyncrasy of capitalising nouns. Noun capitalisation is easy to implement if the part of speech is given (which is the case in our study). However, it is not part of a derivational process, although capitalisation must of course be considered when implementing noun-involving derivation rules.

For such cases, HOFM offers the transformation module (cf. Section 4.1.1): It aggregates important language-specific particularities that are independent of derivation or inflection. Thus, implementing a straightforward uppercasing function in this module is sufficient to handle noun capitalisation.

4.2.2 Implementation of German Derivation Rules in HOFM

This Section describes how we actually implement and instantiate our derivation rules to induce a German derivational lexicon. We employ the standard transformation functions of HOFM as explained in Section 4.1.2 (cf. Table 4.1, and Table 4.2), and additionally expand the framework with two language-specific transformation functions for German, *uml* and *dup*, as shown in Table 4.3. In the following, we specify these language-specific functions and their purpose. Then, we illustrate some rule implementation principles to obtain maximally precise derivation rules while maintaining high coverage. Finally, we outline general insights we gained during the rule implementation procedure.

Function	Description	Example of application
uml	alternate an infix as umlaut shift: $uml = aifx([(a, \ddot{a}), (o, \ddot{o}), (u, \ddot{u})])$	$uml(krumm) = \{krümmen\}$ *curved* → *to curve*
dup	duplicate the last consonant (always occurs with infix changes)	$(aifx(ei, i)\ \&\ dup)(schreiten) = \{Schritt\}$ *to step* → *step*

Table 4.3: Exemplary German-specific transformation functions in HOFM, and their application

Language-specific Transformation Functions. The by far most frequent German stem alternation is the umlaut shift. This special infix replacement is exemplified in the following nominalisation rule for, e.g., $vermacht_A$ – $Vermächtnis_N$ ($bequethed_A$ – $bequest_N$):

$$(sfx(nis)\ \&\ try(uml), \mathcal{A}, \mathcal{N}) \tag{4.11}$$

where *uml* and *sfx* are functions for umlauting an infix vowel and suffixation, respectively, while *try* ensures that the umlaut alternation is performed whenever possible. $\mathcal{A}$ and $\mathcal{N}$ are the part of speech sets for adjectives and nouns for base and derived word, respectively. The umlaut shifting function, *uml*, is German-specific, and facilitates the implementation and readability of corresponding derivation rules significantly: As Table 4.3 specifies, an umlaut shift covers three different infix replacements ($a \rightarrow \ddot{a},\ o \rightarrow \ddot{o},\ u \rightarrow \ddot{u}$), of which only one is applied at once. That is, *uml* subsumes three individual processes which would otherwise need to be implemented in three different derivation rules. We implement two versions of *uml*, one for infix vowels as above, and one for initial vowels (*puml*), e.g., in Arm_N – $\ddot{A}rmel_N$ (arm_N – $sleeve_N$).

The second transformation function that we implemented particularly for German, *dup*, causes the duplication of stem-final consonants. Such duplications only apply to synchronically unproductive stem changes (cf. Section 2.2.1), like $schleifen_V$ – $Schliff_N$ ($to\ sharpen_V$ – $sharpening_N$):

$$\big(rifx(ei, i)\ \&\ dup, \mathcal{V}, \mathcal{N}\big) \tag{4.12}$$

Nonetheless, we include these cases to maximise the size of our derivational families and to avoid fragmentation (cf. Section 2.2.1). Obviously, the duplication transformation is somewhat context-sensitive, i.e., an *unspecified* consonant is duplicated. Although this transformation can be realised with the standard higher-order transformation functions, its syntax would be complicated. Thus, we added *dup* to the set of transformation functions.

These two language-specific transformations, *dup* and *uml*, allow for an appropriate treatment of virtually all German derivation processes. The full set of transformation functions we have used to induce DERIVBASE is listed in Appendix A. Note that the ablaut shift is not represented as a proper transformation function, as it is both essentially less frequent and less systematic. Instead, ablaut stem changes are realised on demand with *rifx* functions.

Implementation Principles for Maximal Coverage and Precision. We found a number of derivational rules in German to be conceptually simple (e.g., verb-noun conversion) so that a substantial amount of derivations can already be covered with a handful of trivial transformation functions. However, there are many complex rules (e.g., suffixation combined with optional stem changes) that in sum affect a considerable number of lemmas, which required us to either implement low-coverage rules or generalise existing rules to achieve the desired level of coverage.

In order to maintain high precision, we restricted the rule application by using *try* instead of *opt* whenever applicable. For example, the following rule:

$$\big(sfx(in)\ \&\ opt(dsfx(e)), \mathcal{N}, \mathcal{N}\big) \tag{4.13}$$

to derive female person-denoting nouns such as $Chef_{Nm}$ – $Chefin_{Nf}$ ($boss_N$ – $female\ boss_N$) and $Türke_{Nm}$ – $Türkin_{Nf}$ ($Turk_N$ – $female\ Turk_N$), can lead to incorrect applications by

retaining the suffixal *-e*, e.g., in the derivation of a proper noun in $Husse_N$ – $Hussein_{NE}$ ($slipcover_N$ – $Hussein_{NE}$), or the derivation of an unrelated lemma in $Geste_N$ – $Gestein_N$ ($gesture_N$ – $rock_N$). To avoid these errors, the rule in (4.13) can be made more selective by using *try* instead of *opt*:

$$\big(sfx(in)\ \&\ try(dsfx(e)), \mathcal{N}, \mathcal{N}\big) \tag{4.14}$$

Additionally, we defined three language-specific limitations concerning the inflectional patterns $\mathcal{P}$:

1., we use gender information from the noun patterns to narrow down the scope of noun-involving derivations: Some rules involving nouns only apply to specific genders (cf. Section 2.2.1). For example, deriving female person-denoting nouns as shown in (4.14), requires a masculine noun as base word and a female noun as derived word:

$$\big(sfx(in)\ \&\ try(dsfx(e)), \mathcal{N}_m, \mathcal{N}_f\big) \tag{4.15}$$

Note that such a treatment generally requires that gender information is available by adequate preprocessing. In order to also accept lemmas for which the gender is unknown (e.g., due to preprocessing issues), we design such constraints generously towards lemmas without gender information. For example, the following rule:

$$\big(sfx(n), \mathcal{N}_m, \mathcal{N}_n\big) \tag{4.16}$$

is used for derivations like $Schwede_{Nm}$ – $Schweden_{Nn}$ ($Swede_N$ – $Sweden_N$), but also accepts nouns without gender specifications, e.g., the nominalisation $Verdächtige_{Nm}$ – $Verdächtigen_N$ ($suspect_N$ – $suspect_N$), where the neuter gender of the derivative might be unknown due to limitations of the preprocessing.

2., we use information about the infinitive suffixes from verb patterns: Some verbalisation rules only produce verbs of a specific infinitive suffix type. For instance, the noun-verb derivation realised by the suffix *-ig* only applies to verbs of the (most frequent) suffix type *-en* (Fleischer and Barz, 2007, p310f), e.g., $Nacht_N$ – $nächtigen_V$ ($night_N$ – $to\ overnight_V$):

$$\big(sfx(ig)\ \&\ try(dsfx(e))\ \&\ opt(uml), \mathcal{N}, \mathcal{V}_{en}\big) \tag{4.17}$$

Here, it is not the transformation functions which mainly restrict the applicability of rule (4.17), but the inflectional pattern $\mathcal{P}_2$, by only allowing for the inflectional infinitive suffix *-en*. Since the verb suffix information is always available, there is no need to relax this constraint for unspecified cases, as we did for the noun gender.

3., we take into account that many prefixations are, linguistically speaking, no transpositions and therefore do not change the part of speech (cf. Section 2.2.1). This property propagates to the noun gender as well as the verb-denoting suffix; both remain unchanged by prefixations. For such cases, we require to retain (*align*) the noun gender and the infinitive suffixes, respectively, between base lemma and derivative. Consider the following

rule:

$$align\big(pfx(um), \mathcal{N}, \mathcal{N}\big) \tag{4.18}$$

The *align* function ensures that this rule is only applied if the gender of the input and the output noun are identical. In other words, it essentially acts like a coindexation on the noun gender. Without this "wrapper", it would be necessary to define three rules instead of the rule in (4.18), one for each gender combination as inflectional patterns, $\mathcal{P}_1 \rightarrow \mathcal{P}_2$ (i.e., $N_f \rightarrow N_f$, $N_m \rightarrow N_m$, $N_n \rightarrow N_n$). Thus, *align* makes the rule grammar more concise: Rule (4.18) keeps the feminine gender in $Welt_{Nf}$ – $Umwelt_{Nf}$ ($world_N$ – $environment_N$), as well as the masculine gender in Bau_{Nm} – $Umbau_{Nm}$ ($construction_N$ – $conversion_N$). As in the first limitation (1.) above, the alignment restriction accepts lemmas without gender specification, e.g., $Siedeln_{Nn}$ – $Umsiedeln_N$ ($settling_N$ – $resettling_N$).

The preference of *try* over *opt*, and these three constraints allow us to narrow down the set of derivable lemmas. As a result, we end up with high-coverage rules, such as derivations of person-denoting nouns ($Schule_N$ – $Schüler_N$ ($school_N$ – $pupil_N$)) as well as high-accuracy rules such as non-transpositional negation prefixes (Pol_{Nm} – $Gegenpol_{Nm}$ ($pole_N$ – $antipole_N$)) which overgenerate as little as possible.

Insights from the Rule Implementation Procedure. Even though we did not focus on the explanatory relevance of rules, we found that the underlying modelling formalism and the methodology used to develop our model, offer substantial linguistic plausibility in practice. The framework seems well-suited for German, because most derivation rules involving Germanic affixes are largely regular and could be implemented straightforwardly. In contrast, derivational transformations that are motivated by Latin or Greek morphology cause less regular elisions or replacements. For example, consider the Latin *de-* prefixation for nouns: $Kompression_N$ – $Dekompression_N$ ($compression_N$ – $decompression_N$), $Interesse_N$ – $Desinteresse_N$ ($interest_N$ – $disinterest_N$). If the base lemma starts with a vowel, the prefix expands to *des-*. We realise this idiosyncrasy with the following rule:

$$\big(pfx(de)\ \&\ try(apfx([(a, sa), (e, se), (i, si), (o, so), (u, su)])), \mathcal{N}, \mathcal{N}\big) \tag{4.19}$$

where we try (*try*) to apply the prefix alternation function (*apfx*) to word-initial vowels, before appending the actual derivational prefix (*pfx*). Appendix B shows that such rules can become much more complicated. In fact, only the combination of Hoeppner (1980), Augst (1975) and Fleischer and Barz (2007), enriched with information from grammis and canoo (cf. Section 4.2.1), lead to a maximal inventory of such idiosyncrasies and semi-regularities, which shows that it is challenging to cover all particularities which can occur in derivational processes.

Most rule descriptions in the mentioned grammar books have the same compactness as our rule implementations in HOFM, i.e., for one rule in the literature, we implemented one rule in the framework. However, the linguistic rules are sometimes more compact than we found it appropriate in HOFM. In other words, we occasionally implemented

several rules for one rule description in the textbooks. The reason is mainly due to the separation of inflectional and derivational aspects in HOFM: The literature typically summarises semi-regular derivations into one rule. Since this is sensible from a theoretic point of view, we generally do the same, but if inflection is involved, such a conflation can reduce precision. For instance, the *ver-* prefixation to derive verbs is applicable for all inflectional verb suffixes: $Klage_N$ – $verklagen_V$ ($suit_N$ – *to* sue_V), $Kabel_N$ – $verkabeln_V$ ($wire_N$ – *to* $wire_V$), $Knochen_N$ – $verknöchern_V$ ($bone_N$ – *to* $ossify_V$). However, only verbs with the *-ern* suffix can (optionally) experience an umlaut shift in the stem. If our rules would not reflect this restriction, they would overgenerate. As a consequence, if the given set of lemma-pattern pairs $\mathcal{L}$ contained spurious lemmas such as **verklägen*, we would end up with this incorrect lemma in the derivational family of $Klage_N$. Thus, we split the *ver-* prefixation into several rules which account for the differences of the inflectional patterns.

In sum, the implementation of the derivation rules was a fairly straightforward task, and the major hurdle was to synchronise the information of the different linguistic textbooks.

4.2.3 Data and Preprocessing

To induce derivational families containing nouns, verbs, and adjectives, we need a lemma list with part of speech information. This information can be gathered from a German text corpus. For an accurate application of nominal derivation rules, i.e., to distinguish masculine, female and neuter nouns for gender-specific derivations, we additionally require gender information. We use a part of speech-tagged and lemmatised version of SDEWAC, a large German-language web corpus containing about 880 million words, from which boilerplate paragraphs, ungrammatical sentences, and duplicate pages were removed (Faaß and Eckart, 2013). For part of speech tagging and lemmatisation, we use TreeTagger (Schmid, 1994) and determine the grammatical gender with the morphological layer of the MATE toolkit (Bohnet, 2010).[5] We decided to combine these two preprocessing tools due to an analysis of a small tagged sample: TreeTagger returns less, but more reliable lemmas and parts of speech than MATE, which is important for the induction. From the corpus preprocessed in this way, we consider only nouns (named entities are also treated like common nouns), verbs, and adjectives for the lexicon induction.

In order to additionally increase the correctness of the lemma-pattern pairs which we pass to the derivational rules, we apply three language-specific filtering steps; these filters are largely based on our observations of characteristics of German nouns, verbs, and adjectives described in Section 2.2.2: First, we discard non-capitalised nominal lemmas. Second, we delete verbal lemmas not ending in verb suffixes. Third, we remove frequently occurring comparative forms of adjectives which remained unlemmatised (usually formed by the suffix *-er*, like *neuer / newer*) by checking for the presence of lemmas without *-er* (*neu / new).*

An additional complication in German concerns prefix verbs, because the prefix can

[5]The morphological gender information is not always available: For about 4% of the nouns, MATE leaves the gender unspecified.

be separated in tensed instances. For example, the 3rd person male singular of *aufhören (to stop)* is *er hört auf (he stops)*. Since most prefixes double as prepositions, the correct lemmas can only be reconstructed by parsing. We parse the corpus using the MST parser (McDonald et al., 2006) and recover prefix verbs by searching for instances of the dependency relation labelled PTKVZ, and re-attaching them to their verbs.

Since SDEWAC, as a web corpus, still contains errors, we only take into account lemma-pattern pairs that occur at least three times in the corpus, a threshold recommended in Evert (2005). This threshold corresponds to 9.8 parts per billion (i.e., 0.0098ppm) of all nouns, verbs and adjectives in the tagged SDEWAC corpus. Given this small ratio, we consider this a conservative filtering step that preserves high recall and provides a comprehensive lemma basis for evaluation. We believe that requiring three instances per lemma-pattern pair is a solid balance between coverage and correctness, which is necessary for a sensible set of lemma-pattern pairs $\mathcal{L}$ for inducing derivational families (cf. Section 4.1.4).

After preprocessing and filtering, we finally run the actual induction as explained in Section 4.1.4 to obtain the DERIVBASE lexicon. Note that the data and preprocessing explained in this Section is also the basis for various data samples that we will use in the following. An overview of all these samples is shown in Figure 4.5.

4.2.4 Rule Development Cycle and Quantitative Rule Analysis

The implementation of the derivation rules that are applied in the lexicon induction took place in several working phases. The following paragraphs explain this development cycle as well as provide statistics about the resulting rule set.

Rule Development Cycle. In the initial rule development phase for DERIVBASE, we implemented 154 rules, which took about 22 person-hours (including familiarisation with HOFM). We then revised the rules in two iterations, with the aim of increasing both precision and recall. To assess the quality and coverage of our rules and to revise them accordingly, we use the lemma list extracted from SDEWAC as explained in Section 4.2.3.

In the first iteration, we were mostly concerned with precision. Based on the set of 154 initial rules, we performed an assessment phase: We sampled a development set comprising 1,000 derivational families induced with our initial rule set, using stratified sampling (i.e., we explicitly picked families of different size). On this set, we inspected the derivational families for false positives, identified the problematic rules, and discovered rules which are not necessary for the minimal rule set which describes the transitive closure, and are therefore redundant. In order to identify the false negatives, we additionally sampled a list of 1,000 lemmas from these 1,000 families, and used string distance measures (cf. Section 4.3.2) to retrieve the 10 most similar words for each lemma not already covered by the derivational families. The refinement process took another 8 person-hours. It revealed three redundant rules and seven missing rules, leading to a set of 158 rules.

The second iteration was more focused on recall. We implemented a large number of additional rules, most of which are prefixation rules. While, in the first iteration, we

Operation	N-N	N-A	N-V	A-A	A-V	V-V	*Total*
Conversion	–	1	4	–	–	–	5
Prefixation	36	–	14	22	10	31	113
+ Stem change	–	–	5	–	3	–	8
Suffixation	20	37	24	1	16	–	98
+ Stem change	7	10	7	–	4	1	29
Circumfixation	–	–	2	–	–	–	2
+ Stem change	1	–	1	–	–	–	2
Stem change	–	1	7	–	–	2	10
Total	64	49	64	23	33	34	267

Table 4.4: Breakdown of derivation rules in DERIVBASE v.1.4.1 by their derivational operation, and by part of speech of base and derived word

concentrated on derivations which maintain the base word's meaning, we now decided to integrate all admissible derivations, in order to reflect the entire spectrum of German derivational processes. Furthermore, we refined a couple of rules because we found them to systematically miss some words, and removed one erroneous rule.[6] In total, the final rule set contains 267 rules.[7] It can be found in Appendix B. The second rule implementation iteration took 15 person-hours. That is, about one working week is enough to establish a well-chosen, revised set of derivation rules for a derivationally productive language.

These two iterations result in two different releases of the lexicon, containing differently shaped derivational families. In the following, we refer to the result of the second iteration, counting 267 rules, and being released, as DERIVBASE v1.4.1. For details about the version of the first iteration, DERIVBASE v1.2, please consult Zeller et al. (2013).

Quantitative Rule Analysis. To give a more detailed picture of the derivational processes covered by DERIVBASE v1.4.1, Table 4.4 shows the distribution of rules with respect to the derivational operations they implement and the part of speech combinations for the base and the derived word. All affixations occur both with and without stem changes, mostly umlaut shifts. Noun-verb derivation is most diverse in terms of derivational operations. Prefixation and suffixation are by far the most frequently used derivation

[6] The removed rule was the *-land* suffixation. Some derivation grammars like canoo denote this word formation a derivation, which is why we implemented the corresponding rule. However, we finally decided that it is a clear case of compounding.

[7] Unfortunately, it is not easy to compare this number with the the amount of corresponding rules in the grammar books, because each of them covers merely a part of the phenomena and necessary information (cf. Section 2.3).

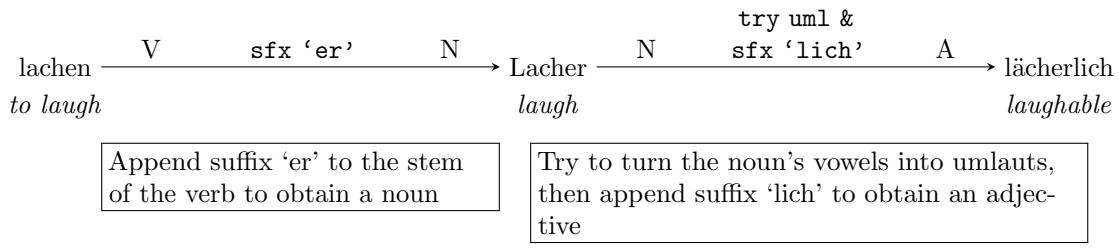


Figure 4.2: Part of a derivational family from DERIVBASE including derivational rules

operations, as well as applied for most part of speech combinations. The conditional transformation, *try*, is mainly used to handle semi-regular adaptations of the base lemma, such as the elision of stem-final characters (50% of cases). The optional transformation, *opt*, is used for umlaut shifts in over 50% of cases. As can be seen from Table 4.4, derivational rules that cross part of speech boundaries are frequent: More than half of the rules derive a lemma with a word class different from that of the base lemma. Such links are particularly valuable, because cross-part of speech information is unavailable in many traditional resources.

During the assessment phase of our development cycle, we gained the impression that the rules differ in their quality: Some rules lead more often to invalid derivations than others. This intuition correlates, on the one hand, with similar findings in the literature (e.g., Šnajder and Dalbelo Bašić (2008, p116)), and on the other hand with our assumption mentioned at the end of Section 2.2.1: We find that stem-changing derivation rules produce errors more often than, e.g., prefixation rules. A separate evaluation of more and less qualitative rules would therefore be interesting. We will return to this point in Section 4.4.1.

Figure 4.2 illustrates a small extract from a derivational family in DERIVBASE with three lemmas induced by two derivational rules, one turning a verb into the corresponding event noun (in this case a semelfactive), and one turning the event into an adjective associated with it. All three lemma pairs are indeed derivationally related. The *derivation rule path* (cf. Section 4.1.4) from $lachen_V$ to $lächerlich_A$ requires two rules, i.e., it has a length of 2.

4.2.5 Statistics on DErivBase

The preparation of the SDEWAC corpus as explained in Section 4.2.3 yields 280,336 lemmas, which we cover with our lexicon. For DERIVBASE v1.4.1 (i.e., using 267 derivation rules, cf. Section 4.2.4), we induced a total of 228,203 derivational families from these lemmas, with 69,437 lemmas being grouped into 17,314 non-singleton families (i.e., 210,889 lemmas are singleton families). For comparison, CatVar v2.1, the only

available derivational lexicon we are aware of, contains 82,676 English lemmas, 44,072 of them being clustered into 13,368 families, and 38,604 of them being singletons.

To get an impression of an entire derivational family, consider the following sample family, also taken from DERIVBASE, which has seven members across all three parts of speech and includes prefixation, suffixation, and infix umlaut shifts:

> $taub_A$ (*numb*$_A$), $Taubheit_{Nf}$ (*numbness*$_N$), $betäuben_V$ (*to anesthetise*$_V$), $Betäubung_{Nf}$ (*anesthesia*$_N$), $betäubt_A$ (*anesthetised*$_A$), $Betäuben_{Nn}$ (*act of anesthetising*), $betäubend_A$ (*anesthetic*$_A$)

All lemmas are morphologically related and, at least with one sense, semantically close to one another.

We examined the families concerning their size. Interestingly, out of the 210,899 lemmas which end up as singletons in DERIVBASE (i.e, for which no derivationally related lemmas were found), 93% are nouns. We contribute this fact to the strong tendency to noun compounding in German. Indeed, in a sample of 100 randomly selected singleton nouns, 84% are compound nouns, such as $Knoblauchduft_{Nm}$ (*smell of garlic*) or $Waldeinsamkeit_{Nf}$ (*woodland solitude*). The remaining singleton nouns are named entities (11%) – a word class which of course occurs frequently in web corpora –, normal nouns (3%), and lemmatisation errors (2%). Conversely, this means that the non-singleton families in DERIVBASE seem to largely cover all German non-compound nouns. Examining the relation between derivational families (e.g., *rain, to rain, rainy*) and compositionally related words (e.g., *raincoat*) would be an additional topic to investigate, but is out of scope of this thesis.

Figure 4.3 shows the frequencies of different family sizes in DERIVBASE; the distribution closely resembles a Zipfian distribution. Most families (over 99.9%) contain less than 25 lemmas. Note that the y-axis of Figure 4.3 is logarithmic, i.e., the number of derivational families with few members is very high. In fact, 11,105 families consist of two lemmas.

On the other hand, the Figure also shows that there exist some extreme outliers, reaching up to one family which counts 238 lemmas. Surprisingly, this huge family is not a result of overgeneration, but contains only morphologically related (albeit, in parts semantically opaque) words, e.g., $setzen_V$ – $sitzen_V$ – $Satz_N$ *(to sit (sb.)*$_V$ – *to sit*$_V$ – *set*$_N$*)*. Nonetheless, many big induced families unfortunately conflate several smaller actual families. For instance, the 12th largest family with 102 lemmas contains, among others, the morphologically unrelated lemmas $malen_V$ – $mauern_V$ – $Meister_N$ (*to paint*$_V$ – *to build*$_V$ – *master*$_N$).

Creating many over-inflated derivational families is a drawback of the approach we use that was already mentioned by Šnajder and Dalbelo Bašić (2010): The derivation rules generate spurious derivations, e.g., due to application to orthographically similar, but morphologically unrelated stems. For instance, the infix replacement rule shown in Table 4.1 is incorrectly applied to the following lemma pair: $zinken_V$ – $Zank_N$ (*to zinc*$_V$ – *quarrel*$_N$). The transitivity assumption used for constructing derivational families additionally contributes to overinflation. Thus, if very pure derivational families are required, there are two options: Either omitting delicate derivation rules and accepting

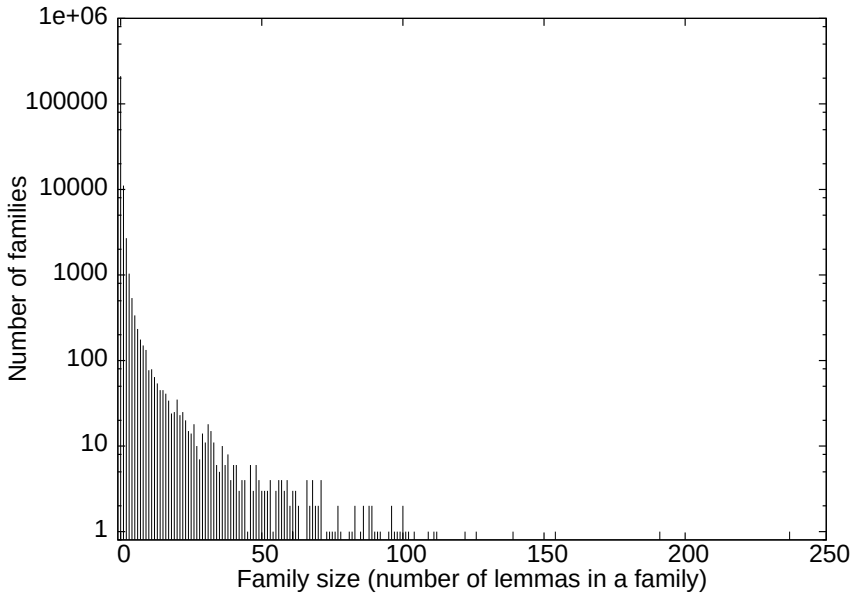


Figure 4.3: Relation of size and number of derivational families

lower coverage, or trying to refine the families by further splitting them. A sensible approach for such a refinement could be, e.g., to reject lemmas according to some semantic similarity measures, or by using information from the internal structure of a family, i.e., the derivation rule paths to connect a lemma pair.

4.3 Intrinsic Evaluation

This Section explains how we evaluate the quality of DERIVBASE. Section 4.3.1 describes the challenges in evaluating derivational families and which evaluation procedure we thus follow. Then, we introduce the baselines against which we qualitatively compare DERIVBASE (Section 4.3.2), and the dataset which we created for this evaluation (Section 4.3.3).

4.3.1 Evaluation Methodology

In this Section, we first explain how we define our evaluation task to assess the quality of DERIVBASE. Then, we describe why the respective datasets cannot be straightforwardly sampled, and how we circumvent these problems.

Evaluation Measures and Format. Generally, the induction of derivational families could be evaluated globally as a clustering problem. Unfortunately, cluster evaluation is a non-trivial task. A number of clustering evaluation metrics have been proposed in the literature. Extrinsic metrics compare the clusters against a gold standard, however, as noted by Amigó et al. (2009), extrinsic cluster evaluation is complicated and there is no consensus on the best approach. Moreover, building a representative gold standard for a cluster-based evaluation of derivational families is in itself not a straightforward task.

For these reasons, we decided to perform our *evaluation at the level of pairs*: We manually judge for a set of lemma pairs whether they are derivationally related or not. That is, we assess our lexicon in a *binary classification task*. Traditional measures used for binary classification are precision, recall, and F_1-score. We employ these measures, and define derivational relatedness as our positive class, i.e., lemma pairs which are derivationally related count as true positives. This procedure produces evaluation results that can be considered an approximation of pairwise F-score, an evaluation measure used for clustering, e.g., by Schulte im Walde (2006) or Hatzivassiloglou and McKeown (1993). Also, our pairwise evaluation based on precision and recall enables us to analyse in more detail the reasons for false positives, i.e., words in a derivational family which are not derivationally related to the other words.

Procedure for the Dataset Sampling. We obtain the gold standard for this evaluation by annotating pairs of lemmas sampled from the SdeWaC lemma list (cf. Section 4.2.3). However, with random sampling, the evaluation would be very unreliable: Most words of a language are not derivationally related. Also, such pairs do not at all exhibit sufficiently high string similarity to be accidentally deemed derivationally related, e.g., $Ente_N$ – $blau_A$ ($duck_N$ – $blue_A$). Consequently, a vast majority of pairs in a random sample would be simply irrelevant for measuring derivational relatedness, as our rules would not match them and correctly recognise them as derivationally unrelated (i.e., true negatives). Moreover, in order to reliably estimate the overall precision of the obtained derivational families, we need to evaluate pairs sampled from these families. On the other hand, in order to assess recall, we need to sample from pairs that are *not* included in our derivational families. That is, we need a method to find, for a given lemma, all lemmas that display a sufficiently high similarity on the string level to be potential candidates for a derivational relation.

To obtain reliable estimates of both precision and recall, we developed a novel evaluation approach specifically designed to assess only subsets of lemmas from a language. Instead of drawing one sample, we draw two different samples: a *P-sample* to estimate precision, drawn from derivational families, and an *R-sample* to estimate recall, drawn from other sources of potentially derivationally related lemmas.

Let us first consider the *P-sample*: We sample a set of lemma pairs from the induced derivational families in DerivBase. As the families contain all lemmas deemed derivationally related in DerivBase (i.e., both true and false positives), sampling from them enables us to estimate precision. A lemma pair (l_1, l_2) is sampled in two steps: First, a lemma l_1 is drawn from a non-singleton family, then the second lemma l_2 is drawn

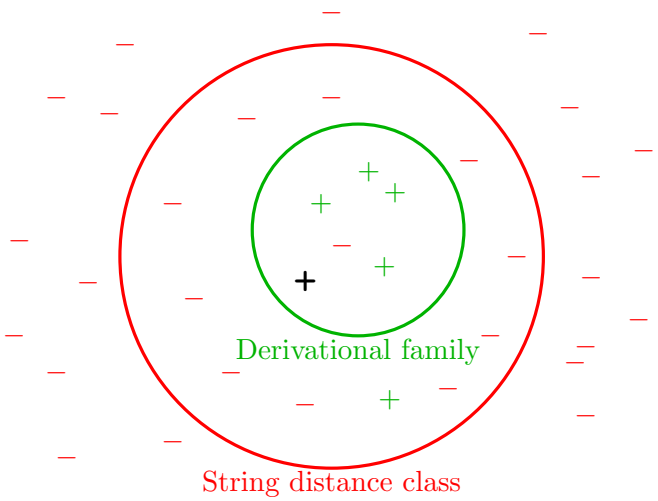


Figure 4.4: The sampling method for the R-sample

from the derivational family of l_1. Due to the rule-based nature of DERIVBASE, the derivational families contain no information about true negatives, which are unnecessary to calculate precision. Thus, the P-sample yields a reliable, focused estimate of precision.

As to the sample to estimate recall, the *R-sample*, we sample a set of lemma pairs from the set of *possibly derivationally related* lemma pairs. Again, a lemma l_1 is drawn from a non-singleton family; then the second lemma l_2 is drawn from the set of lemmas possibly related to l_1. To determine this set, we adopt the conceptual idea introduced by Šnajder and Dalbelo Bašić (2009): They employ two string distance measures to group word forms with relatively similar strings into stem classes. We adopt this approach, using the same string distance measures to achieve a sensibly preselected set of potentially derivationally related lemmas for the R-sample. Figure 4.4 illustrates the intuition behind the idea: For a given lemma l_1 (the bold "+" sign), the set of words with low string distance contains actually related lemmas ("+" signs, mostly contained in the derivational family of l_1), as well as unrelated lemmas ("−" signs within the string distance class). Lemmas with very high string distance to l_1 ("−" signs outside the string distance class) are discarded from the R-sample. Clearly, string distance is not as reliable as our derivation rules, but it gives a sensible approximation to derivation, i.e., it reveals "likely confounders" as well as items missed by DERIVBASE. Such a preselection thus drastically reduces the number of obviously unrelated pairs, which would be classified as (true) negatives from DERIVBASE anyway. In this way, we ensure that the R-sample contains a reasonable number of derivationally related pairs not covered by our lexicon (i.e., false negatives), and returns a reliable estimate of recall. Of course, we cannot guarantee that all actually existing relations are covered. This means that the R-sample measures a *relative recall* of DERIVBASE, i.e., relative to the recall of the string distance measures (Pantel et al.,

2004). As a consequence, evaluations on the R-sample might overestimate recall, but only slightly, so that the estimations are still reliable (provided the R-sample captures almost all true positives).

Let us next describe how we concretely determine the set of possibly related lemmas for the R-sample. We employ the same two string distance measures as Šnajder and Dalbelo Bašić (2009), plus a German stemmer. These three approximations to derivation also serve as baseline methods against which we compare DERIVBASE in Section 4.4; they are explained in more detail in Section 4.3.2. To compile the R-sample, we sample lemmas from the union of the derivational family of l_1, the classes of l_1 obtained with the baseline methods, and k lemmas most similar to l_1 according to the two string distance baseline measures. The addition of these k lemmas is important in two respects: First, sampling only from the lemmas found by the baseline methods would let these methods overfit the data, which is undesirable, as we want to evaluate them as baselines; thus, the addition of k lemmas makes the sampling more robust. Second, we want the R-sample to contain as many potentially derivationally related lemmas as possible to capture all lemmas missed by DERIVBASE. We use $k = 7$ in our experiments; this value is based on preliminary experiments on the development set (cf. Section 4.2.4), which showed that $k = 7$ retrieves about 92% of the related lemmas retrieved for $k = 20$ with a much smaller number of true negatives. In sum, by sampling from this union, the R-sample captures many potential derivational relations without containing overly many simple true negative pairs.

Both P- and R-sample contain 2,400 lemma pairs each. They are further split into three samples to conduct our gold standard annotation, as will be explained in Section 4.3.3. Lemmas included in the development set (Section 4.2.4) were excluded from sampling. Figure 4.5 summarises all samples employed in the induction and evaluation process of DERIVBASE. We will refer to this Figure in the following Sections.

4.3.2 Baselines

We use two baseline types against which we compare the induced derivational families: 1., clusters obtained with the German version of the Porter stemmer (Porter, 1980)[8] and 2., clusters obtained using string distance-based clustering. The purpose of these baselines is to detect derivationally related lemmas that were missed by our rules. To capture all kinds of derivational processes, we use various measures with different characteristics.

The first baseline simply groups all lemmas which can be reduced to the same stem. For instance, the Porter stemmer reduces the following four lemmas to the stem *umfall*: *Umfallen*$_N$ – *Umfall*$_N$ – *umfallen*$_V$ – *umfallend*$_A$ (*the falling over – event of falling over – to fall over – falling over*). Therefore, they constitute a cluster.

Concerning the string distance-based baselines, we have considered a number of string distance measures and tested them on the development set which we used to assess our initial set of derivation rules (cf. Section 4.2.4). More specifically, we experimented with the three measures mentioned in Šnajder and Dalbelo Bašić (2009): The Dice n-gram

[8] `http://snowball.tartarus.org`

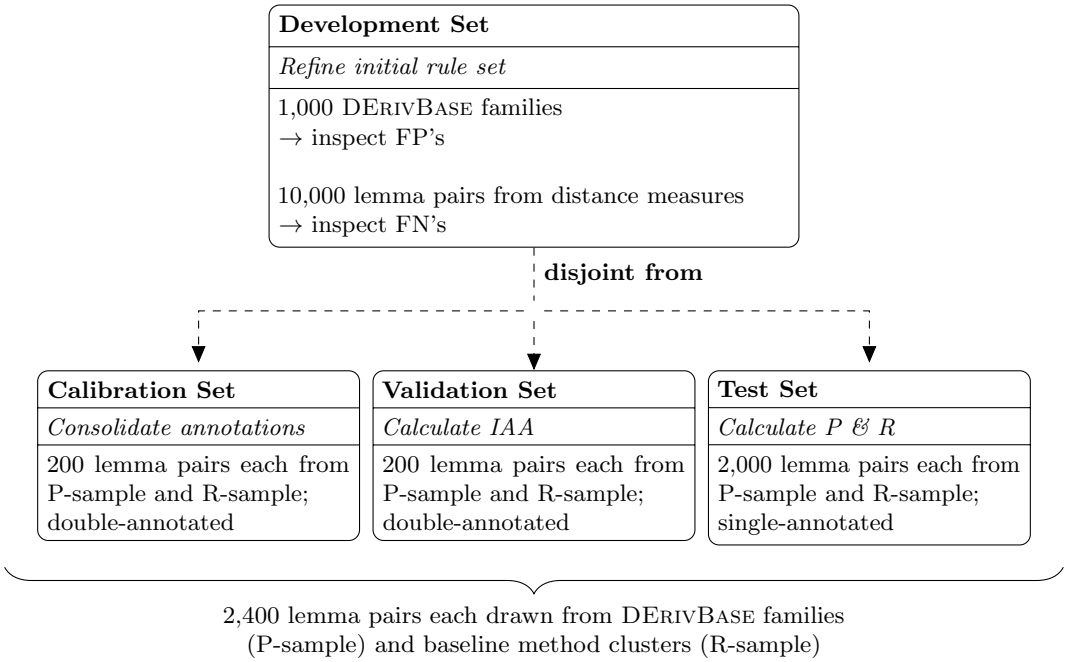


Figure 4.5: Overview of the samples used to induce and evaluate DERIVBASE. The development set is disjoint from the other sets (indicated by the dashed line)

coefficient (Dice, 1945) based on bigrams and trigrams, and two variants of a measure proposed by Majumder et al. (2007); one of these variants favours long matching prefixes, while the other penalises long strings after the first mismatch. All these measures are *distance measures*, i.e., small values for a lemma pair indicate high string similarity.

The latter measure of Majumder et al. (2007) turned out to be the most effective in capturing *suffixal variation*. For words X and Y, it is defined as

$$D_4(X,Y) = \frac{n-m+1}{n+1} \sum_{i=m}^{n} \frac{1}{2^{i-m}} \tag{4.20}$$

where m is the position of the left-most character mismatch, and $n+1$ is the length of the longer of the two strings. If $m = 0$ (i.e., the words X and Y already differ in the first letter), then $D_4 = \infty$. As an example, consider the two lemma pairs *versehen*$_V$ – *verstehen*$_V$ (*to look wrongly – to understand*), and *versehen*$_V$ – *versehentlich*$_A$ (*to look wrongly – by mistake*). For the first, derivationally unrelated pair, $n = 8$, and $m = 5$. Thus, $D_4 = 0.833$. For the second, derivationally related pair, $n = 12$, and $m = 9$, resulting in a lower (less distant) value $D_4 = 0.577$. As these two lemma pairs show, this measure reflects the fact that an appended suffix and, coupled with that, a bigger (Levenshtein-alike) string edit distance do not necessarily imply that the two concerned

lemmas are not derivationally related. For suffixation, such an approach is exactly what we aim at.

To capture *prefixal variation* and *stem changes*, we use the Dice n-gram based measure proposed by Adamson and Boreham (1974):

$$Dice_n(X,Y) = 1 - \frac{2c}{x+y} \tag{4.21}$$

where x and y are the total number of distinct n-grams in X and Y, respectively, and c is the number of distinct n-grams shared by both words. In our experiments, the best performance was achieved with $n = 3$. For illustration, consider the two lemma pairs $geben_V$ – $abgeben_V$ (*to give – to give in*), and $geben_V$ – $Gebende_N$ (*to give – giver*), which are both derivationally related. Both pairs achieve $Dice_3 = 0.333$ ($x = 5$, $y = 7$, $c = 4$), i.e., prefixation is not penalised, which would be the case with the D_4 measure above.

To group lemmas into "baseline" derivational families, we followed previous procedures from the literature (Šnajder and Dalbelo Bašić, 2009, Majumder et al., 2007, Adamson and Boreham, 1974) and used hierarchical agglomerative clustering with average linkage. We reduced the computational complexity by performing a preclustering step by recursively partitioning the set of lemmas sharing the same prefix into partitions of manageable size (1,000 lemmas). Initially, we set the number of clusters to be roughly equal to the number of the induced derivational families in DERIVBASE. For the final evaluation, we optimised the number of clusters based on F_1-score on the calibration and validation sets (cf. Figure 4.5 and Section 4.3.3). This optimisation leads to a balance between precision and recall for the string distance baselines. For details on this method, cf. Šnajder and Dalbelo Bašić (2009).

4.3.3 Gold Standard Annotation

This Section presents the information that we annotated on the P- and R-samples described in Section 4.3.1, and how we conduct the annotation process and calculate annotation quality. Finally, we show how we modify the gold standard annotation to evaluate later DERIVBASE versions.

Annotation Labels. We defined the five categories shown in Table 4.5, into which all lemma pairs are distinctly classified. We defined this annotation scheme, which is more fine-grained than the binary decision between derivational relatedness and unrelatedness, because it provides a more detailed picture of the data: The separation of semantically transparent (**S**), and opaque lemma pairs (**M**) gives deeper insight into the semantics of the derivational families, e.g., what kind of or how many lemmas are morphologically related but semantically unrelated. On the other hand, separating derivationally unrelated lemmas into the three categories **N**, **C** and **L** helps us detecting different reasons for overgenerating derivation rules. Unrelated lemma pairs (**N**) are most crucial here, as such errors might lead to spurious conflations of different derivational families. In contrast,

Label	Description	Example
S	l_1 and l_2 are morphologically and semantically related	$kratzig_A$ – $verkratzt_A$ (*scratchy – scuffed*)
M	l_1 and l_2 are morphologically but not semantically related	$bomben_V$ – $bombig_A$ (*to bomb – smashing*)
N	no morphological relation	$belebt_A$ – $loben_V$ (*lively – to praise*)
C	no derivational relation, but the pair is compositionally related	$Filmende_{Nn}$ – $filmen_V$ (*end of film – to film*)
L	not a valid lemma (mislemmatisation, wrong gender, foreign words)	$Haufe_N$ – $Häufung_N$ (*N/A – accumulation*)

Table 4.5: Categories for lemma pair classification

accidentally included compound relations (**C**) occur rarely in DERIVBASE due to our rule-based procedure, but are frequent for the baseline methods, e.g., $Ehemann_N$ – $Ehefrau_N$ (*husband – wife*). Finally, the **L** category helps to locate lemma pairs where false positives in DERIVBASE do not arise from erroneous or overgenerating rules, but from mistakes in the preprocessing. That is, **L** errors cannot be attributed to our derivation rules.

Nonetheless, we require a binary rather than a five-fold distinction of the data to compute precision and recall. To this end, the five annotation classes can be straightforwardly binarised. For our classification task, which determines – on the purely morphological level – whether a lemma pair is derivationally related or not (cf. Section 4.3.1), we rearrange the five labels as follows: Lemma pairs classified as **S** or **M** hold a derivational relation and thus count as positives, while **N**, **C** and **L** are negatives. Note that with this binarisation, we accept all kinds of derivationally related lemmas as positives. The decision what counts as positive class can look differently in other settings. For instance, if the focus was on derivational *and* semantic relatedness, one would accept only **S** as positive pairs (cf. Section 2.3).

We categorise ambiguous lemmas as positive (**S** or **M**) if there is at least one matching derivation or sense for the lemma pair. For instance, the word pair $Heroin_N$ – $heroisch_A$ ($heroin_N$/$heroine_N$ – $heroic_A$) is labelled **S**, since both lemmas are related with one sense to a courageous person. As mentioned in Section 2.3, disambiguation can be ignored in futher detail, since DERIVBASE operates on the lemma level.

Annotation Procedure and Inter-annotator Agreement. To visualise the annotation procedure of the gold standard, consider Figure 4.5 again. Two German native speakers with strong background in NLP annotated the pairs from the P-sample and R-sample. We first carried out a *calibration* phase in which the annotators double-annotated 200 pairs from each of the two samples (the box in the lower left corner in the Figure). After some discussion about the diverging decisions, the annotation guidelines were refined. For instance, it was crucial to define the differentiation of derivational prefixation and

5-fold decision	Agreement	Cohen's κ
P-sample	0.86	0.70
R-sample	0.85	0.79

2-fold decision	Agreement	Cohen's κ
P-sample	0.93	0.78
R-sample	0.97	0.92

Table 4.6: Inter-annotator agreement on validation sample, with five-fold (left) and two-fold (right) annotation classes

	S	**M**	**N**	**C**	**L**	*Total*
S	**132**	4	3	0	3	142
M	7	**7**	2	0	0	16
N	1	2	**19**	0	2	24
C	1	0	0	**0**	0	1
L	2	1	1	0	**13**	17
Total	143	14	25	0	18	200

Table 4.7: Confusion matrix for the five-fold annotation of the P-sample. The agreements in the diagonal are marked in boldface

composition, which is not always clear from the literature (cf. Section 2.2.1).

In a subsequent *validation* phase, we computed inter-annotator agreements on the annotations of another 200 pairs each from the P- and the R-sample (the bottom centre box in Figure 4.5). We report agreements for two granularity levels: for the five-fold classification according to our categories (**S** vs. **M** vs. **N** vs. **C** vs. **L**), and for the binary classification of derivational relatedness vs. unrelatedness (**S** + **M** vs. **N** + **C** + **L**). Table 4.6 shows, for both granularity levels, the proportion of identical annotations by both annotators as well as Cohen's κ score (Cohen, 1960). On the five-fold annotation task, we achieve high "raw" agreement and substantial agreement for κ (Carletta, 1996, Landis and Koch, 1977). On the binary annotation task, both the raw agreement and κ naturally increase significantly; on the R-sample, both scores attain almost perfect agreement. For both granularity levels, κ is a little lower on the P-sample, because the distribution of the categories is skewed towards **S**: Most lemma pairs sampled from DERIVBASE are actually derivationally and semantically related, which makes an agreement by chance more probable.

Appendix C contains the complete and final version of the annotation guidelines used for this annotation task. Note that especially the decision between **S** and **M** cannot be strictly defined, but requires linguistic intuition; the level of difficulty for distinguishing between these two classes is comparable to fine-grained word sense disambiguation.[9] Therefore, it is crucial that our annotators have strong linguistic expertise.

[9] In fact, our "raw" agreement on the five-fold annotation is higher than that of exemplary word sense disambiguation annotations from the literature: Agreements between 67.8% and 80% are reported on WordNet-level granularity tasks (Palmer et al., 2007, Snyder and Palmer, 2004)

Two pairs of categories show interesting disagreement cases: **M** vs. **S** indicates discrepancies regarding what the annotators accept as semantic relatedness, e.g., for the pair *lieblich*$_A$ – *lieblos*$_A$ (*lovely*$_A$ – *loveless*$_A$). In fact, the distinction between these two classes is most difficult in our annotation task: As the confusion matrix of the annotation agreements on the P-sample in Table 4.7 shows, the **S** vs. **M** distinction leads to the largest share of disagreements.[10] The classes **M** vs. **N**, in turn, show which lemma pairs the annotators accept as derivation at all. This category pair also leads to interesting disagreement cases, e.g., *Klette*$_N$ – *beklettern*$_V$ (*burdock*$_N$ – *to climb_on*$_V$).

Despite such disagreements, we found that the IAA results were sufficiently high to switch to single annotation for the production phase. Here, each annotator annotated another 1,000 pairs from the P-sample and R-sample each, so that the final test set consists of 2,000 pairs from each sample. These final test sets are depicted in the lower right box in Figure 4.5.

Adapting the Gold Standard to the Latest DErivBase Release. The sampling and annotation procedure described in this and the previous Section were used for the evaluation of an earlier release of DErivBase, v1.2, which is described and published in Zeller et al. (2013). But obviously, measuring the precision on the (test) P-sample requires the sample to reflect the entire set of rules used for the lexicon induction. In other words, the P-sample is dependent on the set of rules used for the respective DErivBase version, and its coverage.

As described in Section 4.2.4, there were further rule development phases, until we reached the latest DErivBase version 1.4.1: The size of the derivational families changed substantially between v1.2 and v1.4.1 due to many additional rules. Since a sensible evaluation dataset depends on the predictions of our rule induction, we could not rerun the evaluation without changes. More specifically, in order to properly examine the expansions of DErivBase v1.4.1, we needed a broader P-sample that covers the additionally implemented rules (mostly prefixation). We wanted to avoid additional annotation effort, so we made use of the lemma pairs covered by DErivBase in the (test) R-sample: We added these pairs to our original P-sample, removed duplicate pairs, and consequently obtained a more appropriate basis for calculating precision. The fact that the copied pairs are used in the P- as well as in the R-sample, is conceptually uncritical: The previously used sampling methods can – and actually did – similarly lead to identical pairs in the two samples.

After having changed the scope of the P-sample in this way, it contains a total of 2,545 pairs. The size of the R-sample remains at 2,000 pairs. Table 4.8 shows a breakdown of the two samples per annotation label. Henceforth, we will refer to these test sets as P- and R-sample, if not otherwise specified.

[10] We omit the confusion matrix for the R-sample, as it offers no deeper insight.

Sample	**S**	**M**	**N**	**C**	**L**	*Total*
P-sample	1,899	265	240	8	133	2,545
R-sample	450	122	762	586	80	2,000

Table 4.8: Breakdown of the P- and R-sample per annotation label

4.4 Results

This Section reports the results on the evaluation samples introduced in the previous Section. First, we discuss the general quantitative results with respect to precision and recall (Section 4.4.1). Then, we provide detailed qualitative analyses on the rule level in Section 4.4.2, and on the pair level in Section 4.4.3.

4.4.1 Quantitative Evaluation

First, we examine the distribution of the annotation labels in the P- and the R-sample. Consider Table 4.8 again: It reveals that in the P-sample, 2,164 are positive pairs (**S**, **M**), and 381 are negative pairs (**N**, **L**, **C**), respectively. The R-sample contains 572 positive and 1,428 negative pairs, respectively. As expected, the majority of pairs in the P-sample is positive, while most pairs in the R-sample are negative. The high number of **C** cases in the R-sample suggests that the string distance-based measures are prone to cluster lemmas with a compound relation. In contrast, such erroneous compositionally related pairs are negligible in the pairs sampled from DErivBase. In sum, the quality of the pairs drawn from DErivBase (i.e., the P-sample) is fairly good. Nonetheless, there is still a substantial amount of unrelated lemmas (**N**), and a number of lemmatisation errors (**L**) in our lexicon.

Table 4.9 presents the overall results, i.e., precision and recall on the two samples. We omit the F_1-score because its applicability for precision and recall estimates from different samples is unclear. We evaluate two variants of the induced derivational families: Those obtained before conducting any rule refinement steps as they are described in Section 4.2.4 (DErivBase initial), and those after all rule refinement steps, i.e., DErivBase v1.4.1. The best scores are marked in boldface.

As argued before, we measure the precision of our method on the P-sample and recall on the R-sample, respectively. For the baselines, precision was also computed on the R-sample: Computing it on the P-sample, which is obtained from the induced derivational families, would severely underestimate the number of false positives.

DErivBase v1.4.1 reaches the highest precision (85.0%) and recall (91.4%), notably outperforming the baseline methods on both measures by a large margin. The refinement of the initial model by checking as well as adding rules in several cycles has produced a significant improvement in recall (+ 31 percentage points) without losses in precision. In fact, precision even rises by almost 1% due to the more precise definition of some

Method	Precision P-sample	Recall R-sample
DERIVBASE initial	84.2	60.5
DERIVBASE v1.4.1	**85.0**	**91.4**
	Precision	Recall
	R-sample	
Stemming	69.4	7.5
String distance D_4	37.2	21.2
String distance $Dice_3$	23.7	24.0

Table 4.9: Precision and recall on test samples in different versions

derivation rules that avoids overgeneration. In v1.4.1, precision is lower than recall, meaning that DERIVBASE contains more false positives than false negatives, which supports our claim of having induced a high-coverage resource. Since the precision remained nearly constant between the two DERIVBASE versions, we assume that the false positives mainly arise from the initial rule set (cf. Section 4.2.4). We will go into detail about typical errors in Section 4.4.3.

In contrast, the German version of the Porter stemmer is rather conservative, which fragments the families and leads to a very low recall. In fact, the biggest derivational family in the stemming clusters contains no more than nine lemmas:[11]

Langweile$_{Nf}$, *Langweiligeres*$_{N}$, *Langweiligen*$_{N}$, *Langweiliges*$_{N}$, *Langweilen*$_{Nn}$, *Langweiler*$_{Nm}$, *Langweilige*$_{Nm}$, *Langweiligkeit*$_{Nf}$, *Langweiligste*$_{N}$

The corresponding family in DERIVBASE is more precise (e.g., the lemmatisation error *Langweiligen*$_{N}$ is excluded), but also covers more actual derivations (e.g., *langweilen*$_{V}$ is included):

Langweiligkeit$_{Nf}$, *langweilend*$_{A}$, *Langweilen*$_{Nn}$, *langweilen*$_{V}$, *Langweile*$_{Nf}$, *Langweiler*$_{Nm}$, *langweilig*$_{A}$, *gelangweilt*$_{A}$

That is, even for the biggest stemming cluster, there are still false negatives, because the scope of traditional stemming is too narrow to capture all derivational processes.

The string distance-based approaches achieve more balanced precision and recall scores (recall from Section 4.3.2 that this balance arises from optimising the number of clusters on the F_1-score), however, both are fairly low. The biggest clusters induced by $Dice_3$ and D_4 contain 932 and 462 lemmas, respectively, resulting from massive overgeneration. As mentioned above, both distance measures lack the linguistic information to distinguish

[11] Although this stemming cluster contains only lemmas of the same part of speech, this is not generally the case.

	Accuracy		
Coverage	High	Low	*Total*
High	18	1	19
Low	76	19	95
Total	94	20	114

Table 4.10: Cross-classification of derivation rules according to accuracy and coverage for direct derivations (measured on P-sample)

composition and derivation, and thus many lemma pairs are compositionally related. The largest share of false positives for the string distance measures results from morphologically completely different words with relatively high string similarity due to common prefix or suffix strings, e.g., *Blazer*$_N$ – *Grazer*$_N$ (*blazer*$_N$ – *inhabitant of Graz*). That is, the string distance methods lack the information about admissible derivational processes, as we have implemented in our derivation rules.

4.4.2 Rule-level Analysis

We conduct three analyses on the level of derivation rules: We examine the proportion of frequently and correctly applied rules both on a coarse- and a fine-grained level, and analyse the quality of the derivation rules according to the word classes involved.

Analysis by Frequency and Accuracy. We believe that our derivation rules differ in their ability to produce valid derivations. To investigate this assumption, we carried out an analysis in which we quantify the rules according to their quality and coverage.

We cross-classified our rules according to high/low accuracy and high/low coverage based on the pairs in the P-sample. We only considered directly derivationally related pairs by one single rule ($\rightarrow_{\mathcal{D}}$), amounting to 1,387 pairs related by 114 distinct rules. "High accuracy" and "high coverage" are defined as all rules above the 25th percentile in terms of accuracy and coverage, respectively. That is, we considered high-coverage rules to be the most frequent rules that cumulatively account for 75% of the derivations. Similarly, we considered high-accuracy rules to be the rules that are correctly applied in at least 75% of all cases. The results of this breakdown are shown in Table 4.10: All high-coverage rules except one are also highly accurate. Most rules are accurate but infrequent. Only 20 rules have a low accuracy, but 19 of them apply infrequently. Thereof, six rules are stem changes, which is 60% of all (pure) stem change rules (cf. Table 4.4). This confirms our assumption that stem changes often produce incorrect derivations. In total, however, this Table shows that our procedure to define and apply derivation rules is able to support accuracy as well as coverage.

Analysis by Ranked Rule Quality. The previous analysis has revealed that some rules seem to exhibit higher quality than others. However, only directly related lemma pairs were considered. In this paragraph, we want to further investigate how quality and coverage are distributed over rules when we consider all lemma pairs, i.e., lemma pairs connected by rule paths of any length.

To this end, we simulate a rule ablation test in order to observe how many high-quality rules are needed to achieve a certain precision-recall ratio. From the previous analysis, we expect some rules to produce correct derivations very often, while other rules might match frequently and thus perform poorly, e.g., mislemmatised words or words without derivational relation. Filtering out such unreliable rules would increase the precision of derivational families, but also lower the recall; such a setting might be desirable for applications in which correctness is more important than coverage. To estimate these differences, we analysed how precision and recall behave if we employ only subsets of the implemented rules, selected by their quality.

To attain such subsets, we *ranked the rules* according to two criteria: As a primary sort key, we calculated the quality of rule r as a ratio of correct vs. all applications:

$$quality_r = \frac{|correct\ applications_r|}{|total\ applications_r|} \tag{4.22}$$

In case several rules achieve the same ratio value, we sorted these rules according to a secondary key: by their frequency of correct applications, $|correct\ applications_r|$. That is, if two rules always link **S** or **M** pairs, the more frequently applied rule is ranked higher.

We then measured the precision and recall of subsets of rules, starting with only one – the qualitatively best – rule, and cumulating up to all rules implemented in DERIVBASE. Note that a lemma pair can be connected by more than one rule, e.g., the word pair $Zähe_{Nf}$ – $Zähigkeit_{Nf}$ ($tough\ person_N$ – $toughness_N$) is connected via $zäh_A$ ($tough_A$). For such cases, the lemma pair is only covered by the rule subset when all participating linking rules are included in this rule subset. That is, precision and recall for rule r at a specific rank are calculated as follows:

$$Prec@r = \frac{|correct\ pairs\ involving\ only\ r\ and\ higher\text{-}ranked\ rules|}{|all\ pairs\ involving\ only\ r\ and\ higher\text{-}ranked\ rules|} \tag{4.23}$$

$$Rec@r = \frac{|correct\ pairs\ involving\ only\ r\ and\ higher\text{-}ranked\ rules|}{|all\ correct\ pairs|} \tag{4.24}$$

The intuition behind these calculations is somewhat comparable to that of the Average Precision (AP) measure in Information Retrieval (Manning and Schütze, 1999, p535f).[12]

We determined the ranking keys based on the lemma pairs and their gold annotations

[12]Our measures differ from AP in that they evaluate the ranking of the *rules* that link the lemma pairs in the dataset, while AP would measure the ranking of the *lemma pairs* directly. Additionally, due to the combinability of the rules, we do not necessarily evaluate all instances of a rule at the same time (i.e., at a specific rank), but only those applied up to this point (rank).

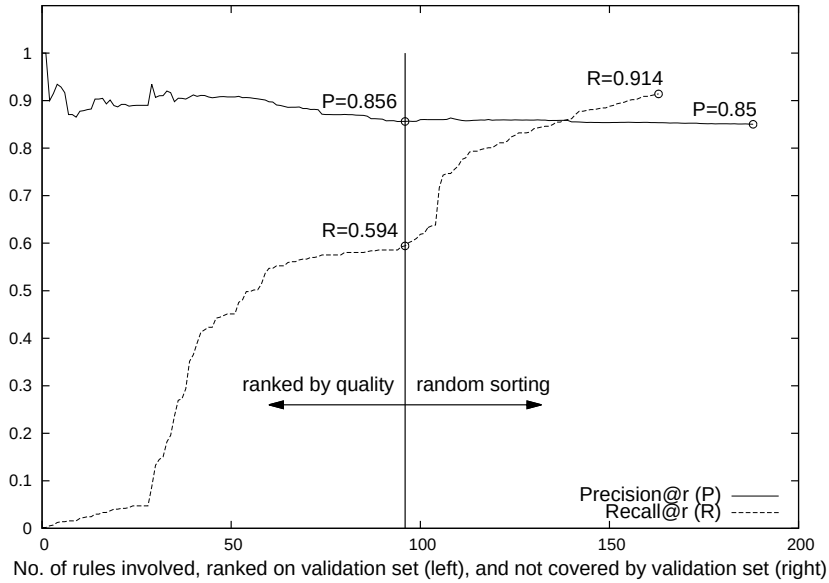


Figure 4.6: Precision and recall for cumulated rules ranked by quality

of the validation set rather than the test set (cf. Section 4.3.3) in order to avoid overfitting. Out of the 400 lemmas in this validation set, 232 are covered by DERIVBASE v1.4.1. Next, we applied the resulting ranking to all lemma pairs in the P-sample and R-sample which are covered by DERIVBASE, and calculated precision and recall as shown in (4.23) and (4.24) on P- and R-sample, respectively. Since the R-sample reflects the fact that DERIVBASE achieves a high, yet imperfect recall compared to an ideal method, recall does not add up to 100% (i.e., only 523 out of 572 derivationally related lemmas are covered).

The validation set covers 97 derivation rules, while the test samples cover 189 rules. Rules which are applied on the test sets, but not covered by the validation set, were appended in random order at the end of the ranking established from the validation set. In that way, these rules are treated like rules of low quality, which satisfies the fact that we could not estimate their actual quality on the validation set.

Figure 4.6 illustrates the precision and recall curves for this rule ranking analysis. We will discuss the curves from left to right.

The high-quality rules, which occupy the top 29 (leftmost) positions, seem to be applied relatively rarely on their own, but require the presence of other, more frequent rules: They lead to very low recall scores on the R-sample (4.7%). This observation is in line with the breakdown shown in Table 4.10. Concerning precision, these rules exhibit high

variance, because the frequency of rule applications is still low and therefore unstable. Nonetheless, precision never falls below 86.5%.

The rules in the medium range of the ranking ($x = 30$ to $x = 60$) provide a great increase in coverage (and consequently recall), while precision remains above 90%. Note, however, that a big jump in the recall curves does not necessarily imply that the corresponding rule in the ranking is exceptionally frequent. Instead, this rule often occurs in combination with higher-ranked rules, which have already been considered before. Relating these numbers to the more coarse-grained analysis of the previous paragraph, we indeed observe that all except three high-coverage/high-accuracy rules from Table 4.10 are situated on ranks between $x = 1$ and $x = 53$. Also, all but four low-coverage/low-accuracy rules are ranked on $x = 57$ and beyond. That is, this rule ranking which considers all rule paths, reflects the tendencies detected on directly related lemma pairs.

When all rules contained in the validation set are considered ($x = 97$), precision amounts to 85.6%, while recall is at 59.4%. The low recall is not surprising, because the validation set covers only 97 out of 267 derivation rules (36%). On the other hand, it is a positive result that the precision curve – except some outliers for the rare high-ranked rules – monotonically decreases with the growing number of rules involved, but only slightly. This behaviour implies that the derivation rules indeed differ in their quality, and can therefore be ranked according to the quality.

Finally, when we randomly add the derivation rules which were not covered by the validation set (from $x = 98$), there is another enormous increase in recall, leading to the final performance of DERIVBASE v1.4.1 of 91.4%.[13] At the same time, the overall precision of 85.0% is achieved, which is only slightly lower than the precision for the 97 top-ranked rules. The flat precision curve between $x = 97$ and $x = 189$ suggests that even rules which have not been previously ranked according to their quality, hardly lead to losses in precision, but can enormously boost recall. This behaviour strengthens our rule-based approach: Even if no information is available to rank these rules, thoroughly developed derivation rules mostly lead to correct derivations.

In sum, Figure 4.6 shows that the derivation rules used for inducing DERIVBASE exhibit different quality. Already a small set of rule applications is enough to establish a sensible qualitative rule ranking, even if recall is expectedly low. The resulting qualitative ranking of the derivation rules allows for building DERIVBASE versions with high precision but low recall by employing only high-quality rule subsets. Such a strategy can be desirable if optimal precision is required.

All subsequent analyses, however, refer to DERIVBASE v1.4.1, i.e., using the whole set of derivation rules, which is the model with the highest recall.

Analysis by Parts of Speech. Table 4.11 shows precision and recall values for different part of speech combinations for the base and derived words, measured on P- and R-sample, respectively. The best scores are marked in boldface. High precision is achieved for A-A as well as N-A derivations, e.g., *kaufbar*$_A$ – *käuflich*$_A$ (*purchasable*$_A$ –

[13] Note that the lemma pairs in the R-sample covered by DERIVBASE involve only 164 distinct rules, which is why the corresponding curve stops earlier.

	P	R		P	R
N-N	0.79	0.83	N-A	**0.90**	0.92
A-A	**0.90**	0.90	N-V	0.82	**0.97**
V-V	0.76	0.94	A-V	0.89	0.96

Table 4.11: Precision and recall across different part of speech combinations for base word and derivative

purchasable$_A$), *Pflanze*$_N$ – *pflanzlich*$_A$ (*plant*$_N$ – *vegetable*$_A$), while N-V derivations have the highest recall, e.g., *Dolch*$_N$ – *erdolchen*$_V$ (*dagger*$_N$ – *to dagger*$_V$). The precision is lowest for V-V derivations, which indicates that some verb deriving rules still overgenerate, e.g., *flechten*$_V$ – *flüchten*$_V$ (*to braid*$_V$ – *to escape*$_V$). In fact, Germanic strong verbs are particularly often affected by – partially hardly predictable – ablaut stem changes (Glück, 2010, p5). Recall is lowest for N-N derivations, suggesting that the derivational phenomena for nouns are not yet covered satisfactorily. For instance, the noun pair *Pathogenität*$_N$ – *Pathologe*$_N$ (*pathogenicity*$_N$ – *pathologist*$_N$) was not retrieved, because we did not implement derivation rules which incorporate combining forms such as *log-* (cf. Section 2.1.3). Notably, we are capable to establish good recall scores for derivational relations across parts of speech.

4.4.3 Pair-level Analysis

In this Section, we further discuss precision and recall errors on the level of annotated pairs. Table 4.12 shows the frequencies of true positives and false positives on the P-sample and false negatives on the R-sample for each annotated category. True negatives are not reported, since their analysis gives no deeper insight.

True Positives. In our analysis, we treated both **S** and **M** pairs as related, but it is interesting to see how many of the true positives are in fact semantically unrelated. In line with the DCA (cf. Section 2.1.3), 88% out of the 2,164 positive pairs are semantically as well as morphologically related (**S**), e.g., *alkoholisieren*$_V$ – *antialkoholisch*$_A$ (*to alcoholise*$_V$ – *nonalcoholic*$_A$), or *Boxer*$_N$ – *boxend*$_A$ (*boxer*$_N$ – *fighting*$_A$). Most **S** pairs result from high-accuracy rules, i.e., conversion, negation prefixation and simple suffixation. The remaining 12% are only morphologically related (**M**), e.g., *beschwingt*$_A$ – *schwingen*$_V$ (*cheerful*$_A$ – *to swing*$_V$), *Stolzieren*$_N$ – *stolz*$_A$ (*strut*$_N$ – *proud*$_A$). In both pairs, the two lemmas share a common semantic concept (i.e., being in motion or being proud) but nowadays' meanings have grown apart from each other. Among the **M** true positives, 73% are prefixation derivations, often involving prefixation at both lemmas, e.g., *Erdenkliche*$_N$ – *bedenklich*$_A$ (*imaginable*$_N$ – *questionable*$_A$). This high ratio of prefixation among **M** pairs reflects the general assumption mentioned in Section 2.2.1 that prefixation often leads to a meaning change. Note that negation prefixation is controversial in this context:

	TPs	FPs	FNs
Label	P-sample	P-sample	R-sample
R	1,899	–	28
M	265	–	21
N	–	240	–
C	–	8	–
L	–	133	–
Total	2,164	381	49

Table 4.12: Predictions over annotated categories

It could be regarded as the most drastic meaning change possible; then, it would have to be categorised as **M**. However, following our definition in Section 2.1.3, we interpret negation derivations as meaning-preserving and assign them the category **S**, because both base and derived word refer to highly related concepts.

False Positives. About 63% of the 381 incorrect pairs are of class **N** (unrelated lemmas). The most frequent errors in this category are due to four phenomena: Short lemmas, named entities, long derivation rule paths, and stem-changing derivations.

Many errors occur in pairs involving short base words, e.g., Gen_N – $genieren_V$ ($gene_N$ – *to be embarrassed*$_V$), where the orthographic context is insufficient to reject spurious derivations. In parallel, about 20% out of the unrelated lemma pairs are a result of derivations between named entities (including proper nouns) and common nouns. For example, the rule to derive nouns denoting a male person incorrectly links the lemma pair $Morse_{NE}$ – $Mörser_N$ ($Morse_{NE}$ – $mortar_N$). Such incorrect named entity derivations happen especially for short base words, as in the the example of *Morse*, or in $Kühn_{NE}$ – $Kühnheit_N$ ($Kühn_{NE}$ – $audacity_N$). However, since named entities also participate in valid derivations (e.g., $Chaplin_{NE}$ – $chaplinesk_A$ ($Chaplin_{NE}$ – $chaplinesque_A$) or $Wien_{NE}$ – $Wiener_N$ ($Vienna_{NE}$ – $Viennese_N$)), we did not exclude them from our derivations. An alternative would be to treat named entities as well as short lemmas specifically.

The third major problem which leads to false positives of category **N** are transitively applied rules that produce incorrect pairs. For instance, $Speiche_N$ – $speicherbar_A$ ($spoke_N$ – $storable_A$) results from the following rule path involving three derivation rules (for details about these rules, cf. Appendix B): $Speiche_N \xrightarrow{NN05} Speicher_N \xrightarrow{NV09} speichern_V \xrightarrow{VA01} speicherbar_A$ ($spoke_N \rightarrow storage_N \rightarrow to\ store_V \rightarrow storable_A$). Longer paths that additionally involve stem changes (e.g., ablaut shifts) can lead to surprising results, e.g., the **N** lemma pair $Erringung_N$ – $rangiert_A$ ($achievement_N$ – $shunted_A$). The fact that about 50% of the **N** pairs concern ablaut and umlaut stem changes, attests again our assertion that involving stem changes is a critical tradeoff decision between coverage and precision (cf. Section 2.2.1): Stem changes are applicable to many base words, but are

synchronically unproductive and do not lead to many correct derivatives. Considering them in our approach leads to a substantial number of false positives, but avoids, on the other hand, fragmented derivational families.

Notably, those four main factors for **N**-type errors already existed in the initial DERIVBASE version, which explains the small difference in precision between the two lexicon versions shown in Table 4.9. An interesting side result regarding the **N** errors is that some pairs judged as unrelated by the annotators might conceivably be weakly related. For example, in the umlaut shift derivation *schlürfen*$_V$ – *schlurfen*$_V$ (*to sip*$_V$ – *to shuffle*$_V$), both base and derived lemma refer to specific long drawn out sounds. Such cases show that it is not always straightforward to draw the line between relatedness and unrelatedness even on the morphological level.

Concerning false positives of class **C**, it is not surprising that such cases hardly occur in the P-sample: We employ a rule-based approach for *derivation* induction, which naturally excludes almost all cases of composition. Nonetheless, there are a few lemma pairs which are compositionally related, e.g., *filmen*$_V$ – *Filmende*$_{Nn}$ (*to film*$_V$ – *end of film*$_N$), or *Gehen*$_{Nn}$ – *vorbeigehen*$_V$ (*walking*$_N$ – *to pass by*$_V$). Here, our derivation rules accidentally match bound morphemes (the building blocks in composition). The derivative of the first lemma pair could actually be a valid derivation (a filming person), if the noun gender was not neuter but masculine. However, the verb suffix *-en* and the derivation suffix *-de* unexpectedly form the German noun *Ende*$_N$ (end). The second lemma pair is incorrectly covered, because the concatenation of the two derivation prefixes *vor-* and *bei-* form an adverb, *vorbei*, which constitutes a compositional word formation process.

Finally, errors of category **L**, which make up 35% of the false positives, are due to incorrect lemmatisation. The most frequent **L** errors are 1., spurious plural endings, e.g., *Abgelehnten*$_N$ or *Messerklingen*$_N$, and 2., incorrect noun genders, often arising from spurious plural endings, such as *Einreisen*$_{Nf}$. **L** errors cannot be attributed to our derivational model, but of course form part of the lexicon and should be avoided wherever possible. Remarkably, many lemmatisation errors, including those mentioned here, are derivatives produced by conversion derivation. Thus, besides using different preprocessing procedures, one could also review how to deal with the respective rules.

False Negatives. Errors of this type can be due to three reasons: missing derivation rules, erroneous rules that leave some derivations undiscovered, or the absence of lemmas in the corpus required for the transitive closure to link two lemmas.

Our inspection of the false negatives revealed that the latter reason – a coverage lack in the corpus – does not apply, which supports the strategy how we selected our German lemma list (cf. Section 4.2.3). Also, erroneous rules are a minor reason: We discovered merely one faulty rule in our rule set, which fails to link some admissible cases of adjectives and their nominalisations, e.g., *Geborene*$_N$ – *geboren*$_A$ (*born*$_N$ – *born*$_A$). Since this rule affects a considerable number of lemmas, we plan to correct it in a future release of DERIVBASE.

In contrast, most false negatives arise from missing derivation rules. For both categories **S** (about 57%) and **M** (about 43%), there are three main types of missing rules: 1.,

irregular derivations, such as *Meute*$_{Nf}$ – *Meuterer*$_{Nm}$ (*mob*$_N$ – *mutineer*$_N$)[14]; 2., irregular foreign derivations, e.g., *olympisch*$_A$ – *Olympionikin*$_{Nf}$ (*Olympic*$_A$ – *female Olympic athlete*) and *Reflex*$_{Nm}$ – *Reflektor*$_{Nm}$ (*reflex*$_N$ – *reflector*$_N$); and 3., different spelling variants, e.g., *Fotograph*$_{Nm}$ – *Fotografie*$_{Nf}$ (*photographer*$_N$ – *photography*$_N$). Additionally, there is a handful of idiosyncratic derivations that involve processes which we did not cover. For instance, we did not consider comparative-involving derivation rules, which were necessary for lemma pairs like *schwerpflegebedürftig*$_A$ – *schwerstpflegebedürftig*$_A$ (indications for two different levels of care dependencies). Similarly, we could not implement rules for cases in which composition and derivation interfere. For instance, the *ge-* prefixation turns into an "infix" for lemma pairs such as *Rechtslehre*$_{Nf}$ – *Rechtsgelehrte*$_{Nm}$ (*jurisprudence*$_N$ – *jurist*$_N$). However, neither such infix insertions exist in German derivation, nor does a bridging verb such as **rechtsgelehren* which would enable to link the lemma pair.

Apart from these rather specific cases of false negatives, our rules display a fairly good coverage, as the low number of only 49 missed pairs and the high recall show.

4.5 Summary

In this Chapter, we have presented how we induced DERIVBASE, a human-readable derivational lexicon for German which covers all admissible derivational operations (suffixation, prefixation, conversion, circumfixation, and stem changes) for verbs, nouns, and adjectives. DERIVBASE is compiled in a different way than, e.g., CatVar (Habash and Dorr, 2003) (cf. Section 3.1.2): Our approach does not collect information from comprehensive knowledge resources. Instead, we employed HOFM, a language-independent rule-based framework, in which derivational rules are implemented via higher-order transformation functions. We expanded the framework with German language-specific phenomena such as the umlaut stem change, or the consideration of noun gender. One working week was enough to implement and thoroughly refine the underlying minimal rule set with the aid of grammar textbooks, which shows that the framework is easy and straightforwardly to use. More than half of our implemented rules cross the part of speech boundary between base word and derivative. We applied the derivation rules to lemmas gathered from a German web corpus, and clustered the resulting lemma pairs into families by computing the transitive, symmetric and reflexive closure of rule applications. The sequence of derivation rules which connects a lemma pair is called a rule path.

In that way, we collected derivational families for over 280,000 lemmas with high accuracy as well as high coverage. Over 69,000 lemmas are grouped into more than 17,000 non-singleton families, while most remaining singleton lemmas are compounds. To assess the quality of DERIVBASE, we have developed an elaborate evaluation method that uses two separate samples to assess precision and recall appropriately. The latest version of DERIVBASE, v1.4.1, achieves a precision and recall score of 85% and 91%, respectively.

The lexicon, our gold standard annotations, and the employed derivation rules[15] as well

[14] The regular derivation of this example would be **Meuter*.

[15] `http://www.ims.uni-stuttgart.de/permalink/56cc6c89-c421-11e4-a5e6-000e0c3db68b.html`; license cc-by-sa 3.0. Last accessed: May 2015

as the grammar framework[16] are freely available. We provide DERIVBASE in *two different formats*. The first format is parallel to that of CatVar, simply containing the induced derivational families. The second format is additionally enriched with information about the derivation rules which connect the lemma pairs within one family. In this way, we do not only provide the result of the induction process, but also the linguistic knowledge with which this result is achieved. As shown in Section 3.1.1, there are various approaches to build derivational lexicons for French, but so far, none of them seems to have become a standard resource for the general public. We hope that the simple lexicon format, and the free access to DERIVBASE eases and increases its usability for other researchers.

Our approach for compiling a derivational lexicon is not restricted to German. In addition to the typologically most similar Germanic and Romance languages, it is also transferable to agglutinative languages like Finnish, or other fusional languages like Russian. The only requirements of our method are a comprehensive set of lemmas for the respective language (optionally with further morphological features such as gender information), and linguistic literature about admissible morphological derivation processes. In fact, Šnajder (2014) applied our procedure to Croatian. In an intrinsic evaluation with a slightly different setting than ours, his lexicon DerivBase.hr achieves precision and recall scores of over 81% and 76%, respectively.

Future Directions. Our analysis in Section 4.4.3 has shown that the recall of DERIVBASE is very satisfactory, but that precision is not: The derivational families contain about 15% false positives (**N**, **C**, **L** cases), mostly arising from overgeneration due to different reasons. Unquestionably, it is desirable to rule out these incorrect lemmas. Addressing the most crucial reasons for false positives mentioned above, it would make sense to design further statistical or rule-based restrictions to the application of derivation rules. For instance, one could add checks for particularly short (and therefore error-prone) base words, or restrict the application of rules that are known to overgenerate (e.g., conversions on lemmatisation errors, or stem-changing rules). Also, named entities which often cause overgenerations, could be specifically treated, e.g., by expanding the inflectional patterns $\mathcal{P}$ accordingly (i.e., the noun patterns would not only reflect gender information, but also whether the corresponding word is a common or a proper noun). Finally, the length of the derivation rule path of a lemma pair could be considered when one determines the probability that the lemma pair is actually related.

Moreover, there is a substantial amount of purely morphologically related words (**M** cases, about 10%) in DERIVBASE. We would expect that, despite the DCA (cf. Section 2.1.3), which relies on the semantic transparency of most derivational relationships, these **M** lemmas might worsen the usefulness of our lexicon when it is applied to semantic NLP tasks. Thus, from a practical point of view, even these **M** pairs should be removed from the families. We would hope that such a semantically sensitive version of the lexicon would perform better. As a side benefit of such a semantic refinement, many actual false positives (categories **N**, **C**, **L**) might also be excluded from the derivational families.

[16] `http://takelab.fer.hr/data/hofm/`; license BSD 3-Clause. Last accessed: May 2015

As we find this question of a semantic refinement very challenging as well as methodologically interesting, we will address it in Chapter 5. In this context, we also consider some of the restrictions proposed above that follow from our false positive analysis.

Alternative Approach: Induction of Derivational Families Using a Morphological Analyser. In Section 3.1.2, we have mentioned that the most traditional topic in morphology is morphological analysis and generation, and presented various lexicon-based analysis tools for German. Thus, the question arises whether the DERIVBASE induction using HOFM that we have presented in this Chapter might be dispensable, and a derivational lexicon could be instead induced by means of such an analyser, and its underlying lexicon; e.g., with SMOR, one of the best German analysers. Such a tool could similarly lead to derivationally related lemmas and thus, to derivational families.

To test this hypothesis, we implemented a wrapper to make SMOR predicting derivational relationships for pairs of lemmas: The SMOR analyser is applied off-the-shelf to both lemmas of a pair (l_1, l_2), and typically returns several analyses per lemma, which differ either in morphological features, or in the assumed base lemma(s).[17] Whenever various analyses with different base lemmas are detected, one can select SMOR's preferred base lemma. For instance, the ambiguous analysis of *abkömmlich*$_A$ (*available*) is:

```
ab<VPART>kommen<V>lich<SUFF><+ADJ><Pos><Adv>
ab<VPART>kommen<V>lich<SUFF><+ADJ><Pos><Pred>
abkömmlich<+ADJ><Pos><Adv>
abkömmlich<+ADJ><Pos><Pred>
```

with the preferred base lemma being *abkömmlich*. To determine derivational relationship, we implemented a transitive link on the analyses of both lemmas (l_1, l_2). We experiment with two link models:

maxLink: Whenever one analysis of l_1 consists of the same base lemmas and parts of speech as one analysis of l_2, we consider this lemma pair as derivationally related via SMOR; otherwise, the lemma pair is not derivationally related.

prefLink: Only if SMOR's preferred base lemma (and part of speech) is able to link l_1 and l_2, we consider this lemma pair as derivationally related via SMOR.

For example, for the lemma *unabkömmlich*$_A$ (*engaged*), SMOR also provides analyses containing both the base lemma *kommen* and *abkömmlich*$_A$ (*available*), the latter again being the preferred one. Thus, both the *maxLink* and the *prefLink* model can establish a derivational relationship to the abovementioned *abkömmlich*, but only the *maxLink* would be able to additionally connect *abkömmlich* with more distant members of the *kommen* family, e.g., *ankommen*$_V$ (*to arrive*). Note that in this process, we cannot provide SMOR with the parts of speech gained in our corpus preprocessing (cf. Section 4.2.3), as this information is not accepted by the analyser. Instead, SMOR looks up the part of speech in its internal lexicon.

[17] An SMOR analysis can consist of several lemma-part of speech pairs, e.g., for compounds.

	Precision	Recall
Method	P-sample	R-sample
SMOR *maxLink*	**93.7**	69.4
SMOR *prefLink*	92.6	38.1
DERIVBASE initial	84.2	60.5
DERIVBASE v1.4.1	85.0	**91.4**

Table 4.13: Precision and recall for different SMOR and DERIVBASE versions

We apply both link models to both our P- and R-sample, and measure precision and recall as explained in Section 4.4.1. Table 4.13 shows the results for SMOR, and repeats the performance of DERIVBASE from Table 4.9.[18] We measure significance of precision and recall differences with bootstrap resampling (Efron and Tibshirani, 1993).

Both SMOR variants achieve a remarkably high precision of over 92%, while the recall is rather low for the *maxLink* model and very poor for the *prefLink* model. The high precision arises from SMOR's low rate of false positives in the P-sample. While some of the correctly rejected true negatives are erroneous rule applications (e.g., a spurious *ver-* prefixation for Arm_{Nm} – $Verarmung_{Nf}$ (arm_N – $impoverishment_N$), the majority of pairs rejected by SMOR involve a lemmatisation error (**L**). More specifically, over 70% of the **L** pairs in the P-sample are correctly rejected, thus dramatically increasing precision. As a consequence, both SMOR models outperform the precision of both DERIVBASE versions significantly at p=0.001. This big performance difference is at least partly due to a technical issue: DERIVBASE, being constructed with a corpus-based approach, relies on the correctness of both the underlying corpus data, and the corpus preprocessing. However, the noun gender predicted by MATE is often incorrect (cf. Section 4.4.3) due difficult input data. In contrast, SMOR, being constructed with a lexicon-based approach, employs its own, to a considerable amount manually compiled morphological information, so that such errors are avoided a priori.

SMOR's high precision comes at the cost of recall, though. For the *prefLink* model, recall is exceptionally low and even significantly below the performance of DERIVBASE initial (at p=0.001), although this is the first, incomplete version of our derivational lexicon. The reason is that SMOR's preferred base lemmas differ for many derivationally related lemma pairs and thus, no transitive link can be established. This fact suggests that, for the purpose of derivational relatedness detection, one should not rely on SMOR's internally preferred analysis, but use the *maxLink* model instead. Here, recall is clearly better, however, still significantly below that of DERIVBASE v1.4.1 (at p=0.001): Surprisingly, many derivationally related words are not derived from the same stem in

[18] Note that a comparison of SMOR and DERIVBASE initial must be treated with caution: This DERIVBASE version only implements a subset of German derivational processes – in particular, very few prefixation rules (cf. Section 4.2.4) – and thus naturally achieves lower recall.

SMOR, thus leading to fragmented derivational families. For instance, the **M** lemma pair *Zweifel*$_{Nm}$ – *verzweifeln*$_V$ (*doubt*$_N$ – *to despair*$_V$) cannot be transitively connected with SMOR, because the only base word retrieved for the former lemma is *Zweifel*, while the analyses of the latter contain *zweifeln*$_V$ (*to doubt*) and *verzweifeln*$_V$. This severe quantitative drawback also concerns entire families. As an example, consider the following derivational family:

hausen$_V$ (*to reside*$_V$), *hausend*$_A$ (*residing*$_A$), *häuslich*$_A$ (*domestic*$_A$),
Haus$_N$ (*house*$_N$), *Häuschen*$_N$ (*small house*$_N$), *Gehäuse*$_N$ (*shell*$_N$)

SMOR outputs three different base lemmas for this family, which are not connected by any morphological analysis: *hausen*$_V$ for the first three lemmas, *Haus*$_N$, for the fourth and the fifth, and *Gehäuse*$_N$ for the last lemma. Thus, two links are missing, and the lemmas cannot be conflated into one family, but end up in three separate families.

In sum, we do not feel that SMOR, although achieving high precision, is competitive with the procedure proposed in this thesis: Its recall is too low to ensure sufficiently high coverage for broad applicability. While the results might be sensible from a word-structure perspective that focuses on the morphemes of a word rather than its connections to morphologically related words, such a behaviour is clearly undesirable for the induction of a derivational lexicon with maximal possible coverage. Furthermore, the construction procedure of SMOR and its underlying lexicon is more time-consuming than that of DERIVBASE, which would require more initial effort when it is transferred to another language: Our approach only requires the implementation of derivation rules in a language-independent, generic framework.

Nonetheless, we believe that SMOR could help to further improve the performance of DERIVBASE: It could be used to reject incorrectly lemmatised corpus data from the very beginning, i.e., as a preprocessing step. As mentioned in Section 3.1.2, such a strategy might achieve synergy effects between the lexicon-oriented, and the word structure-oriented perspective. In the following, we do not consider such an ensemble strategy, but leave it for future work.

5 Semantic Validation of DErivBase

Chapter 4 illustrated our approach to induce a computational derivational lexicon, and which information this lexicon contains: DERIVBASE is a morphological resource, as it consists of families of derivationally related lemmas. Additionally, it provides linguistic information about the derivation rules involved to derive the lemmas within one family. We compiled DERIVBASE as a purely morphological resource, i.e., we concentrated on the main assertion of the Derivational Coherence Assumption (cf. Section 2.1.3) and the intuition of authors of other derivational lexicons (Habash and Dorr, 2003), that derivationally related words are often semantically related and thus, a derivational lexicon is a valuable resource. The appropriateness of this assumption is not completely warranted, though. There is a strong correlation between derivational morphology and semantics, but it is not perfect: On the semantic level, there is a broad continuum of semantic relatedness across derivationally related word pairs. They can be quasi-synonyms (transparent derivations) as well as have fairly different meaning (opaque derivations).[1] The (synchronic) opacity, or *semantic drifts*, of derivation can have a number of reasons (Lieber, 2009), including homography (*arm* as body part or weapon), accidental instantiation of derivational patterns (*corn* – *corner*), and diachronic meaning drift (*dog* (*animal*) – *dogged* (*determined*)). In other words, a substantial number of the lemma pairs in DERIVBASE are false positives regarding the level of semantic relatedness. We believe that capturing these semantic differences is important from a linguistic point of view. For that reason, our goal is to filter the information in DERIVBASE by incorporating a semantic level that sub-groups derivational families into *semantically coherent clusters*, as proposed in Section 2.3. In doing so, we address the restriction mentioned as a last remark of our DCA: Only transparent derivational relationships are expected to contribute valuable semantic information. Thus, for a sensible application in lexical semantics, a derivational lexicon should concentrate on these pairs. In this Chapter, we refine DERIVBASE beyond purely morphological requirements: Not all, but only semantically transparent derivations are considered semantically related.

For an illustration of this problem on the level of families, consider Figure 5.1, an excerpt of Figure 4.1, which explained the DERIVBASE induction. This sample family contains three lemmas, connected by two rules: $eitel_A$ – $vereiteln_V$ – $Vereitelung_N$ ($vain_A$ – *to* $block_V$ – $blocking_N$). The *-ung* suffixation rule between the second and the third lemma is semantically transparent, even across the part of speech boundary; its

[1] Although semantic relatedness is generally distributed on a graded scale and not binary, this operational simplification, which is often used in computational linguistics (cf. Section 2.1.3, is reasonable here: As noted in Section 4.3.3, our manual annotation labels can be binarised with respect to semantic relatedness, and the good inter-annotator agreement (cf. Table 4.6) supports this decision. Thus, we will adopt a binary perspective on semantic relatedness in the following.

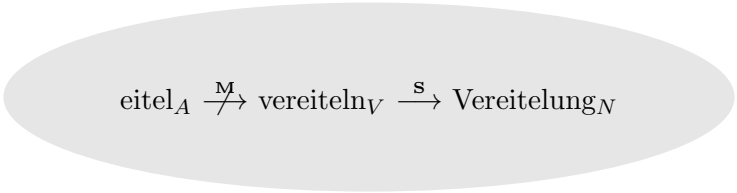


Figure 5.1: Induced sample family

corresponding category is therefore **S** (cf. the annotation categories in Section 4.3.3). In contrast, the *ver-* prefixation applied between the first and the second lemma is opaque and thus labelled **M**. That is, within a derivational family, some lemma pairs are semantically related, while others are not. We tackle this problem by a method we call *semantic validation*: For each related lemma pair in DERIVBASE, we decide whether it is also semantically related (transparent); based on these decisions, we split semantically inconsistent families. In the example, this would mean to detach the lemma *eitel*$_A$ from the other two (as indicated by the deleted arrow), which then constitute a two-member semantic family, and *eitel*$_A$ a singleton family.

This Chapter describes how we incorporate such semantic coherence into DERIVBASE. More specifically, we develop methods to determine, for lemma pairs in the same derivational family, whether they are also semantically related. Recall that DERIVBASE, as it is induced semi-automatically, also contains completely unrelated (**N**) and compositionally related (**C**) lemma pairs as well as lemmatisation errors (**L**). Note that we do not aim at detecting all five relation categories introduced in Section 4.3.3, but treat all derivationally related, but semantically unrelated lemma pairs (**M**, **N**, **C**, **L**) equally.[2] That is, our semantic clusters within derivational families may only contain lemma pairs of the **S** category; all other pair types are discarded. We regard such a filtering as the logical next step to increase the linguistic appropriateness of DERIVBASE.

As described in Chapter 4, we conducted the morphological lexicon induction with a symbolic approach, because derivation can be well generalised and described with rules. In contrast, semantic relatedness is composed of a variety of factors, i.e., there are many reasons for semantic drifts. We think that it would be hard to design a formal, rule-based framework that reflects these interdependencies. Instead, we choose a data-driven method, and conduct semantic validation with machine learning. More specifically, we employ a binary classification model that separates semantically related and unrelated lemmas based on features extracted from two main information sources: On the one hand, we employ distributional semantic models, providing us with information about the semantic relatedness of two words attested on large corpora. We expect this information to be

[2] In fact, compositionally related (**C**) pairs are somewhat semantically related. But since such pairs are extremely rare through the rule-based construction of DERIVBASE, and their semantic transparency is questionable, we do not treat them as semantically related.

valuable for the semantic clustering we intend, similar to other semantic classification studies (Schulte im Walde, 2006, Boleda et al., 2012). On the other hand, we use structural linguistic information encoded in the derivation rules with which we induced DERIVBASE; it helps to examine the behaviour of derivational processes regarding semantic coherence. After analysing to what extent these two sources contribute to semantic validation, we model their characteristics as features to train the classifier.

Distributional similarity as well as relatedness by a derivational rule path refer to pairs of words rather than entire families. However, this is unproblematic: As mentioned in Section 2.3, there are two perspectives on a derivational family; it can either be regarded as a set of lemmas, or as a set of (independent) lemma pairs. For our basic idea of semantic validation, we assume the latter perspective, and determine the distributional similarity as well as the derivational rule information for *pairs of lemmas* taken from the same family. In a subsequent step, we will then propagate this pairwise information to (heuristic) global decisions on the family level.

We start by assessing the quality of DERIVBASE from a semantic point of view, and by reflecting which strategies can be used for its semantic validation (Section 5.1). In Sections 5.2 and 5.3, we analyse the contributions of semantic information from distributional semantics, as well as structural information from our derivational rules. Section 5.4 explains the feature set motivated from these two information sources and the training of the classifier for semantic validation, and reports the performance of our model. Section 5.5 illustrates how we propagate the pairwise information to entire families, including qualitative considerations. Section 5.6 concludes this Chapter.

5.1 Towards Semantic Validation of a Rule-based Derivational Lexicon

In this Section, we assess the validity of our error analysis in Section 4.4, now focusing on semantic aspects. Based on these observations, we establish two working hypotheses with which we define possbile information sources for a semantic validation of DERIVBASE.

5.1.1 Morphological vs. Semantic Relatedness in DErivBase

The evaluation in Chapter 4 is limited in one important respect: It considers all instances of **S** *and* **M** as true positives. In other words, we only evaluated precision and recall for the morphological relatedness of the lemma pairs, but not for their semantic relatedness; opaque and transparent derivations were not distinguished. As we change our perspective in this Chapter, we re-evaluate DERIVBASE with respect to only *transparent* derivations.

Data Basis for Semantic Validation. Recall our P- and R-samples explained in Sections 4.3.1 and 4.3.3, consisting of 2,545 and 2,000 lemma pairs, respectively. Each lemma pair is classified into one of five categories, of which only **S** attests derivational and semantic relatedness. That is, the samples can be interpreted as providing *binary semantic information*: Either a lemma pair is connected by a transparent derivation (**S**),

"Positive" class	Precision %	Recall %
S and **M**	85.1	91.4
S only	74.6	93.8

Table 5.1: Evaluation of DERIVBASE (v1.4.1), once with a morphological and once with a semantic perspective on the "positive" class

or not (**M** – opaque derivation, **N**, **C**, **L** – unrelated; note the difference to the binary partitioning in Section 4.3.3, in which also **M** counted as positive).

Table 5.1 shows a new precision and recall evaluation of DERIVBASE (again measured on the P- and the R-sample, respectively), this time focusing on the **S** instances. Recall that in the sampling procedure, we relied on rule-based derivational families and string distance measures (cf. Sections 4.3.1 and 4.3.3), so that we are unlikely to encounter lemma pairs that are semantically, but not morphologically related (i.e., synonymous pairs such as *couch* and *sofa*).

The first line in the Table, in which **S**- as well as **M**-labelled lemma pairs are accepted as true positives, is a repetition of Table 4.9, i.e., it merely evaluates the morphological quality of DERIVBASE. The scores change substantially when only truly semantically related **S** pairs count as true positives. Recall increases by 2.4%, because the R-sample contains more **M** cases among the lemma pairs not covered by DERIVBASE than **S** cases relative to the total number of each label; as can be seen in Table 4.8, the R-sample contains far less **M** than **S** pairs, so that missed **M** instances have more impact. In contrast, precision drops by 10.5%, as all covered **M** pairs count as false positives: About one quarter of all pairs in the lexicon are *not* semantically related. That is, DERIVBASE cannot be straightforwardly applied to collect information about semantic relatedness, as it provides many opaque derivations. We see here a clear room for improvement of the semantic meaningfulness of the lexicon. As we address such an improvement by filtering the derivational families, it relates to the *precision* of DERIVBASE, while recall issues (i.e., uncovered lemmas) are out of scope. Thus, the P-sample will form the basis of all our observations in this Chapter, while the R-sample is irrelevant.

We aim at achieving the semantic validation with supervised machine learning methods, which means that we need to keep an unseen part of the data for testing. Thus, we randomly divided the P-sample into a development and a test partition (70:30 ratio, corresponding to 1,780 and 763 lemma pairs, respectively[3]). The distribution of the annotation categories in the whole P-sample and in the development and test partitions, respectively, is shown in Table 5.2 (again, the first line is repeated from Table 4.8). The distribution of the labels is fairly similar across development and test set.

[3]We removed two of the 2,545 pairs due to preprocessing issues.

	S	**M**	**N**	**C**	**L**
Frequency overall	1899	265	240	8	131
Frequency on dev. set	1345	184	159	6	86
Frequency on test set	554	81	81	2	45
Percentage overall	74.6	10.4	9.4	0.3	5.2
Percentage on dev. set	75.6	10.3	9.0	0.3	4.8
Percentage on test set	72.6	10.6	10.6	0.3	5.9

Table 5.2: Class distribution in the P-sample

Table 5.2 shows that most semantic errors are **M**- and **N**-labelled pairs: Each class accounts for around 10% of the pairs. **M** mostly refers to semantic drifts or different senses of the same stem, while **N** is often due to rule overgeneration (cf. Section 4.4.3). These two cases are the main focus of our semantic validation, making up over 78% of the false positives; we consider the few **C** cases negligible for semantic validation, and we also do not focus on the lemmatisation errors **L** (about 5%), as they depend on the preprocessing tools and are only partly attributable to the lexicon induction.

5.1.2 Hypotheses for Semantic Validation

The analysis in the preceding Section shows that DERIVBASE contains a substantial number (around one fourth) of semantically unrelated lemma pairs, an issue that we address with *semantic validation*. By semantic validation, we mean computational procedures to determine, for each derivationally related lemma pair, whether it is in fact semantically related (transparent), and to filter out unrelated (opaque) pairs. For now, we do not target the prediction of entire semantically related families, but only of pairs drawn from them. We consider this a first step towards splitting the morphologically motivated families into smaller, semantically coherent clusters; Section 5.5 illustrates how we transfer the pairwise decisions to entire families.

For the semantic validation, we need indications how to separate semantically related and unrelated lemmas. Recall that the only information available from DERIVBASE is:

1. The two lemmas which are members of a common derivational family[4]

2. The sequence of (one or more) derivational rules which connects this lemma pair.

While the mere lemma pair cannot provide indications for semantic validation, we believe that universal semantic information for these pairs taken from distributional similarity (Turney and Pantel, 2010) offers valuable information for our classification.

[4]In fact, we consider pairs of lemma-pattern pairs, as in Chapter 4. In order to avoid confusion of lemma-pattern pairs and sample lemma pairs, we refer to the former as "lemmas".

Also, from the rule path applied per lemma pair, we expect access to indicators for semantic validation: As Sections 4.4.2 and 4.4.3 show, rules behave differently with respect to semantic transparency and correctness. Thus, we base our work for the binary classification on two general hypotheses about these information types:

Hypothesis 1: *Distributional similarity indicates semantic relatedness between derivationally related words.*
The instances of polysemy and semantic drift that we observe – particularly in the **M** class – motivate the use of distributional similarity. As these lemma pairs are semantically fairly different, we expect them also to be distributionally less related than cases of true semantic relatedness.

Hypothesis 2: *Derivational rules differ in their reliability.*
The existence of **M** and **N** pairs indicate that some rules are more meaning-preserving than others. We expect this to be tied to both lexical properties of the rules (e.g., particle verbs are more likely than diminutives to radically change the meaning) and structural properties (e.g., more specific rules are presumably more precise than generic rules).

In the following two Sections, we analyse the lemma pairs in the P-sample with respect to the appropriateness of these assumptions, and then operationalise our hypotheses. It is understood that all subsequent analyses consider only the development portion of the P-sample, while we leave the test portion untouched.

5.2 Analysis 1: Distributional Similarity for Semantic Validation

In distributional semantics, the similarity between two words is deduced from their contextual similarity in large text corpora (Turney and Pantel, 2010). As noted in Section 3.2, distributional information has recently been used to approximate semantic aspects of derivational morphology. In this Section, we investigate whether measuring the similarity of our derivationally related lemma pairs on a standard distributional model provides evidence for their semantic validation. Specifically, we examine whether distributional information helps detecting idiosyncrasies in the semantic drifts that happen through opaque derivation (i.e., errors of the **M** category) as opposed to transparent **S** derivations. Also, we expect a distributional model to identify **N** pairs.

We expect the distributional methods to behave somewhat differently on our lemma pairs compared to other datasets, as most pairs of the P-sample are derivationally related and thus form a specific subsample with respect to the general population of word pairs: Derivationally related words are – using the words of Schütze and Pedersen (1993) – neither syntagmatic associates (co-occurring, like *mashed potatoes*), nor clear paradigmatic parallels (substitutive, such as *cup of tea/coffee*). Also, many distributional approaches investigate sets of words from the same part of speech (e.g., Boleda et al. (2012)), while our lemma pairs exhibit systematical variation in parts of speech. Both

aspects might influence the applicability and performance of distributional methods; to our knowledge, this situation has not been addressed in previous studies.

We begin by explaining how we parametrised the distributional vector space in Section 5.2.1. Then, we elaborate on the two main difficulties we encountered when employing distributional information for semantic validation: Frequency issues (Section 5.2.2), and conceptual issues (Section 5.2.3). Section 5.2.4 proposes an alternative to standard distributional measures that might overcome these issues.

5.2.1 Measuring Distributional Similarity

We examine semantic similarities as predicted by simple bag of words spaces built from the lemmatised SdeWaC (cf. Section 4.2.3). We compute vectors for all lemma-part of speech combinations covered in DErivBase using a window of $\pm$ 5 words within sentence boundaries, and considering the 10,000 most frequent lemma-part of speech combinations of nouns, verbs, and adjectives in SdeWaC as contexts (note that we generalise the inflectional verb and noun patterns from Chapter 4, so that we now use three simple parts of speech). Distributional vectors are built from co-occurrences which are measured with Local Mutual Information (LMI) (Evert, 2005). The semantic similarity is measured by the cosine similarity between the vectors.

Recall from Section 4.2.3 that DErivBase contains all lemmas that occur at least three times in SdeWaC. As a consequence of Zipf's law, the majority of the 280k covered lemmas are very infrequent: About 5% of the lemmas occur three, about 12% up to five, and about 25% up to ten times in SdeWaC, respectively. Due to the versatile inflection in German, it is important to retrieve as many occurrences of each lemma as possible. On the other hand, we aim at acquiring an accurate representation. We thus compare two different lemmatisation techniques:

Conservative lemmatisation: We re-employ the lemmatisation used to induce DErivBase, i.e., we obtain lemmas from the lexicon-based TreeTagger (Schmid, 1994). It lemmatises unrecognised words as $\langle unknown \rangle$, which leads to a sparser, but more reliable vocabulary representation, especially for the open-class words in which we are interested.

Liberal lemmatisation: We use TreeTagger lemmas, but fall back on lemmas and parts of speech produced by the MATE toolkit (Bohnet, 2010) when TreeTagger abstains. MATE is based on a probabilistic model trained on TIGER (Brants et al., 2004). Thus, it has a higher coverage (i.e., it predicts a lemma an part of speech for each token), but is less precise.

5.2.2 Influence of Frequency on Similarity Predictions

Small frequencies are a potential problem when we build distributional representations for all lemmas in DErivBase since it is known from the literature that similarity predictions for infrequent lemmas are often unreliable (Bullinaria and Levy, 2007). Such

unreliabilities even occur in bag of words spaces, where sparsity is less problematic than, e.g., in syntax-based spaces.

To investigate whether the distributional representation of infrequent lemmas is indeed problematic, we manually examine the predictions of our two bag of words models with conservative and liberal lemmatisation: We calculate the cosine similarity for the P-sample lemma pairs on each model, and compare their predictions on the binary gold categories (**S** vs. non-**S**). We also consider the amount of contexts shared by the lemma pair in the respective space as an indication for relatedness. According to Hypothesis 1, we expect semantically unrelated pairs to have lower similarity scores and less shared contexts than semantically related pairs. In the following, we discuss the performance differences of the two lemmatisation strategies.

Frequency Behaviour for Conservative Lemmatisation. Using the bag of words model with conservative lemmatisation, many lemmas of our P-sample pairs share only few or even no dimensions. That is, their cosine is very low or zero, even when they are semantically strongly related. For example, both lemmas in *Drogenverkauf*$_N$ – *Drogenverkäufer*$_N$ (*drug selling – drug seller*) have only nine words as dimensions, and those are completely disjoint.

Another problem is polysemy: If, in a lemma pair (l_1, l_2), l_1 is polysemous, but has a semantic relation to l_2 with its *non-predominant* sense, the shared contexts are not informative enough to reflect this semantic relatedness. For instance, the first lemma of the pair *gravierend*$_A$ – *graviert*$_A$ (*severe/engraving*$_A$ – *engraved*$_A$) has two meanings, where only the predominant sense (*severe*) is very frequent in the corpus. This lemma pair shares a reasonable number of contexts (i.e., 119). However, these dimensions contain intuitively uninformative lemmas (e.g., *to exist*, *broad*, *to link*, *to show*), which is reflected in low LMI scores. Thus, the cosine for the word pair is close to zero.

We observe that such underestimations of semantically related **S** pairs constitute a general trend in the conservative lemmatisation model, assigning very low cosine scores to most **S** pairs. Notably, this phenomenon even affects lemma pairs for which there are sufficient attestations in the corpus. Such low similarity scores are problematic for separating related from unrelated pairs, because their scores are too similar to define a reasonable cutting line between them. Table 5.3 shows that, although the average cosines of semantically transparent and opaque derivations clearly differ, many **S** cosine scores are very low and thus close to **M** predictions in the conservative model.

Frequency Behaviour for Liberal Lemmatisation. In comparison, the bag of words model with liberal lemmatisation returns slightly better similarity estimates. Falling back to MATE when TreeTagger cannot provide a lemma frequently leads to more shared contexts for the lemma pair, and the additional dimensions are often informative. For example, the **S** lemma pair *auferstehen*$_V$ – *auferstehend*$_A$ (*to resurrect*$_V$ – *resurrecting*$_A$) has seven more common dimensions in the liberal model, including the informative words *Jesus*, *Lord*, *myth*, and *suffering*. Correspondingly, the cosine value of this pair rises by 50%. Generally, the amount of zero cosines in the P-sample drops by 45% using the two-

		Conservative lemmatisation	Liberal lemmatisation
S pairs	$cos = 0.0$	5%	3%
	$cos < 0.1$	56%	54%
	$cos < 0.2$	83%	81%
	$cos < 0.3$	96%	95%
	avg. cosine	0.14	0.15
M pairs	$cos = 0.0$	11%	4%
	$cos < 0.1$	92%	93%
	$cos < 0.2$	97%	97%
	$cos < 0.3$	98%	99%
	avg. cosine	0.03	0.03

Table 5.3: Percentage of **S** and **M** lemma pairs with low cosine scores, and average cosine

step lemmatisation compared to conservative one-step lemmatisation. The liberal bag of words model even raises cosine scores for rather infrequent, but semantically transparent lemma pairs like $Wucher_N$ – $wucherisch_A$ ($usury_N$ – $usurious_A$) (cosine 0.28 vs. 0.07) or $welk_A$ – $verwelkend_A$ ($faded_A$ – $fading_A$) (cosine 0.62 vs. 0.23). We also observe decreasing cosines and amounts of shared contexts for semantically unrelated pairs: The cosine score of the **N** lemma pair Arm_N – $Verarmung_N$ (arm_N – $impoverishment_N$) becomes slightly lower (0.006 vs. 0.011), and its shared contexts are drastically reduced (349 vs. 1670) in the liberal model. The fact that **S** pairs are assigned higher similarity scores, but **M** pairs do not, is a desirable behaviour, as it facilitates separating related and unrelated pairs. Table 5.3 shows that the liberal model assigns slightly higher similarity scores to **S** pairs, while the scores for **M** remain stable or slightly decrease (except for $cos = 0$) compared to the conservative model.

The average cosine similarities of **S** and **M** pairs in the two models, however, show similar tendencies. Most improvements achieved by falling back to MATE lemmas are small, and some cosines even changed contrary to our linguistic intuition. For instance, the cosine of the unrelated **N** pair $radeln_V$ – $radial_A$ ($to\ cycle_V$ – $radial_A$) increased from 0.15 to 0.27. Overall, the systematic underestimation of semantically related lemmas and overestimation of unrelated lemmas is alleviated, but only slightly, as the small differences between two models in Table 5.3 show.

In sum, the data confirms our expectations: Infrequent lemmas are indeed problematic for validating the semantic relatedness of our lemma pairs. More specifically, *semantically related* lemmas are systematically underestimated. We believe that the two-step lemmatisation is important for a more robust handling of infrequent words, as it provides at least

word pair l_1, l_2)	context(l_1)	context(l_2)	shared contexts(l_1, l_2)
Überschätzung – überschätzt (*overestimation – overestimated*), $cos = 0.09$	eigen (*own*) warnen (*to alert*) Möglichkeit (*possibility*) führen (*to lead*) Kraft (*force*) Bedeutung (*meaning*) Fähigkeit (*ability*) Leistungsfähigkeit (*capability*) neigen (*to tend*) Einfluß (*influence*)	völlig (*totally*) Problem (*problem*) Gefahr (*danger*) Autor (*author*) weit (*widely*) total (*totally*) ernst (*seriously*) überhöht (*excessive*) gnadenlos (*mercilessly*) Hollywood (*Hollywood*)	völlig (*totally*) Möglichkeit (*possibility*) Bedeutung (*meaning*) Gefahr (*danger*) Einfluß (*influence*) überhöht (*excessive*) Macht (*power*) gnadenlos (*mercilessly*) Kraft (*force*) häufig (*frequent*)
Entertainer – Entertainerin (*entertainer – female entertainer*), $cos = 0.1$	Sänger (*singer*) Schauspieler (*actor*) Musiker (*musician*) Harald (*male given name*) Moderator (*anchorman*) Schmidt (*surname*) groß (*big*) Künstler (*artist*) Talent (*talent*) gut (*good*)	Sängerin (*fem. singer*) Schauspielerin (*actress*) Helga (*fem. given name*) Mutter (*mother*) berühmt (*famous*) brillant (*brilliant*) Lisa (*fem. given name*) Künstlerin (*fem. artist*) verstorben (*deceased*) Talent (*talent*)	Schauspieler (*actor*) Musiker (*musician*) **Talent** (*talent*) bekannt (*well-known*) Sängerin (*fem. singer*) beliebt (*popular*) groß (*big*) berühmt (*famous*) Sportler (*sportsman*) Schauspielerin (*actress*)

Table 5.4: Top ten individual and shared context words for the targets *Überschätzung*$_N$ – *überschätzt*$_A$ and *Entertainer*$_N$ – *Entertainerin*$_N$. Individual context words are ranked by LMI, shared context words by the product of their LMIs for the two target words. Shared context words that occur in the top ten contexts for both words are marked in boldface

a slightly better basis than the conservative lemmatisation for both semantic similarity and dissimilarity estimations. Thus, we rely in the following on the liberal lemmatisation strategy. However, since there are only small gains, a more general question arises; namely whether there are any conceptual difficulties which avoid a beneficial application of distributional knowledge to derivationally related lemma pairs.

5.2.3 Conceptual Influences on Similarity Predictions

In addition to the frequency considerations discussed above, we find three conceptual phenomena that affect distributional similarity independently of frequency aspects: derivation across parts of speech, (semantic) markedness, and subtle semantic drifts.

The first aspect concerns the influence of the parts of speech of base and derived word, respectively, to the usefulness of distributional bag of words model. Our derivation rules comprise a large number of distinct patterns; some of them do not cross part of speech boundaries (gender indicators like *Meister*$_N$ – *Meisterin*$_N$ (*master*$_N$ – *mistress*$_N$) or attenuations like *rot*$_A$ – *rötlich*$_A$ (*red*$_A$ – *reddish*$_A$)), but many do, e.g., *klug*$_A$ – *Klugheit*$_N$

(*clever*$_A$ – *cleverness*$_N$). Naturally, if a derivational rule changes the part of speech of the input lemma, the parts of speech of its context words change as well. Consequently, context overlap decreases. For example, *Überschätzung*$_N$ – *überschätzt*$_A$ (*overestimate*$_N$ – *overestimated*$_A$) is assigned a cosine of merely 0.09. The upper half of Table 5.4 shows the top ten individual and shared context words for this pair, ranked by LMI. The context words of the noun are mainly nominal heads of genitive complements (*overestimation of possibility/force/...*), while the context words of the adjective comprise many adverbs (*totally/widely/... overestimated*). We assume that pragmatic aspects come into play here: If a text aims at evaluating an event, the author may choose an adjectival expression, as it enables an averbial rating. In contrast, the nominal variant may be used in more neutral contexts. Table 5.4 shows that none of the shared contexts ranks among the top ten for both individual target lemmas.[5] This is even more surprising considering that – as opposed to English – German adjectives and adverbs have the same surface realisation (cf. Section 2.2.2). In this way, an accidental match of a context adjective for *Überschätzung* and a context adverb for *überschätzt* would be even more likely.

The second phenomenon that we identified as influencing semantic similarity is *markedness* (Battistella, 1996). A considerable number of derivational rules systematically produce marked terms. A striking example is the feminine suffix *-in* as in *Entertainer*$_N$ – *Entertainerin*$_N$: Although the lemmas are intuitively very similar, their cosine is as low as 0.1. The reason is that the female version tends to be used in contexts where the gender of the entertainer is relevant. This is illustrated in the lower half of Table 5.4. The first two contexts for both *Entertainer* and *Entertainerin* (*singer, actor*) stem from frequent enumerations (*actor and entertainer X*) and are almost identical. But again, the female versions are marked for gender, and therefore inhibit a context match. We also find two female given names in the top ten contexts of *Entertainerin*, and a male given name in the contexts of *Entertainer*. As a result, the target lemmas receive a low distributional similarity.

The third issue are cases of mild semantic drifts at polysemous lemmas that were tagged by the annotators as **S**. The semantic relatedness of such lemma pairs is intuitively clearly recognisable but may be accompanied by pretty substantial changes in the distribution of contexts. Consider, as an example, the semantically related pair *Absteiger*$_N$ – *absteigend*$_A$ (*descender (person)*$_N$ – *descending/decreasing*$_A$). It achieves only a cosine of 0.005, because *Absteiger* is almost exclusively used to refer to relegated sport teams while *absteigend* is used as a general verb of scalar change.

5.2.4 Ranking of Distributional Information

Given the results reported in the preceding analyses of this Section, the standard distributional approach of measuring the absolute amount of co-occurrences does not seem very promising for derivationally related words. We expect other similarity measures than cosine, e.g., the Lin measure (Lin, 1998b), to perform equally poorly since they do not change the fundamental approach. Also, using larger corpora for the semantic space

[5] This trend can be observed even on the top 50 shared contexts.

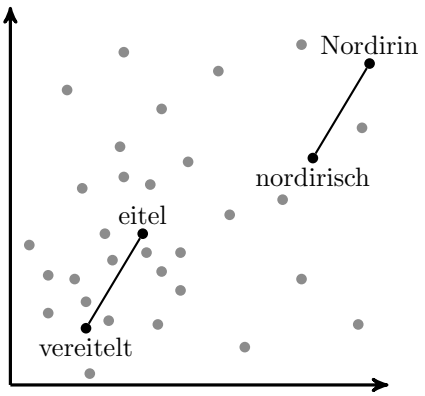


Figure 5.2: Toy space showing differing behaviour for absolute and rank-based measures

construction does not solve the frequency issue, as Zipf's law suggests that exponentially more data would be required for a visible improvement (Lowe, 2001). Thus, we would prefer to make improvements on the modelling side of semantic validation.

To this end, we follow the ideas of Jones et al. (2006), Hare et al. (2009), and Lapesa and Evert (2013), who predict semantic priming using distributional models, but with a slightly different perspective: They propose to consider *semantic similarity in terms of ranks* rather than absolute values. The advantage of rank-based similarity is that it takes not only into account whether a related lemma pair (l_1, l_2) is situated in roughly the same region of the semantic space, but also accounts for the distribution of other words nearby this pair, i.e., the density of regions in the space. That is, low *absolute* similarity does not necessarily indicate low semantic relatedness – provided that the two words are located in a sparsely populated region: *All* similarities involving one of these lemmas might be low, but within those, the score for (l_1, l_2) might be *relatively* high. Conversely, high similarity can be meaningless in a densely populated region.

Figure 5.2 illustrates this phenomenon in an exemplary two-dimensional space (for purposes of illustration, we depict similarity by means of distance). The similarity of the semantically opaque **M** pair $eitel_A$ – $vereitelt_A$ ($vain_A$ – $blocked_A$) and the semantically transparent **S** pair $Nordirin_N$ – $nordirisch_A$ ($Northern\ Irishwoman_N$ – $Northern\ Irish_A$), depicted with a line, is identical:

$$sim(eitel, vereitelt) \equiv sim(Nordirin, nordirisch)$$

However, the pair $eitel_A$ – $vereitelt_V$ is located in a substantially denser region than the pair $Nordirin_N$ – $nordirisch_A$. A rank-based similarity measure takes this difference into account. It assigns a lower rank to the former lemma pair than to the latter, since many

words are closer to the respective target lemma, and thus obtain a higher rank:

$$rank(vereitelt|eitel) = 18$$
$$rank(nordirisch|Nordirin) = 3$$

An additional conceptual benefit of rank-based similarity is that it is *directed*: It is possible to distinguish the "forward" rank (the rank of l_1 in the neighborhood of l_2) and the "backward" rank (the rank of l_2 in the neighborhood of l_1). As to the toy example in Figure 5.2, this means that

$$rank(eitel|vereitelt) = 14 \neq rank(vereitelt|eitel) = 18$$
$$rank(Nordirin|nordirisch) = 4 \neq rank(nordirisch|Nordirin) = 3.$$

The real cosine similarities of these lemma pairs in our distributional model reflect exactly this behaviour: The *raw* cosine is identical, but the **M** pair has a far lower *rank-based* similarity than the **S** pair for both forward and backward rank, indicating its lower semantic relatedness:[6]

$$cos(eitel, vereitelt) = cos(Nordirin, nordirisch) = 0.01$$
$$rank_{cos}(eitel|vereitelt) = 476; \quad rank_{cos}(vereitelt|eitel) = 443$$
$$rank_{cos}(Nordirin|nordirisch) = 227; \quad rank_{cos}(nordirisch|Nordirin) = 367$$

Similarly, the previous studies mentioned above found rank-based similarity to be more beneficial for the prediction of priming effects than absolute cosine. This suggests a refined version of our Hypothesis 1 in Section 5.1.2:

Hypothesis 1': *High **rank-based** distributional similarity indicates semantic relatedness between derivationally related words.*

5.3 Analysis 2: Derivational Rules for Semantic Validation

As discussed in Section 5.1.2, a second source of information is provided by the derivational rules that are encoded in DERIVBASE. More specifically, we want to utilise the linguistic information about derivational processes that is explicitly available in the lexicon. Our intuition is that some rules are always semantically transparent, i.e., that they reliably connect semantically similar lemmas, while other rules tend to cause semantic drifts. To examine this question, we focus our analysis on all lemma pairs in the P-sample that are connected by derivation rule paths of length one, which we call *"simplex paths"*: They are easy to analyse, and make up a majority of the P-sample (about 54%). Longer paths (*"complex paths"*) are briefly considered below.

We find that the rules indeed behave differently. For example, the *-in* female marking rule explained in Section 5.2.3 is very reliable: Every lemma pair connected by this rule

[6] Section 5.4.1 explains in detail how we determine the rank-based similarity.

is semantically related.[7] At the other end of the scale, there are rules that consistently lead to semantically unrelated lemmas, e.g., the *ver-* noun-verb prefixation: *Zweifel*$_N$ – *verzweifeln*$_V$ (*doubt*$_N$ – *to despair*$_V$). Suffixes originating from Latin and Greek like *-ktiv* in *instruieren*$_V$ – *instruktiv*$_A$ (*to instruct*$_V$ – *instructive*$_A$) retain semantic relatedness in most cases, but sometimes link actually unrelated lemmas. For example, *Objektiv*$_N$ – *Objektivismus*$_N$ (*lens*$_N$ – *objectivism*$_N$) is an **N** pair for the suffix *-ismus*. Finally, conversions and very short suffixes are less reliable: Since they easily match (cf. Section 4.4.3), they are often applied to incorrectly lemmatised words (**L**), e.g., the *-n* suffix, which relates nationalities with countries: *Schwede*$_N$ – *Schweden*$_N$ (*Swede*$_N$ – *Sweden*$_N$). It matches many wrongly lemmatised nouns due to its syncretism with the plural dative/accusative suffix *-n*, as in *Schweineschnitzel*$_N$ – *Schweineschnitzeln*$_N$ (*pork cutlet*$_N$ – *pork cutlets*$_{dat/acc\text{-}pl_N}$).

This different behaviour of the rules suggests that *rule-specific reliability* is a promising feature for semantic validation. Thanks to its construction, DERIVBASE provides the rules applied to a lemma pair, so that these reliabilities can be "read off".

For many rules, however, the variance between the individual lemma pairs that instantiate the rule is large, and the question whether the rule is appropriately applied is influenced by the particular combination of rule and lemma pair. We examined various indications for the reliability of such rules, e.g., their overall application frequency, the conducted derivation operation (prefixation, suffixation, etc.), and the part of speech combinations of base and derived lemma. However, none of these properties leads to clear-cut semantic patterns in the data, so that it was not possible to establish concrete directives that would describe a rule's reliability. The fact that the rule occurrences are Zipf-distributed and many rules apply infrequently to the lemma pairs in the P-sample, additionally complicates the analysis.

Regarding word pairs that are linked by "complex paths", i.e., more than one rule (e.g., the lemma pair *eitel*$_A$ – *Vereitelung*$_N$ in Figure 5.1), our main observation is that rule paths clearly show a "weakest link" property. One unreliable rule can be sufficient to cause a semantic drift, and only a sequence of reliable rules is likely to link two semantically related words. Naturally, the longer a derivation rule path is, the less likely the respective lemma pair is semantically related.

In sum, although we could not detect clear-cut patterns from the derivation rules to conduct semantic validation, the rules clearly *do* differ in their reliability. For these reasons, we suggest that distributional knowledge and structural rule information should be combined, a direction that we will pursue in Section 5.4 to approach the semantic validation challenge.

[7] Although many German nouns ending in *-in* do not denote the female analogue of a person-denoting noun, e.g., *Pinguin* (penguin), the corresponding DERIVBASE rule does not match such cases, because there is no lemma **Pingu* from which the allegedly female variant could be derived.

5.4 A Classification Model for Semantic Validation

This Section shows how we implement the semantic validation as a binary classification task. We first present the features used for the semantic validation, motivated from our analyses in the previous Sections (Section 5.4.1), then, we outline the parametrisation of the classifier (Section 5.4.2). Section 5.4.3 reports the results of our classification models.

5.4.1 Features for Semantic Validation

Recall that we classify lemma pairs drawn from derivational families. That is, each instance is a complex object consisting of two lemmas (l_1, l_2). Additionally, we know a derivation rule path connecting the pair. Our analyses of the previous Sections motivate 35 linguistic features that make up the feature vector for a lemma pair in our P-sample. We divide these features into three feature groups: Distributional, derivation rule-based ("structural"), and hybrid features. Table 5.5 lists them and gives a short explanation for each feature. The three groups are explained in detail in the following.

Our observations emerged from three different information levels: lemmas, lemma pairs, and rules. We thus assigned each feature a type, indicating whether it applies to merely one lemma of a pair instance (l), to the lemma pair (p), or whether the feature applies to information from the derivation rules (r).

Some features described in Table 5.5 occur in two variants. There are two reasons for such "twin features": Either there are two different values for the two lemmas that make up the dataset instance (e.g., the frequency of l_1 and l_2, respectively), or the feature concerns rule information, and we treat information from simplex and complex rule paths (cf. Section 5.3) in two ways. Hence, the question arises how to pass these "twin features" to the classifier. Note that the lemmas in our P-sample pairs are not ordered in any respect (for instance, l_1 is not necessarily a derivation rule's base word, and l_2 is not necessarily its derivative). For that reason, we arrange such "twin features" on the basis of their values: One feature defines the minimum of the two scores, and the other defines the maximum. Although the association between a lemma and its respective value gets lost in such a minimum/maximum order (e.g., the higher lemma frequency might originate from l_1 *or* from l_2), we believe that it is crucial to keep the feature vector order-invariant – which is satisfied by such a score-oriented sort.

Distributional Features. Since rule information is not available in our distributional model, all distributional features apply to the lemma or pair level. They are calculated from our BOW model with liberal lemmatisation (cf. Section 5.2.1). As "traditional" features, we use the two normalised *lemma frequencies*, the *absolute cosine similarity*, as well as the normalised *number of shared contexts* for the lemma pair (computed with LMI, cf. Section 5.2.3). According to Hypothesis 1, we expect these features to reflect the following lemma properties: Frequency reflects the strength of lexical association (motivated from observations on the importance of frequency considerations (Bybee (1985, 1988), cf. Section 2.3), cosine reflects the semantic transparency of the pairs, and shared contexts the reliability of this prediction. Additionally, we implement two features

Feature group (# features)	Type	Feature name (# features)	Description
Distributional (6)	l	Lemma frequency (2)	Normalised SdeWaC lemma frequencies of based and derived word
	p	Cosine similarity	Standard cosine lemma similarity
	p	Contexts shared	Number of shared context words
	p	Cos. rank similarity (2)	Rank-based forward and backward similarity
Structural (25)	r	Rule identity (11)	Indicator features for the top ten rules in the dev. set, plus one aggregate feature for all other rules
	r	Rule reliability	Percentage of rule applications to **S** pairs among all applications of the rule in dev. set
	r	Rule frequency rank (2)	Rank-based rule frequency in DErivBase based on simplex and complex paths
	r	Avg. string distance (2)	Minimum and maximum average Levenshtein distance for all rule instances
	p	POS combinations (6)	Indicator features for POS combinations
	p	Path length	Length of the shortest path between the lemmas
	p	String distance (2)	Dice bigram coefficient; Levenshtein distance
Hybrid (4)	r	Average rank sim. (2)	Frequency-weighted average rank similarity of rules (forward and backward)
	p	Rank sim. deviation (2)	Difference between lemma pair rank similarity and average rule rank similarity (forward and backward)

Table 5.5: Features used to characterise derivationally related lemma pairs. "Type" indicates the level at which each feature applies: l=lemma level, p=pair level, r=rule level

for *rank-based cosine similarity* (cf. Section 5.2.4): forward rank similarity (the rank of l_1 in the neighbourhood of l_2), and backward rank similarity (the rank of l_2 in the neighbourhood of l_1). To speed up processing, we compute the forward rank similarity for a lemma pair (l_1, l_2) not on the complete vocabulary, but by pairing l_1 with a random sample of 1,000 lemmas from DErivBase (plus l_2 if it is not included). Backward rank similarity is computed analogously. As Hypothesis 1' states, we expect the rank similarity to provide better similarity predictions than raw cosine.

As indicated above, the "twin" lemma frequency and cosine rank similarity features are ordered according to their scores, i.e., as minimum/maximum lemma frequency and cosine rank similarity, respectively.

Structural Features. The structural features encode properties of the rules and rule paths in DErivBase for the given lemma pair. Most features apply to the level of derivation rules; we will consider those first.

The first feature is the *identity of the rule*: For the ten most frequently applied rules in the development set, we employ (binary) features to indicate their application to a given lemma pair. An eleventh feature denotes the application of other (less frequently applied) rules. With these identity features, we hope to retrieve patterns of semantic behaviour that hold for many lemma pairs, as the respective rules apply frequently.

Next, we implement the *reliability of a rule*, which we estimate as the ratio of its application on **S** pairs among all its applications on the development set. That is, this feature extracts supervised information about the semantic transparency of a rule.

For the other two structural features that apply on the rule level, we take into consideration not only information from the development set, but from all derivationally related lemma pairs in DERIVBASE. On the one hand, we compute a *rule frequency ranking*, representing a rule's total application frequency. That is, we determine how often each rule relates two lemma pairs in DERIVBASE. We expect frequent (i.e., productive) rules being applied more often without semantic drifts (see also the productivity and frequency considerations in Fleischer and Barz (2007, p84f, p224f, p291f)). Application frequency can be easily read off from the DERIVBASE file format that indicates rule paths (cf. Section 4.5). The rule frequency ranking serves as a measure of specificity, since it indicates how often a rule can generally apply to the linguistic material of the German language. We compute it in two versions: on simplex paths, and on all instances (both simplex and complex paths) of the rule in DERIVBASE, trading reliability against potential noise.

On the other hand, we calculate the *average of the Levenshtein distance* between input and output lemmas of all applications of a rule. In doing so, we estimate the complexity of a rule by measuring the average number of string modifications it involves. We assume low string distances to indicate semantic drifts, as simple rules match frequently (cf. Section 5.3).

For lemma pairs linked by complex paths, the question arises how the latter three rule-level features should be computed. Following our observations of the "weakest link" behaviour in Section 5.3, we always select the most pessimistic feature value of the individual rules (i.e., minimum in the case of rule reliability, and rule frequency ranking). This is a cautious choice with which we hope to avoid overestimating complex paths that contain some reliable rules, but still lead to a semantic drift. For average Levensthein distance, we incorporate both minimum and maximum value of the individual rules, as we believe that the range of string modifications might be another useful indicator. As concerns the binary rule identity features, several of these eleven features are set to true for pairs connected by complex paths.

Finally, three more structural features are computed directly at the lemma pair level: their *part of speech combination* (e.g., *"NV"* for $oxide_N$ – $oxidate_V$)[8], the length of the shortest *derivation rule path* connecting the lemma pair, and two string distance measures between the two lemmas: the *Levenshtein and Dice distances*. The intuition behind these features is to find additional indicators for a lemma pair's semantic relatedness that remained undiscovered in our analysis shown in Section 5.3, e.g., interdependencies

[8]We implement six such features, according to the six possible part of speech combinations.

between part of speech combination and rule application frequency with respect to semantic drifts.

Hybrid Features. The hybrid features attempt to combine both distributional and rule-based information. Although such combinations might be learned by the employed classifier, we want to make explicit two of them, one at the rule level and one at the pair level.

The rule-level feature models the *average rank-based similarity* of a specific rule, and is thus a more general counterpart of the purely distributional rank similarity feature on the pair level. To reflect our confidence in the reliability of the output vector, we weight it with the log frequency average over rule instances of the output lemma:

$$avg_{forward\text{-}rank}(rule) = \sum_{(l_1,l_2)\in\, rule} \frac{fwd\text{-}rank(l_2|l_1)\ log(freq(l_2))}{\sum_{(l_1,l_2)\in\, rule} log(freq(l_2))} \tag{5.1}$$

$$avg_{backward\text{-}rank}(rule) = \sum_{(l_1,l_2)\in\, rule} \frac{bwd\text{-}rank(l_1|l_2)\ log(freq(l_1))}{\sum_{(l_1,l_2)\in\, rule} log(freq(l_1))} \tag{5.2}$$

As set of lemma pairs (l_1, l_2) connected by a specific rule, we randomly draw 200 lemma pairs (or less, if the rule has fewer instances) from DERIVBASE that are connected by this rule with a simplex path. The average rank similarity can be considered an unsupervised counterpart to the structural rule reliability feature, as it does not require class labels.

The pair-level feature is the *rank similarity deviation*, i.e., the difference between the rule's average rank similarity and the rank similarity for one specific pair:

$$fwd\text{-}rank\text{-}sim\text{-}deviation(l_1, l_2) = rank_{fwd}(l_2|l_1) - avg_{fwd\text{-}rank} \tag{5.3}$$

$$bwd\text{-}rank\text{-}sim\text{-}deviation(l_1, l_2) = rank_{bwd}(l_1|l_2) - avg_{bwd\text{-}rank} \tag{5.4}$$

That is, this feature measures the rank of a specific pair relative to the rule's "baseline" rank similarity, and indicates how similar and dissimilar a lemma pair is compared to this average. This deviation represents a normalisation of the distributional rank similarity feature on the pair level, as the average rank similarity substracts out outliers caused, e.g., by markedness effects. We thus expect it to constitute another robust similarity measure.

In parallel to the structural features, values for complex rule paths are computed by the most pessimistic rule rank (i.e., the minimum). Since the rank similarity is directional, we compute both hybrid features in two variants, forward and backward. As mentioned in the beginning of this Section, these "twin features" are sorted according to their (minimum/maximum) values in the feature vector.

5.4.2 Classification

Using these 35 features, we train a classifier on the development portion of the P-sample (1,780 training instances, cf. Section 5.1.1), and learn distinguishing semantic relatedness and non-relatedness within derivationally related pairs. For classification, we use a nonlinear model, i.e., a Support Vector Machine (SVM) with a Radial Basis Function (RBF) kernel. Using the RBF kernel allows us to capture potential non-linear dependencies between the features. We rely on LIBSVM (Chang and Lin, 2011), a well-known SVM implementation, within the `scikit-learn` python library (Pedregosa et al., 2011), v0.14. We optimise the C and γ hyperparameters of the SVM model using 3-fold cross-validation on the training data.

5.4.3 Results and Discussion

For evaluation, we applied the trained classifier to the test portion of the P-sample (763 instances; cf. Section 5.1.1). Table 5.6 summarises precision, recall, F_1-score, and accuracy of the classifier for various combinations of features and feature groups. The best results are marked in boldface. Recall that since our motivation is semantic validation, i.e., the removal of false positives, we are in particular interested in improving the *precision* of our predictions. We test significance of F_1 differences among models with bootstrap resampling (Efron and Tibshirani, 1993).

Our baseline is the majority class in the sample, **S**.[9] As we consider the semantic validation as a filtering task and concentrate on the P-sample only, we assume a relative recall, i.e., recall=1.0 if all **S** pairs are retrieved (e.g., by the baseline). Due to the sample's skewed class distribution (cf. Table 5.2), the frequency baseline is fairly competitive (precision 72.6, F_1-score 84.1). We first consider the three most prominent individual features: Distributional similarity measured as cosine and as similarity rank, and rule identity. As expected from our analyses, the cosine similarity on its own is not reliable; in fact, it performs at baseline level. The rank-based similarity already leads to a considerable gain in precision (+5.1%), but only a slight F_1-score increase of 0.6% that is not statistically significant at p=0.05. These results provide empirical evidence for Hypothesis 1' (Section 5.2.4), and emphasise that it is more appropriate than Hypothesis 1 (Section 5.1.2). On the structural side, rule identity alone improves the precision by 2.3%, with an F_1-score increase of even 3.4% (again not significant).

We now proceed to complete feature groups, all of which perform at least 3.4% F_1-score better than the baseline, proving that the features within each group indeed provide valuable information. The hybrid group achieves a remarkably good precision, given that it contains only four features. The gain in F_1 compared to the baseline is statistically highly significant (at p=0.001). The distributional feature group is able to improve 2.1% over the individual rank-based similarity feature in precision (77.7 vs. 79.8), but gains even more in recall (+3.8%). This is sufficient for a significant improvement in F_1 compared to the rank-based similarity alone (+2.8%, significant at p=0.001). Also, the

[9] Note that this baseline is slightly different from the "only **S**" baseline shown in Table 5.1, since we only consider the test part of the sample here.

Validation method	Precision	Recall	F_1	Accuracy
Majority baseline	72.6	**100**	84.1	72.6
Classifier, *only "cosine similarity" feature*	72.6	100	84.1	72.6
Classifier, *only "similarity rank" feature*	77.7	93.1	84.7	75.6
Classifier, *only "rule identity" feature*	74.9	96.9	87.5	74.2
Classifier, *hybrid group*	81.4	94.8	87.6	80.5
Classifier, *distributional group*	79.8	96.9	87.5	79.9
Classifier, *structural group*	82.6	94.0	87.9	81.3
Classifier, *hybrid + distributional groups*	80.7	96.0	87.7	80.5
Classifier, *hybrid + structural groups*	**85.6**	94.0	89.6	84.1
Classifier, *distributional + structural groups*	84.7	97.8	89.4	83.7
Classifier, *all features*	85.5	94.8	**89.9**	**84.5**

Table 5.6: Accuracy, precision, recall, and F_1 for semantic validation on the test portion of the P-sample

structural feature group performs surprisingly well, considering that these features are very simple and most are computed only on the relatively small training set. It yields by far the highest precision (82.6), and even its F_1-score is slightly higher than that of the hybrid group (87.9 vs. 87.6; not significant). We take this as evidence for the usefulness of structural information, as expressed by Hypothesis 2 (cf. Section 5.1.2).

Ultimately, all three feature groups turn out to be complementary. We obtain an improvement in F_1-score and a clear increase in precision for two out of the three feature group combinations. Notably, the combination of hybrid and structural features achieves the best overall precision score, supporting our assumption for the design of the hybrid features: Structural rule information can be propagated to unseen lemma pairs and still remains consistent with the supervised rule-based features. Only the combination of hybrid and distributional features somewhat levels out between the high precision of the hybrid group and the high recall of the distributional group, achieving a slightly lower F_1-score than the best individual group (87.7 vs. the structural group with 87.9).

The best overall result is shown by the combination of all three feature groups. It attains an F_1-score of 89.9, an improvement of 5.8% over the baseline and 2% over the best feature group (both differences significant at $p=0.001$ and $p=0.01$, respectively). Crucially, this model gains 12.9% in precision while losing only 5.2% of recall compared to the baseline. This corresponds to a reduction of false positives in the sample by more than half (from 27.4% to 14.5%) while the true positives were reduced by less than 4% (from 72.6% to 68.8%).

Performance per Annotation Label. We examined the distribution of improvements through the classification for our five gold standard classes. Table 5.7 shows a breakdown of

	S	**M**	**N**	**C**	**L**	total
Gold annotation	554	81	81	2	45	763
Classified as **S**	525	39	18	2	30	614
Classified as **not S**	29	42	63	0	15	149

Table 5.7: Predictions on the test set of the *all features* classifier per annotation class

the predictions by the best *all features* model for each category. Ignoring compounds (**C**), of which there are too few cases for a sensible analysis, we first find that the classifier achieves a high **S** recall. It is also very good in filtering out unrelated cases (**N**), of which it discards around 75%. The model also recognises semantically opaque derivations (**M**) fairly well and manages to remove more than half of these; obviously, the distinction between **S** and **N** is easier than between **S** and **M**, which is why the classifier performs worse here. It has the hardest time with lemmatisation errors (**L**), of which only a third was removed. However, this is not surprising: Lemmatisation errors do not form a coherent category that is easy to retrieve with the features that we have developed. Ideally, lemmatisation errors should be handled in an earlier stage, i.e., during preprocessing.

Classifier Performance vs. Lemma Frequency. In another analysis, we addressed the question whether there are systematic correlations between the performance of our classifier, and the frequencies of the corresponding lemmas in the distributional space. We would hope that the classifier is not sensitive to, e.g., very infrequent lemmas, but that the entire frequency spectrum is equally well classified.

To this end, we compare the SdeWaC frequencies of all lemma pairs in the test set, with respect to whether they are correctly classified by the model, or not. Figure 5.3 depicts these relationships for our best classifier, trained with all 35 features described above.[10] Note that the x- and the y-axis are logarithmic.

The particularly salient lemma pair in the lower left corner is the **L** pair $Herumtapsen_{Nm}$ – $herumtapsen_V$ (*blundering around – to blunder around*) with a frequency of 3 for both lemmas. It is not surprising that it is not correctly classified, since it is only incorrect due to a wrong noun gender for l_1. At the other end of the scale, the particularly high-frequent words at the top and the right margin of the Figures are the **N** pair $Aktivierung_{Nf}$ – $aktuell_A$ ($activation_N$ – $current_A$) with frequencies $l_1 = 4,334$ and $l_2 = 161,088$, respectively, and the **S** pair $Mögliche_{Nn}$ – $möglich_A$ ($possible_N$ – $possible_A$) with frequencies $l_1 = 370,277$ and $l_2 = 3,232$, respectively. Both pairs were correctly classified.

[10] We also examined the correctness of the classifier decisions with respect to all five annotation labels (i.e., correct **S** instances, incorrect **S** instances, correct **M** instances etc.), but this analysis did not yield further insights.

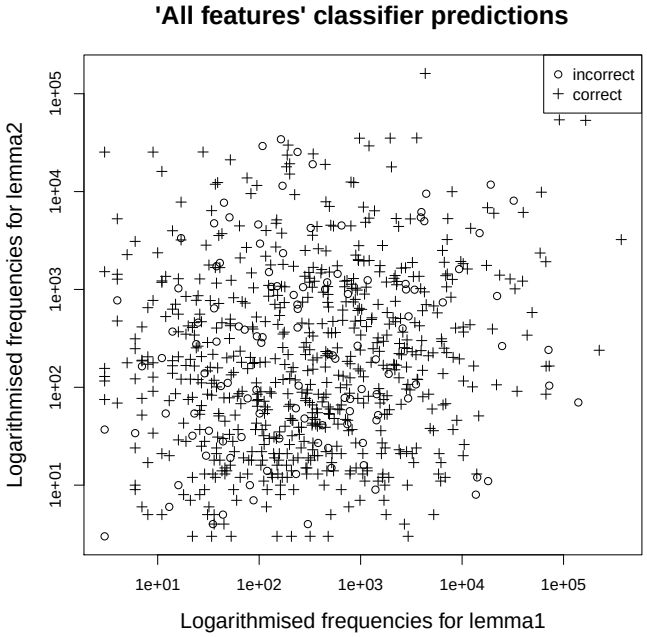


Figure 5.3: Comparison of frequencies for lemma pairs, and the respective decisions of the *all features* classifier

The apparent chaos in this Figure, especially in the middle range, shows that our feature set indeed performs well in all frequency ranges. The only slight tendency one could perceive is that lemma pairs with two highly frequent words are resolved correctly more often; but in sum, our features do not systematically over- or underestimates lemma pairs with a certain frequency. This result again attests the suitability of all our features.

Additional Experiments. We also examined a row of different settings for semantic validation that are not discussed in detail above, but are worthwhile being mentioned:

Other distributional models: For the calculation of the distributional features, we also experimented with a syntax-based space as well as a bag of words model using raw co-occurrence frequencies. However, the former achieved very low coverage, and the latter more unreliable predictions.

Other classification models: Rather than training a binary classification model, we experimented with multi-class classifications on all five labels. However, we found that our dataset contains too few instances of the individual non-**S** classes to properly learn to distinguish these. Also, we tried binary classification with a

linear rather than a nonlinear SVM. The difference to our nonlinear model using *all features* is 1.0% F_1-score, statistically significant at p=0.05.

Other features: We tested the hybrid features based on raw cosine rather than on the rank-based similarity. As expected, they yielded worse results than the rank-based hybrid features so that we discarded them. Additionally, we experimented with two entire feature groups:

HOFM transformation function features: From the transformation functions implemented in HOFM (cf. Appendix A), we extracted various features such as which derivation operations are executed in a rule (e.g., umlaut change or suffixation), or how many operations are involved. In total, we experimented with 17 features (both binary and numeric).

Translation features: The intuition for this feature group is that semantically unrelated words may have completely distinct translations in another language, while words with a similar sense may have similar-looking translations. Consider Example (2.4) on page 20 again: The semantically unrelated lemma pair *einpauken*$_V$ – *Pauke*$_N$ (*to cram into* – *kettledrum*$_N$) has fairly distinct English translations, while the **S** lemma pair *pauken*$_V$ – *Pauke*$_N$ (*to play the kettledrum* – *kettledrum*$_N$) does not. Thus, we implemented three features on the pair level, measuring string distances for the English translations of a lemma pair. The translations were taken from the German-English dictionary `dict.cc`.

In individual tests of these feature groups, we found the translation features to perform only at baseline level, while the transformation function features achieve some improvement over the baseline (precision 79.0; recall 93.7; F_1 85.7; accuracy 77.3). When combining each of these groups with the structural features, results slightly increase or remain stable. In tests with the distributional features, the translation features lead to contradictory decisions (i.e., results remain stable or decrease), while the transformation function features improve the precision slightly (recall decreases, accuracy and F_1 remain stable).

However, none of these two groups lead to improvements exceeding those of the *all features* classifier of Table 5.6. This suggests that they do not contribute additional information to the previous three feature groups. Thus, we refrained from adding the transformation function and the translation features to our feature set.

5.5 From Pairs to Families: Semantic Validation of DErivBase

Having conducted the semantic validation with a machine learning model, and having demonstrated its impact and quality in an intrinsic evaluation, we now take the next step: We move from the classification of word pairs to the segmentation of derivational families in DERIVBASE into smaller, semantically coherent clusters. In this Section, we

explain how we conduct this propagation with clustering methods, pursuing the goal to establish clusters which correspond to the same (transparent) sense.

5.5.1 Clustering Validated Pairs

The goal of clustering is to divide a set of instances into groups (or clusters), so that the instances within the same group are – with respect to some property – more similar to each other than to the instances of different groups (Hastie et al., 2009, 501ff.). A straightforward way to do this is hierarchical clustering, in which a hierarchy of progressively merged clusters is built (for an introduction to this method, cf. Hastie et al. (2009)). The final clusters are determined by a cut-off threshold for the minimum within-cluster similarity. Hierarchical clustering does not require an initial choice of the numbers of clusters (as it is necessary for other clustering strategies), and it is advantageous in one important respect: Due to the resulting hierarchical structure, one can easily test different cuts, corresponding in our task to differently strict definitions of what counts as semantically related.

Acquiring Similarities for Hierarchical Clustering. However, what is needed for hierarchical clustering, is a measure indicating the similarity (or closeness) of two instances, according to which the cluster hierarchy is established. Typically, this measure is some kind of probability score. As described in Section 5.4, we employed an SVM classifier to perform our classification task. SVMs generally do not provide probabilities, but only scores that indicate an instance's distance from the hyperplane. However, there is an algorithm to approximate such probabilities, called Platt scaling (Platt, 1999), which transforms these distances from the hyperplane into a probability distribution over the respective classes by combining SVM and logistic regression methods. The result are scaled probability scores for all instances. Notably, this strategy offers only an approximation to the actual SVM predictions, so that the scaled results might be inconsistent with the original classification for borderline cases. Nonetheless, this method has shown to effectively approximate actual probabilities on various binary classification problems (Niculescu-Mizil and Caruana, 2005).

Platt scaling is implemented in the LIBSVM library we employ. Thus, we can straightforwardly transform the decisions of our binary classifier into probability approximations which indicate the degree of semantic relatedness of a lemma pair.

Compiling Cluster Hierarchies. We apply our trained classification model from Section 5.4 to all derivationally related lemma pairs in DERIVBASE, i.e. all lemma pairs which are in the same derivational family. Version 1.4.1 consists of 716,628 such pairs, for which we calculate the Platt-scaled probabilities.

We apply hierarchical agglomerative clustering (HAC) to these lemma pair probabilities, i.e., we operate bottom-up: We start with singleton clusters and merge, step by step, pairs of clusters until we conflated the singletons to the original derivational families. At each merge, the pair probabilities between the newly added lemmas and the lemmas in

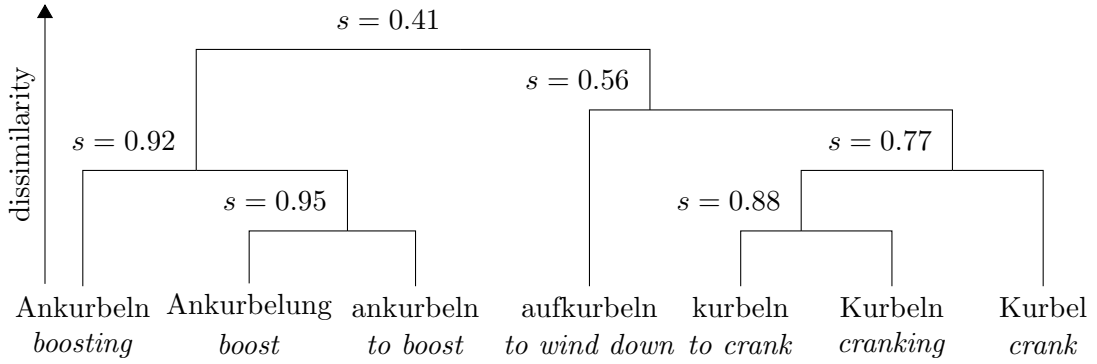


Figure 5.4: Dendrogram of a derivational family in DERIVBASE v1.4.1 (hierarchical agglomerative clustering with average linkage)

the cluster are used to assign the cluster an overall *cluster similarity*.[11] These cluster similarities are calculated differently for different linkage criteria. We experiment with the three most common linkage types: complete, single, and average linkage.

The structure resulting from HAC is usually illustrated by a dendrogram. Figure 5.4 shows the dendrogram of one derivational family. The cluster similarities s quoted at each merge are calculated with average linkage.[12] As can be seen, cluster similarity drops as more lemmas are added. The dendrogram captures the semantics of the family very well: Three lemmas describing a "boost" are clustered together at a high similarity ($s = 0.92$); in fact, we consider them pairs of category **S**. The verb *aufkurbeln* (to wind down) is only moderately related ($s = 0.56$) to three other **S** lemmas describing actions with a "crank". This complies with our linguistic intuition, as *aufkurbeln* is a specific way of cranking, e.g., at a car window, but it can still be regarded as an **S** candidate compared to the "crank" lemmas. Finally, the two clusters describing a boost and actions with a crank are merged with a relatively low similarity ($s = 0.41$), which is a desirable outcome, given the fact that the two clusters are – synchronically – semantically opaque (category **M**).[13]

5.5.2 Building Semantic Clusters

Now that we have hierarchical clusters for each derivational family at hand, we need to divide them into semantically coherent clusters, and evaluate their performance.

Defining the Cut-off Threshold. In order to produce semantic clusters, we need to decide where to apply cuts within the cluster hierarchies. As mentioned in Section 5.5.1,

[11] Note that cluster similarities are not identical with the probabilities for individual lemma pairs.
[12] We omit cluster similarity for singletons, as it is always $s = 1$.
[13] Diachronically, *ankurbeln* was used to describe the act of giving an impulse to, e.g., a motor with a mechanical crank.

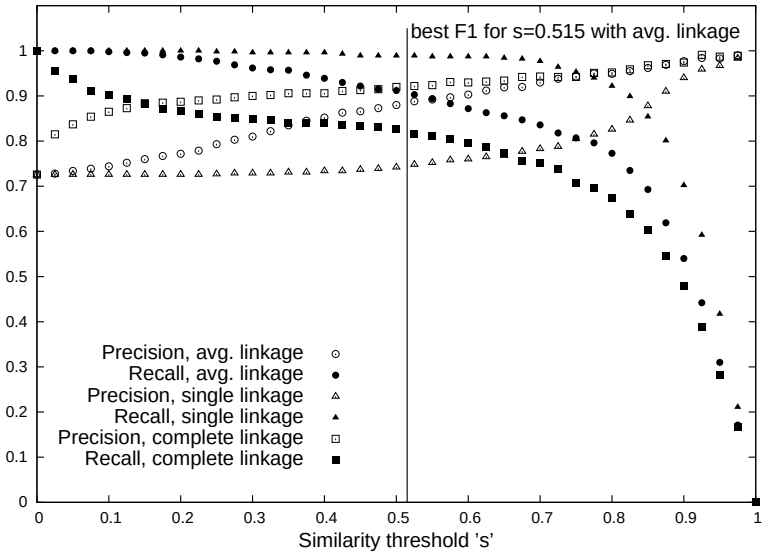


Figure 5.5: Precision and recall curves for different linkage strategies and cluster similarity thresholds (s) on the P-sample test set

HAC offers the possibility to comfortably test various such cuts. Typically, one determines a cut-off threshold for the cluster similarity s, which is optimised on some test data. We do so as well, which means that we define a global threshold that is applied to all derivational families.

To this end, we re-use the test portion of our P-sample (cf. Section 5.1.1): We apply HAC to all 716,628 pairs contained in DERIVBASE, and determine the threshold for s by maximising the F_1-score on the P-sample test data. As mentioned above, we try complete, single, and average linkage, and select the linkage criterion which best separates the **S** and non-**S** pairs of the dataset into different clusters.

Figure 5.5 depicts the precision and recall tradeoff for each linkage criterion across different cluster similarity thresholds.[14] This Figure again illustrates HAC's advantage that it is easy to choose a cut-off threshold depending on whether one focuses on precision or recall. Single linkage has the lowest precision but highest recall, showing that using the nearest neighbour of two clusters to define the cut-off threshold is rather generous. Complete linkage demonstrates the exact opposite behaviour, while average linkage is a moderate balance between the two. We found the best cut-off threshold for cluster similarity to be $s = 0.515$ using average linkage (achieved F_1-score: 89.9%, indicated by

[14]For legibility reasons, we plot s in intervals of .02 rather than in the actually calculated intervals of .005. Precision is not defined for $s = 1$, as there are no singleton clusters in the P-sample.

	Prec.	Rec.	F_1	Acc.
Clustering	88.6%	91.2%	89.9%	85.1%
Classification	85.5%	94.8%	89.9%	84.5%

Table 5.8: Predictions on the test set after clustering vs. with the *all features* classifier

a vertical bar in Figure 5.5). That is, lemma pairs that are assigned a slightly higher probability than equiprobability of semantic relatedness and non-relatedness ($s = 0.5$) after Platt scaling, are considered related. The fact that this threshold is close to 0.5 indicates that the SVM classifier did a good job in estimating a class boundary that optimises both precision and recall. Considering the example in Figure 5.4 again, this threshold indeed performs as desired: It separates the two clearly different senses in this derivational family (the boost and the crank sense), leading to two semantic clusters.

Table 5.8 shows the performance of the clustered DERIVBASE version on our P-sample, compared with the results from the binary classifier (taken from Table 5.6). Recall that classification and clustering differ from each other in that the clustering is based on the classifier's decisions, but that the linkage strategy determines the cluster similarities and thus, the cut-off threshold. As a result, lemma pairs which are close to this threshold may be categorised differently by the clustering and the classifier.

The clustering performance on the F_1-score remains stable compared to the classification. Although it is possible to improve the F_1-score through clustering, it is rather unlikely as we build upon the classifier's predictions. Nonetheless, it is a positive result that F_1 does not decline either. Also, it is particularly encouraging for our goal to build a semantically coherent lexicon that precision rises by another 3.1%, however, at the cost of 3.6% in recall.[15] Overall, the performance of the clustered lexicon – which we henceforth call DERIVBASE v2.0 – looks fairly promising.

DERIVBASE v2.0 consists of 235,287 derivational families, out of which 20,371 are non-singletons that group 65,420 lemmas (i.e., 214,916 are singletons). To compare, v1.4.1 consists of 17,314 non-singleton families that cover 69,437 lemmas, i.e., v2.0 contains less lemmas in more families. In the next paragraph, we examine a couple of entire families.

Qualitative Analysis on the Level of Families. The P-sample test set only evaluates the quality of our semantic validation on the basis of lemma pairs. However, we want DERIVBASE v2.0 to be semantically plausible as a whole. In order to verify whether we achieved this goal with our clustering procedure, we made a qualitative analysis of some medium-size morphological families in v.1.4.1, and their semantic clusters in

[15] We also experimented with the more recall-oriented F_2-score in order to leverage this loss, but obtained exceedingly big clusters, so that we sticked to the F_1 optimisation.

DERIVBASE v2.0.[16]

Figure 5.6 shows the five sample families we selected to give an impression of the quality as well as the problems of the semantic sub-structure in DERIVBASE.

In the first family shown, two occurring **N** lemmas were correctly detected and separated into two different clusters. Two lemmas within the cluster containing lemmas about crafting, *umbasteln* and *zurechtbasteln*, are a little controversial. Their sense is clearly connected to the act of crafting; however, they introduce an additional meaning component of improvisation. We find the decision whether these two lemmas should count as **M** candidates difficult, and would accept the semantic clusters as is.

The second family is not clustered at all, which fully corresponds to our linguistic intuition, as all lemmas are concerned with roughly the same concept. This example shows that the clustering is not prone to "oversplitting", although the precision we attested on the P-sample is fairly high.

In contrast, in the third example, the opposite is the case: The semantically coherent cluster concerning the lemma "beard" is split in two. The clustering correctly separated a proper name (*Bartel*), but leaves another false positive in one of the split beard clusters (*Barte*). A glance at the cluster similarities shows that *Barte* is not scored as exceptionally related ($s = 0.67$), however, considerably above our threshold of $s = 0.515$. We attribute this relatively high score to the homography of the lemma *Barte* with the dative case of *Bart*, which was probably incorrectly lemmatised and assigned substantial relatedness to the "beard" lemmas by the distributional features.

The fourth example illustrates that the clustering has a hard time when it comes to several unrelated lemmas. The morphological family induction severely overgenerated, conflating a derivational family with two members (*Ast*, *Hauptast*) with an unrelated **N** adjective (*astral*), and four named entities (two person names, a flower, and a car model). The false positives are assigned surprisingly high similarities to the "branch" family ($0.6 - 0.9$). The clustering did not succeed in properly resolving all of these inconsistencies, but at least separates three unrelated words from the two-member family.

Finally, in the fifth family, the clustering worked perfectly: The two morphologically distinct families about "boxes" and "boxers" (**N**) are correctly separated, and in addition to that, the morphologically related, but semantically unrelated **M** lemma *durchboxen* is also cut off. An analysis of the dendrogram for this family reveals that the cluster similarity of the boxer sense plus *durchboxen* is only slightly below our threshold ($s = 0.50$). In contrast, the cluster similarity of the entire original family is only at $s = 0.39$. This shows that our cut-off threshold turns out to be reasonable also for unseen cases.

In sum, our analysis suggests that the clustering achieves sensible results not only on the pair, but also on the family level. We feel that the propagation of the semantic validation from pairs to entire clusters worked satisfactorily.

[16] We chose families consisting of seven to nine lemmas. We exclude a detailed description of overinflated families, although their quality on the semantic as well as the morphological level clearly improved, because their size makes them not particularly illustrative. Also, we do not analyse very small families, as they are mostly morphologically and semantically coherent and thus do not lead to striking insights.

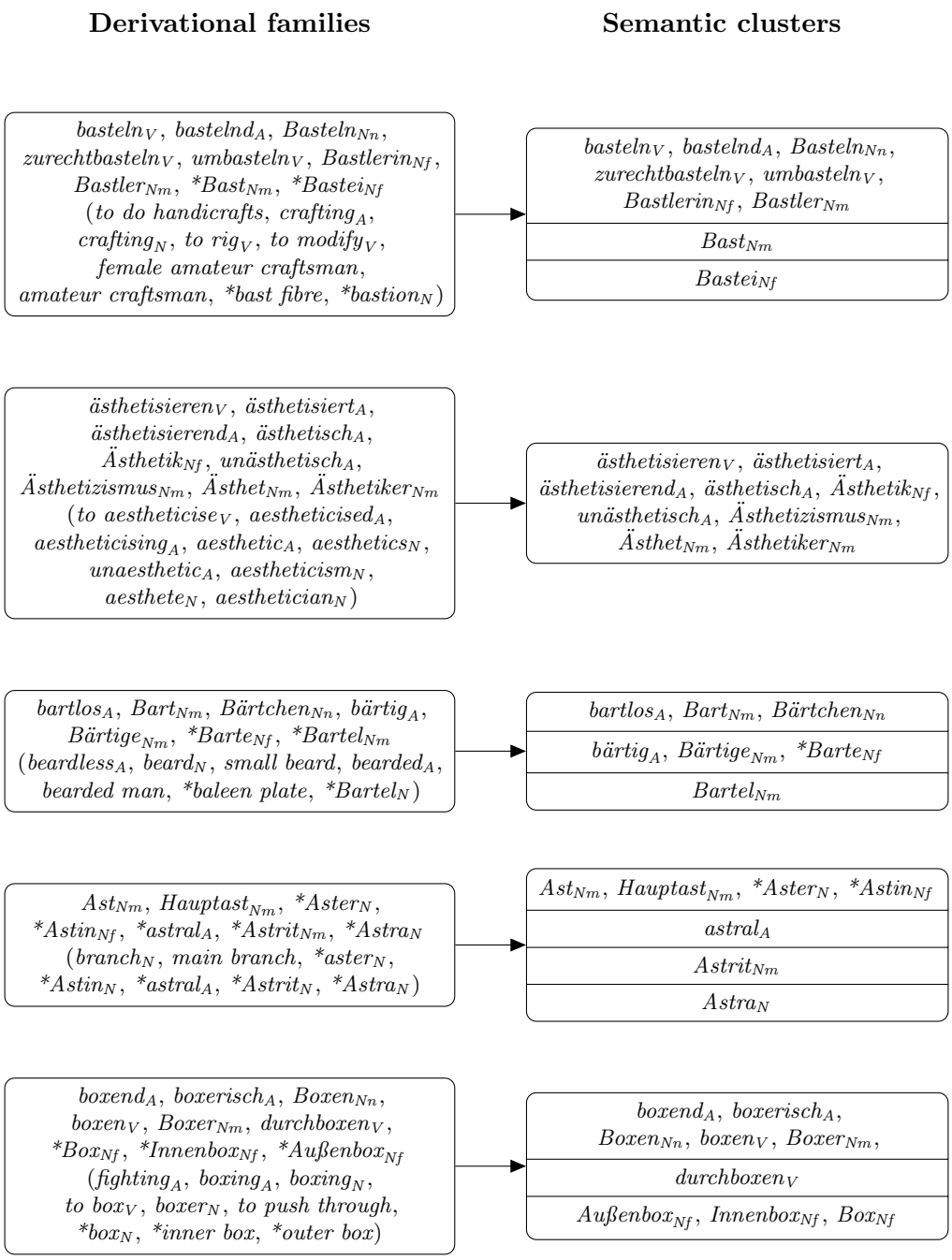


Figure 5.6: Comparison of five purely derivational families, and their respective semantic clusters. False positives are denoted with an asterisk (left: morphologically unrelated wrt. the majority in this family; right: semantically unrelated)

A future direction which one could pursue is to consider soft clustering, i.e., allowing for duplicate entries in various clusters of a family. Consider the last family in Figure 5.6 again: The lemma *Boxer* has (at least) two senses: That of a fighter, and that of a dog. In order to reflect these two senses, a fourth cluster, containing only *Boxer* (in the dog sense) could be added. We think that such a soft clustering would be an interesting endeavour, however, we leave it for future work.

5.6 Summary

Generally, derivational lexicons do not distinguish between only morphologically related and both morphologically and semantically related words. However, such a distinction of opaque and transparent derivations is highly desirable from a methodological point of view. In this Chapter, we have addressed the task of recovering this separation in our German derivation lexicon DErivBase, and called it *semantic validation*. In doing so, we refined our lexicon beyond the Derivational Coherence Assumption (cf. Section 2.1.3) in that not all, but only semantically transparent derivations are considered.

We have made three contributions: 1., we provided a detailed analysis of the information types that are available for this task: distributional similarity and structural information about derivation rules. We also identified problems associated with both sources. 2., we trained a machine learning classifier on a set of linguistically motivated features. The classifier, although not perfect, can substantially improve the precision of the word pairs in DErivBase and thus help to filter the derivational families in the lexicon according to their semantic relatedness. 3., we established a new version of DErivBase, which subdivides morphologically related families into semantically coherent clusters. With this clustering, we successfully separate different senses within one derivational family (**M** cases) as well as accidentally conflated derivational families (**N** cases).

The approach that we have described should transfer straightforwardly to other derivational lexicons and other languages on the conceptual level, e.g., to the Croatian DErivBase version (Šnajder, 2014). The practical requirements are an appropriate corpus (for the distributional features) and derivational rule information (for the structural features). Alternatively, if no derivational rule information is available, the structural and hybrid features that we have proposed can be omitted, and a classifier can be trained with distributional features only. In this way, semantic validation would be also applicable to, e.g., the English CatVar lexicon.

We conducted the semantic validation of DErivBase not only because we find a semantically coherent lexicon more plausible, but also because we hope the validation to improve the usability of derivational knowledge in NLP tasks. In Part III, we will investigate the impact of semantic validation on various applications.

Part III

Using Derivational Knowledge for German

The value of a resource like DERIVBASE is not an end in itself, but is ideally demonstrated by extrinsic evaluations, that is, by showing that it provides benefits to various computational applications. In this Part, we evaluate our lexicon extrinsically by assessing the impact of information added from DERIVBASE in various NLP tasks. Before we present our evaluations in Chapters 6, 7 and 8, we give a brief introduction to how we generally employ our lexicon.

Applying Derivational Knowledge – Rationale

We see potential for applying DERIVBASE in NLP areas related to semantic aspects, since the derivational families incorporate lexical-semantic knowledge. This is not only true for the semantically validated version, but also (to a lesser degree) for purely morphological relatedness, which, according to the DCA, implies semantic transparency in most cases (cf. Section 2.1.3). Thus, we select three – in the broadest sense – semantics-related evaluation topics that cover a range of computational linguistics issues: 1., detection of semantic relatedness; 2., prediction of priming effects; and 3., recognition of textual entailment. In each evaluation, we incorporate DERIVBASE as an additional knowledge resource into an existing basic system that is in principle applicable to the respective task, and demonstrate the system's performance changes. In doing so, we explicitly show the impact of our lexicon, which contrasts with most studies presented in Section 3.1.3. In the following, we briefly outline how we address this incorporation for each task. We conduct all three evaluations on both DERIVBASE v1.4.1 and v2.0 in order to analyse the impact of the semantic validation.

There are many possibilities to apply derivational knowledge. For our evaluations, we experiment with the following three approaches:

1. For semantic relatedness detection, we employ distributional semantic models, and incorporate DERIVBASE by means of *smoothing*: Similarity between two words (l_1, l_2) is not calculated by merely the two vectors representing them, but by the vectors of all words that are in the same derivational families as l_1 or l_2. We propose several parameters to specify the similarity calculation from these sets of of vectors.

2. For a psycholinguistic priming experiment, we again employ a distributional model. Derivational information comes into play as a *morphological generalisation* of the information contained in this model. To this end, we adopt the smoothing idea accordingly: We model priming effects, i.e., the speed-up of a certain representation activation, by incorporating the vector representations of all words in the derivational family of the prime into the similarity calculation. Technically, this method is very similar to derivational smoothing. Conceptually, however, morphological generalisation differs from smoothing in that it is not concerned with the handling of sparse data, but with the activation of morphological representations.

3. Finally, for the recognition of textual entailment, a generic evaluation framework for semantic inference, we integrate knowledge from DERIVBASE into a matching-based

entailment system by means of a *query expansion*: Assuming that derivational relationships between two texts suggest the text pair to be entailing rather than non-entailing, such an expansion increases the chance of a lexical overlap, and could improve entailment predictions.

Our techniques to conduct these three evaluations are similarly applicable to assess other derivational lexicons, such as CatVar. However, we restrict our study on DERIVBASE.

A very different application of a derivational lexicon would be its usage for morphological analysis. Morphological analysers such as SMOR (Schmid et al. (2004); cf. Section 3.1.2) could be used as potential competitors to DERIVBASE (cf. Section 4.5). For instance, one could examine whether the families in DERIVBASE contain all related words that these analysers output; e.g., SMOR's analysis of the lemma *Umverteilung*$_N$ (redistribution) comprises *umverteilen*$_V$ (to redistribute), *Verteiler*$_N$ (distributor), *verteilen*$_V$ (to distribute), and *teilen*$_V$ (to share), and we would expect DERIVBASE to contain at least these lemmas in one family. However, we refrain from such a comparison, because we believe that – as argued in Section 3.1.2 and shown in Section 4.5 – the basic goals of a morphological analyser and a derivational lexicon are too different: Analysers are built to examine the *internal structure* of words, while derivational lexicons model *morphological aspects across words*. Thus, it is questionable whether a comparison of their performance on either task is meaningful.

6 Smoothing Distributional Models for Lexical Semantics with DErivBase

In Section 3.2, we have explained the trade-off between *bag of words* and *syntax-based* distributional semantic models (Turney and Pantel, 2010): While the former is more robust and yields high coverage but only mediocre prediction quality, the latter provides predictions of very good quality, but faces sparsity problems. In this Chapter, we propose a novel strategy for combating sparsity in vector spaces by employing information from derivational lexicons, which we expect to be particularly useful in sparse syntactic spaces. We call our approach *derivational smoothing*. Smoothing is a family of statistical techniques that is used, among others, to provide flexible and robust data statistics by making density estimations more uniform (Simonoff, 1996). In NLP, smoothing is particularly prevalent in language modelling (Chen and Goodman, 1999), where it is employed to improve predictions of statistical models when they run into sparsity problems, i.e., very low or zero counts (Manning and Schütze, 1999, p199).

Our idea of derivational smoothing follows the intuition of the DCA (Section 2.1.3) that derivationally related words are often semantically similar. Consequently, knowledge about derivational relatedness can be used as a fallback for sparse vectors, i.e., for lemmas that occur infrequently in the underlying corpus. For example, the word pair *funnyish – comical* should receive a high semantic similarity, but in practice, the vector of *funnyish* is sparse (in UkWaC (Baroni et al., 2009), a 1.9 billion word corpus, it occurs merely 2 times), which makes its predictions uncertain. Knowing that *funnyish* is derivationally related to *funny* allows us to use the much less sparse vector for *funny* (about 61,800 occurrences) as a proxy for *funnyish*, and thus obtain better similarity estimates from the space. In this way, derivational smoothing brings in information about other members of the concerned derivational family, similar to class-based smoothing in language modelling (Brown et al., 1992). We believe that the high coverage of DErivBase makes it particularly suitable for this use, where it is crucial that the lexicon contains low-frequency lemmas.

We present a set of general methods for smoothing vector similarity computations given our derivational lexicon, and test to what extent they are able to overcome sparsity problems in syntax-based spaces. In a follow-up study, we also investigate whether an even stronger effect can be achieved by complementing (smoothed) syntax-based spaces with (smoothed) word-based spaces. For both studies, we evaluate our models on two German lexical tasks: Similarity prediction and synonym choice. Derivational information is incorporated using DErivBase both as a purely morphological (v1.4.1), and as a semantically validated lexicon (v2.0).

The remainder of this Chapter is structured as follows. Section 6.1 describes our first

study, i.e., how DERIVBASE can be employed to smooth syntactic distributional models. Section 6.2 describes our investigations of the complementarity of information from derivational smoothing and bag of words models. The lexical tasks on which we evaluate both approaches are described in the former, but are equally valid for the latter Section. Section 6.3 summarises our experiments and gives a brief outlook on the possibilities and expectations of derivational smoothing on high-coverage but noisy bag of words models.

Additional Experiments. The rest of this Chapter will concentrate on the smoothing strategy as sketched above. However, we also tried a fairly different approach to improve distributional models with DERIVBASE: Using derivational information for dimensionality reduction. Our intuition was that derivationally related lemmas are similar enough that we could create a new, lower-dimensional vector space by using the set of derivational families as context dimensions, and, for each family, adding the vectors for all members to obtain the family vector. We executed such a reduction on syntax-based as well as word-based matrices, which substantially reduced the matrices' sparsity (i.e., the number of zero values). However, it did not yield clear improvements on the two abovementioned lexical tasks for neither distributional space and neither DERIVBASE version. In this thesis, we do not elaborate on these dimension reduction experiments; they are described in detail (for DERIVBASE v1.4.1) in Kreutzer (2014).

6.1 Study 1: Impact on Syntax-based Models

In this study, we consider to what extent derivational information can alleviate sparsity problems and possibly increase quality in syntax-based distributional vector spaces. We first motivate our choice of the term "smoothing" by relating our study to smoothing approaches in the literature (Section 6.1.1). Next, we explain how we incorporate derivational information into distributional models to conduct derivational smoothing (Section 6.1.2). Section 6.1.3 presents the setup of our experiments on two lexical-semantic tasks, and Sections 6.1.4 and 6.1.5, respectively, report and discuss the results.

6.1.1 Smoothing Techniques in Related Areas

In language modelling, smoothing techniques are well-established to make predictions for unseen data (mostly, unseen n-grams; Chen and Goodman (1999), Dagan et al. (1999)). Although under a different name, the concept of smoothing also exists in other NLP areas, often conducted through the incorporation of additional resources. For instance, studies in Information Retrieval (IR) often describe smoothing as "query expansion", addressing lexical gaps between query and document by integrating data from lexical resources such as WordNet (Voorhees, 1994, Gonzalo et al., 1998, Navigli and Velardi, 2003). In lexical semantics, smoothing is considered a "generalisation" that is achieved by backing off from words to their semantic classes, again adopted from lexical resources (Resnik, 1996) or induced from dependency-parsed data (Pantel and Lin, 2002, Wang et al., 2005, Erk et al., 2010). Both IR and distributional semantics also use SVD (Deerwester et al., 1990,

Landauer and Dumais, 1997), a mathematical method to find low-rank approximations of word vector matrices, i.e., to smooth and to compact them.

Although distributional information is often *employed* for smoothing, to our knowledge there is little work on smoothing distributional models themselves (apart from the general concept of SVD). We see two main precursor studies for our work: Bergsma et al. (2008) build models of selectional preferences that include morphological features such as the consideration of capitalisation or the presence of digits, in order to narrow down the number of potential arguments of specific verbs. However, their approach is task-specific and requires a (semi-)supervised setting. Allan and Kumaran (2003) make use of morphology by building language models for stemming-based equivalence classes. Our approach also uses morphological information, albeit more precise and productive than stemming, as Table 4.9 on page 74 has shown.

6.1.2 Models for Derivational Smoothing

Our derivational smoothing builds on the DCA, i.e., most derivationally related words are assumed to be semantically related. In the following, we describe how we incorporate the semantic knowledge of DERIVBASE into the distributional representation of derivationally related words. The definition of a derivational smoothing algorithm consists of three parts: a *derivation rule path*, a *smoothing trigger*, and a *smoothing scheme*. In all following evaluations, we refer to these parameters using the terminology defined here.

Model Notations. Given a word w, we use $\vec{w}$ to denote its distributional vector and $\mathcal{DF}(w)$ to denote the set of vectors for the derivational family of w. We assume that $\vec{w} \in \mathcal{DF}(w)$. For words that have no derivations in DERIVBASE, $\mathcal{DF}(w)$ is a singleton set, $\mathcal{DF}(w) = \{\vec{w}\}$.

Smoothing Parameters. We first introduce the concepts underlying the three parameters: The binary *derivation rule path* determines whether the rule paths between family members are used to weight the respective word pair in the smoothing procedure. The *smoothing trigger* is a binary function that defines whether a word pair should undergo derivational smoothing, or not. The *smoothing scheme*, in turn, defines the smoothed similarity function, i.e., in which way derivational information from DERIVBASE is included. Formally, smoothing trigger and scheme define the smoothed similarity function of two words, $sim^*(\vec{w}_1, \vec{w}_2)$, by means of the unsmoothed similarity function, $sim(\vec{w}_1, \vec{w}_2)$, on a vector space W, as follows:

$$sim^*(\vec{w}_1, \vec{w}_2) = \begin{cases} sim(\vec{w}_1, \vec{w}_2) \text{ if } smoTr_{sim}(w_1, w_2) = \text{false} \\ smoSc(\vec{w}_1, \vec{w}_2) \text{ otherwise} \end{cases} \tag{6.1}$$

where $smoTr_{sim} : W \times W \to \mathbb{B}$ is the smoothing trigger for a given similarity measure, and $smoSc : W \times W \to \mathbb{R}$ is the smoothing scheme.

Derivation Rule Path. As regards DERIVBASE v2.0, we assume that the derivational families are semantically coherent. However, v1.4.1 is not semantically validated, and we believe that the plain information about membership in the same derivational family can be further refined. We define a score that quantifies our expectations about the probability of semantic relatedness between two words by means of our derivation rules: As described in Section 4.1.4, each related lemma pair is connected by a derivation rule path. We have observed a weakest link behaviour for lemmas connected by a sequence of rules (cf. Section 5.3). Thus, our confidence in a word pair decreases the longer its rule path is. We reflect this fact by assigning each pair a *confidence score*: $\alpha(w, w') = 1/n$, where n is the length of the shortest rule path between w and w'. For example, the lemma pair *bekleiden*$_V$ – *Verkleidung*$_N$ (*to enrobe*$_V$ – *disguise*$_N$) is connected by three rules:[1] *bekleiden*$_V \overset{VV02^*}{\longrightarrow}$ *kleiden*$_V \overset{VV05}{\longrightarrow}$ *verkleiden*$_V \overset{VN07}{\longrightarrow}$ *Verkleidung*$_N$ *(to enrobe*$_V \rightarrow$ *to dress*$_V \rightarrow$ *to disguise*$_V \rightarrow$ *disguise*$_N$*)*. It is thus assigned the confidence $\alpha = 1/3$, while $\alpha(\textit{bekleiden}, \textit{kleiden}) = 1$.

We define two different path information levels that can be extracted from DERIVBASE: *Plain* only considers the membership of two lemmas in the same derivational family (or semantic cluster); *path* additionally considers the confidence score α of two lemmas of the same derivational family (or semantic cluster).

Smoothing Trigger. As there is no guarantee for perfect semantic similarity within a derivational family, smoothing may also drown out valuable information from the distributional model, which is undesirable. To investigate when derivational smoothing is beneficial, we experiment with two triggers: *alwaysSmoTr* performs smoothing for all lemma pairs; *zeroSmoTr* smoothes only when the unsmoothed similarity of a lemma pair, $sim(\vec{w}_1, \vec{w}_2)$, is zero or undefined (due to w_1 or w_2 not being in the model), and is thus more conservative:

$$alwaysSmoTr_{sim}(w_1, w_2) = \text{true}$$
$$zeroSmoTr_{sim}(w_1, w_2) = \text{true iff } sim(w_1, w_2) = 0$$

Smoothing Scheme. The last parameter is how we incorporate information from derivational families. We present three schemes, all of which apply to the level of complete families. Incorporating whole families is again motivated by the network model of Bybee (1985, 1988); cf. Section 2.3. Figure 6.1 illustrates how the schemes are calculated.

The first two schemes are *exemplar-based*: They define the smoothed similarity for a word pair as a function of the pairwise similarities between all words of the two derivational families. The first one, *maxSim*, checks for the most similar words in the families of the two lemmas to be compared. This scheme is useful, e.g., when the original lemmas are semantically very similar, but differ in their part of speech and do not achieve

[1] The rules are listed in Appendix B. An asterisk indicates that the respective rule is inversely applied.

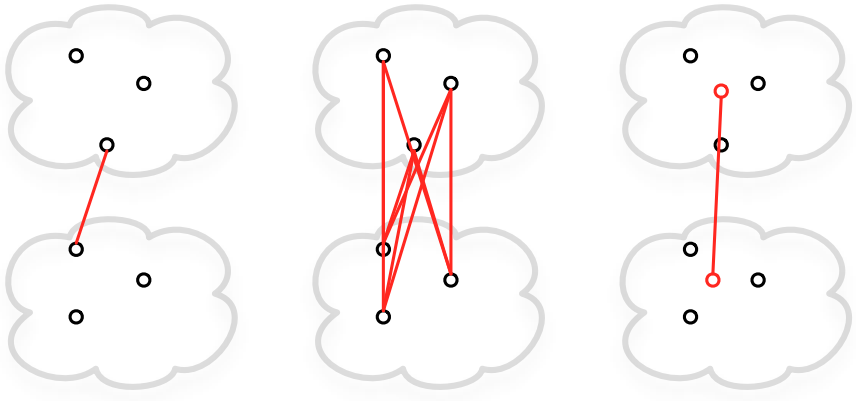


Figure 6.1: Illustration of the three smoothing scheme calculations, given two derivational families. Left: *maxSim*; middle: *avgSim*; right: *centSim*

high similarity due to contextual changes (cf. Section 5.2.3):

$$maxSim(w_1, w_2) = \max_{\substack{\vec{w_1}' \in \mathcal{DF}(w_1) \\ \vec{w_2}' \in \mathcal{DF}(w_2)}} \alpha(w_1, w_1')\, \alpha(w_2, w_2')\, sim(\vec{w_1}', \vec{w_2}') \tag{6.2}$$

where α is the rule path weighting ($\alpha(w, w') = 1$ for *plain* information).

The second smoothing scheme, *avgSim*, computes the average pairwise similarity. Relying not only on one family member, it is less prone to (semantically) incorrect outliers in a family than *maxSim*:

$$avgSim(w_1, w_2) = \frac{1}{N} \sum_{\substack{\vec{w_1}' \in \mathcal{DF}(w_1) \\ \vec{w_2}' \in \mathcal{DF}(w_2)}} \alpha(w_1, w_1')\, \alpha(w_2, w_2')\, sim(\vec{w_1}', \vec{w_2}') \tag{6.3}$$

where N is the number of pairs that can be formed from the two families and have a similarity $\neq 0$. Again, $\alpha(w, w') = 1$ for *plain* information.

The third scheme, *centSim*, is *prototype-based*. It computes a centroid vector for each derivational family, which can be thought of as a representation for the concept(s) that this family expresses:

$$centSim(w_1, w_2) = sim\big(\mathbf{c}(\mathcal{DF}(w_1)), \mathbf{c}(\mathcal{DF}(w_2))\big) \tag{6.4}$$

where $\mathbf{c}(\mathcal{DF}(w)) = \frac{1}{|\mathcal{DF}(w)|} \sum_{\vec{w}' \in \mathcal{DF}(w)} \alpha(w, w')\vec{w}'$ is the centroid vector.[2] *centSim* is similar to *avgSim*, but more efficient to calculate.

[2]Note that the centroid is dependent on the word in question w by means of α (except for the *plain*

In the case of exemplar-based smoothing, the distributional representations of words are unaltered and smoothing is effectively performed on-line. In contrast, for prototype-based smoothing, we first compute a prototype representation for each word by computing the centroid vector of its derivational family, and then compute the similarity between the two prototypes. From a computational perspective, this makes prototype-based smoothing appealing, as it can be processed in advance.

6.1.3 Experimental Setup

This Section describes the two semantic tasks we use as benchmark tests to evaluate our derivational smoothing procedure, and the setup and data basis of our experiments.

Experiments. We evaluate the impact of smoothing on two classical representatives of lexical semantic tasks, where substantial datasets are available for German: semantic similarity prediction, and synonym choice. Both datasets can be downloaded from `http://goo.gl/bFokI`. On these datasets, we measure the performance of a standalone syntax-based model, and to what extent derivational smoothing retains or even improves its predictions while increasing coverage. Synonym choice differs from semantic similarity in two respects: It considers a more narrow scope of similarity (i.e., synonymy), and it requires to rank various possibilities and to choose the best one. We expect differences between the two tasks with regard to the impact of derivational smoothing, since the words within derivational families are generally related but often not synonymous in the strict sense. Thus, semantic similarity judgments should profit more easily from derivational smoothing than synonym choice.

Semantic similarity prediction: The first task is the prediction of semantic similarity between words pairs. We utilise the German GUR350 dataset (Zesch et al., 2007), a set of 350 word pairs with human similarity judgments, created analogously to the well-known Rubenstein and Goodenough (1965) dataset for English. Strictly speaking, the pairs cover semantic *relatedness*, a more general concept than similarity: While similarity defines actual resemblance, relatedness covers a broader scope – including similarity, but also relationships such as meronymy, antonymy, or associations (Budanitsky and Hirst, 2006, Patwardhan et al., 2003). However, the creators note that the GUR350 dataset is biased towards strong classical relations (e.g., hyponymy, synonymy (Morris and Hirst, 2004)), because the word pairs were manually selected and humans tend to select highly related pairs. For this reason, the dataset can be considered a test bed for semantic *similarity* prediction. Eight annotators assessed the relatedness of each word pair, assigning it a score on a five-point Likert scale (Likert, 1932) between fully unrelated (0), and fully related (4). The inter-annotator agreement in this task was 0.69. The following three lemma pairs, including their average relatedness score, are extracted from the dataset:

setting), thereby somewhat abusing the concept of centroids.

Absage$_N$ *ablehnen*$_V$ 3.5 (rejection / to refuse)
Agentur$_N$ *Irrtum*$_N$ 0.0 (agency / mistake)
erklären$_V$ *begründen*$_V$ 2.5 (to explain / to justify)
Note that the GUR350 data contain lemma pairs across parts of speech, a specificity that might be harder to model for syntax-based distributional models.

We manually lemmatised and part of speech-tagged the dataset (since the dataset provides no context, automatic tagging was not possible), and predict semantic similarity as cosine similarity. We make a prediction for a word pair if both words are represented in the semantic space and their vectors have non-zero similarity.

Synonym choice: The second task focusses on synonyms. We conduct this evaluation on the German version of the Reader's Digest WordPower dataset (Wallace and Wallace, 2005), a challenging dataset that is comparable to the synonym choice portion of the English TOEFL test (Landauer and Dumais, 1997), and contains many foreign and rare words, e.g., *Zabaione*. It consists of 984 target words with four synonym candidates each, one of which is correct. Candidate synonyms also include short phrases. One such 5-tuple item, including an index indicating the correct answer, is illustrated in the following:
booten$_V$: *ausschalten*$_V$ / *neu*$_A$ *starten*$_V$ / *Text*$_N$ *eingeben*$_V$ / *speichern*$_V$: 2
(to boot : to switch off / to restart / to enter text / to save : 2)
Again, we manually lemmatised and part of speech-tagged the dataset, and compute semantic similarity as the cosine between target and a candidate vector. From the four pairs per item, we pick the highest-similarity candidate as synonym. For phrase candidates, we compute the similarity between the target and all constituent words, and take the maximum. We make a prediction for an item if at least one target–candidate pair is represented in the semantic space and the respective vector has a non-zero similarity.

Syntax-based Distributional Model. The German syntax-based distributional model that we smooth represents target words by pairs of dependency relations and context words. More specifically, we use the $W \times LW$ matricisation of DM.DE, the German version (Padó and Utt, 2012) of the English Distributional Memory (Baroni and Lenci, 2010) (cf. Section 3.2). DM.DE was induced from the same corpus we have employed to induce DERIVBASE (SDEWAC (Faaß and Eckart, 2013); cf. Section 4.2.3), lemmatised, tagged, and dependency-parsed with the MATE toolkit (Bohnet, 2010).

Baseline. Our baseline is a standard bag of words vector space (BOW), which represents target words by the words occurring in their context. We want the BOW space to behave similar to a syntax-based space in that it should contain rather strict representations of semantic similarity. Thus, we use a relatively small context word window of ± 2, and extract a space using the 10,000 most frequent noun, verb and adjective lemmas as contexts, reduced to 500 dimensions using SVD. The model was created from the same corpus as DM.DE, and preprocessed with the same toolkit.

Evaluation. We evaluate the coverage of our models and the quality of their predictions. In both tasks, coverage is the percentage of items for which we make a prediction, and quality is measured both on covered and on all items, where the uncovered items are treated as false. For the semantic similarity prediction, we measure quality as the Pearson correlation between the model predictions and the human judgments for covered items, r/cov, and for the whole dataset, r/all. For synonym choice, we measure accuracy (acc/cov and acc/all, respectively), and assign partial credit for ties, following the method described by Mohammad et al. (2007). For both tasks, significance testing is performed with bootstrap resampling (Efron and Tibshirani, 1993) on all items.

Derivational Smoothing. We experiment with all combinations of rule path information, smoothing trigger and smoothing scheme, and employ both DERIVBASE v1.4.1 and v2.0 for both evaluation tasks.[3] We expect the rule paths, although representing derivational processes in a very simple way, to generally improve prediction quality. As to the smoothing trigger, *zeroSmoTr* might have little impact if the underlying space has high coverage, but should take into account the semantic information of the proper space more strongly than *alwaysSmoTr*. The effect of the smoothing schemes is hard to predict, but at least, the three fairly different ideas we have implemented should show different trends. Finally, we expect the semantically validated lexicon to perform better in terms of quality (correlation and accuracy), but to have a lower coverage than v1.4.1.

6.1.4 Results

Results for Semantic Similarity Prediction. Table 6.1 shows the results for the first task on the baseline and the unsmoothed and smoothed DM.DE models (identical baseline results are not repeated for different DERIVBASE versions). While the baseline has expectedly high coverage and low quality, the unsmoothed DM.DE attains $r/all = 0.38$, $r/cov = 0.43$, and a coverage of 60.0%. Smoothing increases the coverage substantially to over 90% with DERIVBASE v1.4.1. As expected, the coverage of v2.0 is lower (about 88%).

As regards the quality, DERIVBASE v2.0 is always numerically superior to v1.4.1, showing that the semantic validation indeed yields improvements in this application. Smoothing all DM.DE lemma pairs (*alwaysSmoTr*) with *path* information from v2.0 and using the *maxSim* smoothing scheme achieves the top correlations of $r/all = 0.45$ and $r/cov = 0.48$, respectively (the difference to the unsmoothed DM.DE model is statistically significant at $p = 0.05$, but the difference to the second best model, i.e., the *centSim* scheme for v2.0 using the *zeroSmoTr* trigger, is not significant). The correlations in this top model outperform the same setting using the *plain* information ($r/all = 0.43$, $r/cov = 0.45$; significance at $p = 0.05$), which means that the rule path-related confidence score α provides additional valuable information. The same trend is visible for v1.4.1: Except for the *avgSim* smoothing scheme, quality always improves through the addition

[3] We always instantiate the smoothing triggers with cosine similarity ($alwaysSmoTr_{cos}$, $zeroSmoTr_{cos}$), so that we abbreviate the notation in the following to *alwaysSmoTr* and *zeroSmoTr*.

Rule path	Smoothing trigger	Smoothing scheme	DErivBase v1.4.1 r /all	DErivBase v1.4.1 r /cov	DErivBase v1.4.1 cov %	DErivBase v2.0 r /all	DErivBase v2.0 r /cov	DErivBase v2.0 cov %
plain	Dm.De, *alwaysSmoTr*	maxSim	.31	.32	90.6	.43	.45	87.7
		avgSim	.33	.34	90.6	.36	.38	87.7
		centSim	.36	.38	90.6	.43	.45	87.7
	Dm.De, *zeroSmoTr*	maxSim	.29	.30	90.6	.42	.44	87.7
		avgSim	.43	.45	90.6	.44	.46	87.7
		centSim	.43	.45	90.6	.44	.47	87.7
path	Dm.De, *alwaysSmoTr*	maxSim	.44	.46	90.6	**.45**	**.48**	87.7
		avgSim	.30	.31	90.6	.35	.36	87.7
		centSim	.39	.41	90.6	.44	.46	87.7
	Dm.De, *zeroSmoTr*	maxSim	.43	.45	90.6	.44	.46	87.7
		avgSim	.42	.44	90.6	.43	.45	87.7
		centSim	.44	.46	90.6	.44	.47	87.7
Dm.De, unsmoothed			.38	.43	60.0	–	–	–
Bow baseline			.34	.34	**96.9**	–	–	–

Table 6.1: Results for the semantic similarity prediction (r/cov: Pearson correlation on covered items, r/all: Pearson correlation on all items, cov: coverage). Best results are marked in boldface. Unsmoothed models only shown in DErivBase v1.4.1 column

of the *path*. We assume that the loss for *avgSim* using path confidence arises from big families that contain outliers with long rule paths, and thus impair the similarity of the compared families.

Considering the smoothing schemes, *avgSim* often performs worse than the other two – except for the *zeroSmoTr* trigger on *plain* information –, and is (as just mentioned) the only scheme that does not benefit from *path* information. The *maxSim* and the prototype-based smoothing scheme (*centSim*), in turn, largely perform on par and yield relatively stable results, particularly if the *path* is available.

Surprisingly, the *alwaysSmoTr* trigger performs partly better than *zeroSmoTr* if the *path* is available, meaning that the weighting with α assigns sensible weights particularly for word pairs that are covered by the unsmoothed model. Although this trigger achieves the best results, it is clearly more risky than smoothing only pairs with zero or undefined similarity: Only the radical *maxSim* scheme yields better results when all items are smoothed. On all other combinations of rule path information, smoothing scheme and DErivBase version, however, the more conservative trigger yields better results. The results for the *alwaysSmoTr* trigger on v1.4.1 *plain* show what happens in rather noisy settings: All smoothing schemes achieve fairly low correlations, two of them even below

both the unsmoothed Dm.De model and the bag of words baseline. Consequently, we suggest that smoothing should only be applied to all lemma pairs when sound smoothing information is available (i.e., semantically validated information, and rule path confidence); the more cautious *zeroSmoTr* strategy is generally to be preferred if the quality of the smoothing data is unknown: After all, the unsmoothed Dm.De model itself contains valuable semantic information, and overriding it with derivational smoothing can be counterproductive. The constantly good results of settings using the *zeroSmoTr* trigger show that the model-owned predictions, whenever available, should indeed be used.

Overall, the two DErivBase versions display the same trends, with one exception: While the *maxSim* smoothing scheme performs exceptionally poorly (again, below the Bow baseline) for the *plain* v1.4.1 settings, it achieves fairly good results in *plain* v2.0, and works generally well on the semantically validated data. This difference is reasonable: The clusters in v2.0 typically do not contain outliers but are semantically coherent and rather small. In contrast, the nearest neighbours of two (generally bigger) families in v1.4.1 might be outliers, so that they lead to spurious high similarities.

In sum, for semantic similarity prediction, derivational smoothing clearly profits from high-quality data, i.e., semantic validation and path confidence weights. The optimal choice of smoothing trigger and scheme depends on this quality; generally, the more cautious *zeroSmoTr* trigger and a balanced scheme (*avgSim*, *centSim*) are recommendable.

To get an impression of the effect of derivational smoothing, consider the lemma pair *beschuldigen*$_V$ – *Mitschuld*$_N$ (*to accuse*$_V$ – *share of blame*) with a mid-range human relatedness rating of 2.5. It has a zero cosine in the unsmoothed Dm.De model, but $cos = 0.5$ in our best smoothed model. Notably, the respective model on v1.4.1 assigns a too high similarity, (e.g., $cos = 1.0$ for *maxSim*), since the lemmas are members of the same family, while v2.0 separated them into two subclusters.

The best previous result on this dataset is, to our knowledge, a cross- and multilingual model by Utt and Padó (2014). It reports slightly lower correlations ($r/cov = .47$, $r/all = .42$) and lower coverage (69%) than in our results.

Results for Synonym Choice. The results for the second task are shown in Table 6.2, reporting the same configurations as for the first task. The unsmoothed model achieves accuracies of 48.2 and 59.7 for all and covered items, respectively, and a coverage of 80.8%. Smoothing increases the coverage by +6 and +5 with v1.4.1 and v2.0, respectively. To give an example, a question item with the target lemma *inferior* is additionally covered by backing off to the derivationally related lemma *Inferiorität* (inferiority).

In terms of quality, all smoothed models show a loss in accuracy on all and covered items. We did expect that the more narrow scope of synonymy is not easy to capture with the broader notion of semantic relatedness in the derivational families, however, the decline is more severe than we have thought. Although the difference in accuracy between the best smoothed and the unsmoothed Dm.De model (measured on all items) is not significant at $p = 0.05$, derivational information obviously misleads the unsmoothed model on this task. For this reason, the settings using conservative smoothing (*zeroSmoTr*) lead to less losses than *alwaysSmoTr*. The best smoothed model in terms of accuracy

Rule path	Smoothing trigger	Smoothing scheme	DErivBase v1.4.1 acc /all	DErivBase v1.4.1 acc /cov	DErivBase v1.4.1 cov %	DErivBase v2.0 acc /all	DErivBase v2.0 acc /cov	DErivBase v2.0 cov %
plain	Dm.De, *alwaysSmoTr*	maxSim	44.4	51.2	86.8	47.3	55.1	85.8
		avgSim	41.0	47.2	86.8	44.8	52.3	85.8
		centSim	43.1	49.6	86.8	47.0	54.8	85.8
	Dm.De, *zeroSmoTr*	maxSim	46.8	54.0	86.8	47.5	55.4	85.8
		avgSim	47.1	54.3	86.8	47.6	55.5	85.8
		centSim	47.2	54.4	86.8	**47.7**	**55.6**	85.8
path	Dm.De, *alwaysSmoTr*	maxSim	47.0	54.1	86.8	47.7	55.6	85.8
		avgSim	39.4	45.4	86.8	44.0	51.3	85.8
		centSim	44.0	50.7	86.8	47.2	55.0	85.8
	Dm.De, *zeroSmoTr*	maxSim	46.8	54.0	86.8	47.3	55.1	85.8
		avgSim	47.3	54.5	86.8	47.5	55.4	85.8
		centSim	47.3	54.5	86.8	**47.7**	**55.6**	85.8
Dm.De, unsmoothed (Padó & Utt 2012)			48.2	**59.7**	80.8	–	–	–
Bow baseline			**51.9**	54.5	**95.2**	–	–	–

Table 6.2: Results on the synonym choice task (acc/cov: accuracy on covered items, acc/all: accuracy on all items, cov: coverage). Best results and best smoothing accuracies are marked in boldface. Unsmoothed models only shown in DErivBase v1.4.1 column

uses *zeroSmoTr* triggering and is calculated with DErivBase v2.0, reporting a loss in accuracy of $acc/all = -0.5$ and $acc/cov = -4.1$. This time, however, the best smoothing scheme is *centSim*, and the *path* seems to be irrelevant, as its addition does not change the result.

As to the different smoothing settings, most trends are identical to those examined on the semantic similarity prediction. We again observe particularly poor results for *alwaysSmoTr* on rather low-quality smoothing data (v1.4.1 *plain*), and a general improvement by adding *path* confidences, except for the *avgSim* scheme. This time, the *zeroSmoTr* trigger does not only outperform *alwaysSmoTr* in most settings (like in the first task), but also achieves the "best" smoothing results. The semantic validation of DErivBase v2.0 again improves accuracy over v1.4.1. In fact, all accuracy scores of v2.0 are higher, i.e., provide more robust predictions.

The results for the three smoothing schemes are less clear-cut on this task: While the prototype-based *centSim* scheme achieves mid-range to good results, there are no clear patterns for the relative performance of the two exemplar-based schemes. We attribute this volatility to the fact that the applicability of derivational smoothing for synonym choice is difficult per se, and that the quality of the smoothing data is more important

than the smoothing scheme.

Derivational smoothing is able to trade accuracy against coverage, but does not yield improvements over the unsmoothed DM.DE on this task. What is more, the bag of words "baseline" significantly outperforms all syntactic models – smoothed and unsmoothed – with an almost perfect coverage and substantially higher accuracy on all items. Notably, the BOW model covers a high number of rare and foreign words (which are frequent in the synonym choice data; cf. Section 6.1.3), e.g., *Zabaione*, or *inferior*, and our model construction using SVD and a small context window leads to good predictions for these words. While it is known that word-based spaces can perform fairly well, our results are contrary to those of, e.g., Peirsman (2008) in that our BOW model performs better than DM.DE on strict semantic similarity such as synonymy.

6.1.5 Discussion

The results demonstrate that derivational smoothing is able to improve the performance of a syntax-based distributional model, increasing coverage substantially and also leading to a significantly higher correlation for semantic similarity prediction – although the underlying syntactic model was created from a substantial corpus (880M tokens). We obtained the best results for an approach that smoothes all lemma pairs using high-quality smoothing information from the semantically validated DERIVBASE v2.0, and rule path-related confidence scoring. However, we recommend to employ the more conservative *zeroSmoTr* smoothing approach if the quality of the derivational information is low or unknown. A comparison of prototype- and exemplar-based schemes did not yield a clear winner. In principle, *centSim* seems to be a robust choice, while *maxSim* is advantageous on high-quality smoothing data.

On the synonym choice task, derivational information degrades performance, and a simple, high-coverage BOW model outperforms all syntax-based models. The differences between the two tasks show that the task of estimating generic semantic similarity profits more from derivational smoothing than the task of estimating specific lexical relations such as synonymy. This result is plausible, given the properties of derivational families: They do not only contain words that are highly similar on a very fine-grained level (such as synonyms), but more generally related words, so that DERIVBASE is more useful for semantic similarity prediction.

In sum, our experiments show that syntax-based spaces can be successfully improved with derivational smoothing, but that less sophisticated bag of words spaces perform surprisingly well. Thus, the question arises to what extent distributional models of any kind (syntax- and word-based) and derivational lexicons provide complementary information, and whether they can be combined to better ensemble models. We will examine this question in Section 6.2.

Additional Experiments. We tested various modifications of the three smoothing parameters presented in Section 6.1.2:

Derivation rule path: Apart from the two derivation rule path levels presented above

(*plain* family membership, and additional confidence scores using the rule *path*), a third possibility is to employ the probabilities obtained from the semantic validation classifier (cf. Section 5.5.1) as confidence scores. That is, each lemma pair is weighted with this probability rather than the path length-based score. Results using such a setting were largely comparable to the *path* results.

Smoothing triggers: The smoothing triggers that we have presented are of course not the only possibilities to determine when smoothing should be conducted. One could, e.g., employ thresholds defined for the similarity scores. We also experimented with such thresholds, but found it hard to determine a sensible margin.

Smoothing schemes: We experimented with other smoothing possibilities, e.g., part of speech-restricted smoothing, but did not yield any improvements over the three simpler models we presented above.

Additionally, we experimented with a different prediction strategy on the synonym choice dataset: Instead of making a prediction for an item as soon as one target-candidate pair is covered (as described above), we predicted only items which are fully covered, i.e., where the target word and all four candidates are represented in the model. While accuracies were comparable to the setting reported above, coverage was obviously significantly worse (about 40%).

6.2 Study 2: Complementarity with Word-based Models

The findings of Section 6.1 suggest that the sparsity of syntax-based spaces is a general problem which can be addressed to some extent by derivational smoothing. Another approach would be to alleviate sparsity by combining syntax-based spaces with other spaces with complementary profiles, e.g., less reliable, but high-coverage spaces such as bag of words models. If this strategy indeed reduced sparsity, it would be interesting whether derivational information contains complementary information which can be additionally included to further improve quality and coverage. In this Section, we investigate this question and conduct a follow-up study to our previous smoothing experiments. We combine different distributional models and derivational smoothing.

First, we describe our model combination method, and how we add derivational smoothing to such a combination (Section 6.2.1). Section 6.2.2 explains how we concretely apply this two-level approach to the same benchmark tasks as in Section 6.1, and Sections 6.2.3 and 6.2.4, respectively, report and discuss the results.

6.2.1 Methods for Combining Vector Spaces

Given the complementarities of bag of words (BOW) and syntactic models (cf. Section 3.2), it seems natural to combine them in a beneficial manner. There are at least two general combination possibilities. On the one hand, there is a research tradition that has developed strategies to unify different input vector spaces into a joint output representation,

assuming that the information provided by the spaces is of comparable quality, but contains different types of information, and can therefore be combined on equal footing – e.g., by dimensionality reduction, feature collation, or even just addition (Andrews et al., 2009, Bruni et al., 2011, Fyshe et al., 2013). On the other hand, one can assume that different spaces exhibit different levels of quality, i.e., that some spaces make more reliable predictions than others, depending on their parametrisation and underlying data. Such different profiles often imply an accuracy-coverage tradeoff among the spaces.

Our work assumes the latter perspective, given the well-known variance in the reliability of differently constructed spaces. We combine the models not at the level of co-occurrence information, but at the level of (normalised) predictions, similar to Utt and Padó (2014) who combine cross-lingual with monolingual syntax-based models. They induce a syntactic model cross-lingually by "translating" existing English models with a bilingual lexicon. The filter effect caused by the translation amplifies the properties of syntax-based models, with a still higher prediction quality at even lower coverage. Thus, they combine the cross-lingual with a monolingual syntactic model, which greatly improves coverage at a minimally reduced quality. We follow Utt and Padó's idea and similarly combine two semantic spaces, however, we employ one syntactic, and one word-based space.

Model combination is a fairly different method to improve the performance of distributional spaces than the derivational smoothing presented in Section 6.1: The inclusion of DERIVBASE conducts smoothing by means of relating information *within one space*, i.e., it recombines predictions of the same model. In contrast, the model combination deals with combining the predictions of *two independent spaces*, i.e., it introduces completely new information, potentially leading to very different predictions.

As proposed in Utt and Padó (2014), we conduct this model combination by *score maximisation*. This strategy is based on the assumption that distributional models are more likely to underestimate than to overestimate semantic similarity, as it is rather improbable that vectors of dissimilar words become similar by chance, while many factors (e.g., preprocessing) can make vectors of similar words dissimilar. In consequence, if a model predicts high similarity for two words, this is likely a signal that should be picked up. We define the MAX similarity predicted by a set of models as the maximal score predicted by any of the individual models. By definition, MAX is order-invariant.

Derivational Smoothing of Combined Models. Since the basic ideas of model combination and derivational smoothing are very different, we expect their impact on distributional models to be complementary. Thus, we incorporate derivational relatedness information – in a similar fashion as in Section 6.1 – as an additional level to the model combination: Both combined distributional models m are employed once in plain format (m), and once using derivational smoothing (m_{dsmo}). Thus, we combine up to four models, and examine whether derivational smoothing provides an improvement beyond model combination, or is redundant with that (arguably simpler) method.

Again, we employ both DERIVBASE v1.4.1 and v2.0. As to the three smoothing parameters, we select the following settings (we experimented with all settings, but report only a part of them due to the high number of possible combinations):

Derivation rule path: We use the *plain* information level, since it constitutes a more universal smoothing parameter than *path* information, and the quality differences were unstable in our experimental results in our first study.

Smoothing trigger: We experiment with both the *alwaysSmoTr* and *zeroSmoTr* triggers. Although *alwaysSmoTr* could interfere with high-quality information from the syntax-based models, we expect the MAX model combination to avoid performance declines, since information from DERIVBASE is only added when it leads to the highest similarity prediction.

Smoothing scheme: We employ *maxSim* and *centSim*, the best schemes from Section 6.1.

6.2.2 Experimental Setup

Distributional Models. As word-based and syntax-based spaces, we reuse the BOW and DM.DE models presented in Section 6.1.3, and run derivational smoothing on both models (DM.DE$_{dsmo}$, BOW$_{dsmo}$). The predictions of the four resulting models are normalised by z-score transformation in order to make them comparable. We use the maximal score (MAX) combination strategy to select a prediction for combined models (cf. Section 6.2.1), and test the following four combinations (illustrated in Figure 6.2):

1. The unsmoothed and smoothed syntax-based model (DM.DE+DM.DE$_{dsmo}$): This combination directly compares our previous smoothing strategy and MAX
2. The two unsmoothed models (DM.DE+BOW): This combination compares the performance of derivational smoothing and model combination
3. The two unsmoothed, and the smoothed syntax-based model (DM.DE+DM.DE$_{dsmo}$+BOW): This combination reveals potential complementarities of model combination and derivational smoothing
4. All four models (DM.DE+DM.DE$_{dsmo}$+BOW+BOW$_{dsmo}$): Smoothing the BOW model might lead to additional (presumably small) gains; this setting is added for the sake of completeness

Baselines. We consider two uninformed baselines, based on the assumption that high-frequency lemmas provide more reliable estimates: *random choice* (for synonym choice) and *frequency*. For the semantic similarity prediction, the frequency baseline predicts the smaller of the two words' frequencies, $min(f(w_1), f(w_2))$, thus constituting a lower bound baseline. For synonym choice, the frequency baseline predicts the candidate with the highest corpus frequency.

Prediction and Evaluation. Again, we compute cosine to measure quality and coverage of semantic similarity judgements on the GUR350 and the synonym choice datasets, re-using the setup of Section 6.1.3. For each individual model, we make a prediction if

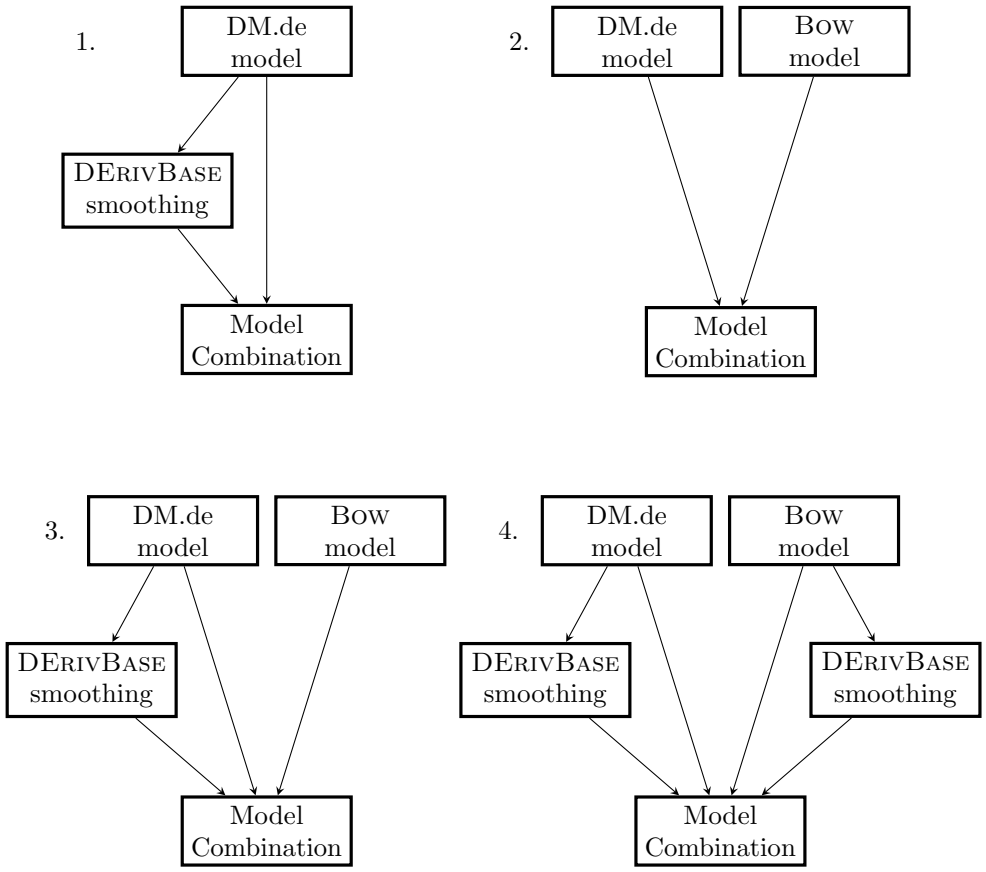


Figure 6.2: Illustration of the four tested settings for model combination and/or derivational smoothing. The numbers correspond to those indicated in Section 6.2.2

both words are represented in the model and their vectors have a non-zero cosine. Again, we test significance with bootstrap resampling (Efron and Tibshirani, 1993) on all items.

6.2.3 Results

Tables 6.3 and 6.4 show, for both tasks, the results of the baselines, the two individual models and their combination, as well as the impact of derivational smoothing on each of these models. Each model combination – also that of an individual model and its derivational smoothing – is conducted with MAX, i.e., the added model is only considered if it increases the predicted similarity; thus, the results of the smoothed individual DM.DE model are not identical to those in Tables 6.1 and 6.2. The Tables indicate the numbers of our four model combinations introduced in Section 6.2.2. As above, identical unsmoothed results are not repeated for different DERIVBASE versions.

Results for Semantic Similarity Prediction. Both individual models outperform the baseline by a large margin. As sketched above, the models differ substantially: The coverage of BOW is higher by almost 37 percentage points. At the same time, the quality of the DM.DE predictions clearly outperforms BOW (e.g., r/cov 0.43 vs. 0.34). The performance of the individual models bears out our assumption about their different profiles (cf. Section 6.2.1).

Comparing the two unsmoothed models and their derivationally smoothed variants, coverage only increases on the sparse syntax-based space (+30%), as BOW already has almost 100% coverage. In consequence, the results on covered items and on all items are almost identical for all models that include the BOW space (i.e., also the combined models), and the *zeroSmoTr* trigger has virtually no effect on the BOW model. As in Section 6.1, DERIVBASE v2.0 boosts coverage slightly less than v1.4.1.

As to the quality, derivational smoothing mostly enhances the unsmoothed models, with two exceptions. The first are losses for smoothing DM.DE using the *maxSim* scheme on DERIVBASE v1.4.1 (e.g., r/all 0.38 vs. 0.36; not statistically significant). This behaviour suggests that semantically not validated derivational information should not be trusted (which happens particularly through *maxSim* and MAX combination). The second exception are minor losses for the *zeroSmoTr* smoothing on BOW with the *centSim* scheme for both DERIVBASE versions. Nonetheless, the *centSim* scheme that actually smoothes over the family members rather than picking the most similar ones, is more suitable for DERIVBASE v1.4.1 – but is also a robust scheme on v2.0. As before, *maxSim*, which is more prone to false positives than *centSim*, profits from semantically validated information, and DERIVBASE v2.0 generally performs better than v1.4.1. Overall, derivational smoothing is more beneficial for the sparser DM.DE than for BOW, both in terms of quality and coverage.

Comparing the derivational smoothing results on DM.DE with those in Table 6.1, we note that the MAX strategy leads to consistently better results than the smoothing we employed in Section 6.1 (i.e., incorporating derivational information whenever the trigger says so). Strikingly, the MAX strategy yields a massive improvement of the *alwaysSmoTr*

Model	Smoothing trigger	Smoothing scheme	DErivBase v1.4.1 r /all	DErivBase v1.4.1 r /cov	DErivBase v1.4.1 cov %	DErivBase v2.0 r /all	DErivBase v2.0 r /cov	DErivBase v2.0 cov %
Frequency baseline	–	–	.13	.13	**100**	–	–	–
Individual models								
Bow (+Bow$_{dsmo}$)	unsmoothed	–	.34	.34	96.9	–	–	–
	alwaysSmoTr	maxSim	.39	.39	96.9	.39	.39	96.9
		centSim	.38	.38	96.9	.38	.38	96.9
	zeroSmoTr	maxSim	.34	.34	96.9	.34	.34	96.9
		centSim	.33	.33	96.9	.33	.33	96.9
Dm.De (+Dm.De$_{dsmo}$; 1.)	unsmoothed	–	.38	.43	60.0	-	–	–
	alwaysSmoTr	maxSim	.39	.41	90.6	**.46**	**.48**	87.7
		centSim	.44	.46	90.6	**.46**	**.48**	87.7
	zeroSmoTr	maxSim	.36	.37	90.6	.44	.46	87.7
		centSim	.45	.47	90.6	.45	.47	87.7
Model combination								
2. Dm.De+Bow	unsmoothed	–	.41	.42	97.1	–	–	–
3. Dm.De +Dm.De$_{dsmo}$ +Bow	*alwaysSmoTr*	maxSim	.40	.41	97.1	.42	.43	97.1
		centSim	.41	.43	97.1	.41	.42	97.1
	zeroSmoTr	maxSim	.36	.37	97.1	.40	.41	97.1
		centSim	.42	.43	97.1	.41	.42	97.1
4. Dm.De +Dm.De$_{dsmo}$ +Bow+Bow$_{dsmo}$	*alwaysSmoTr*	maxSim	.39	.41	97.1	.42	.44	97.1
		centSim	.39	.41	97.1	.41	.42	97.1
	zeroSmoTr	maxSim	.36	.37	97.1	.40	.41	97.1
		centSim	.39	.41	97.1	.39	.40	97.1

Table 6.3: Results for semantic similarity prediction on individual models (above) and model combination (below); r/cov: Pearson correlation on covered items, r/all: Pearson correlation on all items, cov: coverage. Best results are marked in boldface. Unsmoothed models only shown in DErivBase v1.4.1 column

trigger compared to the previous results: It outperforms *zeroSmoTr* on r/all and r/cov in all settings, particularly for the *maxSim* scheme. That is, derivational smoothing and our new maximal score strategy (model combination 1. introduced in Section 6.2.2) fit well together conceptionally, leading to the best overall correlation on all items of $r/all = 0.46$ for the smoothed Dm.De using v2.0 (+.08 over the unsmoothed model; difference stat. significant at $p = 0.05$).

The combination of the two unsmoothed spaces (combination 2.) achieves another slight gain in coverage over BOW (+.02%), and an improvement of +.03 on r/all over the best individual model (DM.DE; stat. significant at $p = 0.05$). This shows that the spaces indeed provide information of different reliability which can be well combined by picking the highest similarity prediction of any model.

Additionally conducting smoothing of DM.DE (3.) leads to small gains of +.01 on covered items in three out of eight settings. That is, smoothing and model combination are indeed to some extent complementary. However, the gains are not statistically significant. High, but incorrect scores of the smoothed DM.DE space using the *maxSim* scheme, the *zeroSmoTr* trigger and v1.4.1 impede the BOW space to improve the results of the syntax-based space. In contrast, *centSim*, which usually does not assign overly high scores, achieves slight improvements using v1.4.1. Although the maximal performance gains of v1.4.1 and v2.0 are similar, the settings including semantically validated data are more stable, suggesting that clean smoothing data is to be preferred to achieve high-quality complementary information from derivational smoothing and model combination.

Finally, let us discuss the model combination using both unsmoothed spaces and their smoothed variants (4.). Once more, correlation slightly rises over the previous setting (3.) by +.01 ($r/cov = .44$) when one chooses the smoothing parameter combination that has proven to be useful before: the *alwaysSmoTr* trigger and the *maxSim* smoothing scheme on DERIVBASE v2.0. However, the difference to the combination in 2. and the best model combination in 3. is again not statistically significant.

As can be seen, the model combinations involving both spaces (DM.DE and BOW) yield improvements over the unsmoothed individual spaces, but they do not outperform the top results achieved by derivational smoothing (setting 1.: smoothing on DM.DE, using DERIVBASE v2.0 and the *alwaysSmoTr* trigger). Nonetheless, the difference between the best combined and the best overall model is not statistically significant.

In sum, the data confirms our assumption from Section 6.2.1 that the MAX combination interacts well with derivational smoothing using the *alwaysSmoTr* trigger, and our general expectation that the two strategies – model combination and derivational smoothing – are to some extent complementary.

Results for Synonym Choice. The results for this task are shown in Table 6.4. Again, all models outperform the baselines, and the two individual models exhibit different profiles: BOW outperforms DM.DE in terms of coverage and – as in Section 6.1 – achieves better quality on all items than the syntax-based space, but DM.DE clearly outperforms BOW on covered items, attaining the highest overall score of $acc/cov = 59.7$.

Derivational smoothing leads to small coverage gains on both individual models, i.e., even DERIVBASE v2.0 adds some items to the BOW space. To give an example, smoothing additionally covers the correct synonym for Suk_N (souk), which is $Markt_N$ (market). Using the *maxSim* scheme, this pair is assigned the highest cosine out of the four candidate pairs, yielding the correct synonym choice as opposed to the unsmoothed BOW model.

Again, using the MAX strategy for derivational smoothing (1.) improves all results on DM.DE compared to those in Table 6.2. In contrast to Table 6.2, accuracy does not

Model	Smoothing trigger	Smoothing scheme	DErivBase v1.4.1 acc /all	acc /cov	cov %	DErivBase v2.0 acc /all	acc /cov	cov %
Random baseline	–	–	25.0	25.0	**100**	–	–	–
Frequency baseline	–	–	31.0	31.0	**100**	–	–	–
Individual models								
Bow (+Bow$_{dsmo}$)	unsmoothed	–	51.9	54.5	95.2	–	–	–
	alwaysSmoTr	maxSim	51.3	53.6	96.9	52.1	54.7	96.8
		centSim	50.9	52.5	96.9	51.2	52.9	96.8
	zeroSmoTr	maxSim	51.9	54.2	96.9	52.0	54.6	96.8
		centSim	50.1	51.7	96.9	50.8	51.8	96.8
Dm.De (+Dm.De$_{dsmo}$; 1.)	unsmoothed	–	48.2	**59.7**	80.8	–	–	–
	alwaysSmoTr	maxSim	46.8	53.9	86.8	48.4	56.4	85.7
		centSim	47.3	54.5	86.8	48.6	56.7	85.7
	zeroSmoTr	maxSim	47.2	54.4	86.8	47.9	55.9	85.7
		centSim	47.4	54.6	86.8	47.9	55.9	85.7
Model combination								
2. Dm.De+Bow	unsmoothed	–	53.7	55.4	97.1	–	–	–
3. Dm.De +Dm.De$_{dsmo}$ +Bow	*alwaysSmoTr*	maxSim	52.1	53.6	97.2	53.1	54.7	97.1
		centSim	52.7	54.3	97.2	**54.1**	55.8	97.1
	zeroSmoTr	maxSim	53.1	54.6	97.2	53.5	55.1	97.1
		centSim	53.4	54.9	97.2	53.5	55.1	97.1
4. Dm.De +Dm.De$_{dsmo}$ +Bow+Bow$_{dsmo}$	*alwaysSmoTr*	maxSim	52.5	54.0	97.2	53.7	55.3	97.1
		centSim	51.5	52.8	97.2	53.3	54.7	97.1
	zeroSmoTr	maxSim	52.8	54.3	97.2	53.4	55.0	97.1
		centSim	51.9	53.2	97.2	52.1	53.4	97.1

Table 6.4: Results for synonym choice task on individual models (above) and model combination (below); acc/cov: accuracy on covered items, acc/all: accuracy on all items, cov: coverage. Best results are marked in boldface. Unsmoothed models only shown in DErivBase v1.4.1 column

always decrease for smoothing Dm.De. More specifically, derivational smoothing leads to gains on all items of up to +.4 (*acc/all* 48.2 vs. 48.6; stat. not significant). Smoothing on Bow increases *acc/cov*, using DErivBase v2.0 and the *maxSim* scheme, while the effect on all items is again small.

As the results show, the behaviour of smoothing schemes and triggers is less clear-cut than for the semantic similarity prediction. Generally, the *zeroSmoTr* trigger seems to be more suitable on DERIVBASE v1.4.1, as it mostly achieves better results than *alwaysSmoTr*, while the exact opposite holds for v2.0. On BOW, the *maxSim* scheme performs better than *centSim*, while the opposite holds for DM.DE. At least, DERIVBASE v2.0 again performs consistently better than v1.4.1, which is plausible given the fact that synonym choice requires semantically more narrow information.

As to the combination of DM.DE and BOW (2.), coverage as well as accuracy on all items increase (cov +1.9% and *acc/all* +1.8 over BOW; difference not statistically significant), again suggesting that the spaces are to some extent complementary.

Including derivational smoothing on DM.DE (3.) leads to the best overall accuracy on all items of $acc/all = 54.1$ for this task (using DERIVBASE v2.0 with the *alwaysSmoTr* trigger and the *centSim* scheme). The quantitative gain by means of derivational smoothing in this setting is identical to that of derivational smoothing only on DM.DE (1.), i.e., +.4 on *acc/all*. While this increase over the pure space combination (2.) is not statistically significant, the difference to the individual BOW model is significant at $p = 0.05$. Unfortunately, this is the only setting in which derivational smoothing on DM.DE actually improves the model combination (3.). Overall, the results using semantically validated smoothing data are more stable, as was the case for the semantic similarity prediction.

Combining all four models (two unsmoothed spaces, and their smoothed variants; 4.) yields mixed results. Only the combination of *maxSim* scheme and *alwaysSmoTr* trigger attains slightly higher (statistically not significant) results than the same parametrisation in the previous setting (3.). This improvement is surprising for v1.4.1, since *maxSim* did not perform well in most other settings using this lexicon version; it might be explicable by the fact that this scheme generally performs better on BOW for synonym choice than *centSim* (see above), so that the four-fold model combination can benefit from *maxSim*.

In sum, derivational smoothing has only a negligible or negative effect on accuracy for synonym choice, even if two distributional spaces are combined. The model combination yields bigger performance gains than our smoothing method. These findings confirm the conclusion of Section 6.1.5: The first task, which requires predicting semantic similarity on a global scale between highly similar and highly dissimilar items, can profit from derivational smoothing. In contrast, the synonym choice involves fine-grained judgments among candidates of a higher similarity degree and does not profit from derivational smoothing. A positive result, however, is that the MAX model combination and derivational smoothing, particularly using the *alwaysSmoTr* trigger on DERIVBASE v2.0, seem to well interact with each other, yielding the (numerically) best accuracy on all items.

6.2.4 Discussion

Our results show that the sparsity of highly accurate syntax-based semantic models can be alleviated by combining them, on the one hand, with models exhibiting complementary profiles such as word-based spaces, and on the other hand with derivational information

by taking into account distributional representations of all lemmas of a derivational family. Both strategies yield coverage improvements and (in large parts) gains in quality. Model combination achieves an improvement over the best individual model of +.03 on *r/all* for the semantic similarity prediction, and of +1.8 on *acc/all* for the synonym choice. Similarly, derivational smoothing improves the best unsmoothed model on *r/all* for the first task by up to +.06, and on *acc/all* for the second task by up to +.2, respectively. Employing both model combination and derivational smoothing achieves the best accuracy on all items for the second task (+2.2 over the best individual unsmoothed model), and a solid Pearson correlation for the first task (+.04 over the best individual unsmoothed model). This suggests that model combination and derivational smoothing provide to some extent complementary information. In parallel to our results in Section 6.1, the synonym choice task does not profit from derivational information alone (nor, as a matter of fact, from model combination alone): Only the interaction of spaces with different profiles and derivational smoothing significantly improve the best individual model.

A surprisingly effective strategy for both tasks is to adopt the maximal prediction made by any model: It clearly improves the derivational smoothing results presented in Section 6.1 in all cases and on both tasks, and it yields the best overall models. We interpret this as evidence for an underlying asymmetry in semantic spaces, where "false positives" (overestimates) are much rarer than "false negatives" (underestimates).

Additional Experiments. We reported only a subset of our experiments with model combination and derivational smoothing. Our major additional experiment was to combine three distributional spaces instead of two: We included the cross-lingual syntactic space of Utt and Padó (2014), tDm, as a very high-quality, very low-coverage model in our combination. Note that this model contains implicit derivational knowledge through the translation step. For instance, the English word *teacher* can be translated into three different German words – *Lehrer* (teacher), *Lehrerin* (female teacher), *Lehrender* (teaching person) – so that the translation step already introduces some derivational smoothing. Still, our smoothing procedure substantially improves the quality of the individual tDm model on both covered and all items on the first task (*r/cov* up to +.15). The quality of the model combinations including tDm rises significantly, suggesting that this space assigns high similarity predictions to actually similar lemma pairs. However, this Chapter focused on the combination of monolingual models in order to clearly differentiate the impact of model combination and derivational smoothing.

As to the interaction of the six models in the three-fold model combination (three unsmoothed and three smoothed models), we also examined the following settings:

Model combination by backoff instead of MAX: We employed the backoff strategy of Utt and Padó (2014), combining the three spaces in the following way: If a word pair is covered by the highest-quality space (tDm), this prediction is used; otherwise, we backoff to the next best space (Dm.De; intermediate quality and coverage) or, as a last resort, use the most noisy space (Bow; highest coverage). However, the Max combination strategy always outperforms backoff.

Derivational smoothing before or after model combination: When using backoff instead of the MAX strategy, the sequence of the model combination is crucial. We tested both derivational smoothing after the distributional spaces have been combined (tDM>DM.DE>BOW>tDM$_{dsmo}$>DM.DE$_{dsmo}$>BOW$_{dsmo}$), and before model combination (tDM>tDM$_{dsmo}$>DM.DE>DM.DE$_{dsmo}$>BOW>BOW$_{dsmo}$). The former approach was superior to the latter on both tasks, suggesting that noisy distributional spaces provide more reliable semantic information than highly accurate spaces expanded by derivational lexicons.

6.3 Summary

The two studies presented in this Chapter investigate how DERIVBASE can be applied to improve distributional semantic spaces: We have proposed a method called *derivational smoothing* which takes into account the distributional representations of all members of a derivational family rather than considering only those of the target words. Our basic idea is that derivational smoothing alleviates the sparsity problem arising particularly for syntax-based spaces. In fact, our smoothing models successfully overcome sparsity in a syntax-based space, and even show quality improvements. Also, derivational information is, to some extent, complementary to the combination of distributional spaces with different profiles, i.e., high-quality syntax-based and high-coverage words-based spaces.

Various parameters determine how the derivational smoothing is conducted. We found a conservative strategy (smoothing only if the two target words to be compared have a zero similarity or are not covered by the model) to work well if this information is mandatorily considered (Section 6.1). However, an even better option is to smooth all target pairs, and to select the maximum of all similarity predictions of the models, i.e., using the MAX combination strategy. The question in which way the incorporation of the derivational family members should be calculated (i.e., which smoothing scheme works best), can not be answered clearly, as this fact also depends on the quality of the derivational information involved, and the task to be solved. Overall, the best and fairly robust results were achieved with the *alwaysSmoTr* trigger, using the DERIVBASE v2.0, and the MAX combination strategy. In sum, DERIVBASE v2.0 achieves better results than v1.4.1, which demonstrates that the semantic validation presented in Chapter 5 is useful for the application of DERIVBASE on semantic tasks.

We have observed that derivational smoothing is not effective for all types of semantic relatedness: Fine-grained relations such as synonymy are too narrow to profit from information from other members of a derivational family, since these families contain generally related words rather than synonyms. Coarser-grained, general relatedness, however, is well reflected in our lexicon, and can be successfully addressed.

Outlook: Smoothing Word-based Models. Since our experiments in Section 6.2 demonstrate that derivational smoothing provides somewhat complementary information to word-based models, another question arises: Can even individual BOW models be effectively smoothed with derivational information to improve their quality? The results for

Bow unsmoothed		Best model shown		Best oracle on Bow	
r/all	r/cov	r/all	r/cov	r/all	r/cov
.34	.34	.46	.48	.76	.80

Table 6.5: Comparison of different models for semantic similarity prediction: The unsmoothed Bow space, the best model of this Chapter, and a smoothing oracle applied to Bow. r/all and r/cov: Pearson correlation on all and covered items

the Bow model variants shown in Tables 6.3 and 6.4 give tentative evidence for such a hypothesis (up to +.05 on r/all for the first, and +.2 on acc/all for the second task, respectively, by means of derivational smoothing), which would be positive particularly for languages for which no adequate parsers are available to construct syntax-based spaces. Generally, we believe that the key issue is how exactly derivational smoothing should be ideally conducted. Although we have tested a variety of parameter combinations, there are still open questions. For instance, the fact that no smoothing scheme consistently yielded the best results suggests that they have complementary strengths and weaknesses.

We made an exploratory study, investigating whether derivational information is capable to smooth word-based spaces when we know which smoothing scheme and trigger to choose. To this end, we implemented a *smoothing oracle*, applied it on the Bow model, and tested it on the Gur350 semantic similarity data. The oracle initiates, for each word pair, the smoothing scheme that most improves the correlation with respect to the dataset's gold annotations, or no smoothing if the unsmoothed Bow prediction best reflects the gold answer. Table 6.5 summarises the performance of the unsmoothed Bow model, the best smoothed model we have presented in this Chapter (derivational smoothing on Dm.De with the *alwaysSmoTr* trigger, DErivBase v2.0 and the Max combination), and the best result achieved by the smoothing oracle (using DErivBase v1.4.1 *plain*). As the numbers show, the oracle boosts correlation dramatically, achieving about +.30 on both r/all and r/cov over our best model.[4] In our test study, we found similar behaviour for predictive distributional models (Mikolov et al., 2013).

That is, derivational information is, in theory, extremely beneficial, but one needs to *learn* when to smooth, and in which way. This fact suggests to employ machine learning methods such as regression or classification models to learn appropriate smoothing operations. We have explored various supervised approaches, but did, thus far, not reveal a clear and promising strategy. Still, we believe that there are ways to learn the appropriateness of smoothing schemes, but this endeavour is out of the scope of this thesis, and we leave it for future work.

[4] Applying the oracle to the Dm.De model achieves merely a maximum of $r/all = 0.59$ and $r/cov = 0.56$, meaning that word-based spaces and derivational information are, in theory, more complementary.

7 Improving Priming Predictions for Psycholinguistics with DErivBase

For our second evaluation, we switch to a very different linguistic field: While Chapter 6 was concerned with lexical semantics, i.e., the meaning of words, we now enter psycholinguistic aspects of derivation, i.e., the perception and representation of derivational processes in the human mind. More specifically, we investigate whether and to what extent a derivational lexicon is helpful for the study of mental representations of morphology.

While there might be many directions in psycholinguistics a derivational lexicon can contribute to, one obvious topic is priming. Priming describes the speed-up effect in the processing of a target word when it is preceded by a related word (the prime); that is, it deals with cognitive and behavioural processes of humans (Meyer and Schvaneveldt, 1971). There are many different priming types, the most important of which is semantic priming, i.e., processing facilitation by means of semantic relatedness of prime and target.

But also, morphologically related words have found to cause priming effects; for instance, the derivative *happiness* "primes" the word *happy*. There is an ongoing debate among psycholinguists about whether only semantically transparent morphological relations (e.g., manage/management; **S** pairs in our terminology, cf. Section 4.3.3), or also opaque relations (e.g., depart/department; **M** pairs) cause priming effects, and what are possible reasons for the respective behaviour. Studies across multiple languages show that overt morphological priming (cf. Section 7.1) leads to a speed-up only for transparent derivations, but not for opaque derivations (e.g., Marslen-Wilson et al. (1994)).

Recently, a controversial study about morphological priming in German was published: Smolka et al. (2014) investigated the overt priming effect of derived prefix verbs (primes) on simplex verbs (targets), e.g., *einfangen*$_V$ – *fangen*$_V$ (*to capture*$_V$ – *to catch*$_V$), considering semantically transparent as well as opaque derivations. They found that both relation types yield significant and, more importantly, comparably strong priming effects, irrespective of other parameters in their experimental setup. In contrast, semantic relatedness did not reliably cause priming effects. These results suggest that German behaves unlike other Indo-European languages in that its lexical representation is 1., based on morphemes rather than lemmas, irrespective of meaning compositionality, and 2., accessed via the base words. Thus, the authors conclude that the organisation of the mental lexicon varies across languages.

Priming experiments have also attracted interest in the computational modelling of psycholinguistic aspects. Many researchers have investigated the modelling of *semantic* priming (Burgess (1998), Landauer and Dumais (1997), McDonald and Lowe (1998), Lowe and McDonald (2000), McDonald and Brew (2004), Jones et al. (2006), to name some

examples from the distributional semantics area). In contrast, *morphological* priming has hardly been addressed so far, which might be partly due to the fact of missing resources for such studies. We think that DERIVBASE can make a valuable contribution in this respect for German: Its representation of derivational relatedness corresponds to the idea of some theories about how the mental lexicon is organised (Bybee (1985, 1988); cf. Section 2.3). Moreover, the two versions of our lexicon reflect the two controversial opinions about the status of opaque relatedness in morphological priming. Thus, we hope that our lexicon can contribute to the computational modelling of priming effects.

In this Chapter, we investigate this aspiration, specifically concentrating on the claim made by Smolka et al. (2014) that German is organised on a morpheme-based level: We present a distributional model particularly designed for priming effects. A distributional space is extended with knowledge about derivational families from DERIVBASE, similar to the models used in Chapter 6. It can be understood as a *morphological generalisation* of the underlying space. We apply this model to Smolka et al.'s dataset, and test whether it can account for the authors' findings, although it does not incorporate any notion of morphemes, which Smolka et al. declared to be essential. In this way, we examine, whether Smolka et al.'s call for novel morpheme-level mechanisms for German, and their claim that cross-lingual differences between morphological systems exist, are justified.

The remainder of this Chapter is as follows. Section 7.1 introduces priming in general, while Section 7.2 specifically focusses on morphological priming. Section 7.3 summarises the experimental study of Smolka et al. (2014) we build upon, and identifies their main results and claims. Section 7.4 describes how we rephrase the idea of derivational smoothing from Chapter 6 as morphological generalisation in order to model the morphological priming effects found by Smolka et al.. Sections 7.5 and 7.6 present our experimental setup and the results, and a discussion, respectively.

7.1 Priming

Priming is a behavioural effect in human language processing: It occurs – in case of positive priming – when the presentation of a stimulus (the *prime*) helps people responding to a second stimulus (the *target*) (Traxler, 2012). This facilitation of subsequent text processing is usually indicated by shorter response times (RTs), and is assumed to result from an activation of a particular mental representation or association. For this reason, priming became a popular method in psycholinguistics to investigate properties of the lexical representations in the human mind, also referred to as the mental lexicon (Jackendoff, 2002). A frequently employed task in priming experiments is the *lexical decision task*, where probands are exposed to a single word as prime, and then have to decide, as quickly as possible, whether the following (single) target is a correct word of the language, or not (Traxler, 2012). For instance, the word *goose* might prime the target word *duck*, as it facilitates access to the respective lexical representation. We will concentrate on the lexical decision task in the following.

There are several parameters in priming experiments that are important influencing factors on the results. Two of them, going hand in hand, are 1., whether the probands

perceive the prime unconsciously or consciously, and 2., the modality how prime and target are presented. As to 1., unconscious perception of the prime is referred to as *masked priming*; it is typically achieved by very short visual exposure of the prime (e.g., on a computer screen, exposure time is measured as "stimulus onset asynchrony", or SOA), and has shown to tap into the very early word processing stage of lexical access (Forster et al., 2003), also called *prelexical stage* (Smolka et al., 2014). In contrast, in *overt priming* experiments, the prime is displayed long enough for the proband to consciously perceive it, thus initiating a different word processing level, the *lexical stage*, where semantic aspects have greater impact (Smolka et al., 2014). Overt priming therefore produces different results than masked priming. The modality, 2., defines the representation format of prime and target. *Visual* and *auditory* modalities are frequently used in priming experiments, i.e., the stimuli are either shown (again, often on a screen), or presented via loudspeakers or headphones. Naturally, auditory modality always involves overt priming. *Cross-modal* priming experiments mix visual and auditory stimuli in order to exclude effects that arise only due to the specific representation (Smolka et al., 2014), e.g., the prime is presented acoustically, while the target is shown on a screen.

Various linguistic aspects are assumed to contribute to priming effects, including semantic, morphological, phonological, orthographical, frequency, or syntactic effects (we will shortly go into detail on some of these aspects). Accordingly, researchers develop priming experiments of very different characteristics to test their hypotheses, and establish theories about the organisation and access of the mental lexicon on the respective findings. Many models – all of them assume semantic associations to play a more or less central role – were proposed according to such theories, e.g., the frequency-ordered bin search model (Forster, 1976), the logogen model (Morton, 1969), the cohort model (Marslen-Wilson, 1987), or the interactive activation and competition network (McClelland and Rumelhart, 1981, Rumelhart and McClelland, 1982). We refrain from an exhaustive discussion of these models, as we are not mainly concerned with the lexical representation, but with the role of morphology in this context.[1] For informative overviews, cf. Traxler (2012), Harley (2008).

Priming Types. As mentioned above, there is a range of linguistic aspects in priming that have been investigated. In this paragraph, we sketch two of them, and then go into detail on morphological priming, the focus of our work.

Most work on priming is concerned with semantic aspects. In their seminal study about *semantic priming*, Meyer and Schvaneveldt (1971) showed two strings simultaneously to the proband, which could be either two words, two non-words, or a word and a nonword. They found that the identification of a word is facilitated if it is preceded by a semantically related word (e.g., *nurse – doctor*), while unrelated primes do not achieve this effect (e.g., *nurse – butter*). Many subsequent studies noticed this effect also in other tasks, modalities, and languages (McNamara, 2005). In this context, it is worthwhile mentioning that there are subtle differences between associativity and semantic relatedness. For instance, the words *cat – mouse* are semantically related as

[1] However, some of these models explicitly take into account morphological aspects.

well as associated to one another, as the one calls into mind the other, whereas the words *fox – camel* are semantically related (both refer to animals), but not associated (Keppel and Postman, 1970). Associated words often cause stronger priming effects than only semantically related words, and most semantic priming experiments work with data that is semantically related as well as associated (cf. Traxler (2012), Harley (2008) for summaries of the discussion of associativity vs. semantic relatedness).

Motivated by the findings of early semantic priming studies, subsequent research identified priming effects also on other linguistic levels. For instance, Bock (1986) observed *syntactic priming*, i.e., that produced sentence structures are guided by the structure of previously perceived sentences. As an example, the probands of this experiment heard and repeated a specific sentence, e.g., *The referee was punched by one of the fans.* Then they were shown a picture that illustrates an event (unrelated to the prime) that they should describe. The produced descriptions followed the syntactic structure of the preceding sentence more frequently than was to be expected – for the above example sentence, a passive construction was framed.

7.2 Morphological Priming: State of the Art

Psycholinguists did not only consider lexical representations of morphologically simple words such as *nurse* or *cat*, but also of complex of words, i.e., inflected words (*spoke*), derivations (*speaker*) and compounds (*loudspeaker*) (Harley, 2008). Addressing the level of morphemes with priming experiments, i.e., investigating speed-up effects by means of morphological relatedness, is referred to as *morphological priming*. In the following, we focus on investigations on *derivational priming*, a subtype of morphological priming. It was found to lead to as strong effects as inflection (Marslen-Wilson et al., 1994, Gordon and Alegre, 1999, Raveh and Rueckl, 2000, Clahsen et al., 2003), e.g., both *believed* and *believer* prime *believe* equally strongly.

Again, there are many theories about how morphologically complex words are mentally organised, the most extreme of which are the "full-listing hypothesis" (Butterworth, 1983), assuming that each word form is stored separately, and the "obligatory decomposition hypothesis" (Smith and Sterling, 1982), assuming a rule-based decomposition (combined with exception lists) of every complex word into stem and affixes. As mentioned above, priming experiments are used to test such theories, which is why morphological priming has become an active research field.

Experimental Setups in Morphological Priming. Typically, morphological priming experiments compare the speed-up effects of prime/target pairs that are morphologically related (*cite – citation*) with pairs that have other relationships, e.g., semantic relatedness (*couch – sofa*), orthographic similarity (*classify – clarify*), or – as a test condition – no relation at all (*bee – piano*). In this context, it is important to keep in mind that morphological relatedness can be either transparent (as in the above example; **S**, cf. Section 4.3.3) or opaque (*depart – department*; **M**). It is then tested whether morphological relatedness facilitates the recognition of words in the lexical decision task. If this was

true, one could assume that complex words are decomposed into their constituents in the lexical representation (Taft and Forster, 1975, Marslen-Wilson et al., 1994).

Elaborate confounders have been used in such experiments, e.g., the combination of actual affixes with real and pseudo-stems (e.g., the prefix *re-* in *rejuvenate* and *repertoire* (Taft and Forster, 1975)), or pseudo-derivations, i.e., word pairs that seem derivationally related, but are not, like *corn – corner* (Longtin et al., 2003).

The choice of masked or overt priming mentioned in Section 7.1 is crucial also for morphological priming, as they address different stages of morphological processing: While masked priming is assumed to activate unconscious (prelexical) processing steps, overt priming is said to activate more conscious (lexical) grammatical and semantic analyses (Smolka et al., 2014).

Basic Findings about Morphological Priming. Although morphological priming is investigated for more than three decades, there are still controversies arising from the experimental results.

For instance, some researchers regard morphology only as an epiphenomenon of form and meaning overlap, which also explains why facilitation effects increase with the degree of semantic transparency of morphologically related words (e.g., Gonnerman and Anderson (2001)). In contrast, other theories claim that morphologically related units are explicitly perceived as such, independently of similar form and meaning (e.g., Marslen-Wilson et al. (1994), Feldman (2000)). In German, earlier studies observed differences between regular and irregular inflections, i.e., that irregular participles such as *geschlafen – schlafen* (*slept – sleep*) do not lead to a speed-up effect, while regular ones, e.g., *geöffnet – öffnen* (*opened – open*), do (Sonnenstuhl et al., 1999). The authors conclude that the mental lexicon is based on a dual system, treating regular and irregular morphological forms differently. However, more recent studies found that irregular and regular participles yield the same priming effect (Smolka et al., 2007), leading to the assumption of a single system, i.e., that morphological structure is always explicitly stored as such.

Also, the question whether morphology is processed in the prelexical (meaning not available) or in the lexical stage (meaning available) is still discussed. Experiments concerning this question yielded, roughly outlined, the following results: It was found for various Indo-European languages (e.g., English, Dutch and French) that morphological decomposition seems to take place in the prelexical stage, because morphological relatedness leads to priming effects in *masked* experiments (e.g., Longtin et al. (2003), Rastle et al. (2004); also supported by overt priming experiments (Kempley and Morton, 1982)). Notably, it is irrelevant in this early stage whether prime and target are semantically opaquely, or transparently related; for instance, both *gaufre – gaufrette* (*wafer – waffle*) and *vigne – vignette* (*vineyard – vignette*) yielded access facilitation in a French lexical decision task. However, also pseudo-derivations (*corn – corner*) lead to speed-up effects. Thus, most studies about masked priming assume that the morphological decomposition happens mostly on an orthographic basis, independently of semantic and actual morphological relatedness. In contrast, findings about the impact of semantic transparency in *overt* priming experiments in Indo-European languages look fairly different: Only

if prime and target are semantically related, morphological priming effects have been observed (e.g., Marslen-Wilson et al. (1994), Longtin et al. (2003)). This means that, at the lexical stage when semantic aspects of a word can be accessed, semantic relatedness becomes more important. In sum, researchers conclude that morphological aspects come into play in both stages: in the prelexical stage by means of decomposition, and in the lexical stage by means of verifying the semantic (and syntactic) compatibility of the morphemes (Taft and Kougious, 2004, Meunier and Longtin, 2007).

Cross-lingual Differences. Many aspects about word formation are assumed to affect morphological priming: For instance, Frost et al. (1997), Smolka et al. (2014) mention as influencing factors a word's frequency and morphemic transparency (the transparency of morpheme boundaries), or a language's morphological productivity (in terms of number of derivations produced for a base word) and richness (in terms of grammaticality expressed by morphology rather than syntax). As these factors differ across languages, morphological priming effects are not identical in different language families:[2] Notably, the abovementioned finding that only transparent morphological relations lead to priming effects under overt conditions was fairly consistent across Indo-European languages. In contrast, Semitic languages such as Hebrew or Arabic, which are morphologically highly complex (Frost et al., 1997), show priming effects also for opaque relations under cross-modal conditions (Frost et al., 2000, Boudelaa and Marslen-Wilson, 2004), suggesting that the mental lexicon in these languages is fully organised by means of morphemes.

7.3 A Recent Study on Morphological Priming in German

As a follow-up to earlier work (Smolka et al., 2009), Smolka et al. (2014) prominently published a controversial study about the status of morphological priming in German. In an overt setup, they analysed morphological priming effects on German prefix verbs, thus concentrating on derivation. Their aim was to reveal whether only semantically transparent derivations (*schließen*$_V$ – *abschließen*$_V$ (*to close*$_V$ – *to lock*$_V$)), or also opaque derivations (*führen*$_V$ – *verführen*$_V$ (*to lead*$_V$ – *to seduce*$_V$)) yield priming effects.

Three Experiments and Their Results. Three overt experimental setups, all of them being lexical decision tasks, were conducted in order to exclude various influencing factors.

The main Experiment 1, using overt visual priming (long prime exposure time of 300ms SOA), involved 40 six-tuples that paired up a target base verb with five prefix verbs as five prime types. An exemplary six-tuple is shown in Table 7.1: Along with Transparent and Opaque Derivations, also purely semantic relations (Synonym), purely orthographic relations (Form), and – as a control relation – verbs without any relation (Unrelated) are considered. Each prime-target pair was assigned a priming signature, indicating the status of morphological, semantic, and form relatedness.[3] The employed verbs were

[2]Similarly, other priming types do not necessarily operate in the same way in all languages.
[3]Note that S and M correspond to our **S** and **M** labels from the previous Chapters.

Prime type	Priming signature	Example for the target *binden (bind)*	RT (ms)
1 Transparent Derivation	M+S+F+	*zubinden (tie)*	563**
2 Opaque Derivation	M+S−F+	*entbinden (give birth)*	566**
3 Synonym	M−S+F−	*zuschnüren (tie)*	580
4 Form	M−S−F+	*abbilden (depict)*	600
5 Unrelated	M−S−F−	*abholzen (fell)*	591

Table 7.1: The five prime types of Smolka et al. (2014) with their priming signatures (M = morphologic relatedness; S = semantic relatedness; F = form relatedness), and experimental average response times (measured in milliseconds). Significance results compared to Unrelated type (** : $p < 0.01$)

normed carefully, e.g., regarding their frequency (based on CELEX) and associativity, in order to exclude confounding factors. For each of the five pair types, the authors measured the average response time (RT) it took the probands to decide whether the target pair is an actual word. Shorter RTs indicated stronger priming effects, where the RT of the Unrelated prime is considered a "no activation" baseline.

As Table 7.1 shows, average RTs for both Derivation primes were clearly shorter than for the Unrelated prime, irrespective of semantic relatedness. Synonym relatedness yielded a weak speed-up, while Form relatedness lead to a delay. The authors measured statistical significance with a one-way ANOVA on the response times, considering six contrasts between the different prime types: Unrelatedness vs. the four other types (Transparent and Opaque Derivation, Synonym, and Form relatedness), Transparent vs. Opaque Derivation, and Transparent Derivation vs. Synonym. Only the RTs of the two Derivation primes were significantly shorter than that of the Unrelated prime, while Synonym relatedness did not yield a statistically significant speed-up. Also surprisingly, the difference between Transparent and Opaque Derivation is not statistically significant, while that between Transparent Derivation and Synonym relatedness is.

Experiment 2 differed from Experiment 1 in that was cross-modal, providing the prime auditorily rather than visually, while the target is presented visually. This change should exclude effects arising from the modality. Almost the same lexical material as in Exp. 1 was used; merely some Form-related primes needed to be adapted for phonological similarity rather than visual similarity. The findings of Exp. 2 were identical to those in Exp. 1, showing that the modality did not affect the results.

Finally, Experiment 3 was conducted to exclude some further influencing factors from the setup: On the one hand, a more refined analysis was conducted to ensure that the data was sensitive to semantic as well as form relatedness, as both did not lead to priming effects in Exp. 1 and 2. On the other hand, nouns were added to the lexical material in order to exclude word class-specific phenomena. Instead of evaluating six-tuples (i.e., five pairings), this experiment considered triples: A target, a related prime and an unrelated

prime, where the related prime varied across the prime types shown in Table 7.1. For instance, a triple for the Semantic relation is *Flut – Ebbe – Dose* (*flood – ebb – tin*). With this design, each of the prime types was separately tested using a two-way ANOVA with the factors prime type and relatedness (related/unrelated). Also in this experiment, both morphological priming types showed robust priming effects independently of transparency. However, this time also the Synonym prime achieved statistically significant speed-up effects, which demonstrates that also semantic priming effects occur in German.

Smolka et al.'s Conclusions. In sum, Smolka et al. (2014) report three main findings: 1., as expected, no priming was observed for Form and Unrelated primes; 2., unexpectedly, also the traditional semantic prime (Synonym) did not yield a facilitation; 3., in contrast, both Transparent and Opaque Derivation lead to significant priming of the same strength.

The morphological effect observed cannot be attributed to an overlap of semantic and form relatedness, since form relatedness did not yield any priming effects. Even more importantly, the findings suggest that morphological priming on German prefix verbs uses a mechanism that is different from semantic priming, which assumes that the strength of the semantic relatedness is the main determinant of priming: Semantic priming would predict finding 1., but neither 2. nor 3. These findings contrast with the overt priming patterns found in similar experimental setups for other Indo-European languages as presented in Section 7.2: In these studies, only transparent morphological patterns were found to be consistent with semantic priming.

Smolka et al. interpret this divergence as evidence for cross-lingual differences and particularly, for a German *Sonderweg* within the Indo-European languages: Its typological properties, such as separable prefixes, productivity, morphological richness, and many opaque derivations, are taken to suggest that German speakers need to handle their mother tongue differently to, e.g., English speakers, namely by a *morpheme*-based organisation of the mental lexicon. More specifically, they assume that the lexicon is related via morphological base forms rather than purely semantic associations. Thus, morphological structures determine German word recognition: Complex words are accessed via their stems, which implies a morphological generalisation process. In this way, the German mental lexicon would be more similar to that of Semitic languages, which have been suggested to be fully based on morphemes, than to that of Indo-European languages.

While the authors claim that semantic priming was hard to demonstrate and might be completely irrelevant in a morpheme-based lexicon, they could not reveal clear factors that actually cause the observed priming effects (Smolka et al., 2009, 2014). Thus, they propose to pursue this question in cross-lingual studies.

7.4 Modelling Morphological Priming

We investigate Smolka et al.'s call for a morpheme-based lexical representation in German by computationally modelling their priming experiment. In this way, we want to make a contribution to the modelling of morphological priming in psycholinguistics: We present a simple model that employs a distributional semantic space combined with

derivational information from DERIVBASE, and examine whether the priming results of Smolka et al. (2014) can be achieved even without morpheme information. Instead, we think that the notion of derivational families in DERIVBASE implements the assumed connections between morphologically related words in the mental lexicon. Adding these connections to a semantic space can be understood as a morphological generalisation of the underlying semantic information, and should yield the morphological priming effects observed by Smolka et al. (2014). If this was true, our computational model would challenge the claim of a morpheme-based lexical representation in German.

We compare three computational models with the results of the original experiments: On the one hand, we consider a standard distributional semantic model, which should model classical semantic priming rather than morphological priming. On the other hand, we generalise a distributional model with information from DERIVBASE, using both its purely morphological and its semantically validated version. We expect v1.4.1 to predict the observed morphological priming effects, i.e., high relatedness for transparent as well as opaque derivations, while v2.0 should perform somewhat in-between a purely semantic model and its generalisation with v1.4.1.

We concentrate on Smolka et al.'s Experiment 1 outlined in Section 7.3, since a standard distributional model cannot make reasonable predictions about Form relatedness/unrelatedness, which the dataset of Exp. 3 would require. In the following, we sketch how we model priming with a distributional model, and how we additionally integrate DERIVBASE.

Distributional Semantics and Priming. Distributional semantic models (cf. Section 3.2) are a classical test bed for semantic priming. Priming has been modelled successfully in a number of studies (cf. the literature mentioned at the beginning of this Chapter), motivated by findings in psycholinguistics that contextual information can yield semantic priming effects (e.g., Schuberth and Eimas (1977)).[4] The results show that this "context effect" (McDonald and Brew, 2004) can be well represented by the vectors in a distributional model. The assumption of this model family, which we call DISTSIM, is that the similarity (typically measured with cosine) of a prime vector $\vec{p}$ and a target vector $\vec{t}$ is a direct predictor of lexical priming:

$$priming_{\text{DISTSIM}}(p,t) \;\propto\; sim\left(\vec{p},\vec{t}\right) \tag{7.1}$$

Regarding morphological priming, this model predicts the result patterns for Indo-European languages such as French or English, but should – according to Smolka et al. (2014) – not be able to explain the German results.

Derivational Morphology in a Distributional Model. As in Chapter 6, we extend distributional models with derivational knowledge – either transparent or opaque – from DERIVBASE. While our motivation in the previous experiments was primarily

[4] Of course, there are also other computational approaches to simulate lexical priming, e.g., traditional neural network models (Cree et al., 1999).

computational (we aimed at improving similarity estimates for infrequent words by taking advantage of the shared meaning within derivational families), the derivational families can be reinterpreted in the psycholinguistic context as driving *morphological generalisation* in priming. That is, words that are derivationally related to the prime are "activated" by means of their membership in the same derivational family. Each of the family members then contributes to the priming effect just like in standard semantic priming. Similar effects are assumed in the human brain by means of spreading activation (Collins and Loftus, 1975), starting from the prime and affecting its related words.

To model this morphological generalisation, we reuse our previously introduced method of derivational smoothing (Section 6.1), and call the resulting distributional model family MorGen. We set the three smoothing parameters as follows:

Derivation rule path: We experiment with both *plain* and *path* information level in order to investigate whether the proximity of derivationally related words in a derivation rule path has an impact on the priming prediction, e.g., by reflecting possible "nearest-neighbour" effects.

Smoothing trigger: We only use the *alwaysSmoTr* trigger, as it – in contrast to the *zeroSmoTr* trigger – reflects the concept of morphological generalisation, i.e., an activation of morphologically related words for *any* perceived word.

Smoothing scheme: We believe that morphological priming neither invokes only the semantically most similar morphologically related word, nor are we, to the best of our knowledge, aware of assumptions in the psycholinguistics literature that morphological priming refers to a prototypical representation of a derivational family. Thus, we employ the *avgSim* scheme, which takes into account all members of a derivational family. We redefine *avgSim* as an asymmetric procedure, since only the derivational family of the prime is to be generalised, but not that of the target:

$$priming_{\text{MorGen}}(p,t) \propto \frac{1}{N} \sum_{p' \in \mathcal{DF}(p)} \alpha(p,p')\ sim\left(\vec{p}', \vec{t}\right) \tag{7.2}$$

where N is the number of pairs that can be formed from the prime family $\mathcal{DF}(p)$ and the target t and have a similarity $\neq 0$, and $\alpha(p,p')$ are the rule path confidences in the derivational family of p. Note that this equation differs from the standard definition of *avgSim* in Equation (6.3) in that only for one lemma of the word pair (i.e., for the prime p), the derivational family is taken into account.

Having set the parameters in this way, our derivational generalisation model operationalises the intuition that every prime activates its complete derivational family, no matter if transparently or opaquely related.

We work with both DErivBase v1.4.1 and v2.0. The MorGen model using v1.4.1 should have a better chance of modelling Smolka et al.'s results than the DistSim model, although it remains completely at the string level, with derivational families as its only

source of morphological knowledge. As to v2.0, we expect it to perform rather at the level of the DistSim model, since the semantic validation should prevent the activation of opaque derivatives, and instead give credits to semantic relatedness.

7.5 Experimental Setup and Results

Distributional Model. For DistSim, we reuse the bag of words model presented in Section 5.2.1 (for the semantic validation of DErivBase). We use the conservative lemmatisation with TreeTagger rather than MATE, since we expect priming to profit from clean information, but assume that a syntactic model as evaluated in Chapter 6 would be too sparse. Similarity is measured with cosine similarity.[5] Note that the interpretation of cosine predictions is inverse to that of Smolka et al.'s response times: Higher cosine scores correspond to stronger priming effects and thus, to shorter RTs.

For the morphological generalisation, we incorporate information from DErivBase as shown in Equation (7.2), which is parallel to the smoothing in Section 6.1: Whenever our families are activated, we select their generalised prediction, irrespective of whether the generalisation increases or decreases the prediction.

Prediction and Evaluation. Following Smolka et al. (2014), we analyse the predictions with a series of one-way ANOVAs, using the prime types of Table 7.1 as factor, and using "Unrelated" as reference level. As appropriate for multiple comparisons, we adopt a more conservative significance level of $p = 0.01$ (Bonferroni, 1936).

Results. Table 7.2 repeats the original experimental results (measured as RT) and reports our model predictions for all settings (measured as cosine similarity) as well as significance of differences. As explained in Section 7.3, Smolka et al. (2014) examine six prime type contrasts: Unrelated vs. the four other types, Transparent vs. Opaque Derivation, and Transparent Derivation vs. Synonym. We also examine these contrasts, showing the former four in the upper part of the Table, and the latter two in the lower part. Model contrasts that match experimental contrasts are marked in boldface.

All models concur with the original experiment in three findings, each of them being statistically significant: 1., Transparent Derivation yields strong priming effects (5 vs. 1); 2., Form relatedness is not sufficient to yield a priming effect (5 vs. 4); 3., the priming effect of Transparent Derivation is stronger than that of the Synonym prime (1 vs. 3).

Apart from that, the DistSim model predicts – as expected – the patterns of classical semantic priming: We observe significant priming effects for Transparent Derivation and Synonymy, and no priming for Opaque Derivation. This is contrary to Smolka et al.'s experimental results.

In contrast, our instances of the MorGen model do a better job. Table 7.2 shows that using DErivBase v1.4.1 and *path* information best reflects the priming effects

[5] We also ran experiments with predictive distributional models (Mikolov et al., 2013). The results were slightly better for DErivBase v1.4.1, but we skip them here for the sake of comparability with the other evaluation chapter.

Prime type	Smolka et al. (RT in ms)	DistSim model (cos.)	MorGen models (cos.)			
			DErivBase v1.4.1		DErivBase v2.0	
			plain	path	plain	path
1 Transparent Deriv.	563**	**0.22***	**0.11***	**0.07***	**0.17***	**0.15***
2 Opaque Deriv.	566**	0.09	**0.11***	**0.06***	**0.06**	**0.05**
3 Synonym	580	0.13***	0.05**	**0.03**	0.08***	0.07***
4 Form	600	**0.03**	**0.02**	**0.01**	**0.03**	**0.02**
5 Unrelated	591	0.03	0.03	0.02	0.03	0.02
Stat. sigf. 1 vs. 2	–	**	–	–	***	***
Stat. sigf. 1 vs. 3	*	*	***	***	***	***

Table 7.2: Top: Average Reaction Times and cosine scores for Smolka et al.'s Exp. 1 dataset. Significance results compared to Unrelated type. Bottom: Significance results for prime types 1 vs. 2 and 1 vs. 3, respectively. Correct contrasts shown in boldface. Legend: * : $p < 0.05$; ** : $p < 0.01$; *** : $p < 0.001$

found in the original study: The difference between its predictions of both Derivation types and the Unrelated type is highly significant ($p < 0.001$), while the difference of the predictions for Synonym and Unrelated is only significant at $p = 0.05$. If the rule path is not taken into account, DErivBase v1.4.1 predicts significant priming for Synonym primes (at $p = 0.01$), which is contrary to Smolka et al.'s findings. This suggests that the rule path information supports the morphological priming effect of the MorGen model. Nonetheless, both MorGen models using v1.4.1 reflect the fact that Transparent and Opaque Derivation primes have equally strong effects.

As to DErivBase v2.0, we still register the three abovementioned findings of all models that comply with the original study, but for the rest, this model rather tends to predict semantic priming: Significance for priming with Opaque Derivations compared to Unrelatedness is weaker than when DErivBase v1.4.1 is used ($p < 0.01$). Coupled with this change, the difference between Transparent and Opaque Derivation primes (1 vs. 2) is now highly significant. This contrasts not only with the findings of Smolka et al., but also shows more drastic differences between Transparent and Opaque Derivation primes than in the DistSim model (significance levels $p = 0.01$ vs. $p = 0.001$). In line with DistSim, but contrary to the original study, MorGen using v2.0 achieves highly significant priming effects for comparing Synonym and Unrelated primes. While such a result is undesirable for the present application, it indirectly supports the semantic validation that we have conducted in order to make DErivBase semantically more coherent. Nonetheless, the difference between Transparent Derivation and Synonym primes (1 vs. 3) remains highly significant for these MorGen models, which shows that this model still incorporates morphological generalisation, though limited to transparent

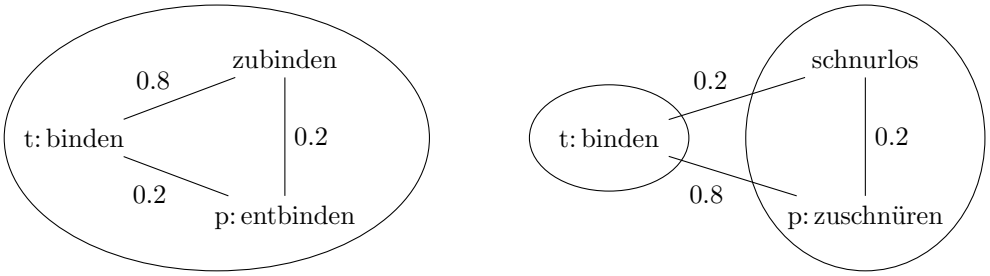


Figure 7.1: Influcence of derivational families on Opaque Derivation primes (left) and Synonym primes (right). The numbers are fictional

derivation. The results of v2.0 are substantially identical for the *plain* and *path* variants.

7.6 Discussion

In sum, our MorGen models can account for the experimental results of Smolka et al. (2014). They conduct a morphological generalisation of the underlying distributional semantic model (DistSim), which leads to effects similar to those observed in the original study: MorGen does not model semantic priming (as DistSim does), but morphological priming. Recall that the main difference between DistSim and MorGen is that the latter includes all members of the prime's derivational family into the prediction of the priming strength. This leads to the following changes compared to DistSim:

1. For Opaque Derivation, MorGen typically predicts stronger priming than DistSim, since prime and target are typically members of the same derivational family (assuming that there are no coverage gaps in DErivBase), and the average similarity between the target and the words in the family is higher than the similarity to the prime itself. Taking the data of Table 7.1 as an example, the Opaque Derivation pair *entbinden* (*to give birth*) – *binden* (*to bind*) is relatively dissimilar, and the similarity increases when other pairs like *binden* (*to bind*) – *zubinden* (*to tie*) are taken into consideration. This fact is illustrated on the left-hand side of Figure 7.1.

2. For Synonymy, MorGen using v1.4.1 typically predicts weaker priming than DistSim, since the average similarity between target and all members of the prime's family tends to be lower than the similarity between target and original prime. For instance, the Synonym pair *zuschnüren* (*to tie*) – *binden* (*to bind*) is relatively similar, while other terms of the derivational family of the prime *zuschnüren* (*to tie*) like *schnurlos* (*cordless*) introduce low-similarity pairs like *schnurlos* (*cordless*) – *binden* (*to bind*). The right-hand side of Figure 7.1 illustrates this behaviour.

Concerning the parametrisations of the MorGen model, there are two main findings:

1. The results clearly show that the derivational information from DERIVBASE v1.4.1 is perfectly suitable to model morphological priming, whereas v2.0 cannot account for the experimental evidence. Considering the construction procedures for the two different versions, this is plausible, and even desirable: The purely morphological version of our lexicon takes into account both opaque and transparent derivations and could thus model the Opaque Derivation primes, while the semantic validation of v2.0 should discard opaque derivations and produce semantically coherent clusters. As the significance tests for v2.0 in the previous Section show, the semantic validation seems to have worked fairly well. In fact, it emphasises semantic relatedness within derivationally related words even more than DISTSIM.

2. The derivation rule path, i.e., a confidence score that assigns higher weights to lemma pairs that are connected by less derivational rules, yields more appropriate morphological generalisation, in particular for v1.4.1. This is the case because both Transparent and Opaque Derivation primes are prefix verbs. To attain the respective target verb, which is the base lemma for the prefixation processes, merely one (inverse) derivation step is typically required. Thus, these lemma pairs are assigned a rule path confidence of $\alpha = 1$. In contrast, many other derivatives of the prime family, beyond which there might be false positives (e.g., the morphologically unrelated verb *verbünden* (*to ally*) as potential prime for *binden* (*to bind*)), are assigned lower α values. On v2.0, the rule path has less impact, because the semantically validated clusters are generally smaller than the morphological families and thus, paths are rather short anyways.

As regards other studies that take a distributional stance towards derivational morphology for psycholinguistics, we are only aware of Marelli and Baroni (2015). They propose a compositional model that computes separate distributional representations for the meanings of stems and affixes, and is able to compute representations for novel, unseen derived terms. Our model is considerably simpler, as it only incorporates knowledge about derivational families, and does not require vector combination. Since Marelli and Baroni's model works on the morpheme level, it corresponds more directly to Smolka et al.'s claim. However, their goal is fundamentally different from the task to be solved on Smolka et al.'s dataset: They aim at finding systematic structures in the meaning construction of derivation rather than developing methods for morphological generalisation on priming.

The fact that our MORGEN models do not contain information about morphemes suggests that, at the very least, morpheme-level processing is not an *indispensable* property of any model that explains Smolka et al.'s experimental results. Thus, the evidence for a special organisation of the German mental lexicon, in contrast to other Indo-European languages, must be examined more carefully.[6]

In fact, our model provides a possible alternative source of explanations for the cross-lingual differences: Since the predictions of MORGEN are directly influenced by the

[6]However, it is unclear whether the rule-based induction method of DERIVBASE might implicitly introduce morpheme-based information into our model; it is a question for future research to assess this potential criticism.

size and members of the derivational families, German opaque morphological priming may simply result from the high frequency of opaque derivations. This high frequency of **M** derivations indeed arises – at least partially – from the German typological particularities mentioned by Smolka et al., e.g., its tendency to employ derivations to express grammatical constructions that are reflected differently in other languages (Smolka et al., 2014, p33). In fact, Schreuder and Baayen (1997) report that, in Dutch, the size of the morphological families negatively correlates with the response latency in priming for simplex words, giving evidence for the family size to have an impact on morphological priming. Additionally, we believe that the distance between two words in terms of their derivational relatedness has an impact on the priming effect: Taking into account the derivation rule path yielded the best MorGen model, suggesting that, the less derivational rules need to be applied to connect two lemmas, the stronger is the priming effect. Observations with a similar basic idea, i.e., that morphological relatedness is a graded scale, have been made, e.g., by Gonnerman and Anderson (2001), Hay and Baayen (2005). Nonetheless, the high productivity of German derivation in general (DErivBase v1.4.1 has rule paths up to a length of 20) might amplify the discrepancy of the effects observed by Smolka et al. compared to other languages. In sum, cross-lingual differences might indeed lead to different morphological priming effects. It would be interesting to check this explanation computationally by applying our MorGen model to other Indo-European languages. However, we leave this study for future work.

8 Recognising Textual Entailment with DErivBase

In this Chapter, we investigate to what extent derivational knowledge from DErivBase helps solving the task of of Recognising Textual Entailment (Rte), a common semantic NLP task. As briefly mentioned in Section 3.1.3, Textual Entailment (Te) systems assess whether an utterance (Hypothesis H) can be inferred from another one (Text T). One common approach, the "matching-based" approach, is to map as much lexical material of T and H as possible, and then to quantify the overlap as a measure of similarity. The underlying hypothesis is that there is a correlation between lexical overlap and relevance: The more lexical material of H is covered by T, the more likely H can be entailed from T. However, there are often lexical gaps, meaning that different lexemes were chosen to formulate the utterances. Naturally, lexical gaps complicate the matching-based approach, and linguistic resources are used to increase the coverage, assuming that they give evidence about meaning-preserving lexical variation, and lead to a better recognition of inference.

We think that Rte is a suitable application for attesting the precision and coverage of DErivBase, since it is a fairly generic semantic task in which any kind of knowledge and approach can be employed. We test to what extent derivational information can be used in order to (partially) close the lexical gap and improve the performance of Te systems. To this end, we integrate DErivBase into an existing system by means of a *query expansion* that adds derivationally related words to the original text. Specifically for German, which is derivationally very productive, we expect some impact.

Section 8.1 introduces Textual Entailment and the Rte task in general, and the role of derivational knowledge in Rte in particular. In Section 8.2, we present the German dataset on which we evaluate the impact of DErivBase (v1.4.1 and v2.0), and Tie, a Textual Entailment system that we employ. Also, we explain how we integrate DErivBase into Tie, and the performance of this integration on the German dataset. The results motivate the creation of a phenomenon-specific dataset (Section 8.3). Finally, we present the results of our approach on this dataset (Section 8.4), and conclude (Section 8.5).

8.1 Recognising Textual Entailment

Textual Entailment is a binary relation between two utterances, called a *Text T* and a *Hypothesis H*, that holds if "a human reading T would infer that H is most likely true" (Dagan et al., 2005). By means of this notion of common sense judgements, Te is intended to be a general processing paradigm that can provide a large part of the semantic

processing needs of various NLP tasks (Dagan et al., 2009, Padó and Dagan, 2016), such as Question Answering (Harabagiu and Hickl, 2006), or Text Summarisation (Harabagiu et al., 2007). One of the main challenges for most NLP tasks is that similar facts can be expressed in many different ways: The potential of lexical variation can make the comparison and matching of semantic aspects between two texts fairly hard. This issue is addressed in TE. Example (8.1) shows a T/H pair with a positive entailment relationship (i.e., T entails H):

(8.1) T: Yoko Ono unveiled a bronze statue of her late husband, John Lennon.
H: Yoko Ono is John Lennon's widow.

Nearly all humans would agree that H can be inferred from T. A simple computational system without world knowledge, however, might not be able to recognise the quasi-synonymy of having a late husband and being a widow. It thus might judge equally probable that T entails the Hypothesis shown above, or the sentence *Yoko Ono is John Lennon's brother*.

T/H pairs as in (8.1) reflect the ubiquitous variability of natural language. That is, they can be similar in their meaning, but differ in their surface realisation (using meaning-preserving linguistic variations), or be similar in their surface realisation, but differ in their meaning. Typical examples for meaning-preserving expressions are sub-sentences[1], paraphrases (*we bought a car – we purchased a car*), or syntactic variations like nominalisations of verbal expressions (*the interest rates decrease – the decrease of the interest rates*). The most prototypical example for a meaning change is negation (*he will attend – he will not attend*). It is the task of the entailment systems to detect which (combinations of) linguistic variations are meaning-preserving, and to correctly predict the entailment relationship.

Various approaches for entailment systems have been proposed. Most entailment decision algorithms can be classified into three groups: 1., matching-based algorithms, 2., transformation-based algorithms, and 3., logics-based algorithms. Logics-based approaches transform T and H into formal expressions and then apply entailment algorithms to these formulas, whereas both matching-based and transformation-based approaches work on raw text. The former maps T and H directly onto each other – possibly using additional knowledge resources –, while the latter translates T into H through a sequence of transformation steps, e.g., syntactic changes from active to passive voice constructions. A comprehensive review of these approaches would go beyond the scope of this Chapter (for an overview, cf. Padó and Dagan (2016)); we focus on the matching-based approach, which is based on the following hypothesis:

Lexical Overlap Hypothesis (LOH): The higher the number of lexical matches between a Text and a Hypothesis, the more likely the T/H pair is entailing rather than non-entailing.

[1] Note, however, that T and H, as we use them, are never identical.

The RTE Datasets. Since 2005, Textual Entailment has its proper evaluation forum, called the PASCAL Recognising Textual Entailment Challenges (Dagan et al., 2005). It is a series of workshops which regularly takes place since then. For every challenge, a dataset is published, consisting of a development and a test set with human annotations for each T/H pair, i.e., the binary decision "entailment" versus "non-entailment". The datasets typically look as shown in Example (8.1) above: T consists of one or more complete sentences, while H is exactly one complete sentence that is usually shorter than the corresponding Text. These datasets were designed in a way that they represent the language variability described above. In the following, we will concentrate on the dataset of the third RTE challenge, RTE-3 (Giampiccolo et al., 2007). It consists of 800 T/H pairs each in the development, and the test set.

The Role of Morphological Derivation in RTE. As just mentioned, the RTE datasets reflect the variability of natural language, which makes the entailment task hard. T and H can be realised as simple sub-sentences or with very different structures like the switch to a passive construction in Example (8.2), where textual similarities are more subtle (this T/H pair is taken from the RTE-3 development set and is assigned an "entailment" relationship):

(8.2) T: The fatal shooting of Steven Charles Jenkins, a terminally ill AIDS patient, by his friend Philip Lee Saylor at Cedars-Sinai Medical Center has again focused attention on the problems of death and dying in our society.

H: Steven Jenkins was shot by a friend.

For matching-based approaches, such linguistic variation is only tractable with adequate linguistic knowledge. Derivation is one such variation phenomenon: In Example (8.2), the two semantically related words *shooting*$_N$ and *to shoot*$_V$, which are split across T and H, can be connected via their derivational family. By aligning this word pair and assigning it a "relatedness" label, the overlap of T and H increases, and the lexical gap between T and H becomes smaller. Derivational lexicons deliver this knowledge and thus might improve the detection of positive entailment relationships. Based on this observation, the following hypothesis, which expands the Lexical Overlap Hypothesis mentioned above, can be formulated:

Derivational Overlap Hypothesis (DOH): If two words w, w' spread across T and H are derivationally related, they should count as a lexical match in the sense of the Lexical Overlap Hypothesis.

Similar assumptions have been made, e.g., by Szpektor and Dagan (2008), who learn paraphrases from CatVar in order to gather new entailment rules for English RTE (for a detailed description, cf. Section 3.1.3). Unfortunately, they did not separately evaluate the impact of the paraphrases induced with CatVar, so that it remains unclear how much derivational information actually helped.

For German RTE, we expect derivational information to be more important than for English due to its higher morphological complexity. Thus, we investigate whether

knowledge from DERIVBASE improves the recognition of entailment and non-entailment relationships on a German RTE dataset. In contrast to Szpektor and Dagan, we integrate the data from DERIVBASE more directly, i.e., via query expansions rather than via induction of entailment rules. Moreover, we explicitly analyse the impact of this query expansion using derivationally related lemmas, thus providing a clearer picture of the appropriateness of the Derivational Overlap Hypothesis. As to our two lexicon versions v1.4.1 and v2.0, this hypothesis naturally suggests the semantically validated variant to perform with higher accuracy.

8.2 Evaluation of DErivBase on the RTE Task

In order to assess the appropriateness of the Derivational Overlap Hypothesis from Section 8.1, we integrate derivational information into TIE, a language-independent, matching-based TE system, and test its impact on a German RTE dataset. This Section describes the German dataset, the TIE system, how we integrate DERIVBASE into TIE, and the system's performance on the dataset. We also discuss problems of using this dataset to evaluate a derivation-specific resource on the RTE task. All significance tests reported in the remainder of this Chapter are χ^2 two-tailed tests measured on accuracy.

8.2.1 Employed Dataset and Entailment System

An RTE Dataset for German. The PASCAL challenges only consider English. Nonetheless, the Textual Entailment community created translations of the English RTE-3 dataset for other languages, one of them being German.[2] Apart from this translation, two domain-specific datasets are available (Zeller and Padó, 2013, Eichler et al., 2014), based on user posts in an online forum, and on customer email requests, respectively.

For our study, we use the German RTE-3 translation, because it most directly reflects the written German standard language. The two other datasets are prone to spelling errors, colloquial language, etc., which might complicate linguistic preprocessing like word segmentation, lemmatisation, part of speech tagging, and syntactic parsing.

TIE, a Textual Entailment System. TIE (Textual Inference Engine) is a matching-based TE system, developed at the Language Technology lab of DFKI GmbH, Germany. It is not publicly available as a standalone system, but it is integrated in the open-source platform of the EXCITEMENT project[3]. Figure 8.1 illustrates TIE's architecture. For linguistic preprocessing of T and H, we employ TreeTagger (Schmid, 1994) as lemmatiser and part of speech tagger, and MaltParser (Nivre et al., 2006) for parsing. TIE works with supervised machine learning techniques: It uses features extracted for each T/H pair that capture common properties as well as mismatches between T and H on the

[2]The datasets are provided in the context of the EXCITEMENT project (Padó et al., 2013a). Downloads: `http://aclweb.org/aclwiki/index.php?title=Textual_Entailment_Resource_Pool`; last accessed: May 2015

[3]`http://hltfbk.github.io/Excitement-Open-Platform/`; last accessed: May 2015

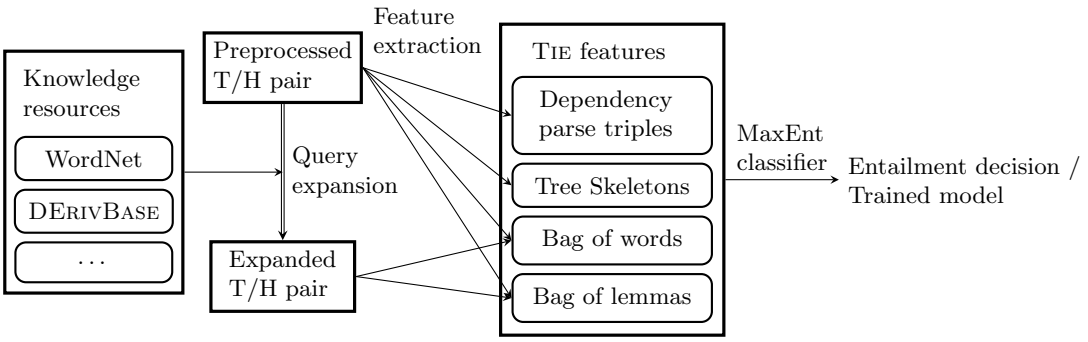


Figure 8.1: Overview of the TIE architecture

text level to train (and apply) a binary Maximum Entropy classifier for the entailment decision. The features hold information of different linguistic levels from various sources. Currently, the following levels of information are implemented:[4]

1. Simple word overlap counts on token and lemma level
2. Syntactic information represented as dependency parse triples, and as abstract dependency sub-trees (called Tree Skeletons) (Wang and Zhang, 2009, Wang and Neumann, 2007)
3. Knowledge extracted from lexico-semantic resources by means of query expansions for T (details follow shortly)

The token and lemma overlap features are considered TIE's baseline functionality.[5] Finally, T/H pairs that are considered sufficiently similar by the classifier, are predicted as positive entailment pairs.

For the following reasons, we employ TIE to evaluate the impact of DERIVBASE on the RTE task: First, TIE's implementation in the EXCITEMENT platform is language-independent and can be applied to German. Second, it is one of the few available entailment systems that deliver state-of-the-art performance for German. Third, its architecture makes the integration of knowledge resources like DERIVBASE very simple.

8.2.2 Integrating DErivBase into TIE

We evaluate DERIVBASE (v1.4.1 and v2.0) using the TIE implementation in the EXCITEMENT platform with all available features in different combinations. We incorporate derivational information in the way such a knowledge integration is specified in TIE: as a *query expansion* for T. The idea behind this query expansion is based on the LOH (cf.

[4]This list refers to TIE's implementation in the EXCITEMENT platform.
[5]In fact, Wang and Neumann (2007) show that this baseline is fairly competitive and hard to beat.

Section 8.1), i.e., increasing the coverage of H by T by adding words which are known to be similar to T's words. In the same fashion, TIE integrates other lexical resources, e.g., taxonomical information from wordnets.[6] Thus, for each lemma-part of speech pair in T, we check whether it is contained in a derivational family in DERIVBASE. If yes, we expand T's bag of lemmas with all other members of the respective family:

$$T \cup \bigcup_{t \in T} \mathcal{DF}(t)$$

where T is the set of words t in the Text, and $\mathcal{DF}(t)$ is the derivational family in DERIVBASE which contains t. This strategy increases the probability of a match (lemma overlap) between T and H, according to the DOH that derivational relationships account for entailing rather than non-entailing T/H pairs (cf. Section 8.1). For our experiments, we employ TIE in four configurations (for details about preprocessing, cf. Section 8.2.1):

BOW: TIE's baseline setting. Word (lemma) overlap features measure the ratio of tokens (lemmas) shared by the bags of words (lemmas) of T and H with respect to the number of words in H.

BOW+DErivBase: Baseline setting plus query expansion with DERIVBASE information as described above.

BOW+DErivBase+SYNT: Baseline setting plus DERIVBASE expansion plus syntactic information. The syntactic features measure the overlap of extracted dependency triples and Tree Skeletons between T and H; these structures match Text and Hypothesis more easily than full syntactic parses.

BOW+SYNT: Baseline setting plus syntactic information.

These settings are defined in configuration files in the EXCITEMENT platform. For illustration, the relevant parts of the BOW configuration are shown in Appendix D.

Additionally to these configurations, we report the performance of a "predict always positive entailment" majority class baseline.[7] We omit the setting of using only DERIVBASE features (without the BOW features), because it holds too little information, and thus performs at the majority baseline level.

8.2.3 Evaluation of DErivBase on RTE with TIE

We evaluate the performance of TIE with the standard measures used in RTE: precision, recall and F_1-score on the positive class (i.e., on the "entailment" decisions), and

[6] We do not make use of these other knowledge resources, but only employ DERIVBASE expansions.

[7] Due to imbalancedness of the original English, as well as the translated German RTE-3 data, there is a slight majority of positive entailment pairs in the dataset; see Giampiccolo et al. (2007, p3), and the README file of the German translation in `http://www.dfki.de/~neumann/resources/RTE3_DE_V1.2_2013-12-02.zip`.

accuracy. Our focus for enhancements is on precision rather than on recall, because an unsophisticated approach like the majority baseline naturally achieves a recall of 100%.

Table 8.1 shows the performance of our four TIE settings, using both DERIVBASE v1.4.1 and v2.0. TIE is both trained and tested on the German RTE-3 dataset (i.e., 800 T/H pairs each). In the following, the best results are always marked in boldface.

Setting (**RTE-dev 800**, **RTE-test 800**)	Acc	P_{pos}	R_{pos}	F_{1_pos}
majority baseline	51.1	51.1	**100**	**67.6**
BOW	61.1	61.4	64.5	62.9
BOW+DERIVBASE v1.4.1	60.8	60.8	65.3	63.0
BOW+DERIVBASE v2.0	61.5	61.6	65.8	63.6
BOW+DERIVBASE v1.4.1+SYNT	**63.5**	**64.2**	64.8	64.5
BOW+DERIVBASE v2.0+SYNT	63.3	63.9	64.5	64.2
BOW+SYNT	63.4	**64.2**	64.1	64.1

Table 8.1: Performance of TIE settings, trained and tested on the whole German RTE-3 dataset

The baseline is slightly above 50% of accuracy, but significantly outperformed by all TIE models ($p < 0.0001$), which, in turn, do not significantly differ among each other. The impact of DERIVBASE is somewhat visible, but not convincing: BOW+DERIVBASE v1.4.1 even shows a small decrease in precision and accuracy compared to BOW, while v2.0 slightly improves results over BOW. The reason is that also incorrect information is added in v1.4.1, mostly due to query expansions with many words added from overinflated derivational families (cf. Section 4.2.5); big families are often triggered because 1., they contain many words, and 2., they contain words that are frequent in the RTE-3 dataset such as *setzen* (*to put*), *legen* (*to put*), or *sprechen* (*to speak*). Up to 1,583 lemmas are added from DERIVBASE for only one T/H pair, including the biggest, and the fourth biggest family (238 and 140 words, respectively).

On the other hand, when DERIVBASE is added to the BOW+SYNT model, performance slightly decreases for v2.0, but increases for v1.4.1 by a small recall gain, achieving the second-best F_1-score (after the baseline). This setting leads to the best overall results in terms of precision improvement (+2.4 percentage points in accuracy, +2.8 percentage points in precision over BOW), which suggests that derivational and syntactic information are, to some extent, complementary and can be well combined.

Nonetheless, including DERIVBASE only resolves one additional T/H pair correctly. TIE cannot optimally profit from the derivation expansion, because it is only relevant for a small subset of T/H pairs. In sum, the information from DERIVBASE is not salient enough on the whole RTE-3 dataset to noticeably improve the classification. This fact might suggest using an RTE sub-dataset which isolates pairs that are relevant to evaluate a derivational lexicon. The next paragraph presents a preliminary analysis with regard to this question.

Measuring the Impact of a Derivational Lexicon on RTE. We analysed the suitability of the German RTE-3 dataset to measure the impact of DERIVBASE, and found that our lexicon is not activated for the vast majority (over 85%) of the T/H pairs: Many pairs do not contain members of the same derivational family spread across T and H, and are thus not relevant. There are a couple of reasons for that, arising from the generation process of the RTE dataset, which of course was not designed for evaluating the performance of one specific resource.

First of all, many H's contain generalised information of T. That is, H repeats one specific aspect of T with simple patterns like "X is Y", or "X is situated in Y", as in the positive entailment pair in Example (8.3):

(8.3) T: U.S. Secretary of State Condoleeza Rice has expressed her anger after a meeting with Sudanese President ...

H: Condoleezza Rice is the U.S. Secretary of State.

Another reason why DERIVBASE tends to be irrelevant for many T/H pairs is that they use completely different lexical material, like the paraphrase pair "is wife of" – "is married to" in the positive entailment in Example (8.4). Although derivational information serves as a source of meaning-preserving paraphrases, it cannot cover such kinds of variation:

(8.4) T: Hughes loved his wife, Gracia, and was absolutely obsessed with his little daughter Elicia.

H: Hughes was married to Gracia.

An important particularity of German which influences the applicability of a derivational lexicon is the tendency to use compounds: English compounds are typically a sequence of individual words, whereas German concatenates them into one single word. Derivational lexicons like DERIVBASE do not cover compositional relations (cf. Section 2.3), which poses problems to detecting derivational relationships. The problem is illustrated in the negative entailment Example (8.5):

(8.5) T: Chirac brauchte von den **Wählern** ein neues Mandat für seine Regierung ...
Chirac needed a new mandate for his government from the ***electorate*** *...*

H: **Parlamentswahlen** führen zur Gründung einer neuen Regierung in Frankreich.
Parliamentary ***elections*** *create new government in France.*

An additional decomposition and compound interpretation step would be necessary to map *Parlamentswahl*$_N$ to *Wähler*$_N$, while such a relationship is straightforward for the English words *election*$_N$ and *electorate*$_N$.

Additionally, we employ the German translation of the original English RTE-3 dataset. This leads, for some sentences, to rather unnatural or infrequent German expressions; that is, the language is not typically represented, known in the literature as "translationese" (for the discussion of this phenomenon and proposals for its automatic recognition, cf.,

e.g., Tirkkonen-Condit (2002), Baroni and Bernardini (2006), Koppel and Ordan (2011)). For example, some expressions remained untranslated, although a corresponding German counterpart would exist, as shown in the English and German version of the positive T/H pair in Example (8.6):

(8.6) T: Sean Brown (geboren am 5. November 1976 in Oshawa, Ontario, Kanada) ist ein **Utility-Player** und Enforcer der National Hockey League.
*Sean Brown (born November 5, 1976 in Oshawa, Ontario, Canada) is a National Hockey League utility **player** and enforcer.*

H: Sean Brown **spielt** in der National Hockey League.
*Sean Brown **plays** in the National Hockey League.*

If the compound noun *utility player* had been translated with a corresponding German expression like *vielseitig einsetzbarer Spieler*, a derivational relationship between Text and Hypothesis would have been established by means of the words $Spieler_N$ and $spielen_V$, as it is the case in English. Such passages which stay too close with the source language, are typical examples for translationese in our dataset.

Another potential artifact of translation in the German RTE-3 data is an – according to our intuition – atypically high number of nouns. We assume that it arises from many nominalisations produced through the translation process. Specifically, the translation of English verbal expressions by using nominalisations in German seems sometimes more intuitive, because nominalisations might allow for a simpler syntax. Example (8.7) shows a positive T/H pair of the German RTE-3 dataset along with its English original:

(8.7) T: Zwei Brüder, die ein Vergoldungsunternehmen in Nord-Hollywood betrieben, ... müssen wegen leichtfertiger Handhabung und **Lagerung** gefährlicher Stoffe eine Haftstrafe antreten.
*Two brothers who operated a North Hollywood plating company ... must serve jail time for recklessly handling and **storing** hazardous materials.*

H: Ein kalifornisches Unternehmen wurde wegen rücksichtsloser **Lagerung** von Chemikalien angeklagt. *A California company was charged with reckless **storage** of chemicals.*

The verb *to store* is translated with the nominalisation *Lagerung* rather than the verb *lagern*, which avoids a more complex adverbial clause construction like *müssen eine Haftstrafe antreten, weil sie gefährliche Stoffe leichtfertig gehandhabt und gelagert haben.*

We found that this is a quantitatively measurable trend: We compared the distribution of nouns in the German RTE-3 dataset to the English RTE-3 dataset, as well as to a German and an English newspaper corpus (the TIGER corpus (Brants et al., 2004) and the Wall Street Journal portion of the Penn Treebank (Marcus et al., 1993); each consists of about 40,000 annotated sentences). The newspaper genre is quite similar to that of the RTE-3 data. We measured the ratio of nouns in each of the text collections.[8] The

[8]Note that the parts of speech annotation was manual on the news corpora, but automatic on RTE-3.

German RTE-3 dataset contains almost 3% more nouns compared to the German news corpus (23.5% vs. 20.7%), and more than 4.5% more nouns than the English RTE-3 dataset (23.5% vs. 18.9%). In contrast, the English news corpus contains about 1.8% more nouns than the English RTE-3 dataset (20.3% vs. 18.9%).[9] The far higher number of nouns in the German RTE-3 data compared to the other text collections suggests that it is idiosyncratic, independently of genre or language specificities. Thus, we assume it is a result of the translation process.

In sum, there are various reasons why DERIVBASE could not show big performance gains on the complete German RTE-3 dataset. Nonetheless, we believe that derivational knowledge can help resolving the RTE task. In the next Section, we thus investigate the situation when a derivation-specific RTE sub-dataset is used.

8.3 Creating a Derivation-specific Sub-dataset

The reasons mentioned in Section 8.2 motivate a German RTE dataset which specifically reflects derivational relationships between Text and Hypothesis, in order to properly quantify the impact of DERIVBASE on the quality of an RTE system. Therefore, we manually selected a subset of the German RTE-3 data. Similar sub-datasets have been developed before in the RTE literature for English. For instance, Dinu and Wang (2009) aimed at verifying the influence of a collection of inference rules (i.e., pairs of phrases which are assumed to hold a directional entailment relationship) drawn from DIRT (Lin and Pantel, 2001) in combination with WordNet. They also employ the TIE system, including the Tree Skeleton component to serve the DIRT patterns. Their resource-specific test dataset (no development set was created) consists of 64 T/H pairs, which all match an inference rule drawn from their rule collection.

We selected the T/H pairs for our sub-dataset in a two-step procedure: First with a manual selection, then using DERIVBASE to retrieve pairs missed in the manual step. Due to a final manual consolidation of these two sets of T/H pairs, the resulting sub-dataset can be considered a gold standard in terms of derivational relations between T/H pairs. We provide both development and test set, where the former consists of the selected pairs of the RTE-3 development set, and the latter of the selected pairs of the RTE-3 test set.

Manual Selection. We used the following guidelines: For each T/H pair, we manually checked whether it contains two distinct derivationally related words spread across Text and Hypothesis, and selected such pairs. Using the terminology of Chapter 4, we accepted **S** as well as **M** pairs as valid links between T and H. Note that the **M** portion is small, since in virtually all T/H pairs the two tests are semantically related in some way (irrespective of entailment). One **M** case is shown in the English positive T/H pair in Example (8.8).

(8.8) T: Many of the Vikings who travelled to Scotland, and other parts of Europe were traders or peaceful settlers looking for **land** to farm.

[9] All differences are statistically highly significant at $p = 0.0001$.

H: Vikings **landed** in Scotland.

During this procedure, we discarded pairs where one of the derivationally related lemmas occurs identically in both Text and Hypothesis, as depicted in the English positive entailment Example (8.9). That is, the derivational link must actually arise from two different lemmas of the same derivational family; no simple lemma identity must lead to a connection. In contrast, T/H pairs like in the negative entailment Example (8.10) are added to our sub-dataset.

(8.9) T: The **sale** was made to pay Yukos' US$ 27.5 billion tax bill, Yuganskneftegaz was originally **sold** for US$ 9.4 billion to a little known company Baikalfinansgroup which was later bought by the Russian state-owned oil company Rosneft.

H: Baikalfinansgroup was **sold** to Rosneft.

(8.10) T: Loraine besides participating in Broadway's Dreamgirls, also participated in the Off-Broadway **production** of "Does A Tiger Have A Necktie". In 1999, Loraine went to London, United Kingdom. There she participated in the **production** of "RENT" where she was cast as "Mimi" the understudy.

H: "Does A Tiger Have A Necktie" was **produced** in London.

Reliability Assessment. To assess the correctness and the coverage of this sub-dataset and, especially, to ensure that the selection was consistent over all 1,600 T/H pairs, we conducted a double annotation of the first 100 T/H pairs in the development set, carried out by the same annotator, and calculated an "intra-annotator agreement". The second annotation took place after having finished the first annotation of the full set of 1,600 pairs, and with a time interval of roughly three months between the annotations, so that the annotator was ensured not to recall their concrete decisions during the first annotation.

For four out of the 100 double-annotated pairs, the decision whether the T/H pair is derivationally related, was different. The differences were two erroneously added T/H pairs containing derivationally related lemmas, but one of them occurring in both T and H (as in Example (8.9)), and two T/H pairs that have simply been overlooked in the other annotation. This agreement rate corresponds to a Cohen's κ value of 0.834 (Cohen, 1960), which is considered as perfect agreement by Landis and Koch (1977). The expected chance agreement in the κ calculation is at $P(e) = 0.759$, meaning that the annotation is essentially more reliable than expected for this dataset.

Automatic Pair Retrieval. Although the intra-annotator agreement suggests that our gold standard sub-dataset is fairly reliable both in terms of precision (i.e., almost all selected T/H pairs indeed have a derivational relationship), and recall (i.e., almost all derivationally related T/H pairs of the RTE-3 dataset have been selected), we added a second, semi-automatic step: We let DERIVBASE v1.4.1 retrieve T/H pairs that contain

lemmas of the same derivational family spread across T and H, manually checked these pairs for consistency with our annotation guidelines, and consolidated them with the manually extracted pairs. We revealed about 60 missed pairs, about 73% of which contain two **S**-connected lemmas, while about 27% were **M** pairs.[10] This consolidated dataset is the basis for all subsequent investigations. Henceforth, we call it the *derivational subset*.

Note that the derivational subset is an actual gold standard: The consolidation discards cases of overgeneration in the application of DERIVBASE. Thus, having used our lexicon for the creation of the derivational subset does not mean that the precision that can be achieved on it is biased towards the lexicon's performance. However, we admit that the recall that our lexicon can yield on this gold standard might be slightly overestimated.

	orig. RTE $_{dev}$	orig. RTE $_{test}$	deriv. RTE $_{dev}$	deriv. RTE $_{test}$
Total pairs	800	800	109	118
Entailment	407	409	66	67
Non-entailment	393	391	43	51
Ratio ent./all	51%	51%	61%	57%
Ratio non-ent./all	49%	49%	39%	43%

Table 8.2: Statistics of German RTE original dataset and derivational subset (development and test each)

Dataset Statistics. Table 8.2 gives a quantitative impression of the compiled derivational subset, compared to the complete German RTE-3 dataset. As can be seen, 109 and 118 T/H pairs have been selected for the development and the test subset, respectively; thus, the test set is roughly 8% bigger than the development set. Compared to the (test) dataset of Dinu and Wang (2009) with 64 T/H pairs, our test set is almost twice as big.

As expected, pairs involving derivation are more often entailing than non-entailing (61% and 57% entailment pairs on development and test set, respectively), however, the difference is rather small, given our Derivational Overlap Hypothesis (Section 8.1) that pairs with a derivational relation across T and H are more likely to be entailing (Section 8.1). We think that this is due to the fact that there are many other linguistic factors (language variability, cf. Section 8.1) which cause a lot of non-entailing pairs.

Coverage of Derivational T/H Pairs by DErivBase. The main goal of our query expansion with DERIVBASE is to increase the margin between entailment and non-entailment pairs by trying to increase the similarity of entailing T/H pairs, operationalised by higher overlap. Thus, an important performance indicator of our lexicon is its coverage of the derivationally related lemmas split across T and H. In other words: Is DERIVBASE capable to improve the performance of an entailment system like TIE on the RTE task, or does it lack too many derivational relations? We evaluated DERIVBASE's coverage

[10] In fact, almost all **M** have been overlooked by the annotator. A plausible reason is that these pairs are semantically very opaque, e.g., *Gesetz*$_N$ – *Umsetzung*$_N$ (*law*$_N$ – *implementation*$_N$).

of the derivational subset by measuring how many of the derivationally related lemmas split across T and H are retrieved using v1.4.1 or v2.0.

	Dev. set	Test set	Dev. & test set, micro averaged
Covered T/H pairs, v1.4.1	88.99%	83.05%	85.90%
Covered T/H pairs, v2.0	59.63%	62.71%	61.23%

Table 8.3: Coverage of the derivational sub-dataset by DERIVBASE

Table 8.3 shows the coverage for the development and the test set, and for the whole derivational subset calculated by micro-average. As expected, DERIVBASE v2.0 yields only low coverage of about 61%, as it does not cover purely morphologically related lemma pairs, and even semantically related lemma pairs might have been split into two semantic clusters. Using v1.4.1, we achieve a fairly large coverage of around 86% which we find a satisfying quantity. However, putting this number the other way round, the coverage of DERIVBASE lacks 14.1% of derivationally related words of the manually annotated dataset. We analysed the T/H pairs where DERIVBASE v1.4.1 missed the match. In the following, we discuss the main reasons for these misses.

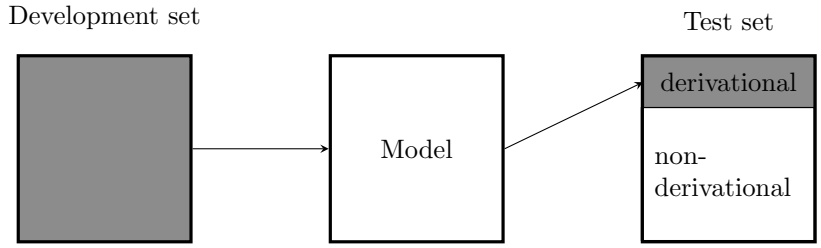
First, we encounter errors which are based, on the one hand, on incorrect translations in the RTE-3 dataset into German, and, on the other hand, on the linguistic data preprocessing. Concerning the translation errors, we found typing errors like **Robbenjagt* instead of *Robbenjagd*$_N$ (*seal hunting*$_N$), or the English spelling *America* instead of the German *Amerika*$_N$. We did not modify the dataset to correct such errors. As to the preprocessing, we discovered problems with phrasal verbs when the sentence structure requires to separate the particle from the main verb. They are not concatenated to the infinitive verb by the lemmatiser and thus do not match the entries in DERIVBASE. For example, both *abreisen*$_V$ (*to leave*$_V$) and *reisen*$_V$ (*to travel*$_V$) are in the same derivational family, but the former cannot be retrieved, because the particle *ab-* is not correctly concatenated with *-reisen*, thus being deemed identical to *reisen*.

Second, and conceptually more important, we observed a couple of uncovered zero derivation nominalisations of adjectives, like *Außerirdische*$_N$ – *außerirdisch*$_A$ (*alien*$_N$ – *alien*$_A$), or *Französische*$_N$ – *französisch*$_A$ (*French*$_N$ – *French*$_A$). We found that this coverage lack is caused by the error in a derivation rule mentioned in Section 4.4.3.

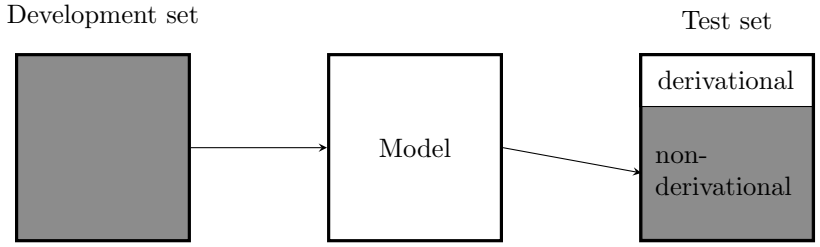
Third, a part of the coverage lack is due to irregularly formed derivations. This happens, e.g., for the derivation of a handful of adjective forms of nationalities like *Kanadier*$_N$ – *kanadisch*$_A$ (*Canadian*$_N$ – *Canadian*$_A$), or for derivation patterns that are historically attested but were not included in DERIVBASE, because they are very unproductive; e.g., *geboren*$_A$ – *Geburt*$_N$ (*born*$_A$ – *birth*$_N$).

Finally, with *Roboter*$_N$ – *robotisiert*$_A$ (*robot*$_N$ – *robotised*$_A$), we encountered one missing derivation pair whose second lemma is too rare to be incorporated in DERIVBASE: It did not occur three or more times in SDEWAC and thus was not taken into account in the DERIVBASE building process (cf. Section 4.2.3).

(A) RTE-dev 800, deriv-test 118



(B) RTE-dev 800, non-deriv-test 682



(C) RTE-dev 800, ensemble test (v1.4.1: 698+102; v2.0: 727+73)

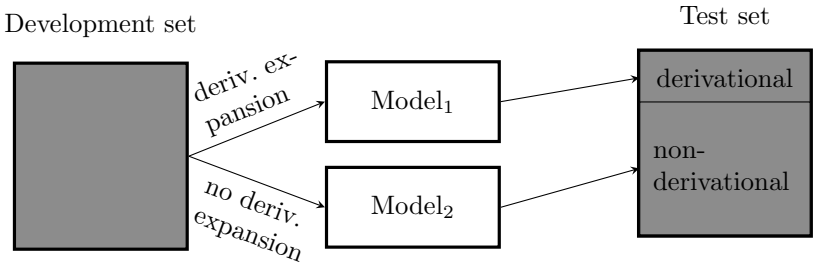


Figure 8.2: Overview of the three setups of used system-dataset combinations

8.4 Evaluation of DErivBase on the Derivational Subset

To evaluate the impact of DERIVBASE on the derivational subset, we assume that training on the whole RTE-3 dataset is sensible, even if we want to measure the quality only on derivationally related T/H pairs, since TIE's base features improve as the training set grows. On the test set, in turn, we differentiate between T/H pairs with and without derivational relations across T and H.

Settings. We examine three different setups; the respective model and dataset combinations are depicted in Figure 8.2. In all three settings, we train the TIE models on the whole RTE-3 development set (denoted with "RTE-dev 800").

(A) We test the four TIE configurations presented in Section 8.2.2 only on the derivational subset consisting of 118 pairs ("deriv-test 118").

(B) We test the four TIE configurations presented in Section 8.2.2 only on T/H pairs which are *not* derivationally related, i.e., on the complement of the test set used in (A), consisting of 682 pairs ("non-deriv-test 682").

(C) We build ensemble models, combining the TIE configurations which appear most suitable for the two complementary data subsets ("ensemble test"): We apply models with derivation expansion to derivationally related T/H pairs (as in (A)), and models without derivation expansion to this subset's complement (as in (B)). In this way, we want to optimally address derivational as well as non-derivational T/H pairs. Similar strategies have been employed in the literature, e.g., by Shen and Lapata (2007), who use semantic role labelling to improve on the QA task: For questions covered by their model, this model is applied; for all remaining questions, a baseline model is used.

For settings involving DERIVBASE, we conduct a query expansion as explained in Section 8.2, using both DERIVBASE v1.4.1 and v2.0.

We conduct settings (A) and (B) for diagnostic purposes: They should indicate the difficulty of the two test sub-datasets (derivational and non-derivational), i.e., how easily the entailment decision can be made. Additionally, (A) should demonstrate the impact of our DERIVBASE integration into TIE when the underlying data can be entirely addressed by derivational knowledge.

In the ensemble setting (C), we aim at modelling a realistic application scenario: We do not deliver the manually annotated derivational and non-derivational subsets to TIE, but let the system separate the data automatically. All pairs for which DERIVBASE detects a derivational relation across T and H, and which are not additionally covered by lemma identity (cf. Section 8.3), are considered part of the derivational subset. All remaining pairs are deemed non-derivational. According to this segmentation, two different models are combined: one including DERIVBASE for the derivation cases, and one without DERIVBASE for non-derivational cases. Note that the T/H pairs retrieved by DERIVBASE do not necessarily correspond to those of the gold standard subset, and that

the two lexicon versions retrieve different sets of T/H pairs (i.e., 102 vs. 73 derivationally related T/H pairs retrieved by v1.4.1 and v2.0, respectively; cf. Figure 8.2).

Setting	Acc	P_{pos}	R_{pos}	F_{1_pos}
RTE-dev 800, **RTE-test 800** (from Table 8.1)				
Best (BOW+DERIVBASE v1.4.1+SYNT)	**63.5**	**64.2**	64.8	64.5
(A) RTE-dev 800, **deriv-test 118**				
majority baseline	56.8	56.8	**100**	**72.4**
BOW	55.9	**64.2**	50.7	56.7
BOW+DERIVBASE v1.4.1	55.1	60.0	62.7	61.3
BOW+DERIVBASE v2.0	56.8	63.3	56.7	59.8
BOW+DERIVBASE v1.4.1+SYNT	**57.6**	63.1	61.2	62.1
BOW+DERIVBASE v2.0+SYNT	55.9	62.7	55.2	58.7
BOW+SYNT	54.2	61.8	50.7	55.7
(B) RTE-dev 800, **non-deriv-test 682**				
majority baseline	50.1	50.1	**100**	**66.8**
BOW	62.0	61.0	67.3	64.0
BOW+DERIVBASE v1.4.1	61.7	61.0	65.8	63.3
BOW+DERIVBASE v2.0	62.3	61.3	67.5	64.3
BOW+DERIVBASE v1.4.1+SYNT	64.5	64.4	65.5	64.9
BOW+DERIVBASE v2.0+SYNT	64.5	64.1	66.4	65.2
BOW+SYNT	**65.0**	**64.6**	66.7	65.6
(C) RTE-dev 800, **ensemble test 698+102 / 727+73**				
BOW_{698} ∘ BOW+DERIVBASE v1.4.1_{102}	61.3	61.0	**67.0**	63.9
BOW_{727} ∘ BOW+DERIVBASE v2.0_{73}	61.4	61.4	65.8	63.5
BOW+SYNT_{698} ∘ BOW+DERIVBASE v1.4.1+SYNT_{102}	**64.0**	**64.4**	66.0	**65.2**
BOW+SYNT_{727} ∘ BOW+DERIVBASE v2.0+SYNT_{73}	63.6	64.3	64.8	64.6

Table 8.4: Performance of TIE settings, trained on the whole development set, tested on different subsets. Best results per setting in boldface

Results. Table 8.4 repeats the best result achieved for training *and* testing on the whole RTE-3 datasets (cf. Table 8.1), and shows the results of the three new setups.

The part of Table 8.4 marked with (A) shows the performance of TIE when it is trained on the whole RTE-3 development set, but tested only on the derivational subset. The majority baseline is higher on this subset than on the whole RTE-3 dataset (cf. Table 8.2). Note that that all TIE models obtain a worse accuracy (even below the baseline) and recall than on the whole RTE-3 test set (Table 8.1). This decrease implies that the derivational subset is more difficult in terms of entailment decision than the standard RTE-3 test dataset: T/H pairs which contain two different members of the same derivational family, tend to differ in their remaining words as well as their syntax, which causes the classifier to reject them more likely. For instance, the T/H pair in Example (8.11) cannot be

resolved correctly as entailing by the BOW model, although the word overlap between T and H is reasonable:

(8.11) T: Washington hat wiederholt versucht, Russland zu versichern, dass es von dem System nichts zu **befürchten** hat.
Washington has repeatedly sought to reassure Russia it has nothing to ***fear*** *from the system.*

H: Russland **fürchtet** sich vor dem System.
Russia ***fears*** *the system.*

When the Hypothesis is modified to a semantically comparable, but lexically much more similar sentence to T like *Russland befürchtet etwas von dem System.*, the BOW model correctly recognises entailment.

Concerning settings including DERIVBASE in part (A) of Table 8.4, the lexicon's impact is more noticeable than in Table 8.1. On this dataset combination, DERIVBASE v1.4.1 again performs worse in combination with BOW, but adding SYNT improves accuracy and precision for both lexicon versions with up to 3.4 and 1.3 percentage points, respectively, over the BOW+SYNT model. In fact, BOW+DERIVBASE v1.4.1+SYNT is the best model on this dataset, being the only to beat the baseline in accuracy, though, not statistically significantly, and achieving the second-best F_1-score (after the baseline). Adding v2.0 to BOW and BOW+SYNT, respectively, always leads to accuracy and recall gains (up to 1.7 and 6.0 percentage points, respectively). However, precision is unstable, meaning that the inclusion of derivationally related pairs makes the entailment decision less clear (e.g, by expanding as many entailment as non-entailment pairs). In general, the semantically validated version provides lower recall than v1.4.1, as expectable due to the smaller families (cf. Table 8.3). In sum, the main insight of our analysis is that the derivational subset is much harder to solve than the whole RTE-3 dataset, even when derivational information is used. We can enhance accuracy (not significantly) over the TIE standard settings, but precision remains best for the BOW model.

Part (B) in Table 8.4 shows the results of our second setup, when the test set consists only of T/H pairs which are *not* derivationally related according to our gold annotation (682 pairs; positive majority baseline: 50.1%). The performance of BOW is a little more recall-oriented now than on the 800/800 dataset combination of Table 8.1, while the accuracy of BOW+SYNT improves by 1.6 percentage points on the 800/682 combination, and achieves the best accuracy and precision. As expected, DERIVBASE v1.4.1 introduces too much noise in this test set, and has a negative impact on the performance of BOW and BOW+SYNT, while v2.0 leads to less losses and even a small accuracy and precision gain when expanding BOW. Overall, the generally higher performance scores demonstrate that this dataset is clearly easier than the derivational subset.

Finally, the (C)-marked part in Table 8.4 shows the performance of our ensemble models when there is no gold standard for derivational relatedness, i.e., no manual data separation, at hand. We apply either BOW or BOW+SYNT to the T/H pairs which are not derivationally related, and BOW+DERIVBASE or BOW+DERIVBASE+SYNT to the derivational subset. Then, we combine either the BOW-, or the BOW+SYNT-involving

models. The results of the two models on the two dataset portions are micro-averaged. As can be seen from the Table, DERIVBASE v1.4.1 and v2.0 recognise derivational relationships for 102 and 73 T/H pairs in the test set, respectively. That is, DERIVBASE retrieves less pairs than the manually annotated subset contains, which corresponds to the fact that roughly 14% and 39%, respectively, of this subset are missed (cf. Table 8.3). Nonetheless, DERIVBASE still overgenerates: v1.4.1 includes 11 spurious pairs (reduced to merely 4 pairs in v2.0), which arise from errors in DERIVBASE (**N** cases; 55%), and preprocessing errors from TreeTagger (45%).

The results of this unsupervised data separation show that the ensemble approach in (C) is indeed beneficial. Notably, the SYNT-involving model combination using DERIVBASE v1.4.1 attains the best overall accuracy and precision scores on the whole RTE-3 test set. With 512 correctly resolved T/H pairs, it recognises four additional pairs correctly compared to the best model in Table 8.1, where only one model was applied to the whole test set. This corresponds to an increase in accuracy of 0.5 percentage points (not statistically significant), and in precision of 0.2 percentage points. F_1-score is, again, only outperformed by the majority baseline, which is an unpreferable option due to many false positives and the uninformedness of the approach. Setting (C) shows that derivational knowledge can improve an RTE system, even if it is unknown which T/H pairs are derivationally related. Unfortunately, v2.0 again performs worse than v1.4.1. Our overall impression is that the query expansion approach to incorporate DERIVBASE into TIE requires high coverage as much as high quality. Thus, the cleaner, but smaller families in v2.0 are sometimes counterproductive. For instance, the positive entailment pair in Example (8.12) was not retrieved because the derivationally related lemmas were split into two semantic clusters:

(8.12) T: Die **Vereinbarung** kam nach etwa 10 Stunden Verhandlungen zwischen Herrn Fradkow und seinem weißrussischen Amtskollegen Sergei Sidorski zustande.
The agreement came after about 10 hours of negotiations between Mr Fradkov and his Belarussian counterpart, Sergei Sidorsky.

H: Herr Fradkow und Sergei Sidorski haben nach 10 Stunden Verhandlungen eine **Einigung** gefunden.
Mr Fradkov and Sergei Sidorsky found an agreement after 10 hours of negotiations.

8.5 Summary

In this Chapter, we have tested whether derivational knowledge helps resolving the task of Recognising Textual Entailment by integrating DERIVBASE into TIE, a standard TE system, with a query expansion approach. Our Derivational Overlap Hypothesis (cf. Section 8.1) was that derivational relationships between T and H suggest the T/H pair to be entailing rather than non-entailing, and that an expansion with derivationally related lemmas increases the chance of a lexical overlap between T and H. We manually compiled

a phenomenon-specific derivational sub-dataset, based on a German translation of the English RTE-3 data, in order to determine the impact of derivational knowledge on TE.

From our experiments, we conclude that the DOH is too simplified to reflect the linguistic variability of the RTE datasets: Entailment relationships involve far more than merely derivational relationship, so that our experiments achieved rather unstable performance results. Nonetheless, DERIVBASE has often a positive, albeit small and statistically not significant impact on the entailment decisions of derivationally related T/H pairs, and does not harm the results. However, derivational information tends to confuse TIE's classifier on T/H pairs without a derivational relationship between Text and Hypothesis. Thus, we suggest to filter T/H pairs for this linguistic phenomenon before the classification, and employ an ensemble of two models for derivational and non-derivational T/H pairs, respectively, as proposed in setting (C) presented in Table 8.4. The ensemble results also underline our finding from Table 8.1 that derivational and syntactic information can be well combined for resolving the RTE task on the whole dataset. Applying the ensemble model with both derivational and syntactic information, we achieve a numerical performance gain of 0.6 and 0.2 percentage points in accuracy and precision, respectively, over using only syntactic features on the standard RTE dataset.

To compare, Dinu and Wang (2009) achieved on the whole English RTE-3 dataset an improvement of 0.63 percentage points in precision over TIE's BOW baseline.[11] However, they use a different knowledge integration strategy than we do: While we combine the TIE-internal features (word overlap and syntax features) with features containing DERIVBASE information for all T/H pairs, they assume that the generated DIRT/WordNet rules lead to entailment whenever they match a T/H pair, and thus assign the decision `entailment = true` to all matching T/H pairs, without considering the BOW baseline's vote. The baseline is, similar to the approach of Shen and Lapata (2007), only used as a fallback for all pairs on which no rule matches. Such an assumption is sensible for a resource like Dinu and Wang (2009) examine, because it is specifically tailored for application in textual entailment. Giving similarly much credit to a derivational lexicon, however, would result in a deterioration of the overall performance.[12] In sum, the quality of our result is roughly comparable to that reported by Dinu and Wang (2009).

Surprisingly, the semantically validated DERIVBASE v2.0 performs not always better than the purely morphological v1.4.1, although its cleaner clusters should work better on the inherently semantic RTE task; v1.4.1 leads to the overall best ensemble setting, and also performs better when no differentiation is made between derivational and non-derivational T/H pairs (Table 8.1). We thus believe that high coverage is at least as important as high quality for the query expansion method that we use to integrate DERIVBASE into TIE. In order to fully take advantage of v2.0, we assume that a different feature extraction strategy would be necessary.

Since TIE is a representative matching-based entailment system, our conclusions should similarly hold for other matching-based TE systems. We also believe that our results can

[11] Accuracy and recall are not reported.

[12] We carried out similar experiments as an alternative to (C), and observed a decline in accuracy from 61.3% on the BOW system to 60.0% on a "rely always on DERIVBASE, else use BOW" setting.

be cautiously interpreted as generally valid for other kinds of entailment algorithms. For example, transformation-based approaches can similarly take advantage of derivational information to transfer one parse tree into another. Such approaches have already been tackled, e.g., by Szpektor and Dagan (2008), Berant et al. (2012), but again, the impact of derivational knowledge was not reported individually.

Additional Experiments. We investigated two further settings for the RTE task that are not reported in detail in this Chapter:

Gold ensemble model: Instead of automatically splitting the dataset into derivational and non-derivational T/H pairs, we employed the gold standard annotation to build the ensemble models. The results are only slightly worse than those of setting (C) (about -0.2 and -0.1 percentage points in accuracy and precision, respectively).

Training on derivational subset: We investigated how results change for setting (A) when the four TIE configurations (Section 8.2.2) are trained only on the 109 derivationally related T/H pairs in the development subset. Again, the trends are similar: BOW and BOW+SYNT are still competitive, but adding DERIVBASE v1.4.1 improves the BOW+SYNT model.

Part IV

Conclusions and Future Directions

9 Conclusions

This Chapter summarises our findings. We resume our contributions, discuss our main insights, and outline promising directions for future research.

9.1 Contributions

In this thesis, we have introduced a two-step methodology for the induction of a derivational lexicon, and its refinement in terms of semantic relatedness. With our approach, we induced DERIVBASE, a derivational lexicon for German that is publicly available. Due to the two-step procedure, DERIVBASE provides two levels of information: 1., purely morphological relatedness (v1.4.1), providing derivational families that include both semantically transparent and opaque relations, and 2., semantically validated relatedness (v2.0), retaining within a cluster only derivationally related lemmas that are semantically transparent. The former information level is acquired by means of a rule-based and data-driven approach, while the latter is attained with a machine learning classifier based on derivation-specific and standard distributional features. The combination of various techniques, including manual and automatic components, leads to a high-quality, but at the same time high-coverage lexical resource, as our evaluations confirm. Our main contributions are the following:

A German Derivational Lexicon. While inflection has received much attention in computational morphology for decades, particular interest in derivation arose only recently. Thus, the number of available derivational lexicons is still small. For German, there existed no such resource before. With DERIVBASE, we created the – to our knowledge – first German derivational lexicon. It satisfies today's standards in terms of coverage, but is also very precise due to the rule-based induction. By providing information about the relatedness of words across parts of speech boundaries, it constitutes a counterpart to most other lexical resources, which typically incorporate only relations within the same word class.

In this context, we have proposed an elaborate evaluation methodology that measures precision and recall on two separate samples, and is thus specifically designed for assessing the performance of a phenomenon-specific resource for which only a subset of a language's word inventory is relevant.

A novelty of DERIVBASE compared to other derivational lexicons is the internal structure of the derivational families that is available in the form of derivation rule paths that relate the lemmas. As our experiments have shown, these paths provide valuable information when it comes to the application in semantic and psycholinguistic tasks.

Our procedure for the lexicon induction and intrinsic evaluation can be transferred to other synthetic languages, provided that knowledge about admissible derivational processes, and a sufficiently big lemmatised corpus are available.

Semantic Validation of a Derivational Lexicon. The need of semantic coherence in derivational lexicons was already observed by other researchers (Jacquemin, 2010, Hathout and Namer, 2014). However, no actual resource with such information has been published so far. The semantic validation of DERIVBASE, conducted with a machine learning classifier, constitutes a new method to introduce a semantic notion into derivational families by separating out opaque lemmas. It achieves considerable improvements in terms of semantic coherence within derivational families.

Our validation method builds upon features extracted from distributional models and the derivation rules that we used to induce DERIVBASE. Nonetheless, it can be similarly applied when no derivational, but only distributional information is available, and still improves the semantic coherence of the derivational families.

Extrinsic experiments have shown that indeed, this refinement yields improvements on different semantic tasks: Although DERIVBASE v2.0 naturally performs worse in reproducing morphological priming effects, and the RTE task seems to depend more on high-coverage resources than on (semantically) high-quality resources, the tasks to detect semantic similarity and synonyms both profit from the semantic validation.

Derivational Information in Distributional Semantics. In Part III, we have presented a novel possibility to employ derivational information in lexical semantics. We have developed a strategy to overcome sparsity in distributional semantic models that we call derivational smoothing. It takes the derivational relatedness between words as evidence for semantic relatedness, and calculates the similarity of two words by considering all members of the respective derivational families.

As our second extrinsic evaluation shows, this notion of applying derivational information is extremely versatile: The derivational smoothing of distributional models can be straightforwardly rephrased for fairly different settings, e.g., the modelling of morphological priming in psycholinguistic experiments by means of morphological generalisation.

Thus, we regard our incorporation of derivation into distributional models as a simple, yet effective enrichment of standard distributional approaches, where the spaces typically have no notion of derivational relatedness.

9.2 Insights

There are some fundamental insights we have gained in our study of German derivation and its application to different tasks. We have evaluated DERIVBASE for its usability in: 1., the application for smoothing distributional semantic models, 2., the replication of morphological priming experiments, and 3., the expansion of an RTE entailment system. The results show that our lexicon can actually yield improvements, but that the impact

essentially depends on the respective task. In the following, we shortly discuss our main findings for each task.

Derivation and Distributional Semantics. On the lexical-semantic tasks in our first evaluation, smoothing a distributional space with DERIVBASE improves the results of an unsmoothed space, particularly when the highest similarity predictions of both spaces per item (MAX) are selected. That is, the semantic similarity of two target items can be well approximated by means of the distributional similarity of their derivational families. We observed that the rule path information that is available from the internal structure of our families improves the semantic similarity predictions, which again shows that our rule-based induction method is beneficial. Notably, the semantically validated lexicon outperforms the purely morphological variant. This behaviour is plausible, since, for v2.0, the target word vectors are only smoothed with vectors of semantically fairly similar (i.e., transparent) words. Both semantic similarity prediction and synonym choice profit from this more focused smoothing.

Also, we compared the quality improvements achieved by derivational smoothing or by combining distributional models with different profiles (i.e., sparse syntax-based and noisy bag of words models): Model combination, although performing better than the individual models, was outperformed by derivational smoothing on the semantic similarity prediction task. Nonetheless, both strategies are, to some extent, complementary, and their combination achieves the best results on the synonym choice task. As for derivational smoothing, model combination (and its combination with derivational smoothing) particularly improves by using the MAX combination strategy.

Overall, our studies have shown that the usability of derivational smoothing crucially depends on the task to be solved: Fine-grained tasks such as the synonym choice profit less from derivational smoothing than coarser-grained similarity tasks, but they can at least take advantage of the complementarity of the two presented improvement strategies.

Finally, we would like to mention that the idea that underlies the MAX combination strategy is worthwhile more thorough investigation: We motivated the usage of MAX with the hypothesis that semantic similarity is more likely to be underestimated than overestimated in distributional models. The results of Utt and Padó (2014) and our study have shown that selecting the maximum prediction of various models indeed yields very good performance, although the profiles of the models used in these two studies were different. Also, our analysis in Chapter 5 revealed underestimations for semantically similar lemma pairs. Thus, this hypothesis might be a fundamental principle in distributional semantics. Yet, this remains to be assessed; a formal examination of this issue could contribute to a theoretical understanding of distributional semantics.

Derivation in Psycholinguistics. Our second extrinsic evaluation makes a contribution to psycholinguistic research in morphological priming: With our strategy of combining a standard distributional semantic model and information extracted from DERIVBASE, we constructed a computational model that could account for morphological priming effects in behavioural experiments. Obviously, the semantically validated DERIVBASE v2.0 is

less appropriate to capture these effects than v1.4.1, as it considers only transparent derivations. Nonetheless, both versions perform more similar to the results of the original study than a model without any information from our lexicon, showing that also v2.0 incorporates morphological aspects. Notably, taking into account the derivation rule path yields the best model. This suggests that the distance between two words in terms of their derivational relatedness has an impact on the morphological processing.

As reported in Chapter 7, Smolka et al. (2014) consider the results of their experimental study as evidence for a morpheme-based organisation of the German mental lexicon, which contrasts with other Indo-European languages. However, our model is able to account for the priming effects found by Smolka et al., although it remains completely at the word level and no morpheme-level knowledge is accessible. Since our model is clearly simpler than what Smolka et al. proposed, we believe that it is a good starting point for investigating other reasons for the specificities in German morphological priming. Particularly, we think that the cross-lingual differences observed by Smolka et al. arise, among others, from the size and the degree of internal semantic relatedness of the derivational families in a language. That is, our results might correspond to that of the original study because our model considers rather large derivational families (in v1.4.1), including many, and notably opaquely related lemmas into the prediction process. In contrast, derivational lexicons for other languages might contain less opaque derivations and thus rather produce semantic priming patterns. We consider this issue an interesting direction for psycholinguists to investigate.

Derivation in Textual Inference. The results for the NLP task of Recognising Textual Entailment are rather mixed and do not provide a clear picture. DErivBase is in fact able to improve the underlying matching-based entailment system when it is only applied to T/H pairs with derivational relationships, however, the gains are small. Unexpectedly, the purely morphological lexicon version performs better than v2.0, although Rte is an inherently semantic task.

We draw two conclusions from these results: 1., it seems that not only semantic coherence, but also high coverage does matter. Rte – at least when it is addressed using a matching-based entailment algorithm – profits more from resources that yield extensive query expansions (i.e., DErivBase v1.4.1) than from semantically clean resources (v2.0). 2., Rte, being an extremely versatile semantic task, can be solved at most partially by addressing only one specific phenomenon – in our case, derivational relationships.

We think that both issues are worthwhile further investigation. Concretely, we propose to either change global parameters in the entailment system (e.g., implementing additional features that capture other linguistic aspects of the T/H pairs, or taking into account the features in a different way than Tie does), or opening the derivational lexicon to additional linguistic phenomena. We will return to the second point in Section 9.3.

In sum, our three evaluations show that DErivBase is applicable to lexical-semantic tasks, and that it can can support the computational modelling of theoretical studies about German morphology. Both variants of our lexicon have proven to be useful for

different purposes. However, its impact is uncertain when it comes to more generic, multi-faceted semantic NLP tasks, where the meaning of entire sentences rather than individual words matter. At this point, we concur with Jacquemin (2010) in that it is challenging to improve an extrinsic task with a resource that is specialised in one linguistic phenomenon.

The main conclusions that we draw from our evaluations is that the usability of a derivational lexicon crucially depends on the kind of task it is applied to. We believe that this variability at least partially arises from the fundamental Derivational Coherence Assumption we have made (cf. Section 2.1.3), i.e., that most derivationally related words are semantically related. Our application of DERIVBASE always relies on this assumption and thus, its success depends on the DCA's appropriateness for the respective task. As is known, semantic relatedness can be understood as a very strict or a fairly general relation, ranging from synonymy to associativity (Budanitsky and Hirst, 2006). Our extrinsic evaluations reflect this broad spectrum fairly well: While the DCA holds for semantic similarity prediction, it is less appropriate for specific semantic relations such as synonymy. Also, since DERIVBASE only refers to the word level, it cannot account for the range of phenomena involved in RTE, where the relatedness between individual words – may it be semantic or derivational – is not sufficient to assess the entailment relationship of two entire utterances.

9.3 Future Directions

We see potential for further improvements of DERIVBASE and starting points for future research particularly in three respects:

Expansion of DErivBase with Compounds: As shown, the impact of our lexicon on versatile tasks such as RTE is unclear – presumably because covering only one phenomenon is not sufficient. While our focus was to assess merely derivational information, DERIVBASE is in principle not restricted to this type of lexical information. Since we have observed that derivational processes interact with other linguistic phenomena, we expect the impact of DERIVBASE to rise when it is opened to these phenomena. Specifically, incorporating compound information might be a reasonable expansion, since it describes another fairly productive morphological process in German that frequently interacts with derivation (cf. Section 8.2.3). Concretely, this expansion could look as follows: Compounds that constitute singleton families could be included into the derivational family with which they share their head word. For instance, the family containing the words *wählen*$_V$ – *Wahl*$_N$ – *gewählt*$_A$ (*to elect*$_V$ – *election*$_N$ – *elected*$_A$), could be expanded by the lemma *Parlamentswahl*$_N$ (*parliamentary election*). To do so, a compound decomposition component would be needed, and one would preferably tag compound relationships as such to indicate different morphological relations. Semantic validation could be similarly conducted on this expanded lexicon, however, derivational rule information would of course not be available for all lemma pairs.

With this expansion, DERIVBASE would be triggered more often, particularly for datasets with many compounds. For instance, the number of T/H pairs in our phenomenon-specific RTE dataset (cf. Section 8.3) would rise by about 16 percentage points, e.g., including Example (8.5) on page 166.

Semantic Validation via Soft Clustering: Due to various reasons, the most important of which is polysemy, the assumption that a lemma can be assigned exactly one semantic cluster in a derivational family, is oversimplified. As mentioned in Section 2.3, a word can have several meanings and thus be part of several semantic clusters, which is ignored by the hard clustering that we have conducted. The solution is to construct soft clusters instead, i.e., to let items participate in several clusters to a certain degree (Xu and Wunsch, 2005, Nock and Nielsen, 2006).

In a tentative study, we analysed the quantitative relevance of soft clustering. We extracted 50 families from v1.4.1, (ranging from two to 156 members) and annotated them according to their semantic coherence: We kept only semantically transparently related lemmas in a cluster, and duplicated lemmas that can participate in several clusters. Indeed, a substantial number of lemmas was duplicated.

Conceptually, soft clustering would lead to a linguistically more plausible representation of semantics in DERIVBASE. From a practical point of view, it would increase the coverage of v2.0, since the soft clusters would become larger than the hard clusters. In this way, the semantically validated version of our lexicon might have more positive impact than v1.4.1 on tasks such as RTE.

Distributional Examination of the Meaning of Derivation: In the recent years, derivational morphology has gained growing interest in computational linguistics, as various proposals to access derivational data (Baranes and Sagot, 2014, Hathout and Namer, 2014, Šnajder, 2014), but also studies that approach the meaning of derivational processes by means of distributional semantics (Lazaridou et al., 2013, Kisselew et al., 2015, Padó et al., 2015, Marelli and Baroni, 2015) suggest. These latter studies investigate to what extent the semantics of derivational affixation can be expressed by vector composition, and what are the influence factors on the performance of such models – e.g., regularities in semantic drifts, or changes of information content through derivation. We concur with these studies in believing that the influence of derivation on word semantics has been linguistically described in detail (cf. Chapter 2), but not yet appropriately operationalised computationally. Using distributional semantic models is a promising direction to investigate these aspects of derivation, and their role in computational semantics.

For such endeavours, DERIVBASE is a valuable resource, providing at the same time information about derivational processes, as well as the "outcome" of such processes in the form of – purely morphological, or semantically validated – derivational families. These information can serve for exploratory single-case studies, but also for the large-coverage examination of the semantics of derivation, and thus help to better understand the impact of derivation on language and language processing.

Appendix

A Employed HOFM Transformation Functions for Derivation

This appendix lists all transformation functions we used to define German derivational processes. We list both language-independent and German-specific rules.

$apfx$**:** Alternate by list of prefix alternations. Corresponds to:
$apfx[(s1, s2), (s3, s4)] = rpfx(s1, s2)$.|. $rpfx(s3, s4)$

$asfx$**:** Alternate by list of infix alternations. Corresponds to:
$aifx[(s1, s2), (s3, s4)] = rifx(s1, s2)$.|. $rifx(s3, s4)$

$dsfx$**:** Delete a suffix. Corresponds to:
$rsfx(s, \text{“ ”})$

*dup***:** Duplicate a final consonant (German-specific). Roughly corresponds to:
$rsfx(c, cc)$

*nul***:** Do nothing (used for conversion)

*opt***:** Optionally perform transformation t. Corresponds to:
t .|. *nul*

pfx**:** Prepend a prefix. Corresponds to:
$rpfx(\text{“ ”}, s)$

*puml***:** Alternate initial vowel to umlaut (German-specific). Corresponds to:
$apfx([(a, \ddot{a}),\ (o, \ddot{o}),\ (u, \ddot{u})])$

$rifx$**:** Replace an infix. Primitive function.

$rpfx$**:** Replace a prefix. Primitive function.

$rsfx$**:** Replace a suffix. Primitive function.

sfx**:** Append a suffix. Corresponds to:
$rsfx(\text{“ ”}, s)$

*try***:** Perform transformation t, if possible. Corresponds to:
t .||. *nul*

*uml***:** Alternate infix vowel to umlaut (German-specific). Corresponds to:
$aifx([(a, \ddot{a}),\ (o, \ddot{o}),\ (u, \ddot{u})])$

B Implemented DErivBase Rules, v1.4.1

This appendix contains the derivation rules covered in version 1.4.1 in DERIVBASE. They are displayed exactly as they are implemented in the corresponding Haskell module.

Each derivation rule has a unique name indicating the part of speech of the base and the derived word, followed by a serial number. For instance, `dNN01` indicates the first noun-noun derivation rule. Every derivation rule is accompanied by German examples in the format `base word -> derivative`, given in comments (lines starting with `--`).

```
-------------------------------------------------------------------------------
-- 1. NOUN DERIVATION
-------------------------------------------------------------------------------

-----------------------
-- 1.1 NOUN TO NOUN
-----------------------

-- Bäcker -> Bäckerei, Rüpel -> Rüpelei, Träumer -> Träumerei, Türke -> Türkei
NN01 = dPattern "NN01"
  (sfx "ei" & try (dsfx "e")) mNouns fNouns

-- Bäcker -> Bäckerin, Idiot -> Idiotin, Türke -> Türkin, Vanille -> Vanillin
NN02 = dPattern "NN02"
  (sfx "in" & try (dsfx "e")) nouns nouns

-- Dieb -> Dieberei, Sklave -> Sklaverei, Abgott -> Abgötterei, Schwein -> Schweinerei
NN03 = dPattern "NN03"
  (sfx "erei" & opt uml & try (dsfx "e")) nouns fNouns

-- Anwalt -> Anwaltschaft, Freund -> Freundschaft, Friede -> Friedschaft
NN04 = dPattern "NN04"
  (sfx "schaft" & try (dsfx "e")) nouns fNouns

-- Schule -> Schüler, Attentat -> Attentäter, Kritik -> Kritiker,
-- Italien -> Italiener, England -> Engländer, Australien -> Australier,
-- Argentinien -> Argentinier, Kanada -> Kanadier Abenteuer -> Abenteurer,
-- Gambia -> Gambier, Ghana -> Ghanaer, Forschen -> Forscher, Garten -> Gärtner
NN05 = dPattern "NN05"
  (sfx "er" & opt uml & try (rsfx "er" "r" .||. dsfx "e"
  .||.  opt (dsfx "en" .|. rsfx "en" "n") .||. try (dsfx "ien" .|. rsfx "ien" "i"))
  & try (rsfx "ia" "i") & opt (rsfx "a" "i")) nouns mNouns

-- Bank -> Bankier, Hotel -> Hotelier, Karabiner -> Karabinier, Kneipe -> Kneipier
NN06 = dPattern "NN06"
  (sfx "ier" & try (dsfx "er" .||. dsfx "e")) nouns mNouns

-- Alkohol -> Alkoholiker, Hygiene -> Hygieniker
NN07 = dPattern "NN07"
  (sfx "iker" & try (dsfx "e" .||. dsfx "er")) nouns mNouns

-- Piano -> Pianist, Kapital -> Kapitalist, Bratsche -> Bratschist,
-- Bhudda -> Bhuddist, Bigamie -> Bigamist
NN08 = dPattern "NN08"
  (sfx "ist" & try (asfx [("a",""), ("e",""), ("i",""), ("o",""), ("u",""), ("ie",""),
  ("ei",""), ("eu",""), ("au",""), ("äu",""), ("ai",""), ("ui","")])) nouns mNouns

-- Alkohol -> Alkoholismus, Anarchie -> Anarchismus, Bhudda -> Bhuddismus
NN09 = dPattern "NN09"
  (sfx "ismus" & try (asfx [("a",""), ("e",""), ("i",""), ("o",""), ("u",""),
  ("ie",""), ("ei",""), ("eu",""), ("au",""), ("äu",""), ("ai",""), ("ui","")]))
  nouns mNouns

-- Direktor -> Direktorium, Privileg -> Privilegium
NN10 = dPattern "NN10"
  (sfx "ium") nouns nNouns

-- Luther -> Lutheraner, Peru -> Peruaner, Angola -> Angolaner
-- Salvador -> Salvadorianer, Nigeria -> Nigerianer, Sizilien -> Sizilianer,
-- Monarchie -> Monarchianer, Mauritius -> Mauritianer, Marokko -> Marrokkaner,
NN11 = dPattern "NN11"
  (sfx "aner" & try (rsfx "or" "ori" .||. rsfx "ia" "i" .||. rsfx "ien" "i" .||.
  rsfx "ie" "i" .||. rsfx "ius" "i" .||. dsfx "o" .||. dsfx "a")) nouns mNouns

-- Asien -> Asiat, Aluminium -> Aluminat, Stipendium -> Stipendiat
NN12 = dPattern "NN12"
  (sfx "at" & try (asfx [("a",""), ("en",""), ("ium",""), ("um","")]))
  nouns (mNouns++nNouns)
```

```
-- Schwede -> Schweden, Pole -> Polen, Bayer -> Bayern
NN13 = dPattern "NN13"
  (sfx "n") mNouns nNouns

-- Ozean -> Ozeanien, Serbe -> Serbien, Rumäne -> Rumänien
NN14 = dPattern "NN14"
  (sfx "ien" & try (dsfx "e")) mNouns nNouns

-- Este -> Estland, Feuer -> Feuerland, Deutsche -> Deutschland
--NN15 = dPattern "NN15"
--  (sfx "land" & try (dsfx "e")) nouns nNouns

-- Jemen -> Jemenit, Israel -> Israelit, Sunna -> Sunnit, Jesus -> Jesuit,
-- Hus -> Hussit, Chlor -> Chlorit
NN16 = dPattern "NN16"
  (sfx "it" & try (asfx [("a",""), ("us","u"), ("us","uss")])) nouns nouns

-- China -> Chinese, Vietnam -> Vietnamese, Togo -> Togolese, Lugano -> Luganese,
-- Bali -> Balinese, Libanon -> Libanese
NN17 = dPattern "NN17"
  (sfx "ese" & try (dsfx "a" .||. rsfx "i" "in" .||. dsfx "on"
  .||. try (rsfx "o" "ol" .|. dsfx "o"))) nNouns mNouns

-- Erfolg -> Misserfolg
NN18 = align $ dPattern "NN18"
  (pfx "miss") nouns nouns

-- Beweis -> Gegenbeweis, Pol -> Gegenpol
NN19 = align $ dPattern "NN19"
  (pfx "gegen") nouns nouns

-- Faschismus -> Antifaschismus
NN20 = align $ dPattern "NN20"
  (pfx "anti") nouns nouns

-- Ruhe -> Unruhe, Kraut -> Unkraut
NN21 = align $ dPattern "NN21"
  (pfx "un") nouns nouns

-- Harmonie -> Disharmonie, Agio -> Disagio
NN22 = align $ dPattern "NN22"
  (pfx "dis") nouns nouns

-- Betrag -> Fehlbetrag
NN23 = align $ dPattern "NN23"
  (pfx "fehl") nouns nouns

-- Wille -> Widerwille
NN24 = align $ dPattern "NN24"
  (pfx "wider") nouns nouns

-- Funktion -> Dysfunktion
NN25 = align $ dPattern "NN25"
  (pfx "dys") nouns nouns

-- Schlag -> Konterschlag
NN26 = align $ dPattern "NN26"
  (pfx "konter") nouns nouns

-- Kompression -> Dekompression, Orientierung -> Desorientierung, Interesse -> Desinteresse
NN27 = align $ dPattern "NN27"
  (pfx "de" & try (apfx [("a","sa"), ("e","se"), ("i","si"), ("o","so"), ("u","su")]))
  nouns nouns

-- Kunst -> Künstler, Sport -> Sportler, Wissenschaft -> Wissenschaftler,
-- Sommerfrische -> Sommerfrischler
NN28 = dPattern "NN28"
  (sfx "ler" & opt uml & try (dsfx "e")) nouns mNouns

-- Teil -> Anteil, Zeichen -> Anzeichen
NN29 = align $ dPattern "NN29"
  (pfx "an") nouns nouns

-- Preis -> Aufpreis, Marsch -> Aufmarsch
NN30 = align $ dPattern "NN30"
  (pfx "auf") nouns nouns

-- Maß -> Ausmaß, Land -> Ausland, Zeit -> Auszeit
NN31 = align $ dPattern "NN31"
  (pfx "aus") nouns nouns

-- Zimmer -> Nebenzimmer
```

```
NN32 = align $ dPattern "NN32"
  (pfx "neben") nouns nouns

-- Geschmack -> Nachgeschmack
NN33 = align $ dPattern "NN33"
  (pfx "nach") nouns nouns

-- Besitz -> Mitbesitz
NN34 = align $ dPattern "NN34"
  (pfx "mit") nouns nouns

-- Abend -> Vorabend
NN35 = align $ dPattern "NN35"
  (pfx "vor") nouns nouns

-- Erwerb -> Zuerwerb, Name -> Zuname
NN36 = align $ dPattern "NN36"
  (pfx "zu") nouns nouns

-- Haus -> Gehäuse, Lumpen -> Gelumpe, Wasser -> Gewässer, Darm -> Gedärme
NN37 = dPattern "NN37"
  (opt (sfx "e") & pfx "ge" & opt (uml) & try (dsfx "n")) nouns nNouns

-- Tasse -> Untertasse, Offizier -> Unteroffizier
NN38 = align $ dPattern "NN38"
  (pfx "unter") nouns nouns

-- Kreis -> Umkreis, Lauf -> Umlauf
NN39 = align $ dPattern "NN39"
  (pfx "um") nouns nouns

-- Bau -> Abbau, Luft -> Abluft, Zeichen -> Abzeichen
NN40 = align $ dPattern "NN40"
  (pfx "ab") nouns nouns

-- Dienst -> Außendienst, Ansicht -> Außenansicht
NN41 = align $ dPattern "NN41"
  (pfx "außen") nouns nouns

-- Blatt -> Beiblatt, Geschmack -> Beigeschmack
NN42 = align $ dPattern "NN42"
  (pfx "bei") nouns nouns

-- Markt -> Binnenmarkt, Zoll -> Binnenzoll
NN43 = align $ dPattern "NN43"
  (pfx "binnen") nouns nouns

-- Bischof -> Erzbischof, Engel -> Erzengel
NN44 = align $ dPattern "NN44"
  (pfx "erz") nouns nouns

-- Sorge -> Fürsorge, Wort -> Fürwort
NN45 = align $ dPattern "NN45"
  (pfx "für") nouns nouns

-- Nahrung -> Grundnahrung, Bestandteil -> Grundbestandteil
NN46 = align $ dPattern "NN46"
  (pfx "grund") nouns nouns

-- Bestandteil -> Hauptbestandteil, Bahnhof -> Hauptbahnhof
NN47 = align $ dPattern "NN47"
  (pfx "haupt") nouns nouns

-- Deck -> Oberdeck, Befehl -> Oberbefehl
NN48 = align $ dPattern "NN48"
  (pfx "ober") nouns nouns

-- Antwort -> Rückantwort, Reise -> Rückreise
NN49 = align $ dPattern "NN49"
  (pfx "rück") nouns nouns

-- Zug -> Sonderzug, Angebot -> Sonderangebot
NN50 = align $ dPattern "NN50"
  (pfx "sonder") nouns nouns

-- Gewicht -> Übergewicht, Mensch -> Übermensch
NN51 = align $ dPattern "NN51"
  (pfx "über") nouns nouns

-- Hemd -> Unterhemd, Führung -> Unterführung
```

```
NN52 = align $ dPattern "NN52"
  (pfx "unter") nouns nouns

-- Zeit -> Zwischenzeit, Wand -> Zwischenwand
NN53 = align $ dPattern "NN53"
  (pfx "zwischen") nouns nouns

-- Dienst -> Innendienst, Welt -> Innenwelt
NN54 = align $ dPattern "NN54"
  (pfx "innen") nouns nouns

-- Mensch -> Urmensch, Enkel -> Urenkel, Aufführung -> Uraufführung
NN55 = align $ dPattern "NN55"
  (pfx "ur") nouns nouns

-- Bauch -> Bäuchlein, Ente -> Entlein, Licht -> Lichtlein, Brunnen -> Brünnlein,
-- Vogel -> Vögelein
NN56 = dPattern "NN56"
  (sfx "lein" & try (dsfx "en" .||. dsfx "e" .||. dsfx "l") & try (uml)) nouns nNouns

-- Schiff -> Schiffchen, Figur -> Figürchen, Rose -> Röschen, Krug -> Krügelchen,
-- Tuch -> Tüchelchen, Sache -> Sächelchen, Haar -> Härchen, Boot -> Bötchen
NN57 = dPattern "NN57"
  (sfx "chen" & try (rsfx "g" "gel" .||. rsfx "sch" "sch" .||. rsfx "ch" "chel")
  & try (dsfx "en" .||. dsfx "e") & try (uml) & try (rifx "aa" "ä" .||. rifx "oo" "ö"
  .||. rifx "uu" "ü")) nouns nNouns

-- Arm -> Ärmel, Kies -> Kiesel, Busch -> Büschel
NN58 = dPattern "NN58"
  (sfx "el" & opt (puml .|. uml)) nouns nouns

-- Kind -> Kindheit, Gott -> Gottheit, Tor -> Torheit, Wesen -> Wesenheit
NN59 = dPattern "NN59"
  (sfx "heit") nouns fNouns

-- Gans -> Gänserich, Weg -> Wegerich, Maus -> Mäuserich
NN60 = dPattern "NN60"
  (sfx "rich" & opt (dsfx "e" .||. sfx "e") & try uml) nouns mNouns

-- Sultan -> Sultanine, Nektar -> Nektarine, Mandola -> Mandoline,
-- Sonate -> Sonatine
NN61 = dPattern "NN61"
  (sfx "ine" & try (dsfx "e" .||. dsfx "a")) nouns fNouns

-- Antiquar -> Antiquariat, Kommissar -> Kommissariat
NN62 = dPattern "NN62"
  (sfx "iat") nouns nNouns

-- Chanson -> Chansonette, Statue -> Statuette
NN63 = dPattern "NN63"
  (sfx "ette" & try (dsfx "e")) nouns fNouns

-- Delphin -> Delphinarium
NN64 = dPattern "NN64"
  (sfx "arium" & try (dsfx "e")) nouns nNouns

-- Division -> Divisionär, Legion -> Legionär
NN65 = dPattern "NN65"
  (sfx "är" & try (dsfx "e")) nouns mNouns


--------------------------
-- 1.2 ADJECTIVE TO NOUN
--------------------------

-- tief -> Tiefe, weit -> Weite (n/f), deutsch -> Deutsche, scharf -> Schärfe,
-- lang -> Länge, eben -> Ebene, erinnernd -> Erinnernde, erinnert -> Erinnerte
AN01 = dPattern "AN01"
  (sfx "e" & try uml) adjectives nouns

-- dunkel -> Dunkelheit, fein -> Feinheit, berühmt -> Berühmtheit
AN02 = dPattern "AN02"
  (sfx "heit") adjectives fNouns

-- wichtig -> Wichtigkeit, traurig -> Traurigkeit
AN03 = dPattern "AN03"
  (sfx "keit") adjectives fNouns

-- dreist -> Dreistigkeit, ernsthaft -> Ernsthaftigkeit
AN04 = dPattern "AN04"
  (sfx "igkeit") adjectives fNouns

-- faul -> Fäulnis, vermacht -> Vermächtnis, geheim -> Geheimnis,
-- finster -> Finsternis
AN05 = dPattern "AN05"
  (sfx "nis" & try uml) adjectives (fNouns++nNouns)
```

```
-- schwanger -> Schwangerschaft, bereit -> Bereitschaft, gefangen -> Gefangenschaft
AN06 = dPattern "AN06"
  (sfx "schaft") adjectives fNouns

-- einverstanden -> Einverständnis, gefangen -> Gefängnis
AN07 = dPattern "AN07"
  (rsfx "en" "nis" & try uml) adjectives (fNouns++nNouns)

-- schwach -> Schwächling, roh -> Rohling, schön -> Schönling
AN08 = dPattern "AN08"
  (sfx "ling" & opt uml) adjectives mNouns

-- absolut -> Absolutismus, exotisch -> Exotismus, klassisch -> Klassizismus,
-- totalitär -> Totalitarismus, industriell -> Industrialismus
AN09 = dPattern "AN09"
  (sfx "ismus" & try (rsfx "ell" "al" .||.rsfx "är" "ar"
  .||. (try (dsfx "isch" .|. rsfx "isch" "iz")))) adjectives mNouns

-- kulant -> Kulanz, rasant -> Rasanz, präsent -> Präsenz, konsequent -> Konsequenz
AN10 = dPattern "AN10"
  (rsfx "t" "z") adjectives fNouns

-- analog -> Analogie, infam -> Infamie, idiotisch -> Idiotie
AN11 = dPattern "AN11"
  (sfx "ie" & try (dsfx "isch")) adjectives fNouns

-- linguistisch -> Linguistik, theatralisch -> Theatralik, problematisch -> Problematik,
-- anglistisch -> Anglistik
AN12 = dPattern "AN12"
  (rsfx "isch" "ik") adjectives fNouns

-- allergisch -> Allergiker, zynisch -> Zyniker
AN13 = dPattern "AN13"
  (rsfx "isch" "iker") adjectives mNouns

-- antibiotisch -> Antibiotikum, charakteristisch -> Charakteristikum
AN14 = dPattern "AN14"
  (rsfx "isch" "ikum") adjectives nNouns

-- aktiv -> Aktivist, amerikanisch -> Amerikanist, linguistisch -> Linguist
AN15 = dPattern "AN15"
  (sfx "ist" & try (dsfx "istisch" .||. dsfx "isch")) adjectives mNouns

-- banal -> Banalität, elektrisch -> Elektrizität, aktuell -> Aktualität,
-- kompatibel -> Kompatibilität
AN16 = dPattern "AN16"
  (sfx "ität" & try (rsfx "isch" "iz" .||. rsfx "ell" "al" .||. rsfx "bel" "bil"))
  adjectives fNouns

-- grau -> Grau, seidenmatt -> Seidenmatt, tannengrün -> Tannengrün
AN17 = dPattern "AN17"
  nul adjectives nNouns

-- dick -> Dickerchen, dumm -> Dummerchen
AN18 = dPattern "AN18"
  (sfx "erchen") adjectives nNouns


----------------------
-- 1.3 VERB TO NOUN
----------------------

-- singen -> Gesang
VN01 = dPattern "VN01"
  (pfx "ge" & rifx "i" "a") verbs mNouns

-- reden -> Gerede, zanken -> Gezanke, tosen -> Getöse, lasen -> Gebläse/Geblase
VN02 = dPattern "VN02"
  (pfx "ge" & sfx "e" & opt uml) verbs nNouns

-- tanzen -> Tänzer, rauchen -> Raucher, lehren -> Lehrer, wildern -> Wilderer,
-- sammeln -> Sammler
VN03 = dPattern "VN03"
  (sfx "er" & opt uml & try (rsfx "el" "l")) verbs mNouns

-- pfeifen -> Pfeife, ernten -> Ernte
VN04 = dPattern "VN04"
 (sfx "e") verbs fNouns

-- for -en verbs: schlagen -> Schlägel, decken -> Deckel
-- for -eln verbs: hebeln -> Hebel, würfeln -> Würfel, segeln -> Segel, handeln -> Handel
VN05 = dPattern "VN05"
 (sfx "el" & opt uml & try (dsfx "el")) [verbEn,verbEln] nouns

-- erleben -> Erlebnis, empfangen -> Empfängnis, ärgern -> Ärgernis
```

```
VN06 = dPattern "VN06"
  (sfx "nis" & try uml) verbs (fNouns++nNouns)

-- rechnen -> Rechnung, entleeren -> Entleerung
VN07 = dPattern "VN07"
  (sfx "ung") verbs fNouns

-- abgeben -> Abgabe
VN08 = dPattern "VN08"
  (rifx "e" "a" & sfx "e") verbs fNouns

-- for -eln and -ern verbs: handeln -> Handeln, segeln -> Segeln,
-- baggern -> Baggern, lagern -> Lagern, säubern -> Säubern
-- for -en verbs: bersten -> Bersten, säugen -> Säugen
VN09 = dPatternLL "VN09"
  nul verbs nNouns

-- kochen -> Koch, fischen -> Fisch, verstecken -> Versteck, schauen -> Schau,
-- einkaufen -> Einkauf, Baggern -> Bagger, Handeln -> Handel
VN10 = dPattern "VN10"
  nul verbs nouns

-- for -en and -ern verbs: wandern -> Wanderschaft, pflegen -> Pflegschaft
VN11 = dPattern "VN11"
  (sfx "schaft") verbs fNouns

-- sieden -> Sud, schließen -> Schluss, fließen -> Fluss, gießen -> Guss
VN12 = dPattern "VN12"
  (rifx "ie" "u" & try (rsfx "ß" "ss")) verbs nouns

-- klingen -> Klang, dringen -> Drang, binden -> Band, zwingen -> Zwang, trinken -> Trank
VN13 = dPattern "VN13"
  (rifx "i" "a") verbs nouns

-- brechen -> Bruch, heben -> Hub, scheren -> Schur, sprechen -> Spruch
VN14 = dPattern "VN14"
  (rifx "e" "u") verbs nouns

-- gebieten -> Gebot, sprießen -> Spross
VN15 = dPattern "VN15"
  (rifx "ie" "o" & try (rsfx "ß" "ss")) verbs nouns

-- treiben -> Trieb, scheiden -> Schied
VN16 = dPattern "VN16"
  (rifx "ei" "ie") verbs nouns

-- reißen -> Riss, streichen -> Strich, schreiten -> Schritt, greifen -> Griff,
-- pfeifen -> Pfiff, kneifen -> Kniff
VN17 = dPattern "VN17"
  (rifx "ei" "i" & (opt dup .|. try (rsfx "ß" "ss"))) verbs nouns

-- springen -> Sprung, finden -> Fund
VN18 = dPattern "VN18"
  (rifx "i" "u") verbs nouns

-- spedieren -> Spediteur, komponieren -> Kompositeur
VN19 = dPattern "VN19"
  (rsfx "ier" "iteur" & try (rsfx "ponier" "posier")) [verbEn] mNouns

-- generieren -> Generator, applizieren -> Applikator, diktieren -> Diktator
VN20 = dPattern "VN20"
  (rsfx "ier" "ator" & try (rsfx "izier" "ikier")) [verbEn] mNouns

-- doktorieren -> Doktorand, operieren -> Operand
VN21 = dPattern "VN21"
  (rsfx "ier" "and") [verbEn] mNouns

-- Variant of -and: promovieren -> Promovend, addieren -> Addend
VN22 = dPattern "VN22"
  (rsfx "ier" "end") [verbEn] mNouns

-- demonstrieren -> Demonstrant, operieren -> Operant
VN23 = dPattern "VN23"
  (rsfx "ier" "ant") [verbEn] mNouns

-- fundieren -> Fundament, temperieren -> Temperament
VN24 = dPattern "VN24"
  (rsfx "ier" "ament") [verbEn] nNouns

-- blamieren -> Blamage, massieren -> Massage
VN25 = dPattern "VN25"
  (rsfx "ier" "age") [verbEn] fNouns

-- tolerieren -> Toleranz, repräsentieren -> Repräsentanz
VN26 = dPattern "VN26"
  (rsfx "ier" "anz") [verbEn] fNouns
```

```
-- signieren -> Signatur, reparieren -> Reparatur
VN27 = dPattern "VN27"
  (rsfx "ier" "atur") [verbEn] fNouns

-- abonnieren -> Abonnement, arrangieren -> Arrangement
VN28 = dPattern "VN28"
  (rsfx "ier" "ement") [verbEn] nNouns

-- absolvieren -> Absolvent, dirigieren -> Dirigent
VN29 = dPattern "VN29"
  (rsfx "ier" "ent") [verbEn] mNouns

-- existieren -> Existenz, präferieren -> Präferenz
VN30 = dPattern "VN30"
  (rsfx "ier" "enz") [verbEn] fNouns

-- exorzieren -> Exorzist, kopieren -> Kopist, komponieren -> Komponist
VN31 = dPattern "VN31"
  (rsfx "ier" "ist") [verbEn] mNouns

-- studieren -> Studium, refugieren -> Refugium
VN32 = dPattern "VN32"
  (rsfx "ier" "ium") [verbEn] nNouns

-- garnieren -> Garnitur, polieren -> Politur
VN33 = dPattern "VN33"
  (rsfx "ier" "itur") [verbEn] fNouns

-- addieren -> Addition, opponieren -> Opposition, komponieren -> Komposition
VN34 = dPattern "VN34"
  (rsfx "ier" "ition" & try (rsfx "ponier" "posier")) [verbEn] fNouns

-- flimmern -> Geflimmer, brabbeln -> Gebrabbel, bellen -> Gebell,
-- spotten -> Gespött, backen -> Gebäck, jodeln -> Gejodel, hören -> Gehör
VN35 = dPattern "VN35"
  (pfx "ge" & opt (uml)) verbs nNouns

-- schreiben -> Geschreibsel, mengen -> Gemengsel
VN36 = dPattern "VN36"
  (pfx "ge" & sfx "sel") [verbEn] nNouns

-- sortieren -> Sortiment, regieren -> Regiment
VN37 = dPattern "VN37"
  (rsfx "ier" "iment") [verbEn] nNouns

-- hypnotisieren -> Hypnotiseur, vulkanisieren -> Vulkaniseur
VN38 = dPattern "VN38"
  (rsfx "ier" "eur") [verbEn] mNouns

-- animieren -> Animation, dekorieren -> Dekoration
VN39 = dPattern "VN39"
  (rsfx "ier" "ation") [verbEn] fNouns

-- animieren -> Animateur, dekorieren -> Dekorateur
VN40 = dPattern "VN40"
  (rsfx "ier" "ateur") [verbEn] mNouns


--------------------------------------------------------------------------------
-- 2. ADJECTIVE DERIVATION
--------------------------------------------------------------------------------

--------------------------
-- 2.1 NOUN TO ADJECTIVE
--------------------------

-- Kunst -> künstlich, Herr -> herrlich, Glück -> glücklich
NA01 = dPattern "NA01"
  (sfx "lich" & try uml) nouns adjectives

-- Künstler -> künstlerisch, Alkohol -> alkoholisch, Stadt -> städtisch,
-- Himmel -> himmlisch, Schule -> schulisch, Brite -> britisch, Ägypten -> ägyptisch
NA02 = dPattern "NA02"
  (sfx "isch" & try (rsfx "el" "l") & try (dsfx "e" .||. dsfx "en") & opt uml)
  nouns adjectives

-- Gesetz -> gesetzmäßig, Instinkt -> instinktmäßig, Arbeit -> arbeitsmäßig,
-- Gewohnheit -> gewohnheitsmäßig
NA03 = dPattern "NA03"
  (sfx "mäßig" & opt (sfx "s")) nouns adjectives

-- Schleier -> schleierhaft, Schatten -> schattenhaft, Sünde -> sündhaft
NA04 = dPattern "NA04"
  (sfx "haft" & try (dsfx "e")) nouns adjectives
```

```
-- Busch -> buschig, Vorrang -> vorrangig, Übermut -> übermütig, Erde -> erdig,
-- Knoten -> knotig
NA05 = dPattern "NA05"
  (sfx "ig" & try (dsfx "e") & try (dsfx "en") & opt uml) nouns adjectives

-- Funktion -> funktional, Zentrum -> zentral, Orchester -> orchestral, Epoche -> epochal,
-- Margo -> marginal, Nomen -> nominal, Vagina -> vaginal, Episkopus -> episkopal, Axis -> axial
NA06 = dPattern "NA06"
  (sfx "al" & try (asfx [("er","r"), ("um",""), ("o","in"), ("en","in"), ("a",""),
  ("us",""), ("e",""), ("is","i")])) nouns adjectives

-- Bestie -> bestialisch, Musik -> musikalisch, Theater -> theatralisch
NA07 = dPattern "NA07"
  (sfx "alisch" & try (rsfx "er" "r" .||. dsfx "um" .||. rsfx "en" "in" .||. dsfx "a"
  .||. dsfx "us" .||. dsfx "e" .||. dsfx "is")) nouns adjectives

-- Afrika -> afrikanisch, Tibet -> tibetanisch, Ecuador -> ecuadorianisch,
-- Brasilien -> brasilianisch
NA08 = dPattern "NA08"
  (sfx "anisch" & try (dsfx "a" .||. rsfx "or" "ori" .||. rsfx "ia" "i"
  .||. rsfx "ien" "i" .||. rsfx "ius" "i" .||. dsfx "o")) nouns adjectives

-- Atom -> atomar, Pol -> polar, Molekül -> molekular, Binokel -> binokular,
-- Lamelle -> lamellar, Linie -> linear, Korona -> koronar
NA09 = dPattern "NA09"
  (sfx "ar" & try (rsfx "ül" "ul" .||. rsfx "el" "ul" .||. rsfx "ie" "e"
  .||. dsfx "e" .||. dsfx "a")) nouns adjectives

-- Defizit -> defizitär, Familie -> familiär, Revolution -> revolutionär,
-- Universität -> universitär, Muskel -> muskulär, Opposition -> oppositär,
-- Station -> stationär, Tempus -> temporär, Primus -> primär
NA10 = dPattern "NA10"
  (sfx "är" & try (dsfx "e" .||. dsfx "ät" .||. rsfx "el" "ul" .||. opt (dsfx "ion")
  .||. rsfx "us" "or") & opt (dsfx "us")) nouns adjectives

-- Disziplin -> disziplinarisch, Legende -> legendarisch, Kalender -> kalendarisch
NA11 = dPattern "NA11"
  (sfx "arisch" & try (dsfx "e" .||. dsfx "ät" .||. rsfx "er" "ar" .||. dsfx "a"))
  nouns adjectives

-- Problem -> problematisch, Klima -> klimatisch, Asien -> asiatisch
NA12 = dPattern "NA12"
  (sfx "atisch" & try (dsfx "a" .||. dsfx "e" .||. dsfx "en")) nouns adjectives

-- Bakterie -> bakteriell, Kultur -> kulturell, Individuum -> individuell,
-- Habitus -> habituell, Tempus -> temporell
NA13 = dPattern "NA13"
  (sfx "ell" & try (dsfx "e" .||. dsfx "um"
  .||. try (rsfx "us" "or" .|. rsfx "us" "u")) & opt (dsfx "us")) nouns adjectives

-- Chaplin -> chaplinesk, Karneval -> karnevalesk, Kafka -> kafkaesk
NA14 = dPattern "NA14"
  (sfx "esk") nouns adjectives

-- Diät -> diätetisch, Energie -> energetisch, Pathos -> pathetisch
NA15 = dPattern "NA15"
  (sfx "etisch" & try (dsfx "ie" .||. dsfx "os")) nouns adjectives

-- Äquator -> äquatorial, Kollege -> kollegial, Tangens -> tangential,
-- Essenz -> essential, Existenz -> existenzial
NA16 = dPattern "NA16"
  (sfx "ial" & try (dsfx "e" .||. rsfx "ens" "ent" .||. opt (rsfx "enz" "ent")))
  nouns adjectives

-- Vektor -> vektoriell, Prinzip -> prinzipiell, Essenz -> essenziell
NA17 = dPattern "NA17"
  (sfx "iell" & opt (rsfx "nz" "nt")) nouns adjectives

-- Byzanz -> byzantinisch, Montenegro -> montenegrinisch, Dalmatien -> dalmatinisch,
-- Menorca -> menorquinisch, Sarde -> sardinisch
NA18 = dPattern "NA18"
  (sfx "inisch" & try (rsfx "nz" "nt" .||. dsfx "o" .||. dsfx "ien"
  .||. rsfx "ca" "qu" .||. dsfx "e")) nouns adjectives

-- Charakter -> charakteristisch, Harmonie -> harmonistisch,
-- Flora -> floristisch, Cello -> cellistisch, Renaissance -> renaissancistisch
NA19 = dPattern "NA19"
  (sfx "istisch" & try (dsfx "ie" .||. dsfx "i" .||. dsfx "a" .||. dsfx "o"
  .||. dsfx "e")) nouns adjectives

-- Affekt -> affektiv, Instinkt -> instinktiv, Intuition -> intuitiv
NA20 = dPattern "NA20"
  (sfx "iv" & try (dsfx "ion")) nouns adjectives
```

```
-- Muskel -> muskulös, Lepra -> leprös, Strapaze -> strapaziös,
-- Tendenz -> tendenziös, Melodie -> melodiös, Religion -> religiös,
-- Mysterium -> mysteriös, Tuberkulose -> tuberkulös, Volumen -> voluminös,
-- Spektakel -> spektakulös, Luxus -> luxuriös, Minute -> minuziös,
-- Monstrum -> monströs
NA21 = dPattern "NA21"
  (sfx "ös" & try (asfx [("a",""), ("ze","zi"), ("z","zi"), ("ie","i"),
  ("ion","i"), ("ose",""), ("en","in"), ("el","ul"), ("us","uri"), ("te","zi"),
  ("um","")])) nouns adjectives

-- Emanzipation -> emanzipatorisch, Illusion -> illusorisch,
-- Provokation -> provokatorisch
NA22 = dPattern "NA22"
  (rsfx "ion" "orisch") nouns adjectives

-- Prozess -> prozessual, Ritus -> ritual
NA23 = dPattern "NA23"
  (sfx "ual" & try (dsfx "us")) nouns adjectives

-- Kontext -> kontextuell, Prozent -> prozentuell
NA24 = dPattern "NA24"
  (sfx "uell" & try (dsfx "us" .||. dsfx "ion")) nouns adjectives

-- Orientierung -> orientiert, Verständigung -> verständigt
NA25 = dPattern "NA25"
  (rsfx "ung" "t") fNouns adjectives

-- Verherrlichung -> verherrlichend, Verständigung -> verständigend,
-- Senkung -> senkend
NA26 = dPattern "NA26"
  (rsfx "ung" "end") fNouns adjectives

-- Urkunde -> urkundlich, Sprache -> sprachlich
NA27 = dPattern "NA27"
  (sfx "lich" & try (dsfx "e")) nouns adjectives

-- Ehre -> ehrsam, Furcht -> furchtsam
NA28 = dPattern "NA28"
  (sfx "sam" & try (dsfx "e")) nouns adjectives

-- Wurzel -> wurzellos, Mühe -> mühelos, Schaden -> schadlos, Freude -> freudlos,
-- Wolke -> wolkenlos, Boden -> bodenlos, Staat -> staatenlos, Kind -> kinderlos,
-- Ausweg -> auswegslos, Mann -> männerlos
NA29 = dPattern "NA29"
  (sfx "los" & opt (dsfx "en" .|. dsfx "e" .|. ((sfx "er" .|. sfx "en")
  & try (dsfx "e")) .|. sfx "er" .|. sfx "s") & opt (uml)) nouns adjectives

-- Blatt -> blätt(e)rig, Loch -> löch(e)rig, Glied -> glied(e)rig, Knochen -> knöchrig
NA30 = dPattern "NA30"
  (sfx "rig" & opt (sfx "e") & try (dsfx "en") & opt (uml)) nouns adjectives

-- Hass -> gehässig, Lehre -> gelehrig, Zahn -> gezahnt, Witz -> gewitzt,
-- Blume -> geblümt, Treue -> getreu,  Lappen -> gelappt, Streifen -> gestreift
NA31 = dPattern "NA31"
  (pfx "ge" & opt (sfx "t" .|. sfx "ig") & try (dsfx "e") & opt (uml))
  nouns adjectives


----------------------------
-- 2.2 ADJECTIVE TO ADJECTIVE
----------------------------

-- rund -> rundlich, wahr -> wahrlich
AA01 = dPattern "AA01"
  (sfx "lich") adjectives adjectives

-- sagbar -> unsagbar, ehrenhaft -> unehrenhaft
AA02 = dPattern "AA02"
  (pfx "un") adjectives adjectives

-- autoritär -> antiautoritär
AA03 = dPattern "AA03"
  (pfx "anti") adjectives adjectives

-- harmonisch -> disharmonisch
AA04 = dPattern "AA04"
  (pfx "dis") adjectives adjectives

-- natürlich -> widernatürlich
AA05 = dPattern "AA05"
  (pfx "wider") adjectives adjectives

-- zentral -> dezentral, armiert -> desarmiert
AA06 = dPattern "AA06"
  (pfx "de" & try (apfx [("a","sa"), ("e","se"), ("i","si"), ("o","so"),
  ("u","su")])) adjectives adjectives
```

```
-- normal -> abnormal, hold -> abhold
AA07 = dPattern "AA07"
  (pfx "ab") adjectives adjectives

-- verständlich -> missverständlich, gelaunt -> missgelaunt
AA08 = dPattern "AA08"
  (pfx "miss") adjectives adjectives

-- katholisch -> erzkatholisch
AA09 = dPattern "AA09"
  (pfx "erz") adjectives adjectives

-- anständig -> grundanständig
AA10 = dPattern "AA10"
  (pfx "grund") adjectives adjectives

-- betrieblich -> zwischenbetrieblich
AA11 = dPattern "AA11"
  (pfx "zwischen") adjectives adjectives

-- betrieblich -> innerbetrieblich
AA12 = dPattern "AA12"
  (pfx "inner") adjectives adjectives

-- geboren -> nachgeboren, industriell ->  nachindustriell
AA13 = dPattern "AA13"
  (pfx "nach") adjectives adjectives

-- faul -> oberfaul
AA14 = dPattern "AA14"
  (pfx "ober") adjectives adjectives

-- natürlich -> übernatürlich
AA15 = dPattern "AA15"
  (pfx "über") adjectives adjectives

-- bewusst ->  unterbewusst
AA16 = dPattern "AA16"
  (pfx "unter") adjectives adjectives

-- bestraft -> vorbestraft, gesehen -> vorgesehen
AA17 = dPattern "AA17"
  (pfx "vor") adjectives adjectives

-- betrieblich -> außerbetrieblich, irdisch -> außerirdisch
AA18 = dPattern "AA18"
  (pfx "außer") adjectives adjectives

-- komisch -> urkomisch, eigen -> ureigen
AA19 = dPattern "AA19"
  (pfx "ur") adjectives adjectives

-- deutsch -> binnendeutsch
AA20 = dPattern "AA20"
  (pfx "binnen") adjectives adjectives

-- sozial -> asozial
AA21 = dPattern "AA21"
  (pfx "a") adjectives adjectives

-- polar -> biploar
AA22 = dPattern "AA22"
  (pfx "bi") adjectives adjectives

-- feudal -> feudalistisch, zentral -> zentralistisch, klassisch -> klassizistisch,
-- dogmatisch -> dogmatistisch
AA23 = dPattern "AA23"
  (sfx "istisch" & try (dsfx "isch" .|. rsfx "isch" "iz")) adjectives adjectives


-------------------------
-- 2.3 VERB TO ADJECTIVE
-------------------------

-- sagen -> sagbar, schließen -> schließbar, wandeln -> wandelbar, filtern -> filterbar
VA01 = dPattern "VA01"
  (sfx "bar") verbs adjectives

-- sinken -> sinkend
VA02 = dPattern "VA02"
  (sfx "end") verbs adjectives

-- auffallen -> auffällig, knacken -> knackig
VA03 = dPattern "VA03"
  (sfx "ig" & opt uml) verbs adjectives

-- akzeptieren -> akzeptabel, praktizieren -> praktikabel
```

```
VA04 = dPattern "VA04"
  (rsfx "ier" "abel" & try (rsfx "izier" "ikier")) [verbEn] adjectives

-- amüsieren -> amüsant, imponieren -> imposant
VA05 = dPattern "VA05"
  (rsfx "ier" "ant" & try (rsfx "ponier" "posier")) [verbEn] adjectives

-- demonstrieren -> demonstrativ, informieren -> informativ, evozieren -> evokativ
VA06 = dPattern "VA06"
  (rsfx "ier" "ativ" & try (rsfx "zier" "kier")) [verbEn] adjectives

-- existieren -> existent, kongruieren -> kongruent
VA07 = dPattern "VA07"
  (rsfx "ier" "ent") [verbEn] adjectives

-- explodieren -> explosibel, konvertieren -> konvertibel, dividieren -> divisibel,
-- komprimieren -> kompressibel, flektieren -> flexibel
VA08 = dPattern "VA08"
  (rsfx "ier" "ibel" & try (rsfx "dier" "sier" .||. rsfx "primier" "pressier"
  .||. rsfx "vidier" "visier" .||. rsfx "ktier" "xier")) [verbEn] adjectives

-- prohibieren -> prohibitiv
VA09 = dPattern "VA09"
  (rsfx "ier" "itiv" & try (rsfx "ponier" "posier")) [verbEn] adjectives

-- adaptieren -> adaptiv, deprimieren -> depressiv, explodieren -> explosiv,
-- produzieren -> produktiv, extrahieren -> extraktiv, deskribieren -> deskriptiv,
-- rezipieren -> rezeptiv, defendieren -> defensiv, destruieren -> destruktiv,
-- agieren -> aktiv, dirigieren -> direktiv, adhärieren -> adhäsiv,
-- subsumieren -> subsumptiv, suggerieren -> suggestiv
VA10 = dPattern "VA10"
  (rsfx "ier" "iv" & try (asfx [("primier","pressier"), ("plodier","plosier"),
  ("duzier","duktier"), ("trahier","traktier"), ("ibier","iptier"),
  ("ipier","eptier"), ("dier","sier"), ("uier","uktier"), ("gier","ktier"),
  ("igier","ektier"), ("rier","sier"), ("umier","umptier"), ("erier","estier")]))
  [verbEn] adjectives

-- beachten -> beachtlich, verantworten -> verantwortlich
VA11 = dPattern "VA11"
  (sfx "lich") verbs adjectives

-- verherrlichen -> verherrlichend, belagern -> belagernd
VA12 = dPattern "VA12"
  (sfx "nd" & opt (sfx "e")) verbs adjectives

-- erfrischen -> erfrischt, belagern -> belagert,
-- verkleiden -> verkleidet, verästen -> verästet
VA13 = dPattern "VA13"
  (sfx "t" & try (rsfx "d" "de" .||. rsfx "t" "te")) verbs adjectives

-- unterhalten -> unterhaltsam, lehren -> lehrsam, folgen -> folgsam
VA14 = dPattern "VA14"
  (sfx "sam") verbs adjectives

-- glauben -> glaubhaft, lachen -> lachhaft, schmeicheln -> schmeichelhaft
VA15 = dPattern "VA15"
  (sfx "haft") verbs adjectives

-- kleben -> klebrig, glitschen -> glitsch(e)rig, schlafen -> schläfrig
VA16 = dPattern "VA16"
  (sfx "rig" & opt (sfx "e") & opt (uml)) verbs adjectives


--------------------------------------------------------------------------------
-- 3. VERB DERIVATION
--------------------------------------------------------------------------------

----------------------
-- 3.1 NOUN TO VERB
----------------------

-- Anspruch -> beanspruchen, Auftrag -> beauftragen, Nebel -> benebeln
NV01 = dPatternSS "NV01"
  (pfx "be") nouns verbs

-- Dolch -> erdolchen, Hitze -> erhitzen, Schauder -> erschaudern, Mangel -> ermangeln
NV02 = dPatternSS "NV02"
  (pfx "er" & try (dsfx "e")) nouns verbs

-- Gleis -> entgleisen, Kraft -> entkräften, Erbe -> enterben
NV03 = dPatternSS "NV03"
  (pfx "ent" & opt uml & try (dsfx "e")) nouns verbs

-- Arm -> umarmen, Ring -> umringen, Mantel -> ummanteln
NV04 = dPatternSS "NV04"
```

```
   (pfx "um") nouns verbs

-- -en verbs: Abschied -> verabschieden, Klage -> verklagen
NV05 = dPatternSS "NV05"
   (pfx "ver" & try (dsfx "e")) nouns [verbEn]

-- -ern verbs: Knochen -> verknöchern, Stein -> versteinern, Donner -> verdonnern,
-- Körper -> verkörpern
NV06 = dPatternSS "NV06"
   (pfx "ver" & sfx "er" & opt uml & try (dsfx "er" .||. dsfx "en")) nouns [verbErn]

-- -eln verbs: Ekel -> verekeln, Doppel -> verdoppeln, Spiegel -> verspiegeln
NV07 = dPatternSS "NV07"
   (pfx "ver") nouns [verbEln]

-- Friede -> befriedigen, Ende -> beendigen, Kost -> beköstigen,
-- Jubel -> bejubeln, Feuer -> befeuern
NV08 = dPatternSS "NV08"
   (pfx "be" & try (rsfx "e" "ig" .||. opt (sfx "ig")) & opt uml) nouns verbs

-- Strand->stranden, Hammer->Hämmern, Haufen->häufen, Computer->computern,
-- Schnorchel->schnorcheln
NV09 = dPatternSS "NV09"
   (try (dsfx "en") & opt uml) nouns verbs

-- -en verbs: Befehl -> befehligen, Angst -> ängstigen, Sünde -> sündigen, Huld -> huldigen
NV10 = dPatternSS "NV10"
   (sfx "ig" & try (dsfx "e") & opt uml) nouns [verbEn]

-- -en verbs: Struktur -> strukturieren, Funktion -> funktionieren,
-- Harmonie -> harmonieren, Flanke -> flankieren, Interesse -> interessieren,
-- Analyse -> analysieren, Fokus -> fokussieren
NV11 = dPatternSS "NV11"
   (sfx "ier" & try (rsfx "sss" "ss") & opt (rsfx "s" "ss") & try (dsfx "ie"
   .||. dsfx "e")) nouns [verbEn]

-- -en verbs: Alkohol -> alkoholisieren, Motor -> motorisieren,
-- Euphorie -> euphorisieren, Hygiene -> hygienisieren
NV12 = dPatternSS "NV12"
   (sfx "isier" & try (dsfx "ie" .||. dsfx "e")) nouns [verbEn]

-- -ern verbs: Wild -> wildern, Geist -> geistern, Rauch -> räuchern
NV13 = dPatternSS "NV13"
   (sfx "er" & try uml) nouns [verbErn]

-- -eln verbs: Kunst -> künsteln, Gift -> gifteln, Stück -> stückeln, Rad -> radeln
NV14 = dPatternSS "NV14"
   (sfx "el" & opt uml) nouns [verbEln]

-- Freund -> anfreunden, Pranger -> anprangern, Bändel -> anbändeln
NV15 = dPatternSS "NV15"
   (pfx "an" & try (dsfx "e")) nouns verbs

-- Sockel -> aufsockeln, Möbel -> aufmöbeln
NV16 = dPatternSS "NV16"
   (pfx "auf") nouns verbs

-- Boot -> ausbooten, Preis -> auspreisen, Tonne -> austonnen, Ufer -> ausufern,
-- Bogen -> ausbogen
NV17 = dPatternSS "NV17"
   (pfx "aus" & try (dsfx "e") & try (rsfx "en" "e")) nouns verbs

-- Kluft -> zerklüften, Matsch -> zermatschen,
NV18 = dPatternSS "NV18"
   (pfx "zer" & opt (dsfx "en") & opt (uml)) nouns verbs

-- Pflicht -> beifplichten, Steuer -> beisteuern, Menge -> beimengen, Wohnen -> beiwohnen
NV19 = dPatternSS "NV19"
   (pfx "bei" & try (dsfx "e" .||. dsfx "en")) nouns verbs

-- Igel -> einigeln, Dose -> eindosen, Delle -> eindellen, Bürger -> einbürgern
NV20 = dPatternSS "NV20"
   (pfx "ein" & try (dsfx "e")) nouns verbs

-- Tunnel -> untertunneln, Keller -> unterkellern
NV21 = dPatternSS "NV21"
   (pfx "unter") nouns verbs

-- Nabel -> abnabeln, Kupfer -> abkupfern, Zweig -> abzweigen
NV22 = dPatternSS "NV22"
   (pfx "ab" & try (dsfx "e")) nouns verbs

-- Brücke -> überbrücken, Winter -> überwintern, Kruste -> überkrusten
NV23 = dPatternSS "NV23"
   (pfx "über" & try (dsfx "e")) nouns verbs
```

```
-- Eile -> durcheilen, Seuche -> durchseuchen, Winter -> durchwintern
NV24 = dPatternSS "NV24"
  (pfx "durch" & try (dsfx "e")) nouns verbs


-------------------------
-- 3.2 ADJECTIVE TO VERB
-------------------------

-- -en verbs: anonym -> anonymisieren, banal -> banalisieren, elektrisch -> elektrisieren
AV01 = dPatternSS "AV01"
  (sfx "isier" & try (rsfx "isch" "")) adjectives [verbEn]

-- -en verbs: aktiv -> aktivieren, kokett -> kokettieren
AV02 = dPatternSS "AV02"
  (sfx "ier") adjectives [verbEn]

-- -eln verbs: fremd -> fremdeln, krank -> kränkeln
AV03 = dPatternSS "AV03"
  (sfx "el" & opt uml) adjectives [verbEln]

-- dicht -> dichten, sicher -> sichern, krumm -> krümmen, dunkel -> dunkeln
AV04 = dPatternSS "AV04"
  (opt uml) adjectives verbs

-- -en verbs: frei -> befreien, schuldig -> beschuldigen, taub -> betäuben,
-- -ern verbs: teuer -> beteuern
AV05 = dPatternSS "AV05"
  (pfx "be" & opt uml) adjectives [verbEn,verbErn]

-- deutlich -> verdeutlichen, einsam -> vereinsamen, breit -> verbreitern, edel -> veredeln
AV06 = dPatternSS "AV06"
  (pfx "ver" & opt (sfx "er")) adjectives verbs

-- möglich -> ermöglichen, heiter -> erheitern
AV07 = dPatternSS "AV07"
  (pfx "er") adjectives [verbEn,verbErn]

-- bieder -> anbiedern, rauh -> anrauhen, reich -> anreichern
AV08 = dPatternSS "AV08"
  (pfx "an" & opt (sfx "er")) adjectives [verbEn,verbErn]

-- heiter -> aufheitern, klar -> aufklaren
AV09 = dPatternSS "AV09"
  (pfx "auf") adjectives [verbEn,verbErn]

-- dünn -> ausdünnen, nüchtern -> ausnüchtern
AV10 = dPatternSS "AV10"
  (pfx "aus" & try (rsfx "ern" "er")) adjectives [verbEn,verbErn]

-- mürbe -> zermürben, klein -> zerkleinern
AV11 = dPatternSS "AV11"
  (pfx "zer" & opt (sfx "er") & try (dsfx "e")) adjectives verbs

-- eng -> einengen, schüchtern -> einschüchtern, schwarz -> einschwärzen
AV12 = dPatternSS "AV12"
  (pfx "ein" & try (rsfx "ern" "er") & try (uml)) adjectives verbs

-- flach -> abflachen, dunkel -> abdunkeln, mager -> abmagern
AV13 = dPatternSS "AV13"
  (pfx "ab" & try (rsfx "ern" "er")) adjectives verbs

-- müde -> übermüden, teuer -> überteuern
AV14 = dPatternSS "AV14"
  (pfx "über" & try (rsfx "ern" "er" .||. dsfx "e")) adjectives verbs

-- düster -> umdüstern
AV15 = dPatternSS "AV15"
  (pfx "um") adjectives verbs

-- quer -> durchqueren
AV16 = dPatternSS "AV16"
  (pfx "durch") adjectives verbs

-- dunkel -> nachdunkeln, braun -> nachbräunen
AV17 = dPatternSS "AV17"
  (pfx "nach" & opt (uml)) adjectives verbs


----------------------
-- 3.3 VERB TO VERB
----------------------

-- husten -> hüsteln, drängen -> drängeln, zucken -> zuckeln
VV01 = dPatternSS "VV01"
  (sfx "el" & opt uml) [verbEn] [verbEln]

-- dienen -> bedienen, heizen -> beheizen, finden -> befinden, schämen -> beschämen,
-- suchen -> besuchen, krabbeln -> bekrabbeln, hindern -> behindern
```

```
VV02 = align $ dPatternSS "VV02"
  (pfx "be") verbs verbs

-- forschen -> erforschen, morden -> ermorden, eilen -> ereilen,
-- fordern -> erfordern, betteln -> erbetteln
VV03 = align $ dPatternSS "VV03"
  (pfx "er") verbs verbs

-- -en verbs: denken -> gedenken, leiten -> geleiten, hören -> gehören
VV04 = align $ dPatternSS "VV04"
  (pfx "ge") verbs [verbEn]

-- ändern -> verändern, kalkulieren -> verkalkulieren, reisen -> verreisen,
-- brühen -> verbrühen
VV05 = align $ dPatternSS "VV05"
  (pfx "ver") verbs verbs

-- leiten -> fehlleiten, schlagen -> fehlschlagen, handeln -> fehlhandeln
VV06 = align $ dPatternSS "VV06"
  (pfx "fehl") verbs verbs

-- legen -> widerlegen, fahren -> widerfahren
VV07 = align $ dPatternSS "VV07"
  (pfx "wider") [verbEn] [verbEn]

-- achten -> missachten, handeln -> misshandeln, verstehen -> missverstehen
VV08 = align $ dPatternSS "VV08"
  (pfx "miss") verbs verbs

-- qualifizieren -> disqualifizieren, fundieren -> diffundieren
VV09 = align $ dPatternSS "VV09"
  (try (rpfx "disf" "diff") & pfx "dis") [verbEn] [verbEn]

-- hydrieren -> dehydrieren, organisieren -> desorganisieren, aktivieren -> deaktivieren
VV10 = align $ dPatternSS "VV10"
  (pfx "de" & try (apfx [("e","se"), ("i","si"), ("o","so"), ("u","su")])) [verbEn] [verbEn]

-- tränken -> trinken
VV11 = align $ dPatternSS "VV11"
  (rifx "ä" "i") verbs [verbEn]

-- legen -> liegen, senken -> sinken, setzen -> sitzen
VV12 = align $ dPatternSS "VV12"
  (opt (rifx "ie" "i") & rifx "e" "ie") verbs [verbEn]

-- backen -> anbacken, häkeln -> anhäkeln, dauern -> andauern
VV13 = align $ dPatternSS "VV13"
  (pfx "an") verbs verbs

-- blasen -> aufblasen, bohren -> aufbohren, schütteln -> aufschütteln
VV14 = align $ dPatternSS "VV14"
  (pfx "auf") verbs verbs

-- beißen -> zerbeißen, bröckeln -> zerbröckeln, schmettern -> zerschmettern
VV15 = align $ dPatternSS "VV15"
  (pfx "zer") verbs verbs

-- fragen -> hinterfragen, gehen -> hintergehen, mauern -> hintermauern
VV16 = align $ dPatternSS "VV16"
  (pfx "hinter") verbs verbs

-- lagern -> zwischenlagern, landen -> zwischenlanden
VV17 = align $ dPatternSS "VV17"
  (pfx "zwischen") verbs verbs

-- biegen -> zurechtbiegen, basteln -> zurechtbasteln
VV18 = align $ dPatternSS "VV18"
  (pfx "zurecht") verbs verbs

-- binden -> zubinden, laufen -> zulaufen, ballern -> zuballern
VV19 = align $ dPatternSS "VV19"
  (pfx "zu") verbs verbs

-- erstatten -> rückerstatten, koppeln -> rückkoppeln, erobern -> rückerobern
VV20 = align $ dPatternSS "VV20"
  (pfx "rück") verbs verbs

-- bringen -> beibringen, füttern -> beifüttern
VV21 = align $ dPatternSS "VV21"
  (pfx "bei") verbs verbs
```

```
-- fallen -> einfallen, schlafen -> einschlafen, buddeln -> einbuddeln
VV22 = align $ dPatternSS "VV22"
  (pfx "ein") verbs verbs

-- bringen -> darbringen, reichen -> darreichen
VV23 = align $ dPatternSS "VV23"
  (pfx "dar") verbs [verbEn]

-- koppeln -> loskoppeln, lösen -> loslösen, fahren -> losfahren
VV24 = align $ dPatternSS "VV24"
  (pfx "los") verbs verbs

-- buttern -> unterbuttern, bringen -> unterbringen, laufen -> unterlaufen
VV25 = align $ dPatternSS "VV25"
  (pfx "unter") verbs verbs

-- fahren -> abfahren, bestellen -> abbestellen
VV26 = align $ dPatternSS "VV26"
  (pfx "ab") verbs verbs

-- behandeln -> vorbehandeln, werfen -> vorwerfen
VV27 = align $ dPatternSS "VV27"
  (pfx "vor") verbs verbs

-- halten -> innehalten, haben -> innehaben
VV28 = align $ dPatternSS "VV28"
  (pfx "inne") verbs [verbEn]

-- altern -> überaltern, streifen -> überstreifen, setzen -> übersetzen,
-- blicken -> überblicken
VV29 = align $ dPatternSS "VV29"
  (pfx "über") verbs verbs

-- fahren -> umfahren, krempeln -> umkrempeln
VV30 = align $ dPatternSS "VV30"
  (pfx "um") verbs verbs

-- atmen -> durchatmen, ackern -> durchackern, bohren -> durchbohren
VV31 = align $ dPatternSS "VV31"
  (pfx "durch") verbs verbs

-- bitten -> fürbitten, sprechen -> fürsprechen
VV32 = align $ dPatternSS "VV32"
  (pfx "für") verbs verbs

-- ahmen -> nachahmen, bereiten -> nachbereiten
VV33 = align $ dPatternSS "VV33"
  (pfx "nach") verbs verbs

-- gewinnen -> wiedergewinnen, erlangen -> wiedererlangen
VV34 = align $ dPatternSS "VV34"
  (pfx "wieder") verbs verbs
```

C Annotation Guidelines

This appendix presents the guidelines used for annotating the five-fold classification of lemma pairs, **S**, **M**, **N**, **C**, **L** (cf. Section 4.3.3). The text is slightly modified for reasons of layout, diction, and consistency with the thesis.

Annotation guidelines

- We annotate pairs of lemmas regarding their semantic and morphological relatedness. The goal is to assess if pairs of lemmas which are put into the same cluster indeed belong there. Examples:
  1. Tanz - tanzen
  2. Tanz - Distanz
  3. tanzen - tanzel

- There are 5 annotation categories, each with a single-letter abbreviation:
  - **S**: The lemmas are **S**emantically and morphologically related (ex. 1. above)
  - **M**: The lemmas are **M**orphologically related, but there is no close semantic relation (e.g., suchen/besuchen)
  - **N**: The lemmas are **N**either morphologically nor semantically related (ex. 2. above)
  - **L**: At least one word is not a valid German **L**emma (ex. 3. above):
    - Inflected words: günstigste (comparatives, superlatives, etc.)
    - Incorrectly stemmed words: Anführungsstrichelch, Wehrlosen_N
    - POS-tagging errors, including:
      - Foreign words tagged as NNs: Kung, Tent
      - Incorrect noun gender: Box_Nm
    - Note: Unspecified noun gender (Box_N) would be accepted
  - **C**: The two lemmas share at least one morpheme, but the "paths" between the lemmas involve **C**omposition. Examples:
    - Kratzer – Wolkenkratzer (via Kratzer)
    - Wolkenkratzer – Schuhkratzer (via Kratzer)
    - Wolkenkratzer – Wolkenkuckucksheim (via Wolke)
    - Wolkenkratzer – abkratzen (via kratzen)

    NB. We do **not** use **C** for proper names or for cases which do not share one morpheme (Gegenrichtung – Titelträger → **N**)
    In case of doubt, distinguish with the following guidelines:

  - The morphemes are complete words with proper semantics: **C**
  - The morphemes have no concrete semantics (be-, -lich): not **C**
  - The morpheme is also used for clear derivation cases: not **C**. Example: Fall → **Bei**fall
  - The morpheme is a rather a prepositional composition: **C**. Example: gehen → **vorbei**gehen

- There are two additional annotation symbols:

  ? Uncertainty about the decision, e.g.,
  M? Vormundschaft_Nf bevormunden_V

  \+ At least one of the lemmas is polysemous, e.g.,
  S + Bank_Nf Banker_Nm

- Decisions in cases of doubt:
  - References:
    If **morphological relatedness** is unclear: Lookup in canoo.net's Wortbildung (`http://canoo.net/` → introduce word → click "Wortbildung")
    If **word existence** is unclear (i.e. for NEs): Lookup in Wortschatz Leipzig (`http://wortschatz.uni-leipzig.de/`)
    However: Linguistic intuition always trumps the use of these resources (e.g., in the case of misspellings listed in Wortschatz Leipzig)!
  - For **ambiguous** lemmas: Accept, if there is a relation in at least one sense
  - Treatment of **named entities** (incl. proper nouns): Accept if actually related (Ungarn – ungarisch); reject if unrelated (Ungarn – garen); pairs of two (mostly unrelated) named entities are rejected
  - Accept present/past **participles** and **conversions** (erfrischend_A, erfrischt_A, Analysieren_Nn)
  - For **compounds**: check for relatedness between the stems of the parts (e.g., Autoteil – Autoteiler → morphological relatedness)

D Abridged TIE Configuration File for Setting BOW

```
<?xml version="1.0" encoding="utf-8"?>
<!--
 Description: Given a certain configuration, the EDA - MaxEntClassificationEDA -
 can be trained over a specific data set in order to optimize its performance
 based on Maximum Entropy Modelling. MaxEntClassificationEDA learns a binary
 classifier for deciding the entailment problem. The supervised learner receives
 a number of features which are provided through the interplay of different
 processing components and knowledge sources. This configuration file specifies
 these components and knowledge sources together with settings of the learner.
-->
<configuration>
  <!-- Platform configuration section -->
  <section name="PlatformConfiguration">
    <!-- The EDA to be used: MaxEntClassificationEDA -->
    <property name="activatedEDA">
     eu.excitementproject.eop.core.MaxEntClassificationEDA</property>
    <!-- The language: DE for German -->
    <property name="language">DE</property>
    <!-- The linguistic annotation pipeline to preprocess the data to be
       annotated: here, the MaltParser dependency parser for DE is selected. -->
    <property name="activatedLAP">
     eu.excitementproject.eop.lap.dkpro.MaltParserDE</property>
  </section>

  <!-- Base processing component for computing a bag of words representation -->
  <section name="BagOfWordsScoring">
  </section>

  <!-- Base processing component for computing a bag of lemmas representation -->
  <section name="BagOfLemmasScoring">
  </section>

  <!-- MaxEntClassificationEDA uses maximum entropy modelling for a learning
     entailment classifier. -->
  <section name="eu.excitementproject.eop.core.MaxEntClassificationEDA">
    <!-- The name of the model -->
    <property name="modelFile">
./src/main/resources/model/MaxEntClassificationEDAModel_Base_DE</property>
    <!-- Location for storing temporary files for training -->
    <property name="trainDir">./target/DE_deriv/dev/</property>
    <!-- Location for storing temporary files for testing -->
    <property name="testDir">./target/DE_deriv/test/</property>
    <!-- Two parameters of the MaxEnt classifier, "max iterations" and "cut off".
       left number ::= iterations - The number of GIS iterations to perform.
       right number ::= cutoff - The number of times a feature must be
       seen in order to be relevant for training. -->
    <property name="classifier">10000,1</property>

    <!-- List of processing components specified above. The order is relevant. -->
    <property name="Components">BagOfWordsScoring,BagOfLemmasScoring</property>
  </section>
</configuration>
```

Bibliography

George W. Adamson and Jillian Boreham. The Use of an Association Measure Based on Character Structure to Identify Semantically Related Pairs of Words and Document Titles. *Information Processing and Management*, 10(7/8):253–260, 1974.

Eneko Agirre and Philip Edmonds, editors. *Word Sense Disambiguation: Algorithms and Applications*, volume 33 of *Text, Speech and Language Technology*. Springer, 2006.

James Allan and Giridhar Kumaran. Stemming in the language modeling framework. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 455–456, Toronto, Canada, 2003. ACM.

Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. A Comparison of Extrinsic Clustering Evaluation Metrics Based on Formal Constraints. *Information Retrieval*, 12(4):461–486, 2009.

Mark Andrews, Gabriella Vigliocco, and David Vinson. Integrating Experiential and Distributional Data to Learn Semantic Representations. *Psychological Review*, 116(3): 463–498, 2009.

Ion Androutsopoulos and Prodromos Malakasiotis. A Survey of Paraphrasing and Textual Entailment Methods. *Journal of Artificial Intelligence Research*, 38(1):135–187, 2010.

Mark Aronoff. *Word Formation in Generative Grammar*. Number 1 in Linguistic Inquiry Monographs. MIT Press, Cambridge, MA, USA, 1976.

Gerhard Augst. *Lexikon zur Wortbildung*. Forschungsberichte des Instituts für Deutsche Sprache. Narr, Tübingen, Germany, 1975.

Necip Fazil Ayan, Bonnie Dorr, and Nizar Habash. Multi-align: Combining linguistic and statistical techniques to improve alignments for adaptable MT. In *Proceedings of the 6th Conference of the Association for Machine Translation in the Americas*, pages 17–26, Washington, DC, USA, 2004. Springer.

R. Harald Baayen, Richard Piepenbrock, and Leon Gulikers. *The CELEX Lexical Database. Release 2. LDC96L14*. Linguistic Data Consortium, University of Pennsylvania, Philadelphia, Pennsylvania, 1996.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The Berkeley FrameNet project. In *Proceedings of the Joint Annual Meeting of the Association for Computational Linguistics and International Conference on Computational Linguistics*, pages 86–90, Montreal, Canada, 1998. Association for Computational Linguistics.

Marion Baranes and Benoît Sagot. A language-independent approach to extracting derivational relations from an inflectional lexicon. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pages 2793–2799, Reykjavik, Iceland, 2014. European Language Resources Association (ELRA).

Marco Baroni and Silvia Bernardini. A New Approach to the Study of Translationese: Machine-learning the Difference between Original and Translated Text. *Literary and Linguistic Computing*, 21(3):259–274, 2006.

Marco Baroni and Alessandro Lenci. Distributional Memory: A General Framework for Corpus-based Semantics. *Computational Linguistics*, 36(4):673–721, 2010.

Marco Baroni, Johannes Matiasek, and Harald Trost. Unsupervised discovery of morphologically related words based on orthographic and semantic similarity. In *Proceedings of the ACL Workshop on Morphological and Phonological Learning*, pages 48–57. Association for Computational Linguistics, 2002.

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.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 238–247, Baltimore, MD, USA, 2014. Association for Computational Linguistics.

Lawrence W. Barsalou. Ad hoc Categories. *Memory and Cognition*, 11(3):211–227, 1983.

Edwin L. Battistella. *The Logic of Markedness*. Oxford University Press, New York, NY, USA, 1996.

Kenneth R. Beesley and Lauri Karttunen. *Finite State Morphology*, volume 18. CSLI Publications, Stanford, CA, USA, 2003.

Jonathan Berant, Ido Dagan, and Jacob Goldberger. Learning Entailment Relations by Global Graph Structure Optimization. *Computational Linguistics*, 38(1):73–111, 2012.

Adam Berger, Rich Caruana, David Cohn, Dayne Freitag, and Vibhu Mittal. Bridging the lexical chasm: statistical approaches to answer-finding. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 192–199, Athens, Greece, 2000. ACM.

Shane Bergsma, Dekang Lin, and Randy Goebel. Discriminative learning of selectional preference from unlabeled text. In *Proceedings of the 13th Conference on Empirical Methods in Natural Language Processing*, pages 59–68, Honolulu, Hawaii, 2008. Association for Computational Linguistics.

Balthazar Bickel and Johanna Nichols. Inflectional Morphology. In Timothy Shopen, editor, *Language Typology and Syntactic Description, Volume III: Grammatical Categories and the Lexicon*, pages 169–240. Cambridge University Press, Cambridge, UK, 2001.

Orhan Bilgin, Ozlem Çetinoğlu, and Kemal Oflazer. Morphosemantic relations in and across Wordnets. In *Proceedings of the 2nd Global WordNet Conference*, pages 60–66, Brno, Czech Republic, 2004. Masaryk University, Brno.

J. Kathryn Bock. Syntactic Persistence in Language Production. *Cognitive Psychology*, 18(3):355–387, 1986.

Bernd Bohnet. Top accuracy and fast dependency parsing is not a contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 89–97, Beijing, China, 2010. Coling 2010 Organizing Committee.

Bernd Bohnet, Joakim Nivre, Igor Boguslavsky, Richárd Farkas, Filip Ginter, and Jan Hajič. Joint Morphological and Syntactic Analysis for Richly Inflected Languages. *Transactions of the Association for Computational Linguistics*, 1:415–428, 2013.

Gemma Boleda, Sabine Schulte im Walde, and Toni Badia. Modeling Regular Polysemy: A Study on the Semantic Classification of Catalan Adjectives. *Computational Linguistics*, 38(3):575–616, 2012.

Carlo E. Bonferroni. Teoria Statistica delle Classi e Calcolo delle Probabilità. *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commerciali di Firenze*, 8:3–62, 1936.

Geert E. Booij. Inflection and Derivation. In Geert E. Booij, Christian Lehmann, and Joachim Mugdan, editors, *Morphologie: Ein Internationales Handbuch zur Flexion und Wortbildung*, volume 1 of *Handbücher zur Sprach- und Kommunikationswissenschaft*, pages 360–369. Mouton de Gruyer, 2000.

Geert E. Booij. *The Grammar of Words: An Introduction to Linguistic Morphology*. Oxford Textbooks in Linguistics. Oxford University Press, New York, NY, USA, 2005.

Sonja Bosch, Christiane Fellbaum, and Karel Pala. Enhancing WordNets with morphological relations: A case study from Czech, English and Zulu. In *Proceedings of the 4th Global WordNet Conference*, pages 74–90, Szeged, Hungary, 2008. University of Szeged.

Jan A. Botha and Phil Blunsom. Compositional morphology for word representations and language modelling. In *Proceedings of the 31st International Conference on Machine Learning*, Beijing, China, 2014. ACM.

Sami Boudelaa and William D. Marslen-Wilson. Abstract Morphemes and Lexical Representation: The CV-Skeleton in Arabic. *Cognition*, 92(3):271–303, 2004.

Sabine Brants, Stefanie Dipper, Peter Eisenberg, Silvia Hansen, 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.

Samuel Broscheit, Massimo Poesio, Simone P. Ponzetto, Kepa J. Rodriguez, Lorenza Romano, Olga Uryupina, Yannick Versley, and Roberto Zanoli. BART: A multilingual anaphora resolution system. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 104–107, Uppsala, Sweden, 2010. Association for Computational Linguistics.

Peter F. Brown, Vincent J. Della Pietra, Peter V. de Souza, and Robert L. Mercer. Class-Based n-gram Models of Natural Language. *Computational Linguistics*, 18(4): 467–479, 1992.

Elia Bruni, Giang Binh Tran, and Marco Baroni. Distributional semantics from text and images. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, pages 22–32, Edinburgh, UK, 2011.

Alexander Budanitsky and Graeme Hirst. Evaluating WordNet-based Measures of Lexical Semantic Relatedness. *Computational Linguistics*, 32(1):13–47, 2006.

John A. Bullinaria and Joseph P. Levy. Extracting Semantic Representations from Word Co-occurrence Statistics: A Computational Study. *Behavior Research Methods*, 39(3): 510–526, 2007.

John A. Bullinaria and Joseph P. Levy. Extracting Semantic Representations from Word Co-occurrence Statistics: Stop-lists, Stemming, and SVD. *Behavior Research Methods*, 44(4):890–907, 2012.

Curt Burgess. From Simple Associations to the Building Blocks of Language: Modeling Meaning in Memory with the HAL Model. *Behavior Research Methods, Instruments, & Computers*, 30(2):188–198, 1998.

Hadumod Bußmann, editor. *Lexikon der Sprachwissenschaft*. Kröner, Stuttgart, Germany, 3rd edition, 2002.

Brian Butterworth. Lexical Representation. In *Language Production*, pages 257–294. Academic Press, London, UK, 1983.

Joan L. Bybee. *Morphology: A Study of the Relation Between Meaning and Form*. John Benjamins Publishing Company, Amsterdam, Netherlands, 1985.

Joan L. Bybee. Morphology as Lexical Organization. In Michael Hammond and Michael Noonan, editors, *Theoretical Morphology*, pages 119–141. Academic Press, San Diego, 1988.

Jean C. Carletta. Assessing Agreement on Classification Tasks: the Kappa Statistic. *Computational Linguistics*, 22(2):249–254, 1996.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems Technology*, 2(3):27:1–27:27, 2011.

Stanley F. Chen and Joshua Goodman. An Empirical Study of Smoothing Techniques for Language Modeling. *Computer Speech and Language*, 13(4):359–394, 1999.

Harald Clahsen, Ingrid Sonnenstuhl, and James P. Blevins. Derivational Morphology in the German Mental Lexicon: A Dual Mechanism Account. In R. Harald Baayen and Robert Schreuder, editors, *Morphological Structure in Language Processing*, pages 125–155. Mouton de Gruyter, Berlin, Germany, 2003.

Peter Clark, Christiane Fellbaum, Jerry R. Hobbs, Phil Harrison, William R. Murray, and John Thompson. Augmenting WordNet for deep understanding of text. In *Semantics in Text Processing. STEP 2008 Conference Proceedings*, Research in Computational Semantics, pages 45–57, Venice, Italy, 2008. College Publications.

Jacob Cohen. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46, 1960.

Allan M. Collins and Elizabeth F. Loftus. A Spreading-activation Theory of Semantic Processing. *Psychological Review*, 82(6):407–428, 1975.

George S. Cree, Ken McRae, and Chris McNorgan. An Attractor Model of Lexical Conceptual Processing: Simulating Semantic Priming. *Cognitive Science*, 23(3):371–414, 1999.

Mathias Creutz and Krista Lagus. Unsupervised Models for Morpheme Segmentation and Morphology Learning. *Transactions on Speech and Language Processing*, 4(1): 3:1–3:34, 2007.

Ido Dagan, Lillian Lee, and Fernando C. N. Pereira. Similarity-Based Models of Word Cooccurrence Probabilities. *Machine Learning*, 34(1–3):43–69, 1999.

Ido Dagan, Oren Glickman, and Bernardo Magnini. The PASCAL Recognising Textual Entailment challenge. In *Proceedings of the 1st PASCAL Challenges Workshop on Recognising Textual Entailment*, pages 177–190, Southampton, UK, 2005.

Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth. Recognizing Textual Entailment: Rational, Evaluation and Approaches. *Journal of Natural Language Engineering*, 15 (4):i–xvii, 2009.

Béatrice Daille, Cécile Fabre, and Pascale Sébillot. Applications of Computational Morphology. In Paul Boucher, editor, *Many Morphologies*, pages 210–234. Cascadilla Press, Somerville, MA, USA, 2002.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society: Series B*, 39(1):1–38, 1977.

Bruce L. Derwing. Morphology and the Mental Lexicon: Psycholinguistic Evidence. In Wolfgang U. Dressler, Hans C. Luschützky, Oskar E. Pfeiffer, and John R. Rennison, editors, *Contemporary Morphology*, number 49 in Trends in Linguistics. Studies and Monographs, pages 249–265. Mouton de Gruyter, Berlin, Germany; New York, NY, USA, 1990.

Magdalena Derwojedowa, Maciej Piasecki, Stanisław Szpakowicz, and Magdalena Zawisławska. Polish WordNet on a shoestring. In *Proceedings of the Biannual Conference of the Society for Computational Linguistics and Language Technology*, pages 169–178, Tübingen, Germany, 2007. Universität Tübingen.

Lee Raymond Dice. Measures of the Amount of Ecologic Association Between Species. *Ecology*, 26(3):297–302, 1945.

Georgiana Dinu and Rui Wang. Inference rules and their application to recognizing textual entailment. In *Proceedings of the 12th Meeting of the European Chapter of the Association for Computational Linguistics*, pages 211–219, Athens, Greece, 2009. Association for Computational Linguistics.

Marc Domenig and Pius ten Hacken. *Word Manager: A System for Morphological Dictionaries*. Olms, Hildesheim, Germany, 1992.

Elke Donalies. *Die Wortbildung des Deutschen: ein Überblick*. Studien zur deutschen Sprache. Narr, Tübingen, Germany, 2005.

Bonnie Dorr, Lisa Pearl, Rebecca Hwa, and Nizar Habash. DUSTer: A method for unraveling cross-language divergences for statistical word-level alignment. In *Proceedings of the 5th Conference of the Association for Machine Translation in the Americas*, pages 31–43, Tiburon, CA, USA, 2002. Springer.

Jean Dubois and Françoise Dubois-Charlier, editors. *Dictionnaire des Verbes Français*. Larousse, Paris, France, 1997. Electronic version.

Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*. Chapman and Hall, New York, NY, USA, 1993.

Kathrin Eichler, Aleksandra Gabryszak, and Günter Neumann. An analysis of textual inference in German customer emails. In *Proceedings of the 3rd Joint Conference on Lexical and Computational Semantics*, pages 69–74, Dublin, Ireland, 2014. Association for Computational Linguistics and Dublin City University.

Katrin Erk. Vector Space Models of Word Meaning and Phrase Meaning: A Survey. *Language and Linguistics Compass*, 6(10):635–653, 2012.

Katrin Erk, Sebastian Padó, and Ulrike Padó. A Flexible, Corpus-driven Model of Regular and Inverse Selectional Preferences. *Computational Linguistics*, 36(4):723–763, 2010.

Stefan Evert. *The Statistics of Word Cooccurrences: Word Pairs and Collocations*. PhD thesis, University of Stuttgart, 2005.

Gertrud Faaß and Kerstin Eckart. SdeWaC – a corpus of parsable sentences from the web. 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 61–68, Darmstadt, Germany, 2013. Springer.

Laurie Beth Feldman. Are Morphological Effects Distinguishable from the Effects of Shared Meaning and Shared Form? *Journal of Experimental Psychology: Learning, Memory and Cognition*, 26(6):1431–1444, 2000.

Christiane Fellbaum, Anne Osherson, and Peter E. Clark. Putting semantics into WordNet's "morphosemantic" links. In *Proceedings of the 3rd Language and Technology Conference*, pages 350–358, Poznań, Poland, 2009. Springer.

Wolfgang Finkler and Günter Neumann. MORPHIX - a fast realization of a classification-based approach to morphology. In Harald Trost, editor, *Proceedings of the 4th Österreichische Artificial-Intelligence-Tagung*, Informatik-Fachberichte, pages 11–19. Springer, 1988.

John R. Firth. A Synopsis of Linguistic Theory 1930-1955. *Studies in Linguistic Analysis*, pages 1–32, 1957.

Arne Fitschen. *Ein computerlinguistisches Lexikon als komplexes System*. PhD thesis, Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart, 2004.

Wolfgang Fleischer and Irmhild Barz. *Wortbildung der deutschen Gegenwartssprache*. Max Niemeyer, Tübingen, Germany, 3rd edition, 2007.

Kenneth I. Forster. Accessing the Mental Lexicon. In Roger J. Wales and Edward Walker, editors, *New Approaches to Language Mechanisms*, pages 257–287. North-Holland Publishing Company, New York, NY, USA, 1976.

Kenneth I. Forster, Kathleen Mohan, and Jo Hector. The Mechanics of Masked Priming. In Sachiko Kinoshita and Stephen J. Lupker, editors, *Masked Priming: The State of the Art*, pages 2–21. Psychology Press, Ltd., 2003.

Ram Frost, Kenneth I. Forster, and Avital Deutsch. What Can We Learn from the Morphology of Hebrew? A Masked-priming Investigation of Morphological Representation. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 23(4):829–856, 1997.

Ram Frost, Avital Deutsch, Orna Gilboa, Michal Tannenbaum, and William D. Marslen-Wilson. Morphological Priming: Dissociation of Phonological, Semantic, and Morphological Factors. *Memory & Cognition*, 28(8):1277–1288, 2000.

Alona Fyshe, Brian Murphy, Partha Talukdar, and Tom Mitchell. Documents and dependencies: an exploration of vector space models for semantic composition. In *Proceedings of the 17th Conference on Natural Language Learning*, pages 84–93, Sofia, Bulgaria, 2013.

Éric Gaussier. Unsupervised learning of derivational morphology from inflectional lexicons. In *Proceedings of the ACL Workshop on Unsupervised Learning in Natural Language Processing*, pages 24–30, College Park, Maryland, USA, 1999. Association for Computational Linguistics.

Alexander Geyken and Thomas Hanneforth. TAGH: A complete morphology for German based on weighted finite state automata. In *Proceedings of the Conference on Finite State Methods and Natural Language Processing, 5th International Workshop*, volume 4002, pages 55–66, Helsinki, Finland, 2005. Springer.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. The third PASCAL Recognising Textual Entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 1–9, Prague, Czech Republic, 2007.

Helmut Glück, editor. *Metzler Lexikon Sprache*. J. B. Metzler, Stuttgart; Weimar, Germany, 4th edition, 2010.

John Goldsmith. Unsupervised Learning of the Morphology of a Natural Language. *Computational Linguistics*, 27(2):153–198, 2001.

Laura M. Gonnerman and Elaine S. Anderson. Graded Semantic and Phonological Similarity Effects in Morphologically Complex Words. In Sabrina Bendjaballah, Wolfgang U. Dressler, Oskar E. Pfeiffer, and Maria D. Voeikova, editors, *Morphology 2000: Selected Papers from the 9th Morphology Meeting*, Amsterdam Studies in the Theory and History of Linguistic Science, pages 137–148. John Benjamins Publishing Company, 2001.

Julio Gonzalo, Felisa Verdejo, Irina Chugur, and Juan M. Cigarrán. Indexing with WordNet synsets can improve text retrieval. In *Proceedings of the COLING-ACL Workshop on Usage of WordNet in Natural Language Processing Systems*, pages 38–44, Montréal, Canada, 1998. Association for Computational Linguistics.

Kira Gor. Beyond the Obvious: Do Second Language Learners Process Inflectional Morphology? *Language Learning*, 60(1):1–20, 2010.

Peter Gordon and Maria Alegre. Rule-based versus Associative Processes in Derivational Morphology. *Brain and Language*, 68(1–2):347–354, 1999.

Rebecca Green, Bonnie J. Dorr, and Philip Resnik. Inducing frame semantic verb classes from WordNet and LDOCE. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 375–382, Barcelona, Spain, 2004.

Joseph Greenberg. Some Universals of Grammar with Particular Reference to the Order of Meaningful Elements. *Universals of Language*, 2:73–113, 1963.

Gregory Grefenstette. SEXTANT: Exploring unexplored contexts for semantic extraction from syntactic analysis. In Henry S. Thompson, editor, *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 324–326, Newark, DE, USA, 1992. Association for Computational Linguistics.

Nizar Habash and Bonnie Dorr. A categorial variation database for English. In *Proceedings of the Joint Human Language Technology Conference and Annual Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 96–102, Edmonton, Canada, 2003.

Christopher J. Hall. Prefixation, Suffixation and Circumfixation. In *Morphology*, volume Part 1 of *Handbooks of Linguistics and Communication Science*, pages 535–545. De Gruyter, 2000.

Harald Hammarström and Lars Borin. Unsupervised Learning of Morphology. *Computational Linguistics*, 37(2):309–350, 2011.

Birgit Hamp and Helmut Feldweg. GermaNet - a lexical-semantic net for German. In *Proceedings of the ACL Workshop on Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, pages 9–15. Association for Computational Linguistics, 1997.

Sanda Harabagiu and Andrew Hickl. Methods for using textual entailment in open-domain question answering. In *Proceedings of the Joint Annual Meeting of the Association for Computational Linguistics and International Conference on Computational Linguistics*, pages 905–912, Sydney, Australia, 2006. Association for Computational Linguistics.

Sanda Harabagiu, Andrew Hickl, and Finley Lacatusu. Satisfying Information Needs with Multi-document Summaries. *Information Processing & Management*, 43(6):1619–1642, 2007.

Mary Hare, Michael Jones, Caroline Thomson, Sarah Kelly, and Ken McRae. Activating Event Knowledge. *Cognition*, 111(2):151–167, 2009.

Trevor A. Harley. *The Psychology of Language: From Data to Theory*. Psychology Press, Ltd., New York, NY, USA, 2008.

Zellig S. Harris. Distributional Structure. *Word*, 10(23):146–162, 1954.

Zellig S. Harris. From Phoneme to Morpheme. *Language*, 31(2):190–222, 1955.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer, New York, NY, USA, 2nd edition, 2009.

Nabil Hathout. Morphonette: A Paradigm-based Morphological Network. *Lingue e linguaggio*, 10(2):245–264, 2011.

Nabil Hathout and Fiammetta Namer. Démonette, a French Derivational Morpho-semantic Network. *Linguistic Issues in Language Technology*, 11(5):125–168, 2014.

Vasileios Hatzivassiloglou and Kathleen R. McKeown. Towards the automatic identification of adjectival scales: Clustering adjectives according to meaning. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 172–182, Columbus, OH, USA, 1993. Association for Computational Linguistics.

Jennifer B. Hay and R. Harald Baayen. Shifting Paradigms: Gradient Structure in Morphology. *Trends in Cognitive Sciences*, 9(7):342–348, 2005.

Marti A. Hearst. Automated discovery of WordNet relations. In Christiane Fellbaum, editor, *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA, USA; London, UK, 1998.

Wolfgang Hoeppner. *Derivative Wortbildung der deutschen Gegenwartssprache und ihre algorithmische Analyse*. Narr, Tübingen, Germany, 1980.

Christina Hoppermann and Erhard Hinrichs. Modeling prefix and particle verbs in GermaNet. In *Proceedings of the 7th Global WordNet Conference*, pages 49–54, Tartu, Estonia, 2014.

Ray Jackendoff. Morphological and Semantic Regularities in the Lexicon. *Language*, 51 (3):639–671, 1975.

Ray Jackendoff. *Foundations of Language: Brain, Meaning, Grammar, Evolution*. Oxford University Press, 2002.

Bernard Jacquemin. A derivational rephrasing experiment for question answering. In *Proceedings of the 7th International Conference on Language Resources and Evaluation*, pages 2380–2387, Valletta, Malta, 2010. European Language Resources Association (ELRA).

Christian Jacquemin. Guessing morphology from terms and corpora. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 156–165, Philadelphia, PA, USA, 1997. ACM.

Michael N. Jones, Walter Kintsch, and Douglas J. K. Mewhort. High-dimensional Semantic Space Accounts of Priming. *Journal of Memory and Language*, 55(4):534–552, 2006.

Lauri Karttunen and Kenneth R. Beesley. *Two-level Rule Compiler*. Xerox Corporation. Palo Alto Research Center, 1992.

Lauri Karttunen and Kenneth R. Beesley. A short history of two-level morphology. Presented at the ESSLLI Special Event "Twenty Years of Finite-State Morphology", 2001.

Lauri Karttunen and Kenneth R. Beesley. Twenty-five Years of Finite-state Morphology. In *Inquiries into Words, Constraints and Contexts. Festschrift for Kimmo Koskenniemi on his 60th Birthday*, pages 71–83. CSLI Publications, Stanford, CA, USA, 2005.

Martin Kay and Gary R. Martins. *The MIND System: The Morphological-analysis Program.* Memorandum RM 6265/2-PR. The RAND Corporation, Santa Monica, CA, USA, 1970.

Steve T. Kempley and John Morton. The Effects of Priming with Regularly and Irregularly Related Words in Auditory Word Recognition. *British Journal of Psychology*, 73(4): 441–445, 1982.

Geoffrey Keppel and Leo J. Postman, editors. *Norms of Word Association.* Academic Press, New York, NY, USA, 1970.

Max Kisselew, Sebastian Padó, Alexis Palmer, and Jan Šnajder. Obtaining a better understanding of distributional models of German derivational morphology. In *Proceedings of the 11th International Conference on Computational Semantics*, pages 58–63, London, UK, 2015. Association for Computational Linguistics.

Anette Klosa, Kathrin Kunkel-Razum, Werner Scholze-Stubenrecht, and Matthias Wermke, editors. *Duden – Deutsches Universalwörterbuch.* Bibliographisches Institut und F.A. Brockhaus AG, Mannheim, Germany, 4th edition, 2001.

Moshe Koppel and Noam Ordan. Translationese and its dialects. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 1318–1326, Portland, OR, USA, 2011. Association for Computational Linguistics.

Kimmo Koskenniemi. *Two-level Morphology: A General Computational Model for Word-Form Recognition and Production.* PhD thesis, University of Helsinki, 1983.

Julia Kreutzer. *Dimensionality Reduction in Semantic Vector Spaces Using a Derivational Resource.* Bachelor's thesis, Heidelberg University, 2014.

Robert Krovetz. Viewing morphology as an inference process. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 191–202, Pittsburgh, PA, USA, 1993. ACM.

Thomas K. Landauer and Susan T. Dumais. A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge. *Psychological Review*, 104(2):211–240, 1997.

J. Richard Landis and Gary G. Koch. The Measurement of Observer Agreement for Categorical Data. *Biometrics*, 33(1):159–174, 1977.

Ronald W. Langacker. *Foundations of Cognitive Grammar: Theoretical Prerequisites*. Number 1 in Foundations of Cognitive Grammar. Stanford University Press, Stanford, CA, USA, 1987.

Gabriella Lapesa and Stefan Evert. Evaluating neighbor rank and distance measures as predictors of semantic priming. In *Proceedings of the 4th Annual Workshop on Cognitive Modeling and Computational Linguistics*, pages 66–74, Sofia, Bulgaria, 2013. Association for Computational Linguistics.

Angeliki Lazaridou, Marco Marelli, Roberto Zamparelli, and Marco Baroni. Compositional-ly derived representations of morphologically complex words in distributional semantics. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1517–1526, Sofia, Bulgaria, 2013. Association for Computational Linguistics.

Omer Levy and Yoav Goldberg. Linguistic regularities in sparse and explicit word representations. In *Proceedings of the 18th Conference on Natural Language Learning*, pages 171–180, Baltimore, MD, USA, 2014.

Gary Libben, Martha Gibson, Yeo Bom Yoon, and Dominiek Sandra. Compound Fracture: The Role of Semantic Transparency and Morphological Headedness. *Brain and Language*, 84(1):50–64, 2003.

Rochelle Lieber. *Morphology and Lexical Semantics*. Cambridge University Press, Cambridge, UK, 2009.

Rochelle Lieber and Pavol Štekauer, editors. *The Oxford Handbook of Derivational Morphology*. Oxford Handbooks in Linguistics. Oxford University Press, New York, NY, USA, 2014.

Rensis Likert. A Technique for the Measurement of Attitudes. *Archives of Psychology*, 22(140):1–55, 1932.

Dekang Lin. Automatic retrieval and clustering of similar words. In *Proceedings of the Joint Annual Meeting of the Association for Computational Linguistics and International Conference on Computational Linguistics*, pages 768–774, Montréal, Canada, 1998a. Association for Computational Linguistics.

Dekang Lin. An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning*, pages 296–304, San Francisco, CA, USA, 1998b. Morgan Kaufmann Publishers Inc.

Dekang Lin and Patrick Pantel. DIRT - discovery of inference rules from text. In *Proceedings of the 7th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 323–328, San Francisco, CA, USA, 2001. ACM.

Catherine-Marie Longtin, Juan Segui, and Pierre A. Hallé. Morphological Priming without Morphological Relationship. *Language and Cognitive Processes*, 18(3):313–334, 2003.

Julie B. Lovins. Development of a Stemming Algorithm. *Mechanical Translation and Computational Linguistics*, 11(1-2):22–31, 1968.

Will Lowe. Towards a theory of semantic space. In Johanna T. Moore and Keith Stenning, editors, *Proceedings of the 23rd Annual Conference of the Cognitive Science Society*, pages 576–581, Edinburgh, UK, 2001.

Will Lowe and Scott McDonald. The direct route: Mediated priming in semantic space. In Lila Gleitman and Aravind Joshi, editors, *Proceedings of the 22nd Annual Conference of the Cognitive Science Society*, pages 675–680, Philadelphia, PA, USA, 2000.

Minh-Thang Luong, Richard Socher, and Christopher D. Manning. Better word representations with recursive neural networks for morphology. In *Proceedings of the 17th Conference on Natural Language Learning*, pages 104–113, Sofia, Bulgaria, 2013.

Catherine Macleod, Ralph Grishman, Adam Meyers, Leslie Barrett, and Ruth Reeves. NOMLEX: A lexicon of nominalizations. In *Proceedings of Euralex98*, pages 187–193, Liège, Belgium, 1998.

Prasenjit Majumder, Mandar Mitra, Swapan K. Parui, Gobinda Kole, Pabitra Mitra, and Kalyankumar Datta. YASS: Yet Another Suffix Stripper. *ACM Transactions on Information Systems*, 25(4):18:1–18:20, 2007.

Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA, 1999.

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, 1993.

Marco Marelli and Marco Baroni. Affixation in Semantic Space: Modeling Morpheme Meanings with Compositional Distributional Semantics. *Psychological Review*, 122(3): 485–515, 2015.

William D. Marslen-Wilson. Functional Parallelism in Spoken Word-Recognition. *Cognition*, 25(1-2):71–102, 1987.

William D. Marslen-Wilson, Lorraine Komisarjevsky Tyler, Rachelle Waksler, and Lianne Older. Morphology and Meaning in the English Mental Lexicon. *Psychological Review*, 101(1):3–33, 1994.

James L. McClelland and David E. Rumelhart. An Interactive Activation Model of Context Effects in Letter Perception: Part 1. An Account of Basic Findings. *Psychological Review*, 88(5):375–407, 1981.

Ryan McDonald, Kevin Lerman, and Fernando Pereira. Multilingual dependency analysis with a two-stage discriminative parser. In *Proceedings of the 10th Conference on Natural Language Learning*, pages 216–220, New York, NY, USA, 2006.

Scott McDonald and Chris Brew. A distributional model of semantic context effects in lexical processing. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 17–24, Barcelona, Spain, 2004. Association for Computational Linguistics.

Scott McDonald and Will Lowe. Modelling functional priming and the associative boost. In Johanna T. Moore and Keith Stenning, editors, *Proceedings of the 20th Annual Conference of the Cognitive Science Society*, pages 675–680, Madison, WI, USA, 1998.

Timothy P. McNamara. *Semantic Priming: Perspectives from Memory and Word Recognition*. Psychology Press, Ltd., New York, NY, USA, 2005.

Wolfang Mentrup. *Zur Pragmatik einer Lexikographie: Von Prinzipien der Sprachforschung zu Prinzipien einsprachiger Lexikographie*. Number 1 in Forschungsberichte. Narr, Tübingen, Germany, 1988.

Marjan Mernik, Jan Heering, and Anthony M. Sloane. When and How to Develop Domain-specific Languages. *ACM Computing Surveys*, 37(4):316–344, 2005.

Fanny Meunier and Catherine-Marie Longtin. Morphological Decomposition and Semantic Integration in Word Processing. *Journal of Memory and Language*, 56(4):457–471, 2007.

David E. Meyer and Roger W. Schvaneveldt. Facilitation in Recognizing Pairs of Words: Evidence of a Dependence between Retrieval Operations. *Journal of Experimental Psychology: General*, 90(2):227–234, 1971.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 27th Annual Conference on Neural Information Processing Systems*, pages 3111–3119, Lake Tahoe, NV, USA, 2013.

Petar Milin, Victor Kuperman, Aleksandar Kostić, and R. Harald Baayen. Paradigms Bit by Bit: An Information Theoretic Approach to the Processing of Paradigmatic Structure in Inflection and Derivation. In James P. Blevins and Juliette Blevins, editors, *Analogy in Grammar: Form and Acquisition*, pages 214–252. Oxford University Press, Oxford, 2009.

George A. Miller and Walter G. Charles. Contextual Correlates of Semantic Similarity. *Language and Cognitive Processes*, 6(1):1–28, 1991.

George A. Miller and Christiane Fellbaum. Morphosemantic Links in WordNet. *Traitement automatique de langue*, 44(2):69–80, 2003.

George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. Introduction to WordNet: An On-line Lexical Database. *International Journal of Lexicography*, 3(4):235–244, 1990.

Jeff Mitchell and Mirella Lapata. Composition in Distributional Models of Semantics. *Cognitive Science*, 34(8):1388–1439, 2010.

Saif Mohammad, Iryna Gurevych, Graeme Hirst, and Torsten Zesch. Cross-lingual distributional profiles of concepts for measuring semantic distance. In *Proceedings of the Joint Conference on Conference on Empirical Methods in Natural Language Processing and Conference on Natural Language Learning*, pages 571–580, Prague, Czech Republic, 2007.

Cristof Monz and Maarten de Rijke. Shallow morphological analysis in monolingual information retrieval for Dutch, German and Italian. In Carol Peters, Martin Braschler, Julio Gonzalo, and Michael Kluck, editors, *Evaluation of Cross-Language Information Retrieval Systems*, volume 2406 of *Lecture Notes in Computer Science*, pages 262–277, Darmstadt, Germany, 2002. Springer.

Taesun Moon, Katrin Erk, and Jason Baldridge. Unsupervised morphological segmentation and clustering with document boundaries. In *Proceedings of the 14th Conference on Empirical Methods in Natural Language Processing*, pages 668–677, Singapore, 2009. Association for Computational Linguistics.

Jane Morris and Graeme Hirst. Non-classical lexical semantic relations. In *Proceedings of the HLT-NAACL Workshop on Computational Lexical Semantics*, pages 46–51, Boston, MA, USA, 2004. Association for Computational Linguistics.

John Morton. The Interaction of Information in Word Recognition. *Psychological Review*, 76(2):165–178, 1969.

William Nagy, Richard C. Anderson, Marlene Schommer, Judith Ann Scott, and Anne C. Stallman. Morphological Families in the Internal Lexicon. *Reading Research Quarterly*, 24(3):262–282, 1989.

Fiammetta Namer. *Morphologie, Lexique et Traitement Automatique des Langues: L'Analyseur DériF*. Hermès Science-Lavoisier, Paris, France, 2009.

Bernd Naumann and Petra M. Vogel. Derivation. In Geert E. Booij, Christian Lehmann, and Joachim Mugdan, editors, *Morphologie: Ein Internationales Handbuch zur Flexion und Wortbildung*, volume 2 of *Handbücher zur Sprach- und Kommunikationswissenschaft*, pages 929–943. Mouton de Gruyer, 2000.

Roberto Navigli and Paola Velardi. An analysis of ontology-based query expansion strategies. In *Workshop on Adaptive Text Extraction and Mining*, Dubrovnik, Croatia, 2003.

Lionel Nicolas, Jacque Farré, and Miguel A. Molinero. Unsupervised learning of concatenative morphology based on frequency-related form occurrence. In *Proceedings of the Morpho Challenge 2010 Workshop*, pages 39–43, Espoo, Finland, 2010.

Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 625–632, Bonn, Germany, 2005. ACM.

Joakim Nivre, Johan Hall, and Jens Nilsson. MaltParser: A data-driven parser-generator for dependency parsing. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, pages 2216–2219, Genoa, Italy, 2006. European Language Resources Association (ELRA).

Richard Nock and Frank Nielsen. On Weighting Clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(8):1223–1235, 2006.

Franz Josef Och and Hermann Ney. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51, 2003.

Sebastian Padó and Ido Dagan. Textual Entailment. In Ruslan Mitkov, editor, *Oxford Handbook of Computational Linguistics*. Oxford University Press, Oxford, UK, 2nd edition, 2016.

Sebastian Padó and Mirella Lapata. Dependency-based Construction of Semantic Space Models. *Computational Linguistics*, 33(2):161–199, 2007.

Sebastian Padó and Jason Utt. A distributional memory for German. In Jeremy Jancsary, editor, *Proceedings of the KONVENS 2012 Workshop on Lexical-semantic Resources and Applications*, pages 462–470, Vienna, Austria, 2012. ÖGAI.

Sebastian Padó, Tae-Gil Noh, Asher Stern, Rui Wang, and Robert Zanoli. Design and Realization of a Modular Architecture for Textual Entailment. *Journal of Natural Language Engineering*, 21(2):1–34, 2013a.

Sebastian Padó, Jan Šnajder, and Britta Zeller. Derivational smoothing for syntactic distributional semantics. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 731–735, Sofia, Bulgaria, 2013b. Association for Computational Linguistics.

Sebastian Padó, Alexis Palmer, Max Kisselew, and Jan Šnajder. Measuring semantic content to assess asymmetry in derivation. In *Proceedings of the IWCS Workshop on Advances in Distributional Semantics*, London, UK, 2015. Association for Computational Linguistics.

Sebastian Padó, Britta Zeller, and Jan Šnajder. Morphological priming in German: The word is not enough (or is it?). In *Proceedings of 1st Conference on Word Knowledge and Word Usage: Representations and Processes in the Mental Lexicon*, pages 42–45, Pisa, Italy, 2015.

Karel Pala. Derivational relations in Slavonic languages. In *Proceedings of the 6th International Conference on Formal Approaches to South Slavic and Balkan Languages*, pages 21–28, Dubrovnik, Croatia, 2008. Croatian Language Technologies Society.

Karel Pala and Dana Hlaváčková. Derivational relations in Czech WordNet. In *Proceedings of the Workshop on Balto-Slavonic Natural Language Processing: Information Extraction and Enabling Technologies*, pages 75–81, Prague, Czech Republic, 2007. Association for Computational Linguistics.

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.

Patrick Pantel and Dekang Lin. Discovering word senses from text. In *Proceedings of the 8th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 613–619, Edmonton, Alberta, Canada, 2002. ACM.

Patrick Pantel, Deepak Ravichandran, and Eduard H. Hovy. Towards terascale knowledge acquisition. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 771–777, Geneva, Switzerland, 2004. Coling 2004 Organizing Committee.

Barbara H. Partee. Lexical Semantics and Compositionality. In Daniel N. Osherson, editor, *An Invitation to Cognitive Science*, volume 1, pages 311–360. MIT Press, 2nd edition, 1995.

Siddharth Patwardhan, Satanjeev Banerjee, and Ted Pedersen. Using measures of semantic relatedness for word sense disambiguation. In Alexander Gelbukh, editor, *Proceedings of the 4th International Conference on Computational Linguistics and Intelligent Text Processing*, volume 2588 of *Lecture Notes in Computer Science*, pages 241–257, Mexico City, Mexico, 2003. Springer.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Yves Peirsman. Word space models of semantic similarity and relatedness. In *Proceedings of the 13th ESSLLI Student Session*, pages 143–152, Hamburg, Germany, 2008.

Yves Peirsman, Kris Heylen, and Dirk Geeraerts. Size matters: Tight and loose context definitions in English word space models. In *Proceedings of the ESSLLI International Workshop on Distributional Lexical Semantics – Bridging the Gap Between Semantic Theory and Computational Simulations*, pages 34–41, Hamburg, Germany, 2008.

Maciej Piasecki and Adam Radziszewski. Morphological Prediction for Polish by a Statistical a Tergo Index. *Systems Science*, 34(4):7–17, 2008.

Maciej Piasecki, Radoslaw Ramocki, and Marek Maziarz. Recognition of Polish derivational relations based on supervised learning scheme. In *Proceedings of the 8th International Conference on Language Resources and Evaluation*, pages 916–922, Istanbul, Turkey, 2012. European Language Resources Association (ELRA).

Frans Plank. *Morphologische (Ir-)Regularitäten: Aspekte der Wortstrukturtheorie*. Studien zur deutschen Grammatik. Narr, Tübingen, Germany, 1981.

John C. Platt. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. In Bernhard Schölkopf Alexander J. Smola, Peter Bartlett and Dale Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, Cambridge, MA, USA, 1999.

Martin Porter. An Algorithm for Suffix Stripping. *Program*, 14(3):130–137, 1980.

Kathleen Rastle, Matthew H. Davis, and Boris New. The Broth in my Brother's Brothel: Morpho-Orthographic Segmentation in Visual Word Recognition. *Psychonomic Bulletin & Review*, 11(6):1090–1098, 2004.

Michal Raveh and Jay G. Rueckl. Equivalent Effects of Inflected and Derived Primes: Long-Term Morphological Priming in Fragment Completion and Lexical Decision. *Journal of Memory and Language*, 42(1):103–119, 2000.

Philip Resnik. Selectional Constraints: An Information-theoretic Model and its Computational Realization. *Cognition*, 61(1-2):127–159, 1996.

Susanne Riehemann. Morphology and the hierarchical lexicon. CSLI Publications, 1994. Manuscript.

Jorma Rissanen. *Stochastic Complexity in Statistical Inquiry Theory*. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 1989.

Herbert Rubenstein and John B. Goodenough. Contextual Correlates of Synonymy. *Communications of ACM*, 8(10):627–633, 1965.

David E. Rumelhart and James L. McClelland. An Interactive Activation Model of Context Effects in Letter Perception: Part 2. The Context Enhancement Effect and Some Tests and Extensions of the Model. *Psychological Review*, 89(1):60–94, 1982.

Benoît Sagot. DeLex, a freely-available, large-scale and linguistically grounded morphological lexicon for German. 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 9th International Conference on Language Resources and Evaluation*, pages 2778–2784, Reykjavik, Iceland, 2014. European Language Resources Association (ELRA).

Magnus Sahlgren. *The Word-Space Model: Using Distributional Analysis to Represent Syntagmatic and Paradigmatic Relations Between Words in High-dimensional Vector Spaces*. PhD thesis, Department of Linguistics, Stockholm University, 2006.

Gerard Salton, A. Wong, and C. S. Yang. A Vector Space Model for Automatic Indexing. *Communications of ACM*, 18(11):613–620, 1975.

Anne Schiller. Deutsche Flexions- und Kompositionsmorphologie mit PC-KIMMO. In Roland Hausser, editor, *Linguistische Verifikation. Dokumentation zur Ersten Morpholympics 1994*, pages 25–35, Tübingen, Germany, 1996. Max Niemeyer.

Anne Schiller, Simone Teufel, Christine Stöckert, and Christine Thielen. Guidelines für das Tagging deutscher Textcorpora mit STTS. Technical report, Institut für maschinelle Sprachverarbeitung, Stuttgart, 1999.

Thea Schippan. *Die Verbalsubstantive der deutschen Sprache der Gegenwart*. Habilitation thesis, Karl-Marx-Universität Leipzig, 1967.

Helmut Schmid. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of the International Conference on New Methods in Natural Language Processing*, pages 44–49, Manchester, UK, 1994.

Helmut Schmid, Arne Fitschen, and Ulrich Heid. SMOR: A German computational morphology covering derivation, composition and inflection. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, pages 1263–1266, Lisbon, Portugal, 2004. European Language Resources Association (ELRA).

Tanja Schmid, Anke Lüdeling, Bettina Säuberlich, Ulrich Heid, and Bernd Möbius. DeKo - Ein System zur Analyse komplexer Wörter. In Henning Lobin, editor, *GLDV-Jahrestagung*, pages 49–57. Gesellschaft für linguistische Datenverarbeitung, 2001.

Patrick Schone and Daniel Jurafsky. Knowledge-free induction of morphology using latent semantic analysis. In *Proceedings of the 4th Conference on Natural Language Learning*, pages 67–72. Lisbon, Portugal, 2000.

Patrick Schone and Daniel Jurafsky. Knowledge-free induction of inflectional morphologies. In *Proceedings of the 2nd Annual Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 1–9, Pittsburgh, PA, USA, 2001. Association for Computational Linguistics.

Robert Schreuder and R. Harald Baayen. How Complex Simplex Words Can Be. *Journal of Memory and Language*, 37(1):118–139, 1997.

Richard E. Schuberth and Peter D. Eimas. Morphological Priming: Dissociation of Phonological, Semantic, and Morphological Factors. *Journal of Experimental Psychology: Human Perception and Performance*, 3(1):27–36, 1977.

Sabine Schulte im Walde. Experiments on the Automatic Induction of German Semantic Verb Classes. *Computational Linguistics*, 32(2):159–194, 2006.

Sabine Schulte im Walde, Stefan Müller, and Stephen Roller. Exploring vector space models to predict the compositionality of German noun-noun compounds. In *Proceedings of the 2nd Joint Conference on Lexical and Computational Semantics*, pages 255–265, Atlanta, GA, USA, 2013. Association for Computational Linguistics.

Hinrich Schütze. Automatic Word Sense Discrimination. *Computational Linguistics*, 24 (1):97–123, 1998.

Hinrich Schütze and Jan Pedersen. A vector model for syntagmatic and paradigmatic relatedness. In *Proceedings of the 9th Annual Conference of the University of Waterloo Centre for the New OED and Text Research*, pages 104–113, Oxford, UK, 1993.

Rico Sennrich and Beat Kunz. Zmorge: A German morphological lexicon extracted from Wiktionary. In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pages 1063–1067, Reykjavik, Iceland, 2014. European Language Resources Association (ELRA).

Dan Shen and Mirella Lapata. Using semantic roles to improve question answering. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Conference on Natural Language Learning*, pages 12–21, Prague, Czech Republic, 2007.

Eyal Shnarch, Jacob Goldberger, and Ido Dagan. A probabilistic modeling framework for lexical entailment. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 558–563, Portland, OR, USA, 2011. Association for Computational Linguistics.

Jeffrey S. Simonoff. *Smoothing Methods in Statistics*. Springer Series in Statistics. Springer, New York, NY, USA, 1996.

Philip T. Smith and Christopher M. Sterling. Factors Affecting the Perceived Morphemic Structure of Written Words. *Journal of Verbal Learning and Verbal Behavior*, 21(6): 704–721, 1982.

Eva Smolka, Pienie Zwitserlood, and Frank Rösler. Stem Access in Regular and Irregular Inflection: Evidence from German Participles. *Journal of Memory and Language*, 57 (3):325–347, 2007.

Eva Smolka, Sarolta Komlosi, and Frank Rösler. When Semantics Means Less than Morphology: The Processing of German Prefixed Verbs. *Language and Cognitive Processes*, 24(3):337–375, 2009.

Eva Smolka, Katrin H. Preller, and Carsten Eulitz. 'Verstehen' ('understand') Primes 'stehen' ('stand'): Morphological Structure Overrides Semantic Compositionality in the Lexical Representation of German Complex Verbs. *Journal of Memory and Language*, 72:16–36, 2014.

Benjamin Snyder and Regina Barzilay. Cross-lingual propagation for morphological analysis. In *Proceedings of the 23rd National Conference on Artificial Intelligence*, pages 848–854, Chicago, IL, USA, 2008. AAAI Press.

Benjamin Snyder and Martha Palmer. The English all-words task. In Rada Mihalcea and Phil Edmonds, editors, *Senseval-3: 3rd International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 41–43, Barcelona, Spain, 2004. Association for Computational Linguistics.

Ingrid Sonnenstuhl, Sonja Eisenbeiss, and Harald Clahsen. Morphological Priming in the German Mental Lexicon. *Cognition*, 72(3):203–236, 1999.

Sylvia Springorum, Sabine Schulte im Walde, and Antje Roßdeutscher. Automatic classification of German "an" particle verbs. In *Proceedings of the 8th International Conference on Language Resources and Evaluation*, pages 73–80, Istanbul, Turkey, 2012. European Language Resources Association (ELRA).

Sylvia Springorum, Jason Utt, and Sabine Schulte im Walde. Regular meaning shifts in German particle verbs: A case study. In *Proceedings of the 10th International Conference on Computational Semantics*, pages 228–239, Potsdam, Germany, 2013. Association for Computational Linguistics.

Richard William Sproat. *Morphology and Computation.* Natural Language Processing. MIT Press, Cambridge, MA; London, UK, 1992.

Angelika Storrer. Funktionen von Nominalisierungsverbgefügen im Text. Eine korpusbasierte Fallstudie. In Kristel Prost and Edeltraud Winkler, editors, *Von der Intentionalität zur Bedeutung konventionalisierter Zeichen. Festschrift für Gisela Harras zum 65. Geburtstag*, pages 147–178. Narr, Tübingen, 2006.

Suriani Sulaiman, Michael Gasser, and Sandra Kübler. Towards a Malay derivational lexicon: Learning affixes using expectation maximization. In *Proceedings of the 2nd Workshop on South Southeast Asian Natural Language Processing*, pages 30–34, Chiang Mai, Thailand, 2011. Asian Federation of Natural Language Processing.

Idan Szpektor and Ido Dagan. Learning entailment rules for unary templates. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 849–856, Manchester, UK, 2008. Coling 2008 Organizing Committee.

Jan Šnajder. DerivBase.hr: A high-coverage derivational morphology resource for Croatian. 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 9th International Conference on Language Resources and Evaluation*, pages 3371–3377, Reykjavik, Iceland, 2014. European Language Resources Association (ELRA).

Jan Šnajder and Bojana Dalbelo Bašić. Higher-order functional representation of Croatian inflectional morphology. In *Proceedings of the 6th International Conference on Formal Approaches to South Slavic and Balkan Languages*, pages 121–130, Dubrovnik, Croatia, 2008. Croatian Language Technologies Society.

Jan Šnajder and Bojana Dalbelo Bašić. String distance-based stemming of the highly inflected Croatian language. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 411–415, Borovets, Bulgaria, 2009. Association for Computational Linguistics.

Jan Šnajder and Bojana Dalbelo Bašić. A computational model of Croatian derivational morphology. In *Proceedings of the 7th International Conference on Formal Approaches to South Slavic and Balkan Languages*, pages 109–118, Dubrovnik, Croatia, 2010.

Jan Šnajder, Bojana Dalbelo Bašić, and Marko Tadić. Automatic Acquisition of Inflectional Lexica for Morphological Normalisation. *Information Processing and Management*, 44(5):1720–1731, 2008.

Jan Šnajder, Sebastian Padó, and Željko Agić. Building and evaluating a Distributional Memory for Croatian. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 784–789, Sofia, Bulgaria, 2013. Association for Computational Linguistics.

Pavol Štekauer and Rochelle Lieber, editors. *Handbook of Word-Formation*, volume 64 of *Studies in Natural Language and Linguistic Theory*. Springer, Dordrecht, Netherlands, 2005.

Marcus Taft and Kenneth I. Forster. Lexical Storage and Retrieval of Prefixed Words. *Journal of Verbal Learning and Verbal Behavior*, 14:638–647, 1975.

Marcus Taft and Paul Kougious. The Processing of Morpheme-like Units in Monomorphemic Words. *Brain and Language*, 90(1):9–16, 2004.

Pius ten Hacken. Word Manager. In Cerstin Mahlow and Michael Piotrowski, editors, *Workshop on Systems and Frameworks for Computational Morphology*, volume 41 of *Communications in Computer and Information Science*, pages 88–107, Zurich, Switzerland, 2009. Springer.

Kapil Thadani and Kathleen McKeown. Towards strict sentence intersection: Decoding and evaluation strategies. In *Proceedings of the ACL Workshop on Monolingual Text-To-Text Generation*, pages 43–53, Portland, Oregon, USA, 2011. Association for Computational Linguistics.

Sonja Tirkkonen-Condit. Translationese – a Myth or an Empirical Fact? *Target. International Journal of Translation Studies*, 14(2):207–220, 2002.

Matthew J. Traxler. *Introduction to Psycholinguistics: Understanding Language Science*. Wiley-Blackwell, 2012.

Harald Trost. Morphology. In Ruslan Mitkov, editor, *Oxford Handbook of Computational Linguistics*, pages 25–47. Oxford University Press, Oxford, UK, 2005.

Peter D. Turney and Patrick Pantel. From Frequency to Meaning: Vector Space Models of Semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188, 2010.

Jason Utt and Sebastian Padó. Crosslingual and Multilingual Construction of Syntax-Based Vector Space Models. *Transactions of the Association for Computational Linguistics*, 2:245–258, 2014.

Martine Vanhove, editor. *From Polysemy to Semantic Change: Towards a Typology of Lexical Semantic Associations*. Number 106 in Studies in Language Companion Series. John Benjamins Publishing Company, Amsterdam, Netherlands, 2008.

Evelyne Viegas, Margarita Gonzalez, and Jeff Longwell. Morpho-semantics and constructive derivational morphology: a transcategorial approach to lexical rules. Technical report, Computing Research Laboratory, New Mexico State University, 1996.

Begoña Villada Moirón and Jörg Tiedemann. Identifying idiomatic expressions using automatic word-alignment. In *Proceedings of the EACL Workshop on Multiword Expressions in a Multilingual Context*, pages 33–40, Trento, Italy, 2006. Association for Computational Linguistics.

Ellen M. Voorhees. Query expansion using lexical-semantic relations. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 61–69, Dublin, Ireland, 1994. ACM.

DeWitt Wallace and Lila Acheson Wallace. *Reader's Digest, das Beste für Deutschland*. Verlag Das Beste, Stuttgart, Germany, 2005.

Géraldine Walther and Lionel Nicolas. Enriching morphological lexica through unsupervised derivational rule acquisition. In *Proceedings of the ESSLLI International Workshop on Lexical Resources*, pages 94–101, Ljubljana, Slovenia, 2011.

Qin Iris Wang, Dale Schuurmans, and Dekang Lin. Strictly lexical dependency parsing. In *Proceedings of the 9th International Workshop on Parsing Technology*, pages 152–159, Vancouver, Canada, 2005. Association for Computational Linguistics.

Rui Wang and Günter Neumann. Recognizing Textual Entailment Using a Subsequence Kernel Method. In *Proceedings of the 22nd National Conference on Artificial Intelligence*, pages 937–943, Vancouver, Canada, 2007. AAAI Press.

Rui Wang and Yi Zhang. Recognizing textual relatedness with predicate-argument structures. In *Proceedings of the 14th Conference on Empirical Methods in Natural Language Processing*, pages 784–792, Singapore, 2009. Association for Computational Linguistics.

Klaus Welke. *Valenzgrammatik des Deutschen: eine Einführung*. De Gruyter Studium. De Gruyter, Berlin, Germany; New York, NY, USA, 2011.

Aris Xanthos, Sabine Laaha, Steven Gillis, Ursula Stephany, Ayhan Aksu-Koç, Anastasia Christofidou, Natalia Gagarina, Gordana Hrzica, F. Nihan Ketrez, Marianne Kilani-Schoch, Katharina Korecky-Kröll, Melita Kovačević, Klaus Laalo, Marijan Palmović, Barbara Pfeiler, Maria D. Voeikova, and Wolfgang U. Dressler. On the Role of Morphological Richness in the Early Development of Noun and Verb Inflection. *First Language*, 31(4):461–479, 2011.

Rui Xu and Donald Wunsch. Survey of Clustering Algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, 2005.

Britta Zeller and Sebastian Padó. A search task dataset for German textual entailment. In *Proceedings of the 10th International Conference on Computational Semantics*, pages 288–299, Potsdam, Germany, 2013. Association for Computational Linguistics.

Britta Zeller, Jan Šnajder, and Sebastian Padó. DErivBase: Inducing and evaluating a derivational morphology resource for German. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1201–1211, Sofia, Bulgaria, 2013. Association for Computational Linguistics.

Britta Zeller, Jan Šnajder, and Sebastian Padó. Towards semantic validation of a derivational lexicon. In *Proceedings the 25th International Conference on Computational Linguistics*, pages 1728–1739, Dublin, Ireland, 2014. Dublin City University and Association for Computational Linguistics.

Torsten Zesch, Iryna Gurevych, and Max Mühlhäuser. Comparing Wikipedia and German Wordnet by evaluating semantic relatedness on multiple datasets. In *Proceedings of the Joint Human Language Technology Conference and Annual Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 205–208, Rochester, NY, USA, 2007. Association for Computational Linguistics.

Andrea Zielinski and Christian Simon. Morphisto - an open source morphological analyzer for German. In *Proceedings of the Conference on Finite State Methods and Natural Language Processing, 7th International Workshop*, pages 224–231, Ispra, Italy, 2008.

Pierre Zweigenbaum and Natalia Grabar. Automatic acquisition of morphological knowledge for medical language processing. In *Proceedings of the Joint European Conference on Artificial Intelligence in Medicine and Medical Decision Making*, volume 1620 of *Lecture Notes in Computer Science*, pages 416–422, Aalborg, Denmark, 1999. Springer.