Parsing German Topological Fields with Probabilistic Context-Free Grammars

by

Jackie Chi Kit Cheung

A research paper submitted in conformity with the requirements
for the degree of M. Sc.
Graduate Department of Computer Science
University of Toronto

Abstract

Parsing German Topological Fields with Probabilistic Context-Free Grammars

Jackie Chi Kit Cheung

M. Sc.

Graduate Department of Computer Science

University of Toronto

2009

Syntactic analysis is useful for many natural language processing applications requiring further semantic analysis. Recent research in statistical parsing has produced a number of high-performance parsers using probabilistic context-free (PCFG) models to parse English text, such as (Collins, 2003; Charniak and Johnson, 2005). Problems arise, however, when applying these methods to parse sentences in freer-word-order languages. Such languages as Russian, Warlpiri, and German feature syntactic constructions that produce discontinuous constituents, directly violating one of the crucial assumptions of context-free models of syntax.

While PCFG technologies may thus be inadequate for full syntactic analysis of all phrasal structure in these languages, clausal structure can still be fruitfully parsed with these methods. In particular, we examine applying PCFG parsing to parse the *topological field* structure of German. These topological fields provide a high-level description of the major sections of a clause in relation to the clausal main verb and the subordinating heads and appear in strict linear sequences amenable to PCFG parsing. They are useful for tasks such as deep syntactic analysis, part-of-speech tagging and coreference resolution.

In this work, we apply an unlexicalized, latent variable-based parser (Petrov et al., 2006) to topological field parsing, and achieve state-of-the-art parsing results on two German newspaper corpora without any language- or model-dependent adaptation.

We perform a qualitative error analysis of the parser output, and identify constructions like ellipses and parentheticals as the chief sources of remaining error. This is confirmed by a

further experiment in which parsing performance improves after restricting the training and test set to those sentences without these constructions.

We also explore techniques for further improving parsing results. For example, discriminative reranking of parses made by a generative parser could incorporate linguistic information such as those derived by our qualitative analysis. Self-training is another semi-supervised technique which utilizes additional unannotated data for training.

Acknowledgements

Many people have contributed to the making of this document, whom I would like to thank here.

First and foremost, I would like to thank my supervisor and mentor, Gerald Penn. I am continually amazed by the breadth and depth of his knowledge. Through this, he has given me a much better perspective of computational linguistics, yet I know that I have much more to learn from him in the future.

I would also like to thank Graeme Hirst for his role as the second reader of this paper. His comments have greatly improved the clarity of expression in this paper.

I feel very fortunate to have found a home in the computational linguistics group in the Department of Computer Science at the University of Toronto. The people I have met and the ensuing discussions (academic and otherwise) have made my experience the opposite of every graduate school horror story I have read or heard about. In particular, I would like to thank my former and current officemates Aida Nematzadeh Chekhoudar, Xuan Le, and Michael Guerzhoy; and fellow research group members Rouzbeh Farahmand, Timothy A. D. Fowler, Michael Reimer, Eric Corlett, and Nicolas Trahan. My dear friends in Vancouver and Toronto and my family also deserve my heartfelt thanks for their faith and confidence in me.

I am grateful to Markus Becker, Anette Frank, Sandra Kuebler, and Slav Petrov for their help in gathering the resources necessary for the experiments. This document has benefited greatly from the careful eye of Sandra Yuen and Timothy A. D. Fowler, who have made valuable suggestions to improve its quality. I would also like to acknowledge the financial support provided by the Natural Sciences and Engineering Research Council of Canada.

Contents

1 Introduction **9**

1.1 Topological Field Model of German 14

1.2 Organization of Paper 16

2 Related Work **18**

2.1 Statistical Parsing 18

2.1.1 Generative Approaches 22

2.1.2 Discriminative Approaches 26

2.2 Topological Field Chunkers and Parsers 27

2.3 Other Computational Models of German Syntax 28

3 Applications **30**

3.1 Part-of-Speech Tagging 30

3.2 HPSG Parsing 31

3.3 Anaphora Resolution 32

3.4 Summary 34

4 Parsing Experiments **35**

4.1 A Latent-Variable Parser 35

4.2 Data 36

4.3 Results 38

4.4 Analysis 41
4.4.1 Category-Specific Results 41
4.4.2 Qualitative Error Analysis 44
4.4.3 Results on Subsets 46
4.4.4 Merging Topological Field Labels 49

5 Improving Topological Field Parsing **50**
5.1 Reranking for Paired Punctuation 50
5.2 Self-Training 53

6 Conclusions **57**

List of Tables

1.1 Topological field model of German 15

4.1 Parsing results for topological fields and clausal constituents in the TüBa-D/Z corpus 38

4.2 Comparison of parsing results 40

4.3 Category-specific results 42

4.4 Types and frequency of parser errors in the fifty worst scoring parses 43

4.5 Subset parsing results 48

4.6 Relaxed labelling parsing results 49

5.1 Effect of constrained reranking, using gold tags and without edge labels 51

5.2 Oracle statistics for 50-best list of TüBa-D/Z test set 53

5.3 Equivalence classes between the TüBa-D/Z and NEGRA annotation schemes . 55

5.4 Parsing results for topological fields and clausal constituents in the TüBa-D/Z corpus's test set after self-training 56

List of Figures

1.1 A German subordinate clause with a discontinuous constituent 13

2.1 Partial listing of rules and probabilities in a sample PCFG 20

2.2 Ambiguous parses from a PCFG 20

4.1 Sample TüBa-D/Z tree with topological field annotations and edge labels . . . 37

Chapter 1

Introduction

One important way in which languages differ from each other is in their syntactic structure. A typical starting point in describing the syntax of a language is to describe the behaviour of the subject and object of the sentence in terms of their position relative to the verb. It is thus customary to classify languages into a language typology based on their "basic" word-order, using some notion of the word "basic", such as by specifying the clause type or grammatical mood.

Many languages can be unproblematically classified in this way; for example, English exhibits SVO order (subject-verb-object), Irish VSO, and so on. This does not mean that these languages adhere to their basic order without exception. For example, English allows objects and adjectives (among other types of constituents) in front of the verb in certain contexts.

(1.1) (a) ***This tie,*** *Fred bought.* (**OSV**) (Cormack and Smith, 2000)

(b) ***So bad*** *was the smell that nobody would go near the room.* (**Adjective-VS**)

Other languages, however, are more problematic and defy an easy classification. For example, Russian and other Slavic languages display considerably more variation in their word orders than English, though they are generally ascribed a SVO basic word order (Siewierska, 1998). An even more extreme case is Warlpiri, an Australian Aboriginal language which has famously been described as *non-configurational* (Hale, 1983), meaning that syntactic relations

such as notions of subject and object are not reflected in the phrase structure of the language. This is illustrated in example 1.2, which shows that all possible orderings of the subject, verb, and object are possible in a simple declarative sentence meaning. The only constant in word order between the sentences is the second position auxiliary particle *ka*.

(1.2) (a) *Ngarrka- ngku ka wawirri panti- rni*
man ERG AUX kangaroo spear NONPAST
'The man is spearing the kangaroo.'

(b) *Wawirri ka panti-rni ngarrka-ngku*

(c) *Panti-rni ka ngarrka-ngku wawirri.*

And so on (Hale, 1983).

Looking beyond the order of the subject, verb, and object, we can also consider the order of other syntactic elements, such as the order of a head noun and its modifiers. It turns out Warlpiri also offers much variation in word order in this respect. The following examples illustrate possible orderings of the elements of the noun phrase 'that kangaroo' *wawirri yalumpu*, where the demonstrative 'that' does not have to occur next to the head noun 'kangaroo', which is an example of a *discontinuous constituent*.

(1.3) (a) ***Wawirri*** *kapi-rna panti- rni* ***yalumpu***
kangaroo AUX spear NONPAST that
'I will spear that kangaroo.'

(b) ***Wawirri*** ***yalumpu*** *kapi-rna panti- rni*
kangaroo that AUX spear NONPAST
(Hale, 1983)

Languages which display such variation in their word order have been called "free-word-order" languages. This label is not unproblematic. First, most "free-word-order" languages, including Warlpiri, as we have seen above, have a privileged second position that is occupied by clitics, auxiliaries, finite verbs, or sometimes also complementizers and subordinating conjunctions. Second, while word order may be variable with respect to grammatical roles, they

may not be completely "free" with respect to other factors, usually related to the information structure of the discourse. Thus, in this paper, we will use the term *freer-word-order* languages, keeping in mind these caveats.

The focus of this paper will be the task of automatically analyzing the clausal structure of Standard German. German is a freer-word-order language in the West Germanic branch of the Indo-European language family, and frequently exhibits freer-word-order phenomena like *topicalization*, *scrambling*, and *extraposition*. Topicalization is the process where a constituent is found at the beginning of a sentence or clause due to its status as the topic of a sentence (loosely, what the sentence is about.) Scrambling refers to variability in the order of the noun phrases, and extraposition is the process in which prosodically heavy elements are optionally placed at the right edge of a sentence.

Unlike English, which relies heavily on word order to indicate grammatical functions, German possesses richer inflectional morphology for this purpose, which allows more latitude for word order variation. Again, this is not to say that German has no constraints on its word order. For example, one characteristic of German and closely related languages such as Dutch is their verb-second word order, which means that the second constituent of a matrix clause in a declarative sentence is a finite verb. Verb-second word order interacts with freer-word-order phenomena to produce many possible renderings of a sentence that differ in word order, but have the same semantic content. Consider the following example, which involves scrambling and topicalization. We first show the sentence in its prescriptively canonical word order.

(1.4) *Die* *Frau* *hat* *dem* *Mann* *das* *Buch* *gegeben.*
the.NOM woman has the.DAT man the.ACC book given.
'The woman gave the man the book.'

All other logical orders of the noun phrases are possible, with the same semantic interpretation, although some of the non-canonical orderings in this and other examples may be considered unusual and pragmatically highly marked.

(1.5) (a) *Die Frau hat das Buch dem Mann gegeben.*

(b) *Dem Mann hat die Frau das Buch gegeben.*

(c) *Dem Mann hat das Buch die Frau gegeben.*

(d) *Das Buch hat die Frau dem Mann gegeben.*

(e) *Das Buch hat dem Mann die Frau gegeben.*

In the above examples, the first position before the finite verb *'hat'* is occupied by a noun phrase. Other constituents may also appear in this position for pragmatic reasons, such as the past participle of the verb or an adverb.

(1.6) (a) *Gegeben hat die Frau dem Mann das Buch.*

(b) *Gestern hat die Frau dem Mann das Buch gegeben.*
yesterday has the.NOM woman the.DAT man the.ACC book given.
'The woman gave the man the book yesterday.'

While these examples differ substantially in word order, some commonalities can be observed. First, the finite verb is always in second position. Also, the past participle *gegeben* always appears at the end of the sentence, except in the case where it was fronted to the first position before the finite verb. In the rest of the sentence, however, there is much variation in the order of the remaining elements.

German sentences can also exhibit discontinuous constituents. We will provide the following example of a German subordinate clause from Duchier and Debusmann (2001). The two syntactic constructions relevant to this example are scrambling, and extraposition.

(1.7) (a) *dass Maria [einen Mann zu lieben] versucht.*
that Maria.NOM a.ACC man to love tries.
'that Maria tries to love a man.'

(b) *dass Maria versucht [einen Mann zu lieben].*

(c) *dass Maria [einen Mann] versucht [zu lieben].*

(d) *dass [einen Mann] Maria [zu lieben] versucht.*

(e) *dass [einen Mann] Maria versucht [zu lieben].*

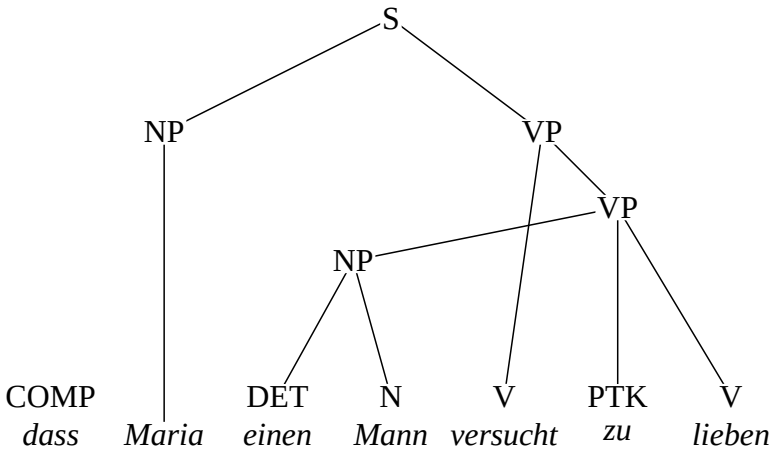


Figure 1.1: A German subordinate clause displaying a discontinuous constituent, *einen Mann zu lieben*

(f) **dass versucht Maria [einen Mann zu lieben].*

1.7(a) shows the word order without scrambling of the noun phrases *Maria* and *einen Mann*. 1.7(b) shows the word order when the verb phrase *einen Mann zu lieben* 'to love a man' is extraposed to the right. (This is also the most common word order of the clause.) In 1.7(c), only the verbal head *zu lieben* part of the verb phrase is extraposed, leading to it being separated from the complement of *lieben*, resulting in a discontinuous constituent. 1.7(d) and 1.7(e) show the resulting word orders when *einen Mann* is placed before *Maria* due to scrambling, without and with the effect of partial extraposition, respectively.

Of these five possible orderings, we see that three of them result in a discontinuous verb phrase. Trying to find a pattern in the word order is more difficult here, but some regularities can be detected. First, the complementizer *dass* always appears in first position. Also, the finite verb *versucht* no longer appears in second position as in the previous example. We will shortly see that these two word-order observations are related. However, there is always at most one constituent to the right of the finite verb, meaning that an ordering such as 1.7(f) is

ungrammatical.

Discontinuous constituents pose a problem for standard *context-free grammar* (CFG) models of language, as well as for existing CFG-based parsing technology. Linguistic structures produced by CFGs have the restriction that they must be *projective*, meaning that crossing branches in the constituency structure are disallowed. However, this is precisely what is needed to model discontinuous constituents. Figure 1.1 shows a constituency tree for example 1.7(b), where the crossing branches are necessary to model the dependencies in the example.

On the other hand, there is utility in simply modelling the clause-level patterns in word order. With this information, we can identify the verbal head in a sentence as well as possible locations for its arguments. This shallower form of parsing would allow us to respect projectivity, because non-projectivity is a result of discontinuous constituents at the sub-clausal level. To this end, we now describe the topological field model of German, which allows us to characterize the observations about patterns in the word order that we have made above.

1.1 Topological Field Model of German

Topological fields are high-level syntactic units which appear in a prescribed linear order. They exist in an enclosing syntactic region (Höhle, 1983), which is the clause in German, and describe the major sections of a German clause by identifying the verbal heads and subordinating conjunctions.

Topological fields may have constraints on the number of words or phrases they contain, and are not required to form a semantically coherent constituent. The main reason that topological fields are a good model of German clausal syntax is that the order of the topological fields is mostly strict and unvarying, in contrast to the relatively free orderings possible at the sub-clausal level.

In the German topological field model, clauses belong to one of three types: verb-last (VL), verb-second (V2), and verb-first (V1), each with a specific sequence of topological fields (Table

Type	Fields
VL	(KOORD) (C) (MF) VC (NF)
V1	(KOORD) (LV) LK (MF) (VC) (NF)
V2	(KOORD) (LV) VF LK (MF) (VC) (NF)

Table 1.1: Topological field model of German. Simplified from the annotation schema of the German newspaper corpus, TüBa-D/Z (Telljohann et al., 2006). Parentheses indicate optional elements.

1.1). VL clauses include finite and non-finite subordinate clauses, V2 sentences are typically declarative sentences and *wh*-questions in matrix clauses, and V1 sentences include yes-no questions, and certain conditional subordinate clauses. Below, we give brief descriptions of the most common topological fields.

- VF (*Vorfeld* or 'pre-field') is the first obligatory constituent in sentences of the V2 type. This is often the topic of the sentence. [1]
- LK (*Linke Klammer* or 'left bracket') is the position for finite verbs in V1 and V2 sentences. It is replaced by a complementizer with the field label C in VL sentences.
- MF (*Mittelfeld* or 'middle field') is an optional field bounded on the left by LK and on the right by the verbal complex VC or by NF. Most verb arguments, adverbs, and prepositional phrases are found here, unless they have been fronted and put in the VF, or are prosodically heavy and postposed to the NF field.
- VC is the verbal complex field. It includes infinite verbs, as well as finite verbs in VL sentences.

[1] An anonymous reviewer to the ACL paper upon which this part of the paper is based pointed out that this position does not correspond to a single function with respect to information structure. The reviewer suggested this case, where VF contains the focus:
—*Wer kommt zur Party?* (Who is coming to the Party?)
—***Peter*** *kommt zur Party.* (**Peter** is coming to the party.)

- NF (*Nachfeld* or 'post-field') contains prosodically heavy elements such as postposed prepositional phrases or relative clauses.

- KOORD (*Koordinationsfeld* or 'coordination field') is a field for clause-level conjunctions.

- LV (*Linksversetzung* or 'left dislocation') is used for resumptive constructions involving left dislocation. For a detailed linguistic treatment, see (Frey, 2004a).

Exceptions to the topological field model as described above do exist. For instance, parenthetical constructions exist as a mostly syntactically independent clause inside another sentence. Consider the following example taken from the German newspaper corpus TüBa-D/Z. In this annotation scheme, parentheticals are attached directly underneath a clausal node without any intervening topological field. In this example, the parenthetical construction is highlighted in bold print. Some clause and topological field labels under the NF field are omitted for clarity.

(1.8) (a) *(SIMPX "(VF Man) (LK muß) (VC verstehen) " , **(SIMPX sagte er)**, " (NF daß diese Minderheiten seit langer Zeit massiv von den Nazis bedroht werden)). "*

(b) Translation: "One must understand," **he said**, "that these minorities have been massively threatened by the Nazis for a long time."

1.2 Organization of Paper

In the following chapters, we will examine the problem of parsing topological field structures in German. First, we will review previous work in statistical methods for constituency parsing in general and topological field chunking and parsing in particular (Chapter 2). Then, Chapter 3 will describe applications of topological fields to other natural language processing tasks like anaphora resolution. Next, we will show that a general statistical parser (Petrov et al., 2006) achieves state-of-the-art performance on two German newspaper corpora, outperforming previous parsers, many of which were tailored to this domain. We will also perform an analysis

of our parsing results by examining some of the remaining errors made by the parser, and consider constructions like ellipses and parentheticals which break the standard prescribed topological field model presented above and affect parsing results (Chapter 4). After that, we will explore reranking and self-training as strategies to improve on the results of the Berkeley parser (Chapter 5). Finally, the last chapter concludes our discussion of German topological fields with remarks on future directions of research.

Chapter 2

Related Work

The problem of phrase structure parsing can be defined as follows. Given an input string of natural language text, return a phrase structure tree of the sentence according to a grammar of the language. As we have seen in the last chapter, linguistic constituents in freer-word-order languages sometimes occur in discontinuous segments in a sentence, which causes problems for many existing parsers that implicitly assume that constituents form continuous substrings. In particular, parsers must deal with the problem of *non-projectivity*, the case of syntax trees having crossing branches, which is classically disallowed by CFGs. Topological field parsing as introduced in the previous chapter allows us to avoid the projectivity problem while still providing us with useful information about the sentence.

In this chapter, we will review existing general statistical parsing methods suitable for topological field parsing. Then we will examine previous work in parsing and chunking topological fields. Finally, we will provide a brief introduction to computational models of German syntax making use of other grammar formalisms.

2.1 Statistical Parsing

We first provide a formal description of context-free grammars. A CFG G can be defined as the following 4-tuple:

$G = (N, \Sigma, R, S)$, where

- N is a set of *nonterminal symbols*.
- Σ is a set of *terminal symbols*.
- R is a set of *rules* or *productions* in the form of $A \rightarrow \beta$, where $A \in N$ and β is an ordered list of symbols drawn from $N \cup \Sigma$.
- S is the starting symbol.

Sentences can be generated from a CFG in the following derivational process. Starting with S, rewrite a nonterminal A by replacing it with the right-hand side of a rule with A on the left-hand side. Repeat this rewriting process until we end up with a string of terminals. One view of parsing is to recover this derivational process for a target output sentence.

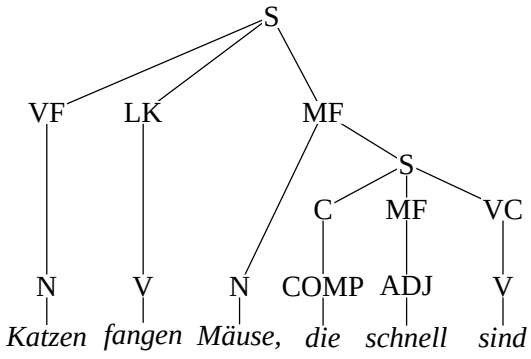
Probabilistic context-free grammars (PCFGs) are an extension of CFGs in which each rule is associated with a probability $p \in [0, 1]$. In a consistent PCFG, the probabilities of all the rules with the same nonterminal on the left-hand side form a probability distribution (i.e., sum to one). Because natural language is highly ambiguous, many CFG parses can result from a single sentence. The main utility of PCFGs is to allow us to select the best parse for a sentence among multiple parses, according to its probability model. Polynomial-time parsing algorithms exist and are well known for CFGs and PCFGs.

The probability of a parse tree in a PCFG is the product of the probabilities of each of the rules used in the parse tree. Consider example 2.1 and the associated sample PCFG (Figure 2.1). Two parse trees can be generated for this sentence, which are shown in Figure 2.1 along with the associated product of rule probabilities used to compute the probability of the parse.

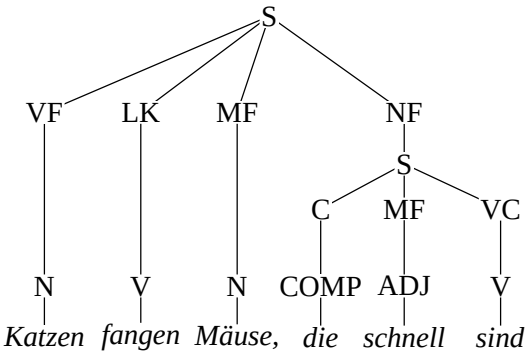
(2.1) *Katzen fangen Mäuse, die schnell sind.*
cats catch mice who fast are.
'Cats catch mice who are fast.'

S → VF LK MF, 0.2	N → *Katzen*, 0.01
S → VF LK MF NF, 0.3	N → *Mäuse*, 0.02
S → C MF VC, 0.1	V → *fangen*, 0.03
VF → N, 0.55	V → *sind*, 0.35
LK → V, 0.95	COMP → *die*, 0.4
MF → N, 0.25	ADJ → *schnell*, 0.05
MF → N S, 0.05	
MF → ADJ, 0.15	...
C → COMP, 0.9	
VC → V, 0.8	
NF → S, 0.7	

Figure 2.1: Partial listing of rules and probabilities in a sample PCFG



(a) Parse probability: $0.2 \times 0.55 \times 0.01 \times 0.95 \times 0.03 \times 0.05 \times 0.02 \times 0.1 \times 0.9 \times 0.4 \times 0.15 \times 0.05 \times 0.8 \times 0.35 = 2.37 \times 10^{-12}$



(b) Parse probability: $0.3 \times 0.55 \times 0.01 \times 0.95 \times 0.03 \times 0.25 \times 0.02 \times 0.7 \times 0.1 \times 0.9 \times 0.4 \times 0.15 \times 0.05 \times 0.8 \times 0.35 = 1.244 \times 10^{-11}$

Figure 2.2: Ambiguous parses for the sentence *Katzen fangen Mäuse, die schnell sind.* from example 2.1 based on PCFG from Figure 2.1

To create a PCFG parsing model then, we need to determine the rules in the grammar and the rule probabilities. This is typically done by training the model on a *treebank*, a collection of parse trees that have been carefully annotated (usually by humans). The simplest method would be to take the set of rules present in the trees in a treebank, and assign to each rule a probability equal to the frequency of their occurrence in the treebank divided by the frequency of the LHS nonterminal. For example, if the treebank contains 110 occurrences of the rule VF $\rightarrow$ N, and VF occurs 200 times in total, the rule would be assigned a probability of 0.55. This method of probability estimation is a case of *maximum likelihood estimation*. We will look at more sophisticated methods of training a parsing model in the next sections.

After training and parsing with a parsing model, we next face the problem of evaluating the output parses. To do this, we compare the generated parses on a test section of the treebank, which we did not use in training or in developing the parser, against a gold standard annotation of the same section. The usual method of doing this comparison is to use a family of constituent-level PARSEVAL measures from Abney et al. (1991).

The most commonly used measures are *precision*, and *recall*. They are defined as would be expected based on their usage in information retrieval:

$$Precision = \frac{\text{\# Correctly parsed constituents}}{\text{\# Constituents identified by parser}}$$

$$Recall = \frac{\text{\# Correctly parsed constituents}}{\text{\# Constituents in gold standard annotation}}$$

Precision and recall can be combined into one score called the F-measure, the harmonic mean of the precision and recall. Different weights can be given to each of precision and recall, but typically, F_1 is used (equal weights on both).

$$F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

A constituent is considered to be correctly parsed if the start and end of the constituent are correctly identified. This is known as *unlabelled* constituent accuracy. One can also impose the

additional requirement that the label (i.e. nonterminal category) assigned by the parser must match the gold standard annotation, which would be the *labelled* constituent accuracy.

Other measures defined by PARSEVAL consider the number of *cross-brackets* in the sentence. A cross-bracket occurs when the elements spanned by a constituent in the gold parse and a constituent in the parser's output partially overlap, but where neither is a subset of the other. That is, for three elements (A B C), one posits a constituent over A and B, ((A B) C), while the other posits one over B and C (A (B C)).

Most work in parsing has used treebanks of newspaper text for training and testing. In English, the most widely used corpus is the *Penn TreeBank* (Marcus et al., 1993), specifically sections of it that are drawn from *Wall Street Journal* text. In German, three corpora are commonly used: NEGRA (Brants et al., 1999), TIGER (Brants et al., 2002), and TüBa-D/Z (Telljohann et al., 2004). NEGRA and TIGER are copora based on the newspaper *der Frankfurter Rundschau*, whereas TüBa-D/Z is based on another newspaper, *die tageszeitung*. The principal difference between these German corpora lies in the type of tree structure that is used. NEGRA and TIGER both emphasize dependency structures, and use crossing branches to model long-distance dependencies. This means that they must be projectivized before they can be used in (P)CFG parsing. TüBa-D/Z, on the other hand, respects projectivity, using additional annotations on the edges between nonterminal nodes to mark long distance dependencies. Another important difference is that only TüBa-D/Z is annotated with topological fields. This will have ramifications for our experiments in the next chapter.

We now review existing techniques for parsing phrase structures. We will divide them into two broad categories based on the type of probability model that they use: *generative* models, and *discriminative* models.

2.1.1 Generative Approaches

In machine learning, learning a generative model involves learning the joint probability distribution $P(x,y)$, where x is the observed input and y is the output. They are so named because

sampling from the joint probability distribution can be done to generate instances of (x,y). In parsing, the input is the sentence, and the output is the parse tree. Finding the best parse for a sentence is then finding $\hat{y}$ such that:

$$\hat{y} = \operatorname*{argmax}_y P(x,y)$$

$$= \operatorname*{argmax}_y P(x|y)P(y)$$

But $P(x|y) = 1$ if y yields x, and 0 otherwise. Thus, the above expression can be rewritten as

$$\hat{y} = \operatorname*{argmax}_{y \text{ s.t. } x=yield(y)} P(y)$$

As we saw in the previous section, one simple way to define a generative probability model over a CFG is to define the probability of a parse tree to be the product of the probabilities of the rules used in the derivation of the tree. One major problem with this approach is that it wrongly assumes that the rule probabilities in the derivation can be determined independently of each other. With a basic set of nonterminal symbols drawn from linguistic theories, this is clearly wrong. Consider example 2.1, specifically the relative clause *die schnell sind*, "that are fast". Relative pronouns in German are marked for gender, case, and number. In this example, *die* marks the plurality and the nominative case. The finite verb in the clause, *sind*, must agree in person and number with the subject of the sentence, which is the relative pronoun in this example. Knowing that the relative pronoun is *die*, as opposed to, say, *das* (neuter, nominative), we can be more confident that the finite verb at the end of the clause is *sind* rather than *ist* (singular subject). This dependence, however, is not reflected in the simple PCFG rule probabilities assigned in Table 2.1.

Strategies to correct this assumption involve refining nonterminal categories into more fine-grained categories. These parsing models are able to deal with lexical and structural dependencies in rule probabilities because the rule probabilities for each split nonterminal are estimated separately from the treebank.

One approach is to incorporate lexical information into the nonterminals, in effect creating many more nonterminal categories. However, this approach creates a much larger model,

and hence requires a much greater amount of data to estimate the probabilities of rules associated with each lexical item. Since most lexicalized rules are encountered a paltry number of times in a treebank, sparsity of data and overfitting becomes a severe issue. The sparsity problem must then be overcome by making other independence assumptions or by smoothing and interpolating rule probabilities, for example towards their part-of-speech.

Collins (1996, 2003) describes three lexicalized statistical parsing models of increasing linguistic sophistication. These models incorporate the parent, the head word and the part-of-speech (POS) tag of the head word into nonterminals, and are simply named Model 1, Model 2, and Model 3. As discussed above, treating the augmented nonterminals as independent labels in a regular PCFG causes severe sparsity problems, hence further independence assumptions must be made. In Model 1, the probability of phrase structure rules is decomposed into the probabilities of generating the head of the right-hand side (RHS), and each of the non-terminals in the RHS, given the parent node and the head node. Model 2 and Model 3 extend Model 1 to model other complex linguistic structures, such as the subcategorization frames of words, and *wh*-movement.

The Charniak parser[1] (1997; 2000) is similar to the Collins Model 1 parser in the way it breaks down PCFG rules into probabilities of generating each produced nonterminal, given the parent and the head of the constituent. It differs in conditioning on other information from the context of the rule application, which is called the "history", such as for example the grandparent label.

While the above results have shown that lexicalization is beneficial for parsing of English, the case is not as clear in German. In Dubey and Keller (2003), PCFG parsing of NEGRA is improved by using sister-head dependencies, which outperforms standard head lexicalization as well as an unlexicalized model. The best performing model with gold part-of-speech tags available during parsing achieves an F_1 of 75.60%. Sister-head dependencies are useful in

[1]Not to be confused with later discriminative reranking parsers based on this generative baseline such as Charniak and Johnson (2005).

this case because of the flat structure of NEGRA's trees. Later, Dubey (2005) show that using morphological information outperforms the lexicalized baseline, with an F_1 of 76.3 on the same corpus using the same training, development, and test sets.

These numbers are much lower than the reported F_1 for labelled English constituency parsing, leading to a belief that parsing in German is harder. However, this may really be a function of the annotation scheme used in the corpus used rather than the language. NEGRA's annotation scheme is very flat in the sense that internal structure of phrases is often omitted. So, for example, the prepositional phrase *auf die Bank* 'to the bank' would not contain a noun phrase constituent *die Bank*. Rather, (the part-of-speech tags of) the three words are attached directly to the PP label. Internal structure, however, can considerably aid parsing. Kübler et al. (2006) show that lexicalization and other parsing techniques used for English do indeed improve parsing on the TüBa-D/Z corpus, which does contain internal structure, yielding F_1 numbers comparable to English parsing. Since the German corpora being compared both contain newspaper text, the large difference in performance is attributed to the annotation scheme.

Lexicalization is not the only way to split nonterminals. Other splits can be found based on linguistic intuitions about subtypes of categories, or by statistical methods that maximize the likelihood of generating the training data. Intuitively, phrases grouped under one nonterminal in standard tagsets can be of many types. For example, the determiner *that* can stand on its own as a noun phrase, or occur with other nominal material as in *that woman*. These different uses of a nonterminal would be expected to have different distributional properties with respect to the rules of a PCFG, and thus can be usefully split.

Using this intuition, Klein and Manning (2003) designed an unlexicalized parser that is competitive with lexicalized ones. They found splits such as the determiner split above manually, based on their linguistic knowledge. Later work by Petrov et al. (2006) automates this process by alternately splitting symbols to increase the expressiveness of the grammar, and merging symbols to keep the problem size manageable and to reduce the risk of overfitting. More details of their parser, called the Berkeley parser, will be provided in section 4.1.

2.1.2 Discriminative Approaches

In contrast to the previous generative approach, discriminative models aim to learn the conditional probability distribution $P(y|x)$ directly. Approaches differ in how they define the probability of a tree given a sentence. Often, the probabilistic interpretation is abandoned entirely, so discriminative parsing boils down to determining the best parse of a sentence by optimizing some arbitrary function. This flexibility in being able to easily incorporate varied knowledge about parse trees into the optimization function is the major advantage of discriminative parsers. There are two main types of discriminative parsers—discriminative reranking parsers, and parsers using dynamic programming approaches.

In discriminative reranking, a number of "good" parses are generated by generative approaches as above. These N-best parse trees are then reranked by statistical methods such as log-linear models, allowing human expert knowledge to be incorporated into the process. The upper-bound of these methods is the maximum achievable parsing performance from the N-best list, known as the *oracle score*. In general English phrase structure parsing, Charniak and Johnson (2005) found that their 50-best oracle F_1-measure is 96.8% on the PennTreebank corpus, which is substantially better than their system's actual performance of 91.0%, indicating potential for further improvement under this approach. We return to reranking in Chapter 4. In dynamic programming approaches, a large number of possible parse trees are represented compactly in a parse tree forest or chart, and the best possible tree is decoded from this representation. One example is the max-margin approach taken by Taskar et al. (2004), which casts parsing as a classification problem of separating the correct parse of a sentence from the other candidate parses. Henderson (2004) uses a left-corner history-based model, in which parsing is viewed as a series of decisions made in sequence, and training a parser is to learn a probability distribution of the next decision conditioned on the previous ones. Since the number of previous decisions in a parse is unbounded, a neural network-based approach is used to learn a finite representation of the previous decisions. The large number of features needed for these approaches means that tractability is typically an issue, so local features which only con-

sider information available at a particular production in the tree are used. More recent work by Huang (2008) introduces *forest reranking*, which allows usage of non-local features in addition to local features by an approximate decoding method. Non-local features are computed in a bottom-up way by reranking subtrees at internal nodes. The performance of this parser on the *Wall Street Journal* corpus is the highest so far for parsers that do not use additional training data, at an F_1 of 91.7%.

2.2 Topological Field Chunkers and Parsers

Existing work in identifying topological fields can be divided into chunkers, which identify the lowest-level non-recursive topological fields, and parsers, which also identify sentence and clausal structure.

Veenstra et al. (2002) compare three approaches to topological field chunking based on finite-state transducers, memory-based learning, and PCFGs respectively. It is found that the three techniques perform about equally well. The finite-state transducer approach provides a F_1 of 94.1% using POS tags from the TnT tagger, and 98.4% with gold tags, and the other approaches provide similar results. In other work by Liepert (2003), a topological field chunker is implemented using a multi-class extension to the canonically two-class support vector machine (SVM) classification framework. Parameters to the machine learning algorithm are fine-tuned by a genetic search algorithm, with a resulting F_1-measure of 92.25%. Training the parameters to SVM does not have a large effect on performance, increasing the F_1-measure in the test set by only 0.11%.

As for parsing, the corpus-based, stochastic topological field parser of Becker and Frank (2002) is based on a standard treebank PCFG model, in which rule probabilities are estimated by frequency counts. This model includes several enhancements, which are also found in the Berkeley parser. First, they use parameterized categories, splitting nonterminals according to linguistically based intuitions, such as splitting different clause types (they do not distin-

guish different clause types as basic categories, unlike TüBa-D/Z). Second, they take into account punctuation, which may help identify clause boundaries. They also binarize the very flat topological tree structures, and prune rules that only occur once. They test their parser on a version of the NEGRA corpus, which has been annotated with topological fields using a semi-automatic method.

Ule (2003) proposes a process termed *Directed Treebank Refinement* (DTR). The goal of DTR is to refine a corpus to improve parsing performance. DTR is comparable to the idea of latent variable grammars on which the Berkeley parser is based, in that both consider the observed treebank to be less than ideal and both attempt to refine it by splitting and merging nonterminals. In this work, nonterminals are split and merged by considering the nonterminals' contexts (i.e., their parent nodes) and the distribution of their productions. Unlike in the Berkeley parser, splitting and merging are distinct stages, rather than parts of a single iteration. Multiple splits are found first, then multiple rounds of merging are performed. No smoothing is done. As an evaluation, DTR is applied to topological field parsing of the TüBa-D/Z corpus. We discuss the performance of these topological field parsers in more detail in Chapter 4.

2.3 Other Computational Models of German Syntax

Although we focus on (P)CFG-based models in this work, German syntax and parsing have been studied using a variety of computational grammar formalisms. Here we briefly mention some of this work. Hockenmaier (2006) has translated the German TIGER corpus (Brants et al., 2002) into a CCG-based treebank to model word order variations in German. Foth et al. (2004) consider a version of dependency grammars known as *weighted constraint dependency grammars* for parsing German sentences. On the NEGRA corpus (Skut et al., 1998), they achieve an accuracy of 89.0% on parsing dependency edges. In Callmeier (2000), a platform for efficient HPSG parsing is developed. This parser is later extended by Frank et al. (2003) with a topological field parser for more efficient parsing of German. The system by Rohrer

and Forst (2006) produces LFG parses using a manually designed grammar and a stochastic parse disambiguation process. They test on the TIGER corpus and achieve an F_1-measure of 84.20%.

Topological field parsing concerns itself with the surface syntactic structure that can be parsed from a sentence using projective syntax trees, but we would also like a deeper level of representation that includes a more complete semantic interpretation of the tree structure. There exist models which maintain both syntactic projectivity and semantic interpretability, by creating one parse tree for each of these components for each sentence. So, one parse tree represents surface order, equivalent to the topological field parsing model described earlier, and requires projectivity. The other is responsible for representing semantic interpretation and dominance relations in the tree which correspond to linguistic realities. They may not require projectivity and may not be fully ordered.

Penn and Haji-Abdolhosseini (2003) provide one such formalism, which combines topological *phenogrammatical structures*, with semantically interpretable *tectogrammatical structures*. They also provide a simple parsing algorithm for this formalism.

Another such formalism is offered by Duchier and Debusmann (2001), using dependency trees to represent the two structures, instead of phrase structure trees. They distinguish *linear precedence* (LP) topological dependency trees, which are partially ordered and projective, from *immediate dominance* (ID) syntax trees, which are unordered and non-projective. In these dependency structures, the edges are labelled with either the grammatical function (for ID trees) as in typical typed dependency trees, or the topological field for LP trees. Unfortunately, parsing in this framework has been shown to be NP-complete (Koller and Striegnitz, 2002).

Chapter 3

Applications

We have seen in the last chapter that topological fields provide clause-level information about the structure of German sentences. Here, we motivate the utility of topological field parsing by examining their application to several NLP problems.

3.1 Part-of-Speech Tagging

In Müller and Ule (2002), part-of-speech tagging and topological field annotation are integrated into a single process, with each component aiding the other. The system begins by tagging an input sentence using the TnT trigram POS tagger (Brants, 2000). Each word is assigned a list of possible POS tags ranked in decreasing order of probability. Then, these tags are fed into a series of transducers which use hand-crafted finite-state grammars to annotate topological fields, embedded clauses, and NP chunks. If the transduction process does not result in a parse, then the next-best POS tag from the ranked list is used and the annotation process is restarted. If this results in a parse, the newly chosen POS tag is considered the correct tag.

This process was found to reduce POS tagging errors, primarily for verbs and complementizers. The tagset used in the experiment distinguishes between finite and infinite verbs, and between different kinds of subordinating complementizers, which can be morphologically identical in German. The clausal context would be able to select the proper category, because

recall that the position of the finite verb depends on the sentence type. The initial POS tagger, however, relies only on the local trigram context. The main benefit of this combined approach is that the topological field annotation process injects clausal context into the POS tagging process to allow proper disambiguation. Overall tagging error is reduced from 2.98% to 2.77%. Another benefit is an increase in parser coverage, though no quantitative evaluation is done to determine if the increased coverage comes at the price of decreased parsing accuracy.

The authors provide the following example (Example 3.1), where the verb *zustimmen* 'agree', occurs in the exact same local context, though it is infinite in 3.1(a), and finite in 3.1(b).

(3.1) (a) *Gestern wollten weder die Konservativen noch die Liberalen dem Antrag zustimmen.*
yesterday wanted neither the conservatives nor the liberals the motion accept-INFINITE
'Yesterday, neither the conservatives nor the liberals wanted to accept the motion.'

(b) *Kommentatoren erwarten, dass weder die Konservativen noch die Liberalen dem Antrag zustimmen.*
commentators expect that neither the conservatives nor the liberals the motion accept-FINITE
'Commentators expect that neither the conservatives nor the liberals will accept the motion.'

3.2 HPSG Parsing

Like other forms of shallow parsing, topological field parsing is useful as the first stage to further processing and eventual semantic analysis. In Frank et al. (2003), topological field information is used to guide parsing in the *head-drive phrase structure grammar* (HPSG) framework.

HPSG is a highly lexicalized grammar theory in which lexical items are feature structures that include detailed information about the feature structure's syntax, semantics, and form, including requirements on how this feature structure combines with other feature structures.

Parsing in HPSG consists of finding a way to combine feature structures such that these requirements are satisfied (through *unification*). From a computational perspective, the main problems in parsing HPSGs are parser efficiency and coverage rather than accuracy, since HPSG grammars are so detailed that any parse for a sentence is likely to be correct.

Also because of the highly lexicalized nature of HPSG, HPSG parsers typically take a bottom-up approach to parsing, where structures are hypothesized for an input sentence starting at the level of the lexical items. The main utility of topological fields is to provide top-down clausal information that can guide the parsing process. They can be useful because topological fields or field sequences often correspond to phrases in an HPSG parse. For example, the *Vorfeld* is an HPSG constituent, and the span from the left bracket containing the finite verb to the end of the clause is also a constituent in HPSG.

Using this knowledge available from the topological fields, they encode a set of soft constraints for the parsing algorithm. In particular, the potential HPSG constituents identified by the topological fields are used to affect the priority of tasks in the chart parsing algorithm used for parsing HPSG structures so that parsing decisions that are consistent with the boundaries of the potential constituent are preferred over those that contradict it.

Testing on the manually annotated test set of Becker and Frank (2002) and using the topological field parser in that work, HPSG parsing performance was sped up by a factor of about 2 with a coverage loss of less than 1%. The authors further show that using NP and PP chunks in a similar fashion do not lead to a speed increase, indicating that their hypothesis about the usefulness of topological fields because of their high-level nature is correct.

3.3 Anaphora Resolution

Anaphora resolution is the task of identifying pairs of linguistic expressions in which one (the anaphor) refers to the other (the antecedent). Strictly speaking, the antecedent must occur before the anaphor; if the order is reversed, the phenomenon is known as cataphora.

Anaphora can take a variety of forms including reflexive pronouns, non-reflexives pronouns, and full noun phrases. These different types of anaphora realization are subject to different syntactic constraints; for example, a regular pronoun cannot be the direct object of the clause in which its antecedent is the subject. (*He_i *saw* him_i, where the indices indicate that *He* and *him* refer to the same person.) In theoretical linguistics, these constraints form the basis of Binding Theory (Haegeman (1994) provides an introduction to the topic), and previous work in computational anaphora resolution made heavy use of syntactic parses using insights drawn from this theory (Hobbs, 1978; Lappin and Leass, 1994).

Becker and Pecourt (2002) present an approach to anaphora resolution which does not make use of full syntactic parses. Rather, the algorithm relies on a topological parser to provide the same type of information regarding the domain that different forms of anaphora can take.

Two levels of domains which can be retrieved from a topological parse are defined: the *local domain*, which is the immediately dominating finite clause, and the *mother domain*, which is the clause immediately dominating the local domain. Then, the following constraints are derived. First, personal pronouns and full noun phrases in the same local domain may not be coreferential. Second, antecedents of reflexive pronouns must be in the same local domain. Finally, the antecedent of relative pronouns must be outside of the local domain, in the mother domain.

The interaction between these three constraints for anaphora resolution can be seen in example 3.2. The reflexive *sich* resolves to *sie* in the same local domain, and the relative pronoun *die* resolves to *die Fragen*. Thus, the personal pronoun *sie* must resolve to *Die Studenten*, since it may not resolve to the same antecedent as the relative pronoun which is in the same local domain, as that would render the two coreferential.

(3.2) *[Die Studenten]*$_1$ *formulierten [die Fragen]*$_2$*, [die]*$_2$ *[sie]*$_1$ *[sich]*$_1$ *gestellt hatten.*
the students formulated the questions, that they themselves asked had.
'The students formulated the questions which they had asked themselves'

Unfortunately, this work lacks a quantitative analysis, so it is unclear how large a contribution topological field information made.

3.4 Summary

In this chapter, we have seen three tasks which benefit from topological field information: part-of-speech tagging, HPSG parsing, and anaphora resolution. The common thread in how topological fields aid in these tasks is that they provide high-level information which supplements the available local information. In POS tagging, they provide clausal context to disambiguate the finiteness of verbs, and the type of complementizer; in HPSG parsing, they provide top-down information to guide a chart parser to select likely edges to expand; and in anaphora resolution, they demarcate relevant boundaries for constraints on different kinds of anaphora.

Another area where topological fields may prove to be useful is computational discourse. Topic-focus ordering in German is known as correlate with topological field structure (Frey, 2004b). One fruitful area of future research would be to explore if topological information could be helpful for identifying sentential topics and hence aid in modelling the information structure of a passage.

Chapter 4

Parsing Experiments[1]

We have seen that topological fields are a useful model of German clausal syntax which can be handled by a phrase structure parser. In this chapter, we describe our experiments to parse the TüBa-D/Z and NEGRA newspaper corpora using an unlexicalized latent-variable parser, and show that we achieve results that are better than the previous state-of-the-art with minimum domain-specific adaption. We also provide an in-depth analysis of the results and show that systematic exceptions to the topological field model including elliptical and parenthetical constructions are the main source of remaining errors.

4.1 A Latent-Variable Parser

For our experiments, we used the *latent-variable*-based Berkeley parser (Petrov et al., 2006). Latent-variable parsing assumes that an observed treebank represents a coarse approximation of an underlying, optimally refined grammar which makes more fine-grained distinctions in the syntactic categories. For example, the noun phrase category *NP* in a treebank could be viewed as a coarse approximation of two noun phrase categories corresponding to subject and object, *NP^S*, and *NP^VP*.

[1] Parts of this chapter and the next chapter have been previously published as (Cheung and Penn, 2009).

The Berkeley parser automates the process of finding such distinctions. It starts with a simple X-bar grammar style backbone (that is, binarized with intermediate 'bar' levels), and goes through iterations of splitting and merging nonterminals, in order to maximize the likelihood of the training set treebank. In the splitting stage, an Expectation-Maximization algorithm is used to find a good split for each nonterminal. In the merging stage, categories that have been oversplit are merged together to keep the grammar size tractable and reduce sparsity. Finally, a smoothing stage occurs, where the probabilities of rules for each nonterminal are smoothed toward the probabilities of the other nonterminals split from the same syntactic category.

The Berkeley parser has been applied to the TüBaD/Z corpus in the constituent parsing shared task of the ACL-2008 Workshop on Parsing German (Petrov and Klein, 2008), achieving an F_1-measure of 85.10% and 83.18% with and without gold standard POS tags respectively. This evaluation considered all nodes, not just topological fields, and considered grammatical functions as well as the syntactic category. We chose the Berkeley parser for topological field parsing because it is known to be robust across languages, and because it is an unlexicalized parser. Lexicalization has been shown to be useful in more general parsing applications due to lexical dependencies in constituent parsing (e.g. (Kübler et al., 2006; Dubey and Keller, 2003) in the case of German). However, topological fields explain a higher level of structure pertaining to clause-level word order, and we hypothesize that lexicalization is unlikely to be helpful. Furthermore, lexicalized parsing models require a notion of headedness for each constituent, which may be difficult to define for topological fields like the *Mittelfeld* which do not form a semantically coherent constituent[2].

4.2 Data

For our experiments, we primarily used the TüBa-D/Z (Tübinger Baumbank des Deutschen / Schriftsprache) corpus, consisting of 26116 sentences (20894 training, 2611 development,

[2]Thanks to Christopher Manning for bringing this point to our attention.

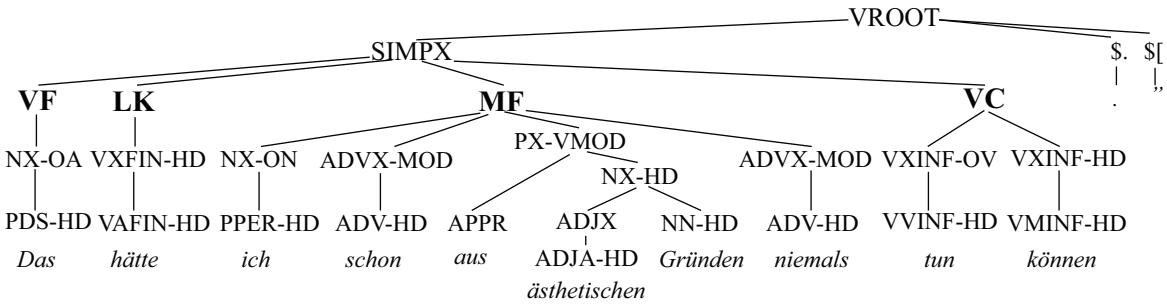


Figure 4.1: "I could never have done that just for aesthetic reasons." Sample TüBa-D/Z tree with topological field annotations. Edge labels appear after a node label, separated by a hyphen. Topological field layer in bold.

2089 test, with a further 522 sentences held out for future experiments)[3] taken from the German newspaper *die tageszeitung*. The corpus consists of four levels of annotation: clausal, topological, phrasal (other than clausal), and lexical. The annotation scheme is based on a context-free backbone, contains no traces or empty categories, and encodes grammatical functions. See Figure 4.1 for an example of a tree in the TüBa-D/Z corpus. We define the task of topological field parsing to be recovering the first two levels of annotation, following Ule (2003).

We also tested the parser on a version of the NEGRA corpus derived by Becker and Frank (2002), in which syntax trees have been made projective and topological fields have been automatically added through a series of linguistically informed tree modifications. All internal phrasal structure nodes have also been removed. The corpus consists of 20596 sentences, which we split into subsets of the same size as described by Becker and Frank (2002)[4]. The set of topological fields in this corpus differs slightly from the one used in TüBa-D/Z, making no dis-

[3]These are the same splits into training, development, and test sets as in the ACL-08 Parsing German workshop. This corpus does not include sentences of length greater than 40.

[4]16476 training sentences, 1000 development, 1058 testing, and 2062 as held-out data. We were unable to obtain the exact subsets used by Becker and Frank (2002). We will discuss the ramifications of this on our evaluation procedure.

Gold tags	*Edge labels*	*LP%*	*LR%*	F_1%	$\overline{CB}$	*CB0%*	$CB \leq 2\%$	*EXACT%*
–	–	93.53	93.17	93.35	0.08	94.59	99.43	79.50
+	–	**95.26**	**95.04**	**95.15**	**0.07**	**95.35**	**99.52**	**83.86**
–	+	92.38	92.67	92.52	0.11	92.82	99.19	77.79
+	+	92.36	92.60	92.48	0.11	92.82	99.19	77.64

Table 4.1: Parsing results for topological fields and clausal constituents in the TüBa-D/Z corpus. CB refers to cross-bracketing. Best results in bold.

tinction between clause types, nor consistently marking field or clause conjunctions. Because of the automatic annotation of topological fields, this corpus contains numerous annotation errors. Becker and Frank (2002) manually corrected their test set and evaluated the automatic annotation process, reporting labelled precision and recall of 93.0% and 93.6% compared to their manual annotations. There are also punctuation-related errors, including missing punctuation, sentences ending in commas, and sentences composed of single punctuation marks. We test on this data in order to provide a better comparison with previous work. Although we could have trained the model in Becker and Frank (2002) on the TüBa-D/Z corpus, it would not have been a fair comparison, as the parser depends quite heavily on NEGRA's annotation scheme. For example, TüBa-D/Z does not contain an equivalent of the modified NEGRA's parameterized categories; there exist edge labels in TüBaD/Z, but they are used to mark head-dependency relationships, not subtypes of syntactic categories.

4.3 Results

We first report the results of our experiments on the TüBa-D/Z corpus. For the TüBa-D/Z corpus, we trained the Berkeley parser using the default parameter settings. The grammar trainer attempts six iterations of splitting, merging, and smoothing before returning the final

grammar. Intermediate grammars after each step are also saved. There were training and test sentences without clausal constituents or topological fields, which were ignored by the parser and by the evaluation. As part of our experiment design, we investigated the effect of providing gold POS tags to the parser, and the effect of incorporating edge labels into the nonterminal labels for training and parsing. In all cases, gold annotations which include gold POS tags were used when training the parser.

We report the standard PARSEVAL measures of parser performance in Table 4.1, obtained by the `evalb` program by Satoshi Sekine and Michael Collins. This table shows the results after five iterations of grammar modification, parameterized over whether we provide gold POS tags for parsing, and edge labels for training and parsing. The number of iterations was determined by experiments on the development set. In the evaluation, we do not consider edge labels in determining correctness, but do consider punctuation, as Ule (2003) did. If we ignore punctuation in our evaluation, we obtain an F_1-measure of 95.42% on the best model ($+$ *Gold tags*, $-$ *Edge labels*).

Whether supplying gold POS tags improves performance depends on whether edge labels are considered in the grammar. Without edge labels, gold POS tags improve performance by almost two points, corresponding to a relative error reduction of 33%. In contrast, performance is negatively affected when edge labels are used and gold POS tags are supplied (i.e., $+$ *Gold tags*, $+$ *Edge labels*), making the performance *worse* than not supplying gold tags. Incorporating edge label information does not appear to improve performance, possibly because it oversplits the initial treebank and interferes with the parser's ability to determine optimal splits for refining the grammar.

To facilitate a more direct comparison with previous work, we also performed experiments on the modified NEGRA corpus. In this corpus, topological fields are *parameterized*, meaning that they are labelled with further syntactic and semantic information. For example, VF is split into VF-REL for relative clauses, and VF-TOPIC for those containing topics in a verb-second sentence, among others. All productions in the corpus have also been binarized. Tuning the

Parser	*LP%*	*LR%*	F_1%
TüBa-D/Z			
This work	95.26	95.04	95.15
Ule	unknown	unknown	91.98
NEGRA—from Becker and Frank (2002)			
BF02 (len. $\leq$ 40)	92.1	91.6	91.8
NEGRA—our experiments			
This work (len. $\leq$ 40)	90.74	90.87	90.81
BF02 (len. $\leq$ 40)	89.54	88.14	88.83
This work (all)	90.29	90.51	90.40
BF02 (all)	89.07	87.80	88.43

Table 4.2: BF02 = (Becker and Frank, 2002). Parsing results for topological fields and clausal constituents. Results from Ule (2003) and our results were obtained using different training and test sets. The first row of results of Becker and Frank (2002) are from that paper; the rest were obtained by our own experiments using that parser. All results consider punctuation in evaluation.

parameter settings on the development set, we found that parameterized categories, binarization, and including punctuation gave the best F_1 performance. First-order horizontal and zeroth order vertical markovization after six iterations of splitting, merging, and smoothing gave the best F_1 result of 91.78%. We parsed the corpus with both the Berkeley parser and the best performing model of Becker and Frank (2002).

The results of these experiments on the test set for sentences of length 40 or less and for all sentences are shown in Table 4.2. We also show other results from previous work for reference. We find that we achieve results that are better than the model in Becker and Frank (2002) on the test set. The difference is statistically significant (p = 0.0029, Wilcoxon signed-rank).

The results we obtain using the parser of Becker and Frank (2002) are worse than the results described in that paper. We suggest the following reasons for this discrepancy. While the test set used in the paper was manually corrected for evaluation, we did not correct our test set, because it would be difficult to ensure that we adhered to the same correction guidelines. No details of the correction process were provided in the paper, and descriptive grammars of German provide insufficient guidance on many of the examples in NEGRA on issues such as ellipses, short infinitival clauses, and expanded participial constructions modifying nouns. Also, because we could not obtain the exact sets used for training, development, and testing, we had to recreate the sets by randomly splitting the corpus.

Comparing across the two corpora, there also is a difference in performance of the parser on TüBa-D/Z and on NEGRA. Aside from a slightly different set of topological and clausal categories, the principal difference between the two corpora is that TüBa-D/Z contains internal phrasal annotation, while NEGRA does not. Our results indicate that internal annotation can improve parsing results on topological fields and clause labels, which complements the result of Kübler (2005) that parsing topological field labels also improves the parsing performance of internal phrasal structure.

4.4 Analysis

While PARSEVAL measures provide a good aggregate picture of the performance of a parser, we also need a more in-depth error analysis in order to identify remaining errors made by the parser. We now return to the TüBa-D/Z corpus for a series of such analyses.

4.4.1 Category-Specific Results

We first examine the category-specific results for our best performing model (+ *Gold tags*, − *Edge labels*). Overall, Table 4.3 shows that the best performing topological field categories are those that have constraints on the type of word that is allowed to fill it (finite verbs in

Topological Fields				
Category	#	*LP%*	*LR%*	F_1%
PARORD	20	100.00	100.00	100.00
VCE	3	100.00	100.00	100.00
LK	2186	99.68	99.82	99.75
C	642	99.53	98.44	98.98
VC	1777	98.98	98.14	98.56
VF	2044	96.84	97.55	97.20
KOORD	99	96.91	94.95	95.92
MF	2931	94.80	95.19	94.99
NF	643	83.52	81.96	82.73
FKOORD	156	75.16	73.72	74.43
LV	17	10.00	5.88	7.41
Clausal Constituents				
Category	#	*LP%*	*LR%*	F_1%
SIMPX	2839	92.46	91.97	92.21
RSIMPX	225	91.23	92.44	91.83
PSIMPX	6	100.00	66.67	80.00
DM	28	59.26	57.14	58.18

Table 4.3: Category-specific results using grammar with no edge labels and using gold POS tags

Problem	*Freq.*
Misidentification of parentheticals	19
Coordination problems	13
Too few SIMPX	10
Paired punctuation problem	9
Other clause boundary errors	7
Other	6
Too many SIMPX	3
Clause type misidentification	2
MF/NF boundary	2
LV	2
VF/MF boundary	2

Table 4.4: Types and frequency of parser errors in the fifty worst scoring parses by F_1-measure, using parameters ($+$ *Gold tags*, $-$ *Edge labels*)

LK, verbs in VC, complementizers and subordinating conjunctions in C). VF, in which only one constituent may appear, also performs relatively well. Topological fields that can contain a variable number of heterogeneous constituents, on the other hand, have poorer F_1-measure results. MF, which is basically defined relative to the positions of fields on either side of it, is parsed several points below LK, C, and VC in accuracy. NF, which contains different kinds of extraposed elements, is parsed at a substantially worse level.

Poorly parsed categories tend to occur infrequently, including LV, which marks a rare resumptive construction; FKOORD, which marks topological field coordination; and the discourse marker DM. The other clause-level constituents (PSIMPX for clauses in paratactic constructions, RSIMPX for relative clauses, and SIMPX for other clauses) also perform below average.

4.4.2 Qualitative Error Analysis

As a further analysis, we extracted the worst scoring fifty sentences by F_1-measure from the parsed test set ($+$ *Gold tags,* $-$ *Edge labels*), and compared them against the gold standard trees, noting the cause of the error. The major mistakes made by the parser are summarized in Table 4.4.

Misidentification of Parentheticals Parenthetical constructions do not have any dependencies on the rest of the sentence, and exist as a mostly syntactically independent clause inside another sentence. They can occur at the beginning, end, or in the middle of sentences, and are often set off orthographically by punctuation. The parser has problems identifying parenthetical constructions, often positing a parenthetical construction when that constituent is actually attached to a topological field in a neighbouring clause. The following example shows one such misidentification in bracket notation. Clause-internal topological fields are omitted for clarity.

(4.1) (a) TüBa-D/Z: (SIMPX *Weder das Ausmaß der Schönheit noch der frühere oder spätere Zeitpunkt der Geburt macht einen der Zwillinge für eine Mutter mehr oder weniger echt / authentisch / überlegen*).

(b) Parser: (SIMPX *Weder das Ausmaß der Schönheit noch der frühere oder spätere Zeitpunkt der Geburt macht einen der Zwillinge für eine Mutter mehr oder weniger echt*) **(PARENTHETICAL / *authentisch / überlegen.*)**

(c) Translation: “Neither the degree of beauty nor the earlier or later time of birth makes one of the twins any more or less real/authentic/superior to a mother.”

We hypothesized earlier that lexicalization is unlikely to give us much improvement in performance, because topological fields work on a domain that is higher than that of lexical dependencies such as subcategorization frames. However, given the locally independent nature of legitimate parentheticals, a limited form of lexicalization or some other form of stronger contextual information might be needed to improve identification performance.

Coordination Problems The second most common type of error involves field and clause coordinations. This category includes missing or incorrect FKOORD fields, and conjunctions of clauses that are misidentified. In the following example, the conjoined MFs and following NF in the correct parse tree are identified as a single long MF.

(4.2) (a) TüBa-D/Z: *Auf dem europäischen Kontinent aber hat (FKOORD* ***(MF kein Land und keine Macht ein derartiges Interesse an guten Beziehungen zu Rußland)*** *und* ***(MF auch kein Land solche Erfahrungen im Umgang mit Rußland))*** *(NF wie Deutschland).*

(b) Parser: *Auf dem europäischen Kontinent aber hat* ***(MF kein Land und keine Macht ein derartiges Interesse an guten Beziehungen zu Rußland und auch kein Land solche Erfahrungen im Umgang mit Rußland wie Deutschland)***.

(c) Translation: "On the European continent, however, no land and no power has such an interest in good relations with Russia (as Germany), and also no land (has) such experience in dealing with Russia as Germany."

Other Clause Errors Other clause-level errors include the parser predicting too few or too many clauses, or misidentifying the clause type. Clauses are sometimes confused with NFs, and there is one case of a relative clause being misidentified as a main clause with an intransitive verb, as the finite verb appears at the end of the clause in both cases. Some clause errors are tied to incorrect treatment of elliptical constructions, in which an element that is inferable from context is missing.

Paired Punctuation Problems with paired punctuation are the fourth most common type of error. Punctuation is often a marker of clause or phrase boundaries. Thus, predicting paired punctuation incorrectly can lead to incorrect parses.

Other Issues Other incorrect parses generated by the parser include problems with the infrequently occurring topological fields like LV and DM, inability to determine the boundary

between MF and NF in clauses without a VC field separating the two, and misidentifying appositive constructions. Another issue is that although the parser output may disagree with the gold standard tree in TüBa-D/Z, the parser output may be a well-formed topological field parse for the same sentence with a different interpretation, for example because of attachment ambiguity. Two judges including the author independently checked the fifty worst-scoring parses, and determined whether each parse produced by the Berkeley parser could be a well-formed topological parse. Where there was disagreement, the judges discussed their judgments until they came to a consensus. Of the fifty parses, it was determined that nine, or 18%, could be legitimate parses. Another five, or 10%, differ from the gold standard parse only in the placement of punctuation. Thus, the F_1-measures we presented above may be underestimating the parser's performance.

4.4.3 Results on Subsets

Another method of examining the results of the parser is to remove suspected problematic cases—in this domain, sequences of topological fields which do not follow the canonical model described in section 1.1—and determine whether performance improves on the remainder. If performance is substantially improved, then the filtered-out constructions may be considered unexplained variance from the standard topological field model, which is currently not well accounted for. We now define and motivate four levels of restrictions on the topological field sequences in the TüBa-D/Z corpus. They are presented in increasing order of restrictiveness.

Model A: Full Model This model consists of the full topological field annotations found in TüBa-D/Z without any restrictions.

Model B: Clauses with Finite Verbs and Infinitival Clauses In this model, all clauses must be headed by a main verb, either a finite one in the LK or VC field, or an infinitival one in infinitival clauses. Excluded are sentences like elliptical constructions where a verb is omitted.

Model C: Only Clauses with Finite Verbs This model also disallows infinitival clauses. Infinitival clauses do not strictly follow the topological field model, because although they are assigned a clause-label by TüBa-D/Z's annotation scheme, no finite verb heads the clause, and no complementizer or C field is found, even though they are verb-last constructions.

Model D: "Textbook" Model This model is the most restrictive. Only sentences which meet the following criteria are permitted:

- Clauses follow either the C (MF) VC (NF) verb-last pattern of topological fields, or the (VF) LK (MF) (VC) (NF) pattern for verb-initial and verb-second clauses.
- Sentence contains at least one clause (this eliminates headlines and meta-data).
- Sentence contains exactly one clause label at the top level, with nothing else other than punctuation (this weeds out more headlines and parenthetical constructions at the top level).
- The clause node SIMPX must be a child of the top level, or NF, or VF (this weeds out more parenthetical constructions).

To create a parser based on each model, we filtered out sentences in the training set that do not conform to the restrictions imposed by each model, then trained the Berkeley parser on the remaining sentences. Similarly, we created four test sets by filtering out sentences that do not fit the model. We tested the four grammars on each of the test sets (Table 4.5).

In general, restricting the power of the model leads to an improvement in parsing performance on the corresponding test set. In the full test set (Test Set A), the original model gives the best parsing performance. As we increase the level of restrictiveness of the test set, however, this is no longer the case. Test Set B is about equally well parsed by Models A and B, with the difference not being statistically significant (two-tailed Wilcoxon signed-rank $p > 0.99$). For Test Set C, Models A and B still tie for best performance up to statistical significance ($p > 0.9$), and interestingly, they outperform Model C itself ($p = 0.013$). Finally, in the most restrictive

Test Set \ *Model*	*Training Set A* M = 20894	*Training Set B* M = 19326	*Training Set C* M = 17922	*Training Set D* M = 10949
Test Set A	95.26/95.04	94.07/93.71	91.82/89.49	81.49/82.78
N = 2089	***95.15***	*93.89*	*90.64*	*82.13*
Test Set B	95.78/96.08	95.87/96.08	93.98/91.95	82.53/85.54
N = 1945	*95.93*	***95.98***	*92.95*	*84.01*
Test Set C	95.97/96.41	96.08/96.41	95.65/95.85	81.96/87.31
N = 1802	*96.19*	***96.24***	*95.75*	*84.55*
Test Set D	97.99/97.90	97.61/97.54	96.31/95.68	98.43/98.42
N = 1076	*97.95*	*97.58*	*96.00*	***98.42***

Table 4.5: Labelled constituency results on subsets of the test set, using parameters (+ *Gold tags*, − *Edge labels*), after five iterations of splitting and merging. First row shows labelled precision/recall, second row in italics shows F_1 (in %). M is training set size, N is test set size.

Test Set D, Model D outperforms the other models ($p < 0.03$ for each of the models to Model D).

These results support the conclusion that non-canonical sequences of topological fields present in the full test set are not well accounted for by the current topological field model. As we remove constructions such as parentheticals and ellipses, the F_1 of the best model increases from 95.15% from Test Set A to 98.42% for Test Set D. Also, the significance tests show that the restrictive models we designed, especially the textbook model, are meaningful subsets of the original corpus. These results corroborate the findings of the analysis of the most frequent error types in the fifty worst parses in section 4.4.2.

4.4.4 Merging Topological Field Labels

In a similar vein, we can also determine whether the parser is consistently confusing topological field labels by performing an experiment where we merge labels in the training set and in the evaluation. For example, we can determine how widespread confusion between MF and NF is. We define the following three models, decreasing in their level of discrimination of topological field labels:

Full Model This is again the original TüBa-D/Z model of topological fields.

VF/MF/NF Merged In this model, the three topological fields which can contain a heterogeneous mix of constituents, VF, MF, and NF, are merged into a single TOPONODE.

All Merged In this model, all topological field labels are renamed TOPONODE.

Once again, we trained grammars using the Berkeley parser after merging these labels in the training set, and performed an evaluation using the same set of topological fields that the model was trained on.

We see that performance does not improve when we merge topological field labels in training and testing. In fact, the full model is statistically significantly better than the unlabelled model ($p = 0.0251$). These results show that topological field distinctions are useful to the parser, and that it is the identification of topological field boundaries, rather than identification of the field labels, that is the leading source of remaining errors.

Test Set	**Full Model**	**VF/MF/NF Merged**	**All Merged**
LP/LR	95.26/95.04	94.98/95.12	94.67/94.58
F_1	95.15	95.05	94.63

Table 4.6: Parsing results after selectively merging labels in the test set by labelled precision, recall, and F_1-measure (in %), using parameters ($+$ *Gold tags*, $-$ *Edge labels*), after five iterations of splitting and merging

Chapter 5

Improving Topological Field Parsing

While we have achieved good parsing results from the Berkeley parser in the previous chapter, we have also identified some of the remaining types of errors in the parse output, indicating that further performance gains are possible. In this chapter, we explore two approaches to this end—reranking, and self-training.

5.1 Reranking for Paired Punctuation

As a general method, reranking has become popular in a variety of NLP problems outside of parsing, including machine translation and sentence boundary detection (Shen et al., 2004; Roark et al., 2006). We have also presented some work on reranking in section 2.1.2 which uses a discriminative statistical model together with a large number of features (for example, 1.3M in Charniak and Johnson (2005)) to improve parsing results.

Here, we take a far simpler approach of reranking based on one feature using a hard constraint. The main result of this section is not primarily the improvement in parsing performance that this simplistic procedure produces, but rather it is to show that reranking is a promising area for further research.

The feature over which we perform reranking concerns punctuation which occurs in pairs, such as quotation marks, parentheses, and brackets. While experimenting with the development

Set	*N*	*LP%*	*LR%*	F_1%	$\overline{CB}$	*CB0%*	$CB \leq 2\%$	*EXACT%*
All Unreranked	2089	95.26	95.04	95.15	0.07	95.35	99.52	83.86
All Reranked	2089	95.39	95.09	95.24	0.07	95.55	99.57	83.92
Problematic Unreranked	38	81.56	82.84	82.19	0.61	60.53	94.74	28.95
Problematic Reranked	38	85.35	84.20	84.77	0.39	71.05	97.37	31.58

Table 5.1: Effect of constrained reranking, using gold tags and without edge labels. Problematic here refers to sentences where paired punctuation does not occur in the same clause in the initial parser output.

set of TüBa-D/Z, we noticed that the parser sometimes returns parses in which paired punctuation is not placed in the same clause—a linguistically implausible situation. In these cases, the high-level information provided by the paired punctuation is overridden by the overall likelihood of the parse tree. To rectify this problem, we performed a simple post-hoc reranking of the 50-best parses produced by the best parameter settings ($+$ *Gold tags*, $-$ *Edge labels*), selecting the first parse that places paired punctuation in the same clause, or returning the best parse if none of the 50 parses satisfy the constraint. This procedure improved the F_1-measure to 95.24% (LP = 95.39%, LR = 95.09%).

Overall, 38 sentences were parsed with paired punctuation in different clauses, of which 16 were reranked. The reranked results for both the entire test set as well as the 38 sentences are shown in Table 5.1. Of the 38 sentences, reranking improved performance in 12 sentences, did not affect performance in 23 sentences (of which 10 already had a perfect parse), and hurt performance in three sentences. A two-tailed sign test suggests that reranking improves performance ($p = 0.0352$).

Example 5.1 illustrates how reranking can improve performance. Here, the parser's best parse predicts a spurious SIMPX clause spanning the text of the entire sentence, but this causes the second pair of quotation marks to be parsed as belonging to two different clauses. The

parser also predicts an incorrect LV field. Using the paired punctuation constraint, our reranking procedure was able to correct these errors.

(5.1) (a) *“ Auch (SIMPX wenn der Krieg heute ein Mobilisierungsfaktor ist) ” , so Pau , “ (SIMPX die Leute sehen , daß man für die Arbeit wieder auf die Straße gehen muß) . ”*

(b) Parser: ***(SIMPX** “ **(LV** Auch (SIMPX wenn der Krieg heute ein Mobilisierungsfaktor ist**))** ” , so Pau , “ (SIMPX die Leute sehen , daß man für die Arbeit wieder auf die Straße gehen muß**))** . ”*

(c) Translation: “Even if the war is a factor for mobilization,” said Pau, “the people see, that one must go to the street for employment again.”

Surprisingly, there are cases in which paired punctuation does not belong inside the same clause in the gold parses. These cases are either extended quotations, in which each of the quotation mark pair occurs in a different sentence altogether, or cases where the second of the quotation mark pair must be positioned outside of other sentence-final punctuation due to orthographic conventions. Sentence-final punctuation is typically placed outside a clause in this version of TüBa-D/Z.

More work can be done to find and motivate features for reranking, and to employ a reranking method more sophisticated than using a hard constraint. For example, more features could be designed based on the qualitative error analysis done in the previous section to deal with problematic constructions like parentheticals and ellipses. To investigate the upper-bound in performance that this type of N-best reranking is able to achieve, we present some statistics on our ($+$ *Gold tags*, $-$ *Edge labels*) 50-best list. We found that the average rank of the best scoring parse by F_1-measure is 2.61, and the perfect parse is present for 1649 of the 2088 sentences at an average rank of 1.90. The oracle F_1-measure is 98.12%, indicating that a more comprehensive reranking procedure might allow further performance gains. Table 5.2 shows more statistics on the characteristics of the best parse in terms of F_1 in the 50-best list.

N	2088
Oracle Labelled Recall	98.01
Oracle Labelled Precision	98.23
Oracle Labelled F_1	98.12
Mean best rank	2.61
Median best rank	1
Standard dev. of best rank	6.29
# Perfect rerank possible	1649
# Best rank is not the first parse	278
Mean best rank when not the first parse	13.13
Median best rank when not the first parse	7
Standard dev. when best is not the first parse	13.04

Table 5.2: Oracle statistics for 50-best list of TüBa-D/Z test set

5.2 Self-Training

Another area worth exploring for improving parsing performance is to utilize external resources for training outside of the corpus from which the training and test sets are drawn. Currently, the best parser performance on the standard test section of the *Wall Street Journal* treebank in English uses the method of *self-training* to make use of more training data (McClosky et al., 2006). Self-training is a kind of semi-supervised method in which a learning algorithm is trained on its own output. Basically, the parser is first trained on an initial training set, is then used to parse a new unlabelled data set, and then retrained on the original training set together with the recently parsed, additional data set. In McClosky et al. (2006), the standard training sections of the WSJ ($\sim$40k sentences in total including training and test sections) is used as the initial training set, and a section of the much larger, but unlabelled, North American News

Text corpus (Graff, 1995) is used as the additional data set (2M sentences from the *LA Times* section). Using the Charniak parser with reranking (Charniak and Johnson, 2005), labelled F_1 performance on the test set improves from 91.3% without self-training to 92.1%.

A related method is that of *co-training*, where multiple "views" of a problem cooperate to produce annotations for unlabelled data, which can then be used as training data. Let us illustrate with the procedure described by Sarkar (2001). In this work, a small set of labelled data is used to train two "views" into the parsing problem. The first (called H1) is a supertagging model that assigns elementary trees in the *Lexicalized Tree-Adjoining Grammar* (LTAG) formalism to input words. The second (called H2) is a model that creates attachments between elementary trees to produce a parse for the sentence.

Co-training proceeds as follows. First, H1 and H2 are trained on the labelled data set. Then, a small portion of the unlabelled data set is processed through H1 and H2. The most probable *n* sentences of this portion that H1 and H2 annotate are added to the labelled set, and the process iterates. After each iteration, *n* increases, in effect relaxing the confidence threshold necessary for a parse to be added to the labelled set. After all the unlabelled data is exhausted, the combined originally labelled and newly labelled set can be used to train a final parser. On *WSJ*, this procedure improves parsing performance from 70.6% F_1 before co-training to 79.8%.

While these semi-supervised methods are able to reduce reliance on labelled data, they can also easily magnify errors in the parser's model. McClosky et al. (2006) find that self-training does not work well in conjunction with the base generative parser alone, and only improves the performance of the reranking parser. Sarkar (2001)'s improvement in performance can be attributed to the small initial labelled training set that he uses (9695 sentences). Our goal then is to expand the availability of training data without compromising on the data quality.

We overcome this problem by using information from another labelled treebank, NEGRA. As mentioned in Chapter 4, we have obtained a version of NEGRA that has been made projective and semi-automatically annotated with topological fields. After parsing NEGRA with our initial parser trained on TüBa-D/Z data, we can then use the NEGRA annotations as a guide

From NEGRA	From TüBa-D/Z
CL	SIMPX, RSIMPX, PSIMPX
LK	C
VF	
MF	MF, MFE
RK	VC VCE
NF	
DF	DM, LV

Table 5.3: Equivalence classes between the TüBa-D/Z and NEGRA annotation schemes for topological fields and clause labels. Each row defines one equivalence class.

to control the quality of the parses by filtering out parses where the parse and the annotations disagree.[1]

We use the following filtering mechanism. First, we define a set of equivalence classes over the sets of topological fields and clausal labels found in the two corpora in order to enable a comparison over the different annotation schemes used (Table 5.3). Then, we require that the topological field and clause nodes in the NEGRA annotation and the initial parse made by the model trained on TüBa-D/Z agree in terms of their start, end, and label according to the equivalence classes defined. Any mismatch results in the parse being excluded from the self-training training set.

We tested four versions of the self-training procedure by varying two binary parameters: whether we use filtering or not, and whether we used all of NEGRA, or one quarter of the sentences randomly drawn from the full set. Of the 16476 sentences in the full NEGRA, 6488 passed the filtering mechanism. Of the one quarter subsection, 1621 sentences of 4119 passed

[1]The other logical possibility is to use TüBa-D/Z as a guide to parsing NEGRA. However, as we noted earlier, NEGRA's topological field annotations were done semi-automatically and contain noise and errors, and are thus not as reliable an evaluation standard.

Proportion Added	*Filtering*	*LP%*	*LR%*	F_1%	$\overline{CB}$	*CB0%*	$CB \leq 2\%$	*EXACT%*
No self-training		95.26	95.04	95.15	**0.07**	**95.35**	**99.52**	83.86
1/4	–	94.99	94.99	94.99	0.08	94.78	99.14	83.53
1/4	+	95.32	**95.21**	95.26	**0.07**	95.07	99.43	83.63
Full	–	95.26	95.00	95.13	0.08	94.59	99.23	83.87
Full	+	**95.43**	95.18	**95.31**	0.08	94.83	99.43	**84.16**

Table 5.4: Parsing results for topological fields and clausal constituents in the TüBa-D/Z corpus's test set after self-training. Best results in bold.

filtering. The results of self-training after incorporating these sentences are presented in Table 5.4. All results use gold POS tags, no edge labels, and are based on the Berkeley parser after five iterations of splitting and merging.

Although the differences in parsing performance are not significant between the four models using self-training and the original without self-training, we can comment on some trends in the results. These results seem to confirm the finding by McClosky et al. (2006) that self-training by itself is not useful for improving performance over a baseline generative parser, reducing F_1 from 95.15% to 95.13% using the full NEGRA corpus. On the other hand, performance improves slightly to 95.31% if only filtered sentences are added to the training set. These results suggest that some source of information outside of the generative parsing process is necessary for self-training to be successful. This could be a reranker, which considers additional features that can be designed by human experts, or filtering, which acts as a kind of quality control mechanism for the new trees to be added to the training set.

Much work remains to be done in working out the specific filtering settings that would maximize performance gains, as well as in determining a more sophisticated weighting system that would better reflect the higher confidence in the TüBa-D/Z training sentences over the parsed sentences from NEGRA.

Chapter 6

Conclusions

In this paper, we examined applying the latent-variable Berkeley parser to the task of topological field parsing of German, which aims to identify the high-level surface structure of sentences. Without any language or model-dependent adaptation, we obtained results which compare favourably to previous work in topological field parsing. We further examined the results of doing a simple reranking process, constraining the output parse to put paired punctuation in the same clause. We also considered self-training as a method of gathering more training data, using a version of NEGRA with topological field annotation as a filter to assure quality of the additional parses that we are feeding back into the parser. Finally, we considered some applications of topological fields.

The following is a summary of the contributions of this paper in more detail with comments on areas of future work.

The Berkeley parser for parsing topological fields We have shown that the Berkeley parser is a good generative model of parsing German topological field and clause structure. The parser performs extremely well in identifying the traditional left and right brackets of the topological field model; that is, the fields C, LK, and VC. The parser achieves basically perfect results on these fields in the TüBa-D/Z corpus, with F_1-measure scores for each at over 98.5%. These scores are higher than previous work in the simpler task of topological field chunking.

Noise in the TüBa-D/Z corpus Through a qualitative analysis of the worst parsed sentences and through experiments on subsets of the corpus, we have found that infrequently occurring topological fields and constructions are poorly parsed and not well accounted for by the canonical topological field model. Parenthetical constructions and ellipses are of particular concern.

Reranking and self-training While we have explored simple reranking and self-training methods which have given small accuracy improvements, much more work can be done to refine the proposals in this paper. A more comprehensive discriminative reranking of the parser output would be able to incorporate more contextual information; for example, it would be able to include features dealing with parentheticals and ellipses, which are problematic as we have discussed. Also, a more sophisticated self-training process utilizing other German corpora could provide a valuable source of additional training data.

Applications We have looked at three applications of topological fields in the literature: part-of-speech tagging, deep parsing, and anaphora resolution. The utility of topological fields in these applications is due to their high-level clausal nature which complements local information available for these tasks. Future work on applying topological fields will also likely share this property. In addition, topological fields may be useful in computational discourse, due to their correlation with the information structure of a passage.

Bibliography

S. Abney, S. Flickenger, C. Gdaniec, C. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, et al. Procedure for quantitatively comparing the syntactic coverage of English grammars. In *Proceedings of the 1991 Workshop on Speech and Natural Language*, pages 306–311. Association for Computational Linguistics Morristown, NJ, USA, 1991.

M. Becker and A. Frank. A stochastic topological parser for German. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 71–77, 2002.

M. Becker and E. Pecourt. Anaphora Resolution Using a Topological Parser. In *Proceedings of the Fourth Discourse Anaphora and Anaphor Resolution Colloquium*, 2002.

S. Brants, S. Dipper, S. Hansen, W. Lezius, and G. Smith. The TIGER Treebank. In *Proceedings of the 2002 Workshop on Treebanks and Linguistic Theories*, pages 24–41, 2002.

T. Brants. TnT–a statistical part-of-speech tagger. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, pages 224–231, 2000.

T. Brants, W. Skut, and H. Uszkoreit. Syntactic annotation of a German newspaper corpus. 1999.

U. Callmeier. PET–a platform for experimentation with efficient HPSG processing techniques. *Natural Language Engineering*, 6(01):99–107, 2000.

E. Charniak. Statistical Parsing with a Context-Free Grammar and Word Statistics. In *Proceedings of the 14th National Conference on Artificial Intelligence*, pages 598–603, 1997.

E. Charniak. A maximum-entropy-inspired parser. In *Proceedings of the First Meeting of the North American Chapter of the Association for Computational Linguistics*, volume 4, pages 132–139, 2000.

E. Charniak and M. Johnson. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 173–180. Association for Computational Linguistics Morristown, NJ, USA, 2005.

J.C.K. Cheung and G. Penn. Topological Field Parsing of German. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing*, pages 64–72, 2009.

M. Collins. Head-Driven Statistical Models for Natural Language Parsing. *Computational Linguistics*, 29(4):589–637, 2003.

M.J. Collins. A new statistical parser based on bigram lexical dependencies. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pages 184–191. Association for Computational Linguistics Morristown, NJ, USA, 1996.

A. Cormack and N. Smith. Fronting: The Syntax and Pragmatics of “Focus” and “Topic”. Technical report, UCL Working Papers in Linguistics, 2000.

A. Dubey. What to do when lexicalization fails: parsing German with suffix analysis and smoothing. *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, 100, 2005.

A. Dubey and F. Keller. Probabilistic parsing for German using sister-head dependencies. In

Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, pages 96–103, 2003.

D. Duchier and R. Debusmann. Topological Dependency Trees: A Constraint-Based Account of Linear Precedence. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, volume 39, pages 180–187, 2001.

K.A. Foth, M. Daum, and W. Menzel. A broad-coverage parser for German based on defeasible constraints. *Constraint Solving and Language Processing*, 2004.

A. Frank, M. Becker, B. Crysmann, B. Kiefer, and U. Schaefer. Integrated shallow and deep parsing: TopP meets HPSG. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 104–111, 2003.

W. Frey. Notes on the syntax and the pragmatics of German Left Dislocation. In H. Lohnstein and S. Trissler, editors, *The Syntax and Semantics of the Left Periphery*, pages 203–233. Mouton de Gruyter, Berlin, 2004a.

W. Frey. A medial topic position for German. *Linguistische Berichte*, 198:153–190, 2004b.

L. Haegeman. *Introduction to government and binding theory*. Blackwell Publishers, 1994.

K. Hale. Warlpiri and the grammar of non-configurational languages. *Natural Language & Linguistic Theory*, 1(1):5–47, 1983.

J. Henderson. Discriminative training of a neural network statistical parser. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics Morristown, NJ, USA, 2004.

J.R. Hobbs. Resolving pronoun refereces. *Lingua*, 4(4):311–338, 1978.

J. Hockenmaier. Creating a CCGbank and a Wide-Coverage CCG Lexicon for German. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 505–512, 2006.

T.N. Höhle. *Topologische Felder*. PhD thesis, Köln, 1983.

L. Huang. Forest reranking: Discriminative parsing with non-local features. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2008.

D. Klein and C.D. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430. Association for Computational Linguistics Morristown, NJ, USA, 2003.

A. Koller and K. Striegnitz. Generation as dependency parsing. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, volume 2, pages 17–24, 2002.

S. Kübler. How do treebank annotation schemes influence parsing results? Or how not to compare apples and oranges. *Proceedings of the 2005 International Conference on Recent Advances in Natural Language Processing*, 2005.

S. Kübler, E.W. Hinrichs, and W. Maier. Is it really that difficult to parse German? In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, 2006.

S. Lappin and H.J. Leass. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561, 1994.

M. Liepert. Topological Fields Chunking for German with SVM's: Optimizing SVM-parameters with GA's. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, 2003.

M.P. Marcus, M.A. Marcinkiewicz, and B. Santorini. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

D. McClosky, E. Charniak, and M. Johnson. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, 2006.

F.H. Müller and T. Ule. Annotating topological fields and chunks: and revising POS tags at the same time. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7. Association for Computational Linguistics Morristown, NJ, USA, 2002.

G. Penn and M. Haji-Abdolhosseini. Topological parsing. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics*, pages 283–290, 2003.

S. Petrov and D. Klein. Parsing German with Latent Variable Grammars. In *Proceedings of the ACL-08 Workshop on Parsing German (PaGe-08)*, pages 33–39, 2008.

S. Petrov, L. Barrett, R. Thibaux, and D. Klein. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440, Sydney, Australia, July 2006. Association for Computational Linguistics.

B. Roark, Y. Liu, M. Harper, R. Stewart, M. Lease, M. Snover, I. Shafran, B. Dorr, J. Hale, A. Krasnyanskaya, et al. Reranking for sentence boundary detection in conversational speech. In *Proceedings of the 2006 IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 57–60. Citeseer, 2006.

C. Rohrer and M. Forst. Improving coverage and parsing quality of a large-scale LFG for German. In *Proceedings of the 2006 Language Resources and Evaluation Conference*, Genoa, Italy, 2006.

A. Sarkar. Applying co-training methods to statistical parsing. In *Proceedings of the Second*

Meeting of the North American Chapter of the Association for Computational Linguistics, pages 95–102, 2001.

L. Shen, A. Sarkar, and F.J. Och. Discriminative reranking for machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 177–184, 2004.

A. Siewierska. *Constituent order in the languages of Europe*. Mouton de Gruyter, 1998.

W. Skut, T. Brants, B. Krenn, and H. Uszkoreit. A Linguistically Interpreted Corpus of German Newspaper Text. *Proceedings of the ESSLLI Workshop on Recent Advances in Corpus Annotation*, 1998.

B. Taskar, D. Klein, M. Collins, D. Koller, and C. Manning. Max-margin parsing. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, 2004.

H. Telljohann, E. Hinrichs, and S. Kubler. The TüBa-D/Z treebank: Annotating German with a context-free backbone. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation*, pages 2229–2235, 2004.

H. Telljohann, E.W. Hinrichs, S. Kubler, and H. Zinsmeister. Stylebook for the Tubingen Treebank of Written German (TüBa-D/Z). *Seminar fur Sprachwissenschaft, Universitat Tubingen, Tubingen, Germany*, 2006.

T. Ule. Directed Treebank Refinement for PCFG Parsing. In *Proceedings of 2003 Workshop on Treebanks and Linguistic Theories*, pages 177–188, 2003.

J. Veenstra, F.H. Müller, and T. Ule. Topological field chunking for German. In *Proceedings of the Sixth Conference on Natural Language Learning*, pages 56–62, 2002.