PCFG-based parsing of German compound words

Bachelor Thesis in
Computational Linguistics

Philipp Gawlik
pgawlik@uni-potsdam.de
Student ID: 745046

University of Potsdam
Linguistics Department
Berlin, April 05^{th} 2017

Supervisors:
Thomas Hanneforth
Jean Vancoppenolle

Contents

Zusammenfassung

Wir präsentieren einen Ansatz zum Parsen deutscher Kompositumnomen basierend auf einer PCFG. Als Grundlage des Trainings der PCFG haben wir das COMPOUNDTREE Korpus erstellt, welches handannotierte (Wort, Baum)-Paare enthält. Darüber hinaus evaluieren wir den Effekt verschiedener Annotationschemata auf die Leistung der PCFG beim Parsen von Kompositumnomen.

Abstract

We present a parsing approach of German compound nouns based on a PCFG. The PCFG is trained on our COMPOUNDTREE corpus that contains a manually annotated set of word-trees pairs. Furthermore, we will evaluate the effect of various annotation schemes on the performance of the PCFG when parsing German compound nouns.

1 Introduction

Compound words are discussed in various fields of natural language processing (NLP). In speech recognition, compound words are emitted by a recogniser as a stream of non-compound units and must be reassembled into compound words. To do so, Spies (1995) and Stolcke and Shriberg (1995) propose probability based language models. Koehn and Knight (2003) also show that syntactic information can be supportive in machine translation of noun phrases.
Motivated by these findings, we will further investigate the parsing of German compound nouns in this work, with the goal of contributing further insights that could be of support in various NLP tasks.
As Steiner and Ruppenhofer (2015) write, one of the bottlenecks for the automatic processing of German language data is the word form productivity. The compound is exemplary for the German word productivity and in contrast to other languages like English where compound words are often separated by spaces or hyphens, compound words in German are written as a single orthographic word. German compound words include the concatenation of words that can require link elements. We will give our linguistic definition of a German compound word in Chapter 3.1.3 but for now we illustrate our notion of a short two word compound with Figure (1.1) and our notion of a long five word compound with Figure (1.2). Furthermore, we will orient our notion of the word arity of a compound word towards the style of the n-gram concept and denote a two word compound as 2-compound, a three word compound as 3-compound and so forth. An n-compound denotes a compound with an arity n whereas $1 \leq n$ and $n \in \mathbb{N}$.

(1) Museum+s+abend
museum+s+evening
"an evening in the museum"

(2) Bus+fahrer+butter+brot+beutel
bus+driver+butter+bread+bag
"a bus driver's bag for buttered bread"

Figure 1: Example of a 2-compound with link element in (1) and a 5-compound in (2).

If we consider the 2-compound noun "Museums+abend" in terms of the hierarchical order of the concatenated two words only one binary tree structure (see Figure 2) is possible where the two words are on the same level. Ambiguity in terms of the hierarchical structure is inherent to compounds with an arity higher than two. For the 3-compound "Butterbrotbeutel" two possible

tree structures exist (see Figure 2) and the 5-compound noun in (1.2) allows up to 14 binary tree structures and gives no affix or link elements that could be of support in the disambiguation task. The increase of the arity of a compound leads to an exponential growth of the number of possible binary tree representations that follows the catalan numbers (see Koshy (2008)). Example (1.2) is pathological in the way that it is not solvable without further semantic knowledge about the concatenated words (see Jones (1983)). The necessary semantic knowledge is usually not available to us in a computer processable form. In what follows we will refer to this problem as **hierarchical ambiguity**.

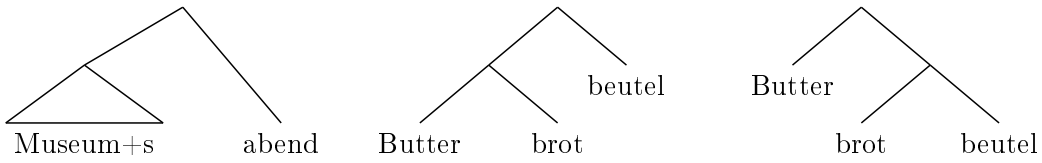

Figure 2: Example of a tree for the 2-compound "Museumsabend" (left) and the two tree variations for the 3-compound "Butterbrotbeutel" (middle, right)

Further complexity is added to the parsing problem of compound words by the morphological structure of the words that are concatenated to build a compound word. We investigated more than 39.000 compounds in the DWDS[1] corpus (more on DWDS in Geyken (2007)) and we found a frequent occurrence of affixation and link elements (see Chapter 3.1 for more information). In addition, we found 5-compounds like (1.2) being rare ($< 1\%$) and 2-compounds (84%) and 3-compounds (14%) being more common. For this work, we sampled and annotated a subset of 4.000 compound nouns from the DWDS corpus without any modification of the proportion of various length compounds to prevent a loss of generality. We generated a set of possible tree structures for each compound noun sample and hand-picked a well-formed tree for each one. The assignment of each compound sample with our hand-picked tree representation of the sample will be called COMPOUNDTREE corpus hereafter.
Subsequently, we created a 4-fold training partition of the COMPOUNDTREE corpus and trained a probabilistic context-free grammar (PCFG) for each of the annotated versions of the partition. We evaluated the performance of the PCFG by parsing and evaluating the respective 4-fold test partition of the COMPOUNDTREE corpus.
We decided to use a PCFG for our approach to choose from the set of possible tree structures for a compound noun while parsing. By adding probabilities to the productions of the grammar, the PCFG allows us to distinguish between rare and frequent morphological constructions. Furthermore, we applied different annotation schemes for the compound trees which are then used to train the PCFG. This allowed us to study non local effects in terms of the rule boundaries on the performance of the PCFG.

In Chapter 2 we will present the work of others that is related to our study of compound nouns. Chapter 3 will clarify basic linguistic concepts that are important for this thesis and describe the COMPOUNDTREE corpus that was created for an empirical investigation of morphological compound noun structures. We will also describe the software that we wrote and third-party software we used. The concept of a PCFG and the annotation schemes that we used will be in-

[1] `https://dwds.de/`

troduced in Chapter 4. Chapter 5 will present the outcome of our empirical study and a related discussion. Chapter 6 will summarise our work and give a prospect about future work.

2 Related work

Our approach is concerned with the disambiguation of flat morphological segmentations of a word by generating a tree structure. The initial segmentation of the German word can be produced by morphological analysers like Morph (Hanrieder (1991) and Hanrieder (1996)), Gertwol (Haapalainen and Majorin (1995)) and Smor (Schmid (2004)). In the present approach we follow Würzner and Hanneforth (2013) and use compound nouns that were segmented with the Tagh system as described in Geyken and Hanneforth (2006).
Our work differs from the work of Würzner and Hanneforth in that they concern themselves with the problem of full morphological parsing while we are only concerned with the parsing part of the approach and exclude the task of morphological segmentation. The grammar in the work of Würzner and Hanneforth, like our grammar, is trained on hand labeled trees but they restrict the corpus to adjectives where we investigate compound nouns. We will compare the results of Würzner and Hanneforth (2013) with our results in Chapter 5.
Schmid (2005) uses a head lexicalised Pcfg for the disambiguation of German word structures. His work differs from our work by using the Smor system (Schmid (2004)) for the morphological segmentation and by the use of the Inside-Outside algorithm following Lari and Young (1990) for training the Pcfg. But an interesting similarity to our work is the use of plain Pcfgs as well as head lexicalised Pcfgs (see Chapter 4.2). Therefore we will consider the results of Schmid (2005) in our evaluation in Chapter 5.
Lauer (1995) investigates English noun compounds by the use of a dependency model with 83% accuracy. Fujisaki et al. (1994) analyse the structures of Japanese compound nouns by using both word collocation statistics and a thesaurus and reach an accuracy of 80%. Besides the language difference, the last two approaches are not suitable for comparison with our approach because of the different models in use. Another interesting approach with some distance to our work is Green et al. (2011) that extends Pcfgs to identify and tag multiword expressions like compound nouns in the french language.
In this study we investigate the effect of different tree representations on the Pcfg training to increase the parsing results of the German compound nouns (see Chapter 4.2). Further work on the variation of Pcfgs tree representations can be found in Johnson (1998) and Schmid (2005) and two of our annotation schemes are inspired by their work.

3 Preliminaries

3.1 Compound words

In this section we will discuss concepts for the description and definition of compounds and give examples of several types of compound words and their underlying structure.
In Chapter 3.1.1 we will introduce the morphological categories *stem*, *rest*, *link*, *konfix* and *affix*. We will introduce the morphological functions we need to define our notion of a compound word in Chapter 3.1.2. And finally, in Chapter 3.1.3 we will give an overview of the types of compound words in the German language and introduce the prototype structure of determinative

compounds as guidance for the discussion of the annotations in the CompoundTree corpus. In the following discussion of the linguistic concepts of compound words we will follow the terminology introduced in Eisenberg (2006) and to illustrate these concepts we will use selected compound words from our CompoundTree corpus as examples hereafter where available.

3.1.1 Foundations

Every word can be partitioned into a sequence of basic grammatical units called **morph**s, e.g. "holz", "gart", "en", "wald" and so forth. For our definition of a compound word we introduce the morph categories *stem*, *affix*, *konfix*, *link* and *rest* and will denote these categories with the generic term **morpheme**.

(3)	(4)	(5)	(6)
Holz	Gart+en	wald+ig	Zahl+ung
wood	garden+rest	wood+affix$_{\text{adjective}}$	pay+affix$_{\text{noun}}$
"wood"	"garden"	"woodsy"	"payment"

Figure 3: Examples of stems and stem groups.

A **stem** (abbr. **St**, also "lexeme", ger. "Stamm") differs from other morphemes like affixes or rest units by possessing a lexical meaning. Some stems form a word on their own as in "Holz" and are independent in that sense. Other stems need to be bound with another morpheme to form a word, e.g. stem+affix as in "Garten".
In contrast to stems, the morpheme **affix** (abbr. **Af**) describes morphs that are not independent and can not form an independent word by concatenation with other affix morphs. Affixes do not posses a lexical meaning but hold grammatical information that they add when concatenated to a stem, for example to cast a noun stem into an adjective stem group (e.g. "waldig") or to cast a verb stem into a noun stem group (e.g. "Zahlung").

(7)	(8)	(9)
Bio+lehrer	Trepp+e	Trepp+chen
bio+teacher	stairs+rest	trepp+affix
"biology teacher"	"stairs"	diminutive of "stairs"

Figure 4: Examples of stems and stem groups.

In between the concepts of a stem and an affix lies the **konfix** (abbr. **Kf**) morpheme. Konfixes used in the German language often originate from Latin as "bio" in "Biolehrer" or from English as "cyber" in "Cyberattacke". Konfixes differ from stems by their lack of independence and they differ from affixes by possessing lexical meaning. As Eisenberg (2006) points out, there are problems of demarcation to other morphemes but we will follow Eisenberg in introducing the category konfix.
Morphs belonging to the **rest** morpheme (abbr. **Rst**) can be concatenated with stems and form an exception because of their neutrality in the concatenation process. A stem concatenated with a rest results in another stem, e.g. the stem morph "trepp" concatenated with the rest morph "e" forms the new stem "Treppe". On one hand, the rest morph "e" adds only phonological information and has no effect on the constituent's category, grammar or semantic. So "Treppe" seems to be a fixed stem. On the other hand, examples like "Treppchen" give evidence for the

stem morph “trepp”. For this reason, morphs like “e” in “Treppe” are necessary and are classified as the rest morpheme.

(10)	(11)	(12)	(13)
Physi(+)o+therapeut	Hilf+s+angebot	Wolle+(-e)+decke	Neu+∅+wahl
physi(+link)+therapist	help+link+offer	wool+link+blanket	new+link+vote
“physiotherapist”	“offer of help”	“wool blanket”	“new election”

Figure 5: Examples of 2-compounds containing a link morpheme.

A **link** (abbr. **Lk**) morpheme is a morphological alternation of a stem which can occur when it is concatenated with another stem. Table 1 gives an overview of link morphs found in the CompoundTree corpus with their frequency and some examples. Foreign link morphs like “o” as in “Physiotherapeut” where missing[2] in the CompoundTree corpus. Some link morphs just occur in the context of an Umlaut as in “Bücherregal”, some link morphs replace phonological elements of the preceding morph as in “Hilfsangebot” (compare *“Hilfeangebot”) and some link morphs even subtract a phonological element as in “Wolldecke” (compare *“Wolledecke”). Eisenberg (2006) also introduces a zero link (ger. "Nullfuge") that we will express with the symbol ∅.

Link Morph	Frequency	Example
s	797	Museum+s+abend, Leben+s+rechner, Weihnacht+s+glitzer
n	414	Sonne+n+hügel, Ochse+n+lenker, Gruppe+n+lesung
en	131	Dorn+en+krone, Terrorist+en+zentrum, Autor+en+alphabet
e	62	Abschieb+e+praxis, Werb+e+slogan, Schweig+e+gelübde
es	47	Meer+es+stille, Bund+es+verfassung, Berg+es+last
er	40	Weib+er+stadt, Kind+er+soldat, Rind+er+dieb
ens	5	Herz+ens+frage, Schmerz+ens+tropfen, Herz+ens+ziel
∅	3117	Schinken+∅+bein, Neu+∅+wahl, Gegen+∅+pol

Table 1: Link morphs occuring in the CompoundTree corpus with the related frequency and examples.

Regarding the morphological structures involving a link morpheme, we associate the link morph with the preceding morpheme. We do so based on the German hyphenation that associates the link with the preceding element as in “Hilfs-angebot”. This results in a constituent structure as displayed in Figure 6. A more detailed discussion of the link category can be found in Eisenberg (2006).

[2]Link morphs as “ial” and “o” do not occur as detached constituent in German (see Eisenberg (2006))

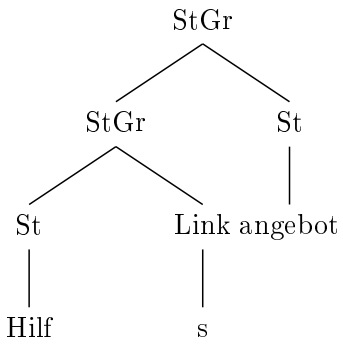

Figure 6: Structure of a compound noun including a link.

3.1.2 Morphological relations

Based on the morphemes we introduces in Chapter 3.1.1 we will go on to discuss the morphological relations that we need to define our notion of a compound word. We will start with adding a structure to the morphemes we introduced so far.
The structure consists of simple **constituents** like the morphemes stem, affix, rest, konfix and link associated with a morph by a unary branch. Binary branches between these simple constituents form complex constituent and we will call these complex constituents **groups** (abbr. **Gr**). Figure 7 gives examples of constituent structures for stems and a stem group by visualising the unary and binary branches between constituents as trees.

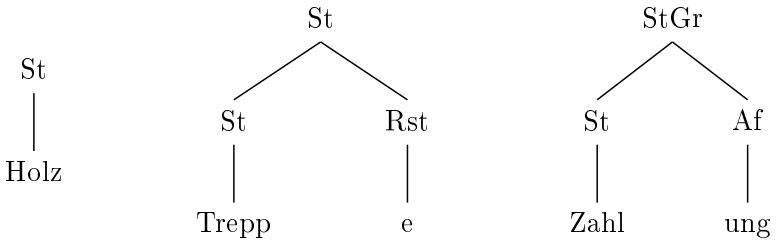

Figure 7: Example of a stem (left), a rest structure (middle) and a stem group (right).

The first morphological relation we introduce is the **head** relation. The head denotes the constituent of the relation that determines the grammatical properties of the group. In the German language the head usually is the right-hand constituent of the relation (so-called Right-Hand Head rule). Therefore "ung {Noun}" is the head of "Zahlung {Noun}" instead of "zahl {Verb}" because the category of the word is defined by "ung {Noun}".
In the unary branch example for "Holz" (see Figure 7) the constituent is its own head. The structure of the example "Treppe" demonstrates the neutrality of the rest morph by leaving the grammatical properties of the stem unchanged. Otherwise, as can be seen in the example structure for "Zahlung", the relation of the head and its complement result in a complex group constituent built out of subordinated morphs.
The second morphological relation necessary for the description of a compound word is concerned with the complement of the head and we will call it **nucleus** following Eisenberg (2006). The nucleus holds the semantic information and undergoes the cast triggered by the head as "zahl" in the example "Zahlung". If a group has no discrete head constituent then head and nucleus fall on the same morph as it is the case with single stem constituents. For a detailed discussion of

the nucleus see Eisenberg (2006). We will not go deeper into this topic here because the concept of the nucleus is not crucial to the description of our analysis.
More important is the **modifier** (abbr. **mod**) relation. It describes a constituent that modifies the nucleus of its direct complement. As a handle for the modifier relation we introduce the notion of determinatum and determinans. The **determinatum** labels the constituent of the relation that includes the head/nucleus and the **determinans** includes the modifier. The determinans modifies the determinatums semantic without changing the grammatical category given by the head. This segmentation is common with German compound nouns. As Eisenberg (2006) states, in German compound word structures the modifier is usually the left-hand constituent in the relation and is marked with the phonological stress.

(14)	(15)	(16)	(17)
bittersüß	Hosen+rock	Nordrhein+Westfalen	Süß+wasser
bitter+sweet	pants+skirt	north+rhine westphalia	sweet+water
"bitter sweet"	"culottes"	"north-rhine westphalia"	"fresh water"

Figure 8: Examples of paratax and hypotax modifications.

Now we can describe the nature of the modification the determinans applies to the determinatum. It can be paratax or hypotax. A **paratax** describes a relation where the determinans and the determinatum are neither superior nor inferior but coordinate as in "bittersüß", "Hosenrock" and "Nordrhein-Westfalen". In contrast to the paratax, a relation is called **hypotax** if the determinans is subordinated to the determinatum as in "Süßwasser" where the determinatum "wasser" is further determined by the determinans "süß".

(18)	(19)	(20)
Marzipan+ferkel	Innen+ohr	un+schön
marzipan+piglet	inner+ear	un+pleasant
"sweet of marzipan in the shape of a piglet"	"inner ear"	"unpleasant"

Figure 9: Examples of exocentric and endocentric modifications.

However, the modifier relation can not be classified solely as the concepts hypotax and paratax. "Marzipanferkel" for example seems to be hypotax but does not determine a specific type of pig but a sweet. To capture the modification of "Marzipan" to "ferkel" we need the distinction of exocentric and endocentric. "Marzipanferkel" is **exocentric** because it is not included in the base concept of pigs, but points to something beyond this concept. In comparison, the "Innenohr" is an **endocentric** modification because it specifies the concept of the ear by referring to certain a part of it.
Regarding the concepts of hypotax and paratax, we leave it open to discussion if "unschön" is paratax or hypotax, as "un" modifies "schön" by negating it.
More examples of the morphological relations we introduced so far can be found in Chapter 3.1.3.

3.1.3 Composition

Apart from derivation composition is one of the main methods for building new words. Composition connects words or independent stems to form new words and differs from the method of

derivation which is concerned with affixation. We will call a word that is formed by composition a **compound word** and follow Eisenberg (2006) in defining it as given in definition 1.

Definition 1 (Compound Word) *A compound word is a stem group that includes at least two subordinated stems.*

In the following, we will briefly introduce the different types of compound words. We will also show restrictions that arise from their semantic properties. We then discuss the implications of these restrictions on the morphological structure of compound words and the problem of hierarchical ambiguity.
The compound word in German can be classified by the semantic properties of the modifier relation we introduced in Chapter 3.1.2. We distinguish the types determinative, possessive, copulative, and governing compound word where governing and possessive compound words are a subset of the determinative compound word.
The **copula compound word** is a stem group in terms of Definition 1 where the determinans and determinatum are in paratax relation (see examples in Figure 10). If the composition is not lexicalised then the determinans and determinatum are exchangeable.

(21)	(22)
Nord+ost	Schürzen+kleid
north+east	apron+dress
"northeast"	"apron dress"

Figure 10: Examples of copula compound words.

A **determinative compound word** is a hypotax composition in terms of the modifier relation between the determinans and determinatum. The determinans further specifies the determinatum in the determinative compound word. Besides the determinative compound noun there also exists the determinative compound adjective, verb and adverb (for a full discussion see Fleischer and Barz (1992) and Höhle (1982)). For our discussion of the morphological structure of compound nouns we will follow Eisenberg (2006) in regarding the determinative compound noun as prototypical structure.
The simplest example of a determinative compound noun involves a noun, adjective, verb or preposition as the determinatum and a noun as determinans. Some examples taken from the CompoundTree corpus can be found in Figure 11.

(23)	(24)	(25)	(26)
Schaum+kanone	Weich+spüler	Schlaf+lied	Gegen+pol
foam+cannon	soft+scourer	sleep+song	against+pole
"foam cannon"	"fabric softener"	"lullaby"	"anti pole"

Figure 11: Determinative compound noun examples left to right: noun+noun, adjective+noun, verb+noun and preposition+noun.

Exemplary structures of determinative compound nouns can be found in Figure 12. Note that these examples are all endocentric compound nouns.
By giving examples for exocentric compound nouns we can introduce the notion of possessive and governing compound nouns.

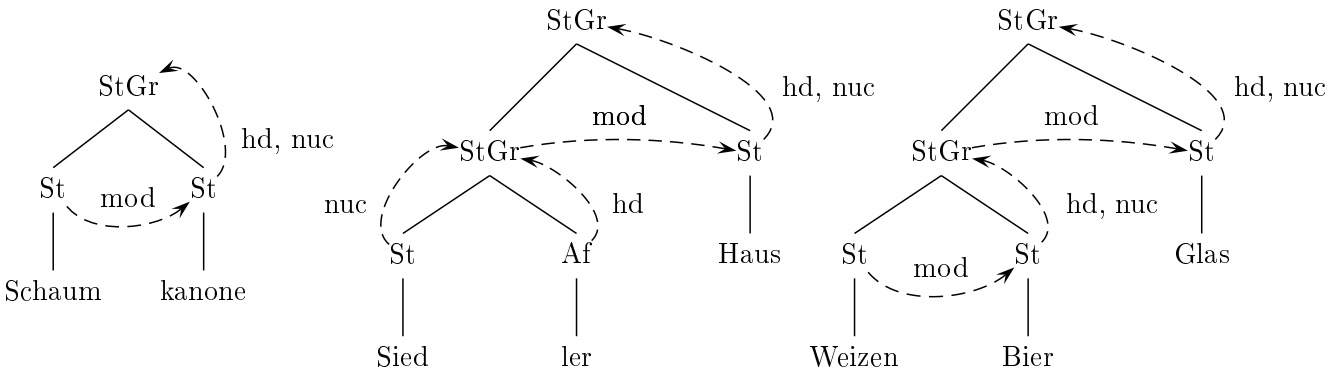

Figure 12: Example structure of determinative compound nouns with annotated morphological relations as dashed lines.

Possessive compound nouns describe a property situation and follow the *pars pro toto* principle ($\approx$ *A part represents the whole*). That means, that the possessive compound word contains a part (e.g. a body part, see in Figure 13) as determinatum and as a whole describes the entity that possesses the part. In the common German saying "Du bist ja ein Hasenfuß!" a "Hasenfuß" is not a specific kind of foot but identifies the owner of the foot as a coward. The quality of cowardliness lies beyond the subset of feet and therefore the underlying modifier relation is exocentric. The same holds for "Milchgesicht" and "Dickwanst" in Figure 13.

(27)	(28)	(29)
Hasen+fuß	Milch+gesicht	Dick+wanst
rabbit+foot	milk+face	thick+belly
"coward"	"milksop"	"tubby"

Figure 13: Examples of possessive exocentric compound nouns.

With **governing compound nouns** too exocentric examples can be found - for instance with determinative compound nouns that take an accusative preposition as determinans as in Figure 14. Clearly a "Untertasse" is not a specific "Tasse" but a plate and "Übersee" is not a specific kind of sea but the land beyond the sea. Therefore, the given examples are exocentric. Regimen compound words are named following the case constraint that is put on the composition. E.g. the compound "Geldwäscherfahndung" involves the verb "fahnden" that possesses a dative object argument that is inherited by the nominalisation with "ung". The inherited dative object argument of the verb is satisfied by the composition with "Geldwäscher". See Eisenberg (2006) for further discussion.

(30)	(31)	(32)
Unter+tasse	Über+see	Geldwäscher+fahndung
below+cup	over+seas	money launderer+manhunt
"saucer"	"overseas"	"manhunt for money launderers"

Figure 14: Examples of two exocentric governing compound nouns (left, middle) and a governing compound noun (right).

Further types of compound nouns exist in the CompoundTree corpus like the **number compound** and the **name compound** (see Figure (15)). We will not discuss these other types of compounds in detail here, but rather discuss the semantic aspect of the problem of hierarchical

ambiguity that we mentioned in Chapter 1.

(33) Vier+takt+diesel+motor
four+stroke+diesel+engine
"diesel powered four-stroke engine"

(34) Büchner+preis+träger
büchner+award+honoree
"honoree of the Büncher award"

Figure 15: Examples of compounds involving a number (left) and a name (right).

The application of Definition 1 can form left-branching, right-branching or mixed left- and right-branching trees as in Figure 16, 17 and 18. But not every binary sub-tree of two adjacent stems results in a meaningful compound word even though a well-formed compound word can contain the two adjacent stems. The StGr "Bezirksjahreshauptversammlung" contains the stems "Bezirk", "Jahr", "Haupt" and "Versammlung". The StGr "Hauptversammlung" and "Jahreshauptversammlung" exist but "Bezirksjahres" and "Jahreshaupt" do not (see ill-formed structure examples in Figure 18). Therefore, the compound word's morphological structure has to account for a well-formed hierarchical structure of the words. Knowledge about the semantic restrictions can be helpful for ruling out ill-formed tree structures. As mentioned before, the knowledge about the meaningful hierarchical combinability is not available to us in a computer processable form.

(35) Berg+bau+wissenschafts+studium
mountain+mining+science+studies
"Mining Studies"

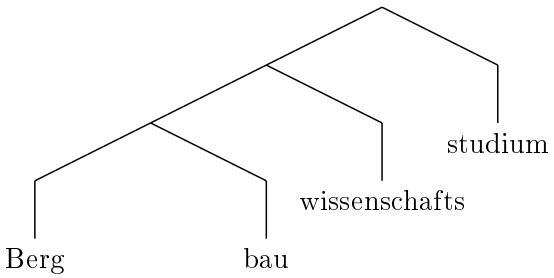

Figure 16: Example of a left-branching determinative compound noun structure (example taken from Eisenberg (2006)).

(36) Bezirks+jahres+haupt+versammlung
district+year+main+meeting
"district annual main meeting"

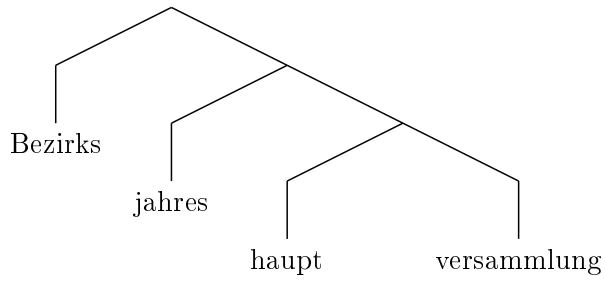

Figure 17: Example of a right-branching determinative compound noun structure (example taken from Eisenberg (2006)).

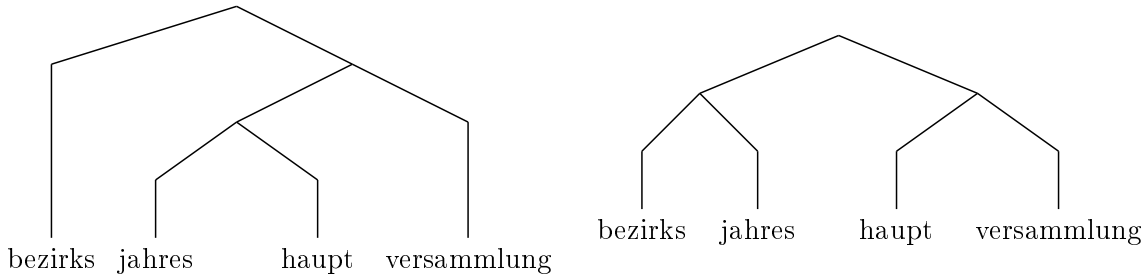

Figure 18: Examples of ill-formed determinative compound noun tree structures.

3.2 CompoundTree corpus

The CompoundTree corpus consists of an assignment of 4.000 German compound nouns - 3.632 2-compounds, 354 3-compounds, 13 4-compounds and 1 5-compound - with their respective binary parse trees. The 4.000 compound nouns where sampled from the Dwds[3] corpus (see also Geyken (2007)) without any modification to the proportion of the various length compounds to prevent a loss of generality and a disproportional share of parsable or pathological samples. The segmentation and tokenisation of the compound nouns was carried out by Thomas Hanneforth[4] using the Tagh morphology (Geyken and Hanneforth (2006)), a comprehensive computational morphology system for the German language based on weighted finite-state transducers. Figure 19 shows a tokenised 3-compound where the symbol in angle brackets denotes the word category, # marks a morpheme boundary and $\sim$ marks the start of a suffix.

$Arbeit\langle N\rangle\backslash s\langle l\rangle\#Zeit\langle N\rangle\#find\langle V\rangle \sim ung$
work+link+time+find+affix$_{\text{noun}}$
"to find time to work"

Figure 19: Tokenisation of the word "Arbeitszeitfindung".

We annotated the coresponing tree for each tokenised compound noun by generating a permutation of trees and subsequently hand-picking a correct tree for every compound noun by visualising it with the Dot-language (see Chapter 3.3.2). The generated trees are already binarised and free of ϵ-productions.
The tree formalism in use follows the one used by Würzner and Hanneforth (2013) to ensure the compatibility with Würzner and Hanneforth's corpus for possible future work. The approach presented in this work is solely based on the CompoundTree corpus material.
The helper scripts for the preparation of the CompoundTree corpus as well as the corpus itself can be accessed through git[5]. The next chapter will describe the technical aspects of the processing pipeline.

3.3 Software

In this chapter we will present an overview of some of the helper scripts we wrote and the third-party software we used to process and study compound words in this approach. For vividness

[3] `https://dwds.de/`
[4] University of Potsdam Department of Computational Linguistics
[5] `https://github.com/PhilippGawlik/compoundtree`

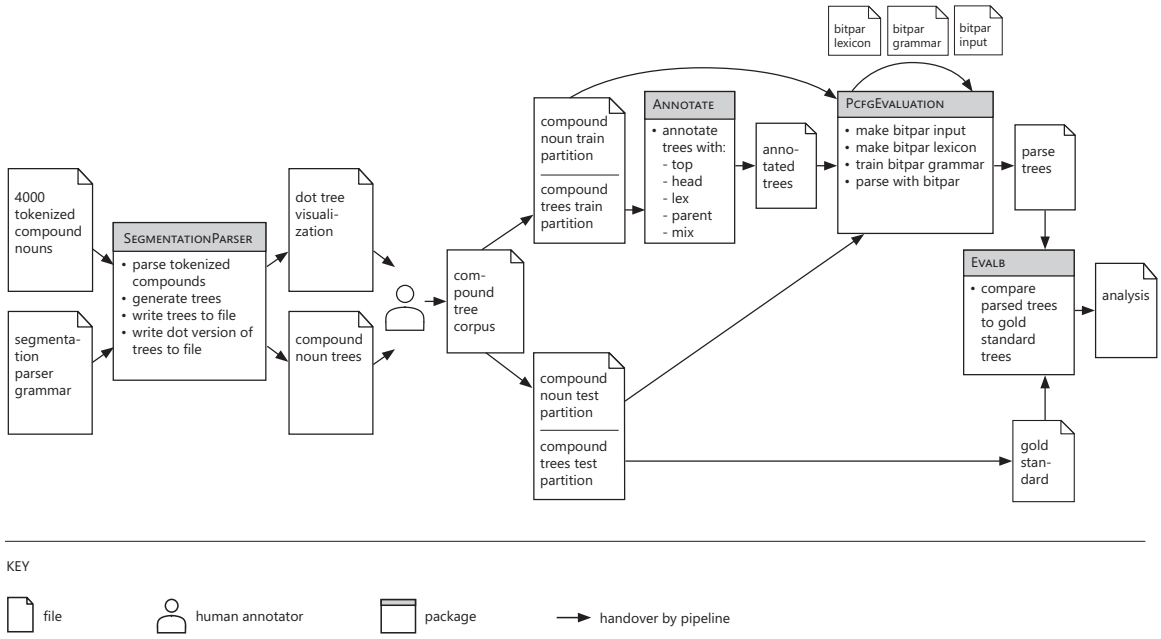

Figure 20: Simplified process diagram to give an overview about the CompoundTree pipeline.

we present a diagram overview of the simplified process pipeline in Figure 20 and we will briefly go through of the diagram from left to right. The next sections will look at each component in more detail.

The pipeline starts with a set of 4.000 tokenised compound words (see Chapter 3.2) and is parsed by the SegmentationParser package that generates a set of trees for every compound noun as described in Chapter 3.3.1.

In the next step of the processing pipeline, a human is required to hand-pick one of the generated trees of every compound word with the support of Dot visualisations to avoid ill-formed structured trees as described in 3.1.3 and 3.3.2. More information about the third-party software Dot is given in Chapter 3.3.2.

From the resulting corpus of compound-tree alignments a test corpus of (compound,tree)-pairs is split. Its compound words are used as evaluation input for the Bitpar parser and its trees are used as gold standard for the Evalb analysis (see Chapter 5) of the Bitpar tree parses.

Subsequently, the trees in the training partition of the CompoundTree corpus are annotated by the Annotate module as described in Chapter 4.2. Next, the PcfgEvaluation package constructs a lexicon and a Pcfg for the Bitpar parser (see 3.3.3) from the training partition. The compound words and trees for testing are not included in the training of the Pcfg but in the generation of the lexicon to prevent unknown words during Bitpar parsing.

Next, the compound nouns in the test partition are parsed by the Bitpar parser based on the generated Pcfg. The resulting tree parses are then analysed against the gold standard by Evalb.

3.3.1 Segmentation parser

The SegmentationParser parses tokenised compound words as described in Figure 19 and generates binary constituent-tree structures as in Figure 25. This is done in three steps.

In step one each morph is matched with its category constituent and cast to unary terminal trees

as shown in Figure 21.

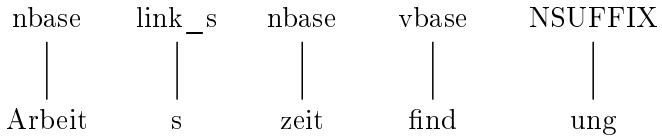

Figure 21: Unary terminal trees for word "Arbeitszeitfindung".

In the second step, the trees for the compound's subordinated words are generated as in Figure 22.

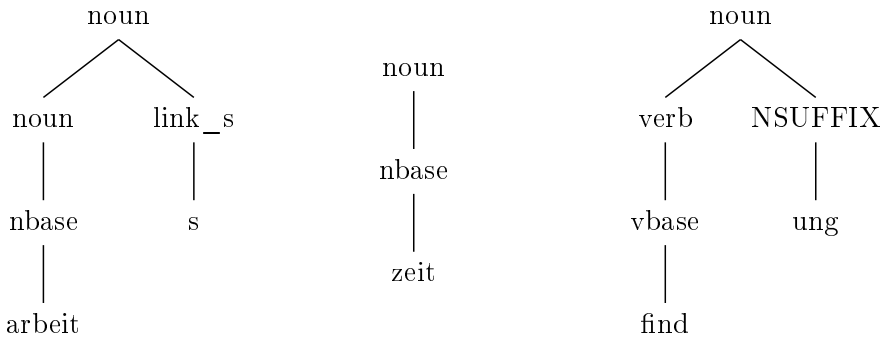

Figure 22: Sub-trees for the word "Arbeitszeitfindung".

In step three the possible binary tree variations are generated by merging the binary sub-trees following the algorithm in Figure 24. The result of step three can be found in Figure 23.

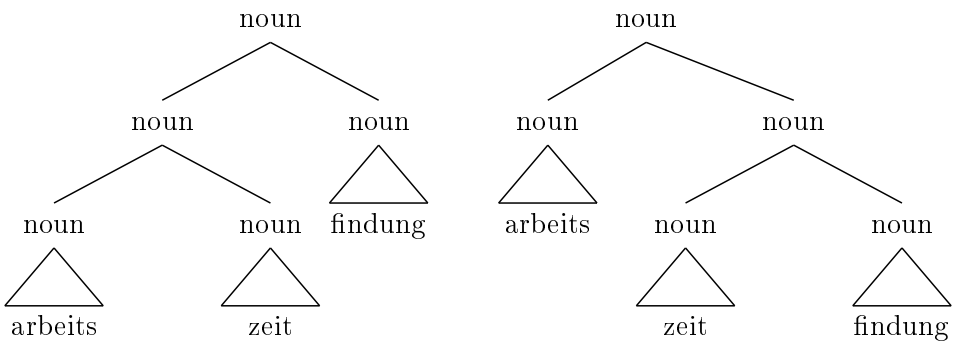

Figure 23: Set of possible trees for the word "Arbeitszeitfindung" as generated by the algorithm in Figure 24.

The algorithm in Figure 24 starts working with an InitialSubtreeList containing sub-trees as given in Figure 22. The InitialSubtreeList is put into the BufferList and the algorithm stops when the BufferList is empty. Until then, each element of the BufferList represents a tree (see line (4)). If the tree consists of just one sub-tree then the tree is complete and saved (see step (6) and (7)). Otherwise, the tree is copied $n-1$ times where n is the number of sub-trees in the tree. Then, an index tuple $(i, i+1)$ with $0 \leq i < n$ is iterated over the copies of the tree (see step (11)-(17)) and merges a distinct pair of sub-trees in every copy. Each copy is then appended to the BufferList. This procedure generates all binary tree variations of the compound noun's

```
SubTreePermutation(InitialSubtreeList)
1:   BufferList := [InitialSubtreeList];
2:   FinishedTrees := [];
3:   while BufferList;
4:       Tree := BufferList.pop(0);
5:       Length := Tree.length();
6:       if Length == 1:
7:           FinishedTrees.append(Tree);
8:       fi;
9:       else:
10:          CopyTrees := [make Length-1 copies of Tree];
11:          Index := 0;
12:          while CopyTrees;
13:              Tree := CopyTrees.pop(0);
14:              Tree.merge(Index, Index+1);
15:              BufferList += Tree;
16:              Index += 1;
17:          elihw;
18:      esle;
19:  elihw;
20:  return FinishedTrees;
```

Figure 24: Algorithm for generating all binary tree structures.

subordinated words and collects those variations in the list FinishedTrees.

3.3.2 Dot-language and dot command line tool

By parsing the tokenised compound nouns, we generated a set of possible trees for every word. Subsequently, we had to hand-pick the generated parse trees that comply with the constraints described in Chapter 3.1. We simplified this task by translating the internal tree representation to the DOT-LANGUAGE[6] and by using the command line tool DOT to generate tree visualisations that we could browse to pick the correct tree.

The DOT-LANGUAGE is a plain-text graph description language. The DOT command line tool is included in the GRAPHVIZ[7] package that contains open source graph visualisation software initiated by AT&T.

Figure 25 gives two visualised trees for the compound in Figure 26. For a native speaker of the German language it is easy to see that the right-hand tree in Figure 25 is ill-formed because the word "Lieblingsfernseh" does not exist in the German language. The tree on the left-hand side is the correct one.

[6] `http://www.graphviz.org/doc/info/lang.html`

[7] `www.graphviz.org`

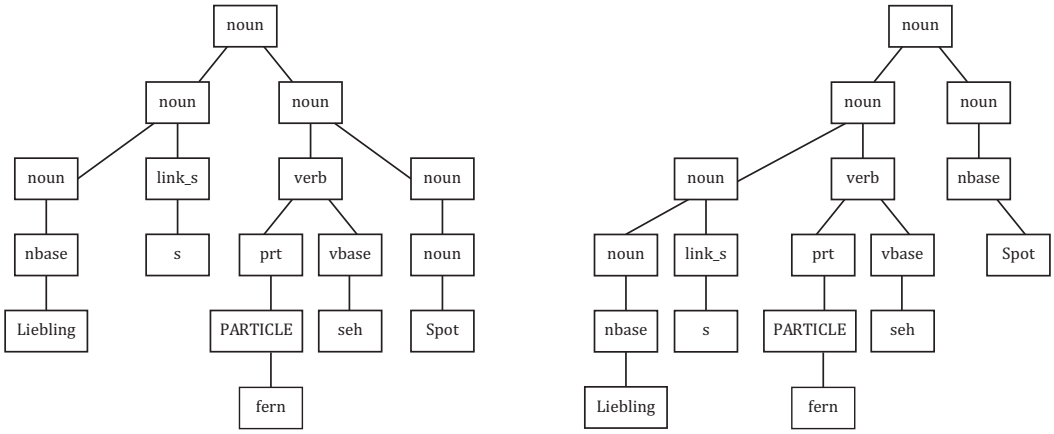

Figure 25: Example tree visualisations generated by using the DOT-LANGUAGE and the DOT command line tool.

(37) Liebling+s+fern+seh+spot
favourite+link+far+see+spot
"favourite television spot"

Figure 26: Decomposition of the word "Lieblingsfernsehspot".

3.3.3 Bitpar

BITPAR[8] (Schmid (2004)) is a bit-vector-based implementation of the CYK algorithm (see Aho and Ullman (1972) on the algorithm and Chapter 4.3 on parsing) for large, highly ambiguous grammars. BITPAR follows the CYK parsing routine in the first processing step but only recognises the set of constituents to which the input string can be reduced and writes it into a chart. In the second step the chart is filtered in order to eliminate constituents which are not part of the complete analysis of the input string. Subsequently, the Viterbi probabilities are calculated and finally the best parse is identified and returned.

3.3.4 Evalb

We use EVALB[9] (Sekine and Collins (1997)) to evaluate the various PCFGs that we trained from the COMPOUNDTREE corpus (see Chapter 3.2). EVALB is a bracketing scoring metric that we use in particular to compare the output trees of the PCFG parsing process to our gold standard trees. From the evaluation terms provided by EVALB we consider the following in Chapter 5:

1. *Bracket Precision*: number of correct constituents divided by number of constituents in parsed file
2. *Bracket Recall*: number of correct constituents divided by number of constituents in gold standard
3. *Bracket Fmeasure*: F-score calculated from bracketing precision and bracketing recall

[8] `http://www.cis.uni-muenchen.de/~schmid/tools/BitPar/`
[9] `http://nlp.cs.nyu.edu/evalb/`

4. *Complete Match*: percentage of sentences where recall and precision are both 100

5. *Tag Accuracy*: percentage of correct POS tags

4 PCFG

In chapter 4.1 we will introduce PCFGs as an extension of the context-free grammar formalism and explain how we used it for modeling and disambiguation of the morphological tree structures. In chapter 4.2 we will motivate the investigation of different tree representations in the process of training and introduce the annotation types we used. Following this, we will give a brief introduction of the usage of context-free grammars in parsing in chapter 4.3.

4.1 Foundations

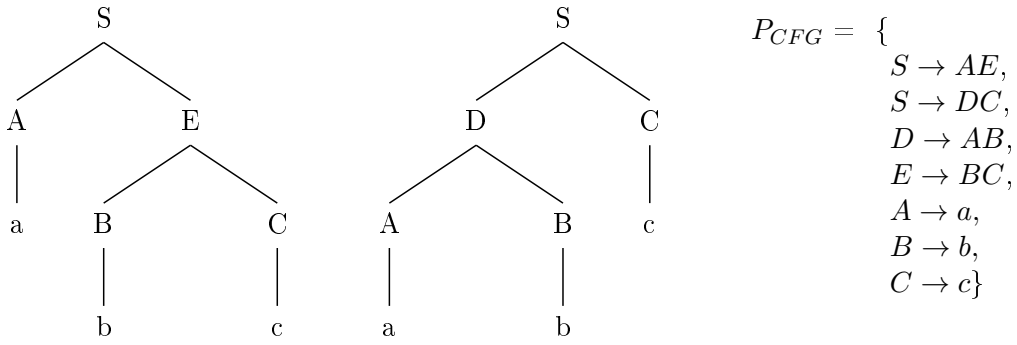

Figure 27: Examples of two tree derivations of the string "abc" and the set of productions P of the underlying grammar $G_{PCFG} = (N, \Sigma, P, S, D)$.

Following Jurafsky and Martin (2000), a probabilistic context-free grammar is a 5-tuple as in Definition 2 and serves the purpose of deriving all the strings (compound nouns in our application) of a target language.

Definition 2 *Probabilitic context-free grammar*

$G_{PCFG} =$ (N, Σ, P, S, D) *where*

N *is a set of non-terminal symbols*

Σ *is a set of terminal symbols (disjoint from* N*)*

P *is a set of productions, each of the form* $A \to \alpha$

where A *is a non-terminal and* α *is a string of symbols*

from the infinite set of strings $(\Sigma \cup N)^*$

S *is the start symbol*

D *is a function assigning a probability to each production in* P

The productions used for the derivation of a specific string α can be represented as a parse tree (see Bondy and Murty (2008) for more information about trees and Figure 27 for an illustration

of a set of productions and related parse trees). For a well-formed derivation of a PCFG at least one (tree, string)-pair can be given. Figure 27 gives a second parse tree for the example string "abc" to illustrate the problem of ambiguous grammars.
To resolve the problem of ambiguous grammars, a probability p is assigned to each production $r \in P$ and $P \in\ G_{PCFG}$ as in (1). The probability p of a production expresses the preference of expanding the production's left hand side A into the right hand side α.

$$A \rightarrow \alpha\ [p] \tag{1}$$

One way to calculate the probability p of a production $r \in P$ is to use a corpus of already parsed (string, tree)-pairs. We do so in the present attempt by using the CompoundTree corpus and (2) (compare Jurafsky and Martin (2000)) where γ represents an arbitrary expansion of A in the grammar G_{PCFG} and the function $C()$ counts the number of occurrences of the given argument. $C(A \rightarrow \alpha)$ counts the frequency of the specific production $A \rightarrow \alpha$ and $\sum_\gamma C(A \rightarrow \gamma)$ sums up the frequencies of all productions with left hand side non-terminal A.

$$p(A \rightarrow \alpha \mid A) = \frac{C(A \rightarrow \alpha)}{\sum_\gamma C(A \rightarrow \gamma)} = \frac{C(A \rightarrow \alpha)}{C(A)} \tag{2}$$

Another way to calculate the production's probabilities is to use a CFG in combination with the Inside-Outside algorithm. We will not follow this attempt here but for more information on this see Baker (1979) and Manning and Schütze (1999).
By utilising the production probabilities, we can now calculate the probability of the (tree-string)-pair (T, S) with the tree T and the string S. To do so, we multiply the probabilities p of all productions involved in a specific tree representation of the string derivation as in (3).

$$p(T, S) = \prod_{n \in T} p(r(n)) \tag{3}$$

Here $p()$ gives the probability of the production that is obtained by using $r()$ on a tree's node n. As Jurafsky and Martin (2000) point out, $p(T, S)$ is both the joint probability of the parse and the sentence, and also the probability of the parse $p(T)$. This follows from (4) of the joint probability (can be obtained by reformulating Bayes rule). The parse tree includes productions ending in S's sub-strings and therefore also includes S. So the probability of the string S given the tree T is always $p(S \mid T) = 1$ and can therefore be ignored as in (5) which results in the parse tree probability given in (6).

$$\begin{aligned} p(T, S) &= p(T)p(S \mid T) && (4)\\ &= p(T) \cdot 1 && (5)\\ &= p(T) && (6) \end{aligned}$$

The disambiguation of derivations of a string S based on an ambiguous PCFG can then be done by picking the parse tree $\hat{T}(S)$ of the set of S's disjoint parse trees $\tau(S)$ that has the highest probability as given in (7). Following the definition of Bayes rule we replace $p(T \mid S)$ as in (8) and since we are maximising over all parse trees for the same sentence, $p(S)$ will be constant for each tree and can be eliminated as in (9). Finally, we can integrate step (4) to (6) to give us

(10).

$$\hat{T}(S) = \underset{T \in \tau(S)}{argmax}\ p(T \mid S) \tag{7}$$

$$\hat{T}(S) = \underset{T \in \tau(S)}{argmax}\ \frac{p(T,S)}{p(S)} \tag{8}$$

$$\hat{T}(S) = \underset{T \in \tau(S)}{argmax}\ p(T,S) \tag{9}$$

$$\hat{T}(S) = \underset{T \in \tau(S)}{argmax}\ p(T) \tag{10}$$

(10) establishes the parsing problem that we will discuss further in Chapter 4.3. Before that, we will introduce the different annotation schemes used for the parsing task.

4.2 Annotation

> The kinds of tree representations [...] can have a dramatic effect on performance of a parser based on the PCFG estimated from a corpus[...]
>
> Johnson (1998)

In our work we investigated the effect of different tree representations on the PCFG training. To do so, we varied the tree format in the COMPOUNDTREE corpus from only mandatory tree annotations to experimental annotations that serve the purpose of increasing the parsing performance for German compound nouns.
As mentioned in chapter 3.2 the trees of the COMPOUNDTREE corpus are already binarised and free of ϵ-productions and we call this tree annotation scheme our **gold standard**. This is the base line for the scheme comparison in our evaluation in Chapter 5.
However, the definition of a PCFG (see Definition 2) requires further mandatory annotations which are described in chapter 4.2.2.
Furthermore, we performed empirical experiments with node relabeling of the tree annotation schemes motivated by the remark in Johnson (1998) that precedes this chapter. Some thoughts about independence assumptions that are implicit in the annotations schemes are given in 4.2.1. The chapters 4.2.3 to 4.2.5 introduce annotation schemes that are used individually and in combination for the empirical experiments.
All of the mentioned annotations will be removed from the predicted morphological structures before the evaluation with EVALB (see Chapter 3.3.4) to restore the compatibility with the gold standard trees.

4.2.1 Bias vs. variance

The trees in the COMPOUNDTREE corpus embody an **independence assumption**. This is concerned with the sensitivity to non-local relationships between nodes in a tree and can be illustrated by comparing a flat tree to a gold standard tree as in Figure 28. The flat tree contains one production $S \rightarrow speise\ eis\ forsch\ er$ and its probability can be directly calculated based on the occurrence frequency of the string "Speiseeisforscher" in the corpus. But in the deep structure

of the tree as in the gold standard tree in Figure (28), the productions become independent of the underlying string and to each other and can be calculated across multiple trees in the corpus, e.g. production *noun* → *nbase noun*. One could assume that the fewer non-terminal nodes are contained in the tree the weaker the independence assumption gets.
However, with our annotation approach we plan to weaken the implicit independence assumption of the trees in the CompoundTree corpus without reducing the number of nodes in the gold standard trees. We will encode more information in each node's label to increase the productions sensitivity to non-local relationships between nodes, e.g. add the parent or terminal node's label. In doing so, the morphological structure of the modeled trees stays the same except for the relabeling of the nodes.
This annotation method allows us to investigate the effect of **bias vs. variance**. A weakened independence assumption increases the number of productions in our Pcfg. For a big corpus with a wide distribution of examples this could cause a increase in accuracy because of an inclusion of specific morphological structures in the grammar (≈ reduced bias). But for a narrow distribution of examples the weakened independence assumption could cause a decrease in accuracy because of a number of unnecessary productions that have to be estimated (≈ increased variance). More on the topic of bias vs. variance can be found in Johnson (1998) and Geman et al. (1992).
The outcome of our empirical study of fitting a magnitude for our independence assumption within the CompoundTree corpus can is presented in chapter 5.1.

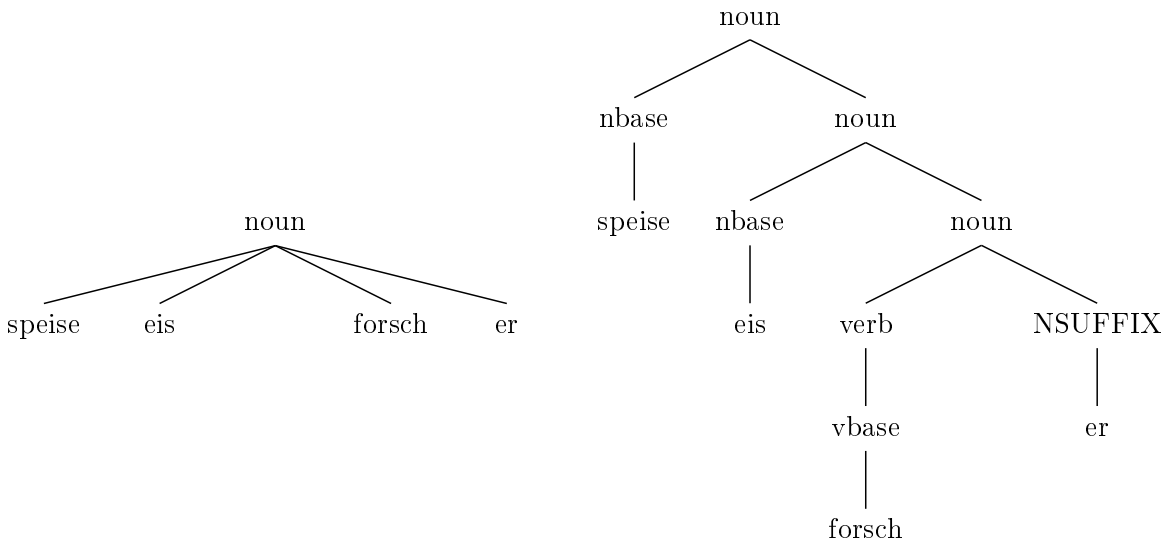

Figure 28: Example of a flat tree (left) and a gold standard tree (right)

4.2.2 Mandatory annotation

The definition of a Pcfg (Definition 2) requires a consistent start symbol S. A Pcfg directly trained on the CompoundTree corpus would have multiple start symbols because of the multiple root node labels of the trees in the corpus, which are equivalent to a start symbol. Therefore, we introduced a new start symbol "top" that we chose following examples in the literature (e.g. Collins (2003)) as displayed in Figure 29.
To keep the binary structure and to mark the beginning and end of a compound noun, we introduced the caret marker "^" for the beginning and the dollar marker "$" for the end of a compound

noun. Both markers possess their own parent constituent "car" and "dol" as in Figure 29. We chose symbols for the annotation that do not occur in the CompoundTree corpus.
Because of the annotation's influence on the overall performance of the Pcfg compared to the same Pcfg without start symbol as well as start and end marker, we decided to exclude the annotations mentioned in this chapter from the evaluation against the gold standard. But all of the annotations in chapter 4.2.3 to 4.2.5 include the start symbol and the start and end marker annotation out of the necessity to follow the Pcfgs requirement of a uniform start symbol.

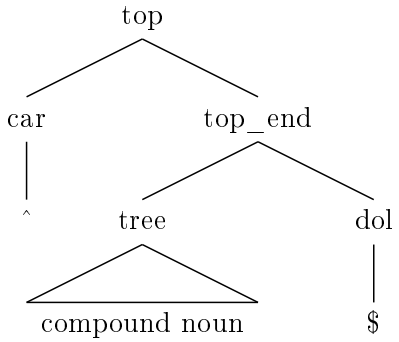

Figure 29: Example annotation of start symbol "top", start marker "^" and end marker "$".

4.2.3 Parent annotation

Following Johnson (1998), the parent annotation relaxes the independence assumption by adding the parent node label to the label of each non-root and non-preterminal node. An example can be found in 30. This annotation scheme increased the number of productions in the grammar by about 41.

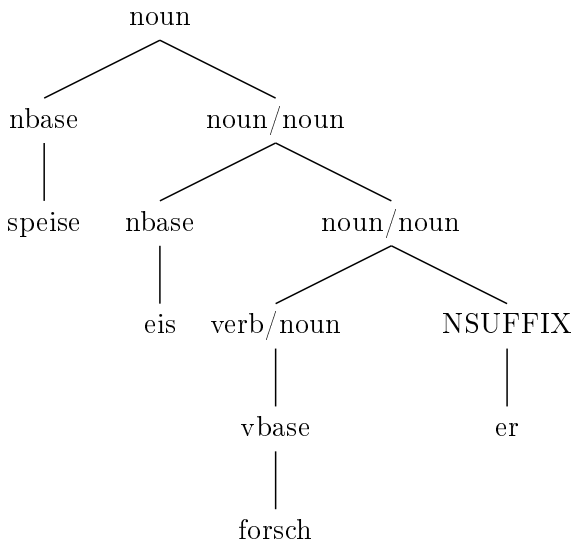

Figure 30: Example of parent annotation (annotation of start symbol "top", start marker "^" and end marker "$" are omitted for clarity).

4.2.4 Head annotation

The head annotation adds the part-of-speech (POS) tag of the head child's branch of the tree to every non-terminal node label. In binary branching structures on sentence level the head can be difficult to determine (see Collins (2014)). On the morphological level we follow the Right-Hand Head Rule (see 3.1.2) and pass the head of the right child upwards towards the root node while annotating the visited nodes on the way as can be seen in Figure 31.
The head annotation increased the number of productions in the grammar about 52.

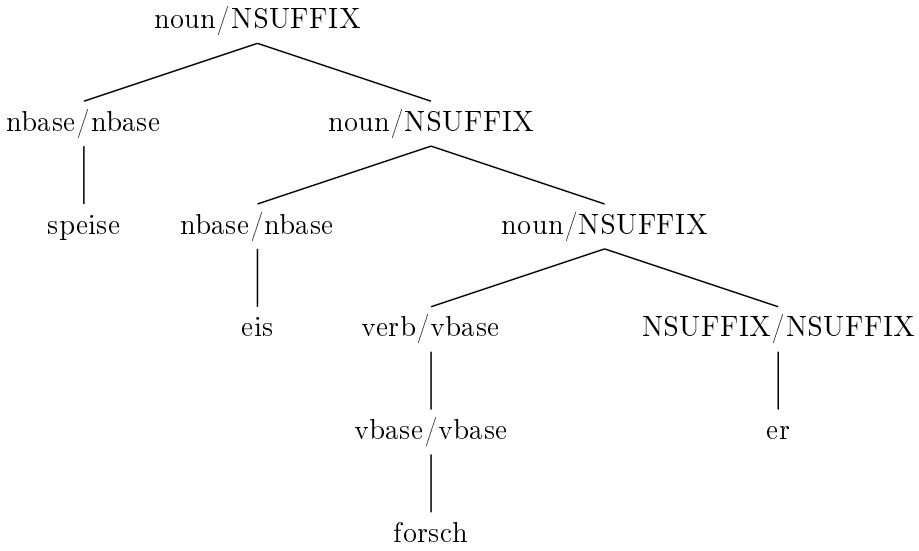

Figure 31: Example of head annotation (annotation of start symbol "top", start marker "^" and end marker "$" are omitted for clarity).

4.2.5 Lexicalisation

Following Schmid (2005) we lexicalised our PCFG by associating a morph and a part-of-speech tag pair with each node's non-terminal label in the tree (see Figure 32). Each of these nodes is associated with the (morph,POS)-pair it dominates. As with the head annotation, in a binary branching situation the (morph,POS)-pair from the child node that is the head gets passed upwards in direction of the root of the tree, annotating the visited node labels on the way.
The lexicalisation increases the number of productions in the grammar about 8.900.

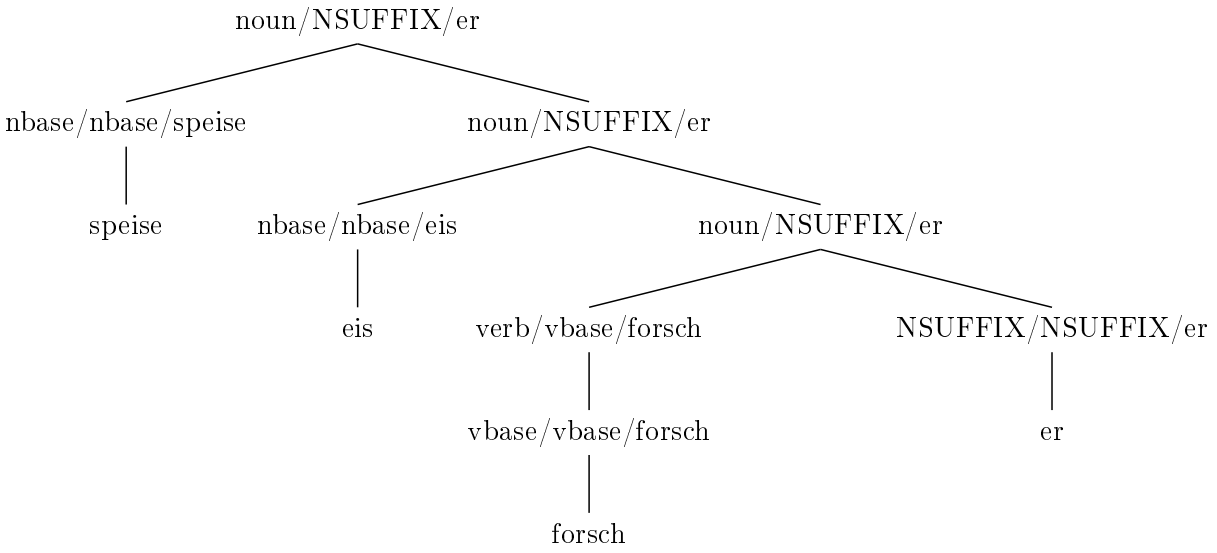

Figure 32: Example of lexicalised annotation (annotation of start symbol "top", start marker "^" and end marker "$" are omitted for clarity).

4.3 Parsing

Large context-free grammars can achieve high coverage and accuracy in parsing (Jurafsky and Martin (2000)). But difficulties can arise in case of massive inherent ambiguities (see figure 27 for an ambiguity example). As we mentioned in chapter 4.1, we can use probabilities to resolve the ambiguity problem of CFG's. To do so, we have to compute formula (11) to find the most-likely parse $\hat{T}(S)$ for a given string S.

$$\hat{T}(S) = \underset{T \in \tau(S)}{argmax}\ p(T) \tag{11}$$

Several algorithms have been introduced to efficiently calculate $\hat{T}(S)$. The probabilistic Early algorithm (see Stolcke (1995)) and the probabilistic Cocke-Younger-Kasami (CYK) algorithm (Ney (1991)) are two of them. For our survey, we used the BITPAR parser implementation (see Chapter 3.3.3), which is based on the CYK algorithm.

In the following, we give a brief introduction of the probabilistic **CYK** algorithm as described by Aho and Ullman (1972). The probabilistic CYK algorithm is a bottom-up parsing algorithm that parses an input string $w_1 \dots w_n$ based on a PCFG in Chomsky normal form. A grammar is in Chomsky normal form if it is ϵ-free and if every production is either of the form $A \rightarrow BC$ or $A \rightarrow a$.

While inductively parsing the input string, the CYK algorithm uses a dynamic programming table to store intermediate results consisting of the non-terminal, a span denoting the sub-string of the input string it derives and the related probability. A successful parse of the input string consists of a parse trees root symbol S, two indices that span the entire input string and the maximum probability of the parse. In the extended probabilistic version of the CYK algorithm a collection of backpointers can be used to restore the parse tree with the maximum probability from the parse table in a top-down fashion. The pseudo code and a detailed discussion of the CYK algorithm can also be found in Jurafsky and Martin (2000).

5 Results and discussion

In this section we present the results of our PCFG-based parsing approach of German compound nouns. In chapter 5.1 we compare the influence of several annotation schemes on the parsing performance and in chapter 5.2 we will discuss the results of this work.

5.1 Annotation schemes

We used the annotation schemes described in chapter 4.2 to alternate the COMPOUNDTREE based PCFG to investigate the effect on the parsing performance. The arithmetic mean of bracket precision, bracket recall, bracket Fmeasure as well as complete match and tag accuracy were calculated via held out validation (75% training data, 25% test data) and a 4-fold cross validation. We also give the standard deviation for the 4-fold results to show the magnitude of the variation between them.
All annotations were removed from the parsed test corpus results for the evaluation. Therefore, the evaluation is solely based on the comparison of the set of gold standard trees and the parsed and cleaned set of trees from the annotated PCFGs. The outcome of the evaluation is displayed in Table 2.

Annotation Scheme	Baseline		+parent		+head		+parent+head	
Measure	mean	std	mean	std	mean	std	mean	std
Bracket Precision	95.08	0.38	95.62	0.48	95.12	0.35	**96.05**	0.27
Bracket Recall	95.18	0.46	95.61	0.54	95.24	0.42	**96.12**	0.3
Bracket Fmeasure	95.13	0.44	95.61	0.51	95.16	0.39	**96.09**	0.28
Complete Match	85.73	1.18	86.85	1.66	86.1	1.12	**88.3**	1.23
Tag Accuracy	96.39	0.41	96.3	0.66	97.05	0.33	**97.16**	0.3

Table 2: Key figures about the parsing quality of PCFGs based on several annotation schemes. The abbreviations in use stand for the arithmetic **mean** and the **st**an**d**ard deviation.

Table 2 clearly shows an effect of the several annotation schemes on the parsing performance. In comparison with our baseline the +parent annotation increased the arithmetic mean of the complete match by 1.12%. We found, that the lexicalisation scheme lowers the parsing performance and is therefore not included in Table 2. The highest scoring annotation scheme is the combination of +parent+head annotation which increases the arithmetic mean of the complete match by 2.57%.

Annotation Scheme	Baseline	+parent	+head	+lex	+parent+head
Number of lexicon entries	3866	3866	3866	3866	3866
Number of productions	79	120	131	8900	184
Mean freq. of productions	235	156	140	2	184

Table 3: PCFG related frequencies.

The amount of productions is closely connected to the annotation scheme in use. As we described in Chapter 4.2.1, our annotation schemes led to a specialisation of the productions towards non-local relationships between the nodes. With this specialisation, the number of productions increases as well. By including parent information, the number of productions increased by 41,

by adding head information, the number of productions increased by 52, by including lexical information, the number of productions increased by 8.821 and by combining the head and parent information the number of productions increases about 105 productions. The mean frequency of the productions describes the number of occurences of each production in the grammar. As Table 3 shows, the baseline approach PCFG contains the lowest number of productions which occur with the highest mean frequency. For the other approaches the ratio between the number of productions and their mean frequency is relatively balanced, except for the lexicalisation annotation scheme. In contrast to the baseline approach, the lexicalised PCFG has the highest number of productions with a very low mean frequency.

For more insights into the effect of the annotation schemes on the parsing performance, we analysed the system's parsing errors related to the arity of the compound nouns. An overview of the parsing errors can be found in Table 4 where the percentage of parses with less than 100% in precision, recall and tag accuracy is accounted. Note that these figures are accumulated across the 4-fold cross-validation and the lowest error percentages are written bold. We find that the inclusion of non-local information into a PCFG has a positive effect on parsing 2-compounds and 3-compounds, and that the annotation scheme +parent+head obtains the lowest percentage of precision, recall and tagging errors.

For compounds with an arity of four or higher, the annotation schemes are ineffective or even increase the number of errors. We excluded the error percentage for 5-compounds because of the small amount of samples in our corpus. The +parent annotation scheme shows no effect on the error percentage of our baseline approach in precision, recall and tagging accuracy. The error percentage increased by the use of +head and +parent+head annotation schemes.

Scheme	baseline error in %			+parent error in %			+head error in %			+parent+head error in %		
Measure→ Arity ↓	prec	rec	acc	prec	rec	acc	prec	rec	acc	prec	rec	acc
2	.096	.095	.079	.084	.083	.081	.077	.076	.07	**.071**	**.068**	**.068**
3	.589	.589	**.116**	.595	.603	.15	.728	.725	.127	**.555**	**.561**	.127
4	**.563**	**.563**	**.125**	.563	.563	.125	.813	.813	.125	.688	.688	.125

Table 4: Percentage of bracket **prec**ision, bracket **rec**all and tag **acc**uracy errors in relation to the compounds arity and the annotation scheme in use. Lowest values are bold.

In the individual comparison of the erroneous parse trees to the gold standard trees we found branching tendencies in the PCFGs. We analysed in particular differing parse trees that where generated for the same compound word but by the use of PCFGs with varying annotation schemes. We found that the parent annotated PCFG and the head annotated PCFG share a branching direction bias in situations of hierarchical ambiguity (see Table 5). That branching bias dissolves with the combined +Parent+Head annotation scheme, which alternates left and right branching disambiguation.

Scheme	Branching Tendency
Baseline	no clear bias
+Parent	right
+Head	right
+Parent+Head	no clear bias

Table 5: Overview of the branching direction tendency of the annotation schemes that are measured for compound nouns where the annotation scheme's predictions differ.

5.2 Discussion

Table 2 shows good results and a positive effect of the tree annotation on the performance of the parser. The complete match of the +parent+head annotation schemes achieve 88.3%.
As we pointed out in chapter 3.2, the 4.000 compound nouns of the CompoundTree corpus were sampled without any modification to the proportion of various length compounds to prevent a loss of generality. With this sampling strategy, we hope to approximate a real distribution a compound parser might be deployed to. However, the observation in Table 4 shows that the problem of hierarchical ambiguity in compounds with an arity of three or higher remains unsolved.
Therefore, the good overall result is caused by the CompoundTree corpus bias towards 2-compounds (90%) which are relatively easy to parse. The error percentage increases to more than 50 percent with 3-compounds and keeps increasing with compounds of a higher arity. We interpret this behaviour as a consequence of the hierarchical ambiguity which the Pcfg fails to solve without semantic information about the combinability of the independent words or morphs that are included in the compounds.

The two approaches known to us that are closest to our approaches are Schmid (2004) and Würzner and Hanneforth (2013). In chapter 2 we discussed the differences between our and their approaches. Because of these differences, we will only consider the effect of the annotation schemes on the Pcfg-based results of their work and ours.
The annotation scheme shared by the three approaches is the lexicalisation, which has a significant positive effect on the Inside-Outside algorithm based training in Schmid (2004), but did neither improve our parsing results nor the results in Würzner and Hanneforth (2013) (with an exception in the context of coordinate structures). As we discussed in chapter 4.2.1, by applying the lexicalisation we weaken the independence assumption and increase the number of productions. Relative to the size of the training corpus, the lexicalisation can result in a reduction of the bias or an increase of the variance. In Schmid (2004), the training corpus consisted of 300 million words, Würzner and Hanneforth (2013) based their training on 4.000 adjectives whereas we based our training on 3.000 compounds. The difference in the size of the training corpora could be responsible for the different effects of the lexicalisation annotation on the results. With the large corpus of Schmid's approach the lexicalisation could reduce the bias and trigger an increase of the accuracy. With the relatively small corpora of Würzner and Hanneforth's approach as well as our approach, the lexicalisation could cause a higher variance and even decrease the accuracy as in our case. Especially if we consider that the lexicalisation results in the weakest independence assumption judging by the number of 8.900 productions and the drastic decrease

of the mean production frequency to 2 occurences in our work.
Similar to Würzner and Hanneforth (2013) the parent annotation adds a little performance to the grammar and we agree with their explanation that the reason are the simple structures of compounds (relative to sentence parsing). The same applies to the head annotation, which simply follows the rightmost branch because of the Right-Hand Head Rule in compounds as opposed to sentence parsing where the head can be difficult to determine (see 4.2.4). The Right-Hand Head Rule is also the most obvious explanation for the right branching tendency in the head annotation.
In contrast to Würzner and Hanneforth (2013) POS-tags are not included in our grammar and therefore in our approach the accuracy of tag prediction is not always 100% as table 2 shows. This leaves room for improvement in future work on this topic.

6 Conclusion

This work contributes to the analysis of German compound words using probabilistic context-free grammars. The impact of various tree annotation schemes was considered theoretically and investigated empirically by the creation, processing and evaluation of the German compound noun corpus CompoundTree which comprises the alignment of 4000 compound nouns and their related syntactic tree. For the creation of the CompoundTree corpus a variety of trees were generated for each compound noun in an automatic procedure and subsequently one fitting tree was hand-picked by an human annotator.
The best scoring annotation scheme +parent+head achieved an f1score of 96.09 and a complete match percentage of 88.3. We conclude that further work on the topic of Pcfg-based parsing of German compound nouns needs to be concerned with the problem of hierarchical ambiguity. We belived that the inclusion of semantic information into the annotation of the Pcfg will be helpful to constrain the variety of possible tree structures for a compound noun.
Unfortunately, within this work we have not had the time or team of human annotators to create a more comprehensive corpus. But a broader coverage of morphological structure analysis of German compounds would be of support in fields like speech recognition and statistical machine translation.

Acknowledgements

I want to thank my supervisors Thomas Hanneforth and Jean Vancoppenolle for their advice and suggestions and their interest in my project. In addition I want to thank Kay-Michael Würzner and Kai Zimmer from the Berlin-Brandenburg Academy of Sciences and Humanities for giving me an introduction to Dlexdb that unfortunately was not used in the present work because of the small time frame.

References

A. V. Aho and J. D. Ullman. *The Theory of Parsing, Translation, and Compiling.*, volume 1. Prentice-Hall, Englewood Cliffs, NJ., 1972.

J. K. Baker. Trainable Grammars for Speech Recognition. In *Speech Communication Papers for the 97th Meeting of the Acoustical Society of America*, pages 547–550, 1979.

J. A. Bondy and U. S. R. Murty. *Graph Theory.* Springer-Verlag, London, 2008. ISBN 978-1-84628-969-9.

M. Collins. Head-Driven Statistical Models for Natural Language Parsing. *Computational Linguistics*, 29:589–637, 2003.

M. J. Collins. Head Annotation Description. `http://www.cs.columbia.edu/~mcollins/papers/heads`, 2014.

P. Eisenberg. *Das Wort: Grundriss der deutschen Grammatik.* Verlag J.B. Metzler Stuttgart-Weimar, 2006.

W. Fleischer and I. Barz. *Wortbildung der deutschen Gegenwartssprache.* Niemeyer, Tübingen, 1992.

T. Fujisaki, F. Jelinsek, J. Cocke, E. Black, and T. Nishino. Analysis of Japanese Compound Nouns using Collocational Information. In *Proccedings of COLING-94*, pages 865–869, Kyoto, Japan, 1994.

S. Geman, E. Beinenstock, and R. Doursat. Neural Networks and the Bias/Variance Dilemma. *Neural Computation*, 4(1):1–58, 1992.

A. Geyken. The DWDS corpus: A reference corpus for the German language of the 20th century. In C. Fellbaum, editor, *Collocations and Idioms: Linguistic, lexicographic, and computational aspects.*, pages 23–41. London: Continuum Press., 2007.

A. Geyken and T. Hanneforth. TAGH: A Complete Morphology for German based on Weighted Finite State Automata. In *Finite State Methods and Natural Language Processing*, volume 4002, pages 55–66. Berlin, Heidelberg. Springer., 2006.

S. Green, M.-C. de Marneffe, J. Bauer, and C. D. Manning. Multiword Expression Identification with Tree Substitution Grammars: A Parsing Tour De Force with French. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 725–735, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

M. Haapalainen and A. Majorin. GERTWOL und morphologische Disambiguierung für das Deutsche. In *Proceedings of the 10th Nordic Conference on Computational Linguistics, Helsinki, Finland.*, 1995.

G. Hanrieder. Robustes Wortparsing. Lexikonbasierte morphologische Analyse (komplexer) deutscher Wortformen. Master's Thesis, Universtität Trier., 1991.

G. Hanrieder. MORPH - Ein modulares und robustes Morphologieprogramm für das Deutsche in Common Lisp. In R. Hauser, editor, *Linguistische Verifikation Dokumentation zur Ersten Morpholymics 1994*, pages 53–66. Niemeyer, Tübingen, 1996.

T. N. Höhle. Über Komposition und Derivation: Zur Konstituentenstruktur von Wortbildungsprodukten im Deutschen. *Zeitschrift für Sprachwissenschaft*, 1(1):76–112, 1982.

M. Johnson. PCFG Models of Linguistic Tree Representations. *Computational Linguistics*, 24 (4):613–632, Dec. 1998.

K. S. Jones. Compound Noun Interpretation Problems. In F. Fallside and W. Woods, editors, *Computer Speech Processing*, pages 363–81. Prentice-Hall, NJ., 1983.

D. Jurafsky and J. H. Martin. *Speech and Language Processing.* Prentice-Hall Inc.; Pearson Higher Education, 2000. ISBN 0-13-095069-6.

P. Koehn and K. Knight. Feature-Rich Statistical Translation of Noun Phrases. In *Proceedings of ACL*, pages 347–254, 2003.

T. Koshy. *Catalan Numbers with Applications.* Oxford University Press, 2008. ISBN 9780195334548.

K. Lari and S. Young. The Estimation of Stochastic Context-Free Grammars using the Inside-Outside Algorithm. *Computational Speech and Language Processing.*, 4:35–56, 1990.

M. Lauer. Corpus Statistics meet the Noun Compound: Some Empirical Results. In *Proceedings of the 33rd Annual Meeting of the ACL*, pages 47–54. Massachusetts Institute of Technology, Campbridge Mass., 1995.

C. D. Manning and H. Schütze. *Foundations of Statistical Language Processing.* MIT Press, Cambridge, MA., 1999.

H. Ney. *Dynamic Programming Parsing for Context-Free Grammars in Continuous Speech Recognition.* IEEE Transactions on Signal Processing, 39(2), 1991.

H. Schmid. Efficient Parsing of Highly Ambigouous Context-Free Grammars with Bit Vectors. In *Proceedings of the 20th International Conference on Computational Linguistics*, 2004.

H. Schmid. Disambiguation of Morphological Structure Using a PCFG. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 515–522, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics. doi: 10.3115/1220575.1220640.

S. Sekine and M. J. Collins. evalb - Bracket Scoring Program. `http://nlp.cs.nyu.edu/evalb/`, 1997.

M. Spies. A Language Model for Compound Words. In *Proceedings of Eurospeech*, pages 1767–1779, 1995.

P. Steiner and J. Ruppenhofer. Growing Trees from Morphs: Towards Data-Driven Morphological Parsing. In *Proceedings of the Int. Conference of the German Society for Computational*

Linguistics and Language Technology, pages 49–57. Gesellschaft für Sprachtechnologie and Computerlinguistik, 2015.

A. Stolcke. An Efficient Probabilistic Context-free Parsing Algorithm That Computes Prefix Probabilities. *Computational Linguistics*, 21(2):165–201, June 1995. ISSN 0891-2017.

A. Stolcke and E. Shriberg. Automatic Linguistic Segmentation of Conversational Speech. In *Proceedings of ICSLP, Philadelphia, PA, USA*, pages 1005–1008, 1995.

K.-M. Würzner and T. Hanneforth. Parsing Morphologically Complex Words. In *Proceedings of the 11th International Conference on Finite State Methods and Natural Language Processing*, 2013.

Declaration of Authorship

I, Philipp Gawlik, declare that the thesis "PCFG-based parsing of German compound words" and the work presented in it are my own and has been generated by me as the result of my own original research.
I further confirm that:

1. This work was done while in candidature for a research degree (Bachelor of Science) at the University of Potsdam.
2. Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
3. Where I have consulted the published work of others, this is always clearly attributed.
4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
5. I have acknowledged all main sources of help.
6. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.
7. None of this work has been published before submission.

Signed: Place and Date: