What to do when lexicalization fails: parsing German with suffix analysis and smoothing

Amit Dubey
University of Edinburgh
Amit.Dubey@ed.ac.uk

Abstract

In this paper, we present an unlexicalized parser for German which employs smoothing and suffix analysis to achieve a labelled bracket F-score of 76.2, higher than previously reported results on the NEGRA corpus. In addition to the high accuracy of the model, the use of smoothing in an unlexicalized parser allows us to better examine the interplay between smoothing and parsing results.

1 Introduction

Recent research on German statistical parsing has shown that lexicalization adds little to parsing performance in German (Dubey and Keller, 2003; Beil et al., 1999). A likely cause is the relative productivity of German morphology compared to that of English: German has a higher type/token ratio for words, making sparse data problems more severe. There are at least two solutions to this problem: first, to use better models of morphology or, second, to make unlexicalized parsing more accurate.

We investigate both approaches in this paper. In particular, we develop a parser for German which attains the highest performance known to us by making use of smoothing and a highly-tuned suffix analyzer for guessing part-of-speech (POS) tags from the input text. Rather than relying on smoothing and suffix analysis alone, we also utilize treebank transformations (Johnson, 1998; Klein and Manning, 2003) instead of a grammar induced directly from a treebank.

The organization of the paper is as follows: Section 2 summarizes some important aspects of our treebank corpus. In Section 3 we outline several techniques for improving the performance of unlexicalized parsing without using smoothing, including treebank transformations, and the use of suffix analysis. We show that suffix analysis is not helpful on the treebank grammar, but it does increase performance if used in combination with the treebank transformations we present. Section 4 describes how smoothing can be incorporated into an unlexicalized grammar to achieve state-of-the-art results in German. Rather using one smoothing algorithm, we use three different approaches, allowing us to compare the relative performance of each. An error analysis is presented in Section 5, which points to several possible areas of future research. We follow the error analysis with a comparison with related work in Section 6. Finally we offer concluding remarks in Section 7.

2 Data

The parsing models we present are trained and tested on the NEGRA corpus (Skut et al., 1997), a hand-parsed corpus of German newspaper text containing approximately 20,000 sentences. It is available in several formats, and in this paper, we use the Penn Treebank (Marcus et al., 1993) format of NEGRA.

The annotation used in NEGRA is similar to that used in the English Penn Treebank, with some differences which make it easier to annotate German syntax. German's *flexible word order* would have required an explosion in long-distance dependencies (LDDs) had annotation of NEGRA more closely resembled that of the Penn Treebank. The NEGRA designers therefore chose to use relatively flat trees, encoding elements of flexible word order us-

Proceedings of the 43rd Annual Meeting of the ACL, pages 314–321,
Ann Arbor, June 2005.

ing grammatical functions (GFs) rather than LDDs wherever possible.

To illustrate flexible word order, consider the sentences *Der Mann sieht den Jungen* ('The man sees the boy') and *Den Jungen sieht der Mann.* Despite the fact the subject and object are swapped in the second sentence, the meaning of both are essentially the same.[1] The two possible word orders are disambiguated by the use of the nominative case for the subject (marked by the article *der*) and the accusative case for the object (marked by *den)* rather than their position in the sentence.

Whenever the subject appears after the verb, the non-standard position may be annotated using a long-distance dependency (LDD). However, as mentioned above, this information can also be retrieved from the grammatical function of the respective noun phrases: the GFs of the two NPs above would be 'subject' and 'accusative object' regardless of their position in the sentence. These labels may therefore be used to recover the underlying dependencies without having to resort to LDDs. This is the approach used in NEGRA. It does have limitations: it is only possible to use GF labels instead of LDDs when all the nodes of interest are dominated by the same parent. To maximize cases where all necessary nodes are dominated by the same parent, NEGRA uses flat 'dependency-style' rules. For example, there is no VP node when there is no overt auxiliary verb. category. Under the NEGRA annotation scheme, the first sentence above would have a rule S→NP-SB VVFIN NP-OA and the second, S→NP-OA VVFIN NP-SB, where SB denotes subject and OA denotes accusative object.

3 Parsing with Grammatical Functions

3.1 Model

As explained above, this paper focuses on unlexicalized grammars. In particular, we make use of probabilistic context-free grammars (PCFGs; Booth (1969)) for our experiments. A PCFG assigns each context-free rule LHS $\rightarrow$ RHS a conditional probability $P_r(\text{RHS}|\text{LHS})$. If a parser were to be given POS tags as input, this would be the only distribution required. However, in this paper we are concerned with the more realistic problem of accepting text as input. Therefore, the parser also needs a probability distribution $P_w(w|\text{LHS})$ to generate words. The probability of a tree is calculated by multiplying the probabilities all the rules and words generated in the derivation of the tree.

The rules are simply read out from the treebank, and the probabilities are estimated from the frequency of rules in the treebank. More formally:

$$P_r(\text{RHS}|\text{LHS}) = \frac{c(\text{LHS} \rightarrow \text{RHS})}{c(\text{LHS})} \quad (1)$$

The probabilities of words given tags are similarly estimated from the frequency of word-tag co-occurrences:

$$P_w(w|\text{LHS}) = \frac{c(\text{LHS}, w)}{c(\text{LHS})} \quad (2)$$

To handle unseen or infrequent words, all words whose frequency falls below a threshold Ω are grouped together in an 'unknown word' token, which is then treated like an additional word. For our experiments, we use $\Omega = 10$.

We consider several variations of this simple model by changing both P_r and P_w. In addition to the standard formulation in Equation (1), we consider two alternative variants of P_r. The first is a *Markov* context-free rule (Magerman, 1995; Charniak, 2000). A rule may be turned into a Markov rule by first binarizing it, then making independence assumptions on the new binarized rules. Binarizing the rule $A \rightarrow B_1 \ldots B_n$ results in a number of smaller rules $A \rightarrow B_1 A_{B_1}$, $A_{B_1} \rightarrow B_2 A_{B_1 B_2}$, ..., $A_{B_1 \ldots B_{n-1}} \rightarrow B_n$. Binarization does not change the probability of the rule:

$$\begin{aligned} & P(B_1 \ldots B_n|A) \\ = & \prod_{n}^{i=1} P(\rightarrow B_i|A, B_1, \ldots, B_{i-1}) \end{aligned}$$

Making the 2^{nd} order Markov assumption 'forgets' everything earlier then 2 previous sisters. A rule would now be in the form $A_{B_{i-2}B_{i-1}} \rightarrow B_i A_{B_{i-1}B_i}$, and the probability would be:

$$\begin{aligned} & P(B_1 \ldots B_n|A) \\ \approx & \prod_{n}^{i=1} P(B_i|A, B_{i-2}, B_{i-1}) \end{aligned}$$

[1] Pragmatically speaking, the second sentence has a slightly different meaning. A better translation might be: 'It is the boy the man sees.'

The other rule type we consider are linear precedence/immediate dominance (LP/ID) rules (Gazdar et al., 1985). If a context-free rule can be thought of as a LHS token with an ordered list of tokens on the RHS, then an LP/ID rule can be thought of as a LHS token with a *multiset* of tokens on the RHS together with some constraints on the possible orders of tokens on the RHS. Uszkoreit (1987) argues that LP/ID rules with violatable 'soft' constraints are suitable for modelling some aspects of German word order. This makes a probabilistic formulation of LP/ID rules ideal: probabilities act as soft constraints.

Our treatment of probabilistic LP/ID rules generate children one constituent at a time, conditioning upon the parent and a multiset of previously generated children. Formally, the the probability of the rule is approximated as:

$$\begin{aligned} & P(B_1 \ldots B_n | A) \\ \approx & \prod_{n}^{i=1} P(B_i | A, \{B_j | j < i\}) \end{aligned}$$

In addition to the two additional formulations of the P_r distribution, we also consider one variant of the P_w distribution, which includes the suffix analysis. It is important to clarify that we only change the handling of uncommon and unknown words; those which occur often are handled as normal. suggested different choices for P_w in the face of unknown words: Schiehlen (2004) suggests using a different unknown word token for capitalized versus uncapitalized unknown words (German orthography dictates that all common nouns are capitalized) and Levy and Manning (2004) consider inspecting the last letter the unknown word to guess the part-of-speech (POS) tags. Both of these models are relatively impoverished when compared to the approaches of handling unknown words which have been proposed in the POS tagging literature. Brants (2000) describes a POS tagger with a highly tuned suffix analyzer which considers both capitalization and suffixes as long as 10 letters long. This tagger was developed with German in mind, but neither it nor any other advanced POS tagger morphology analyzer has ever been tested with a full parser. Therefore, we take the novel step of integrating this suffix analyzer into the parser for the second P_w distribution.

3.2 Treebank Re-annotation

Automatic treebank transformations are an important step in developing an accurate unlexicalized parser (Johnson, 1998; Klein and Manning, 2003). Most of our transformations focus upon one part of the NEGRA treebank in particular: the GF labels. Below is a list of GF re-annotations we utilise:

Coord GF In NEGRA, a co-ordinated accusative NP rule might look like NP-OA →NP-CJ KON NP-CJ. KON is the POS tag for a conjunct, and CJ denotes the function of the NP is a coordinate sister. Such a rule hides an important fact: the two co-ordinate sisters are also accusative objects. The Coord GF re-annotation would therefore replace the above rule with NP-OA → NP-OA KON NP-OA.

NP case German articles and pronouns are strongly marked for case. However, the grammatical function of all articles is usually NK, meaning noun kernel. To allow case markings in articles and pronouns to 'communicate' with the case labels on the GFs of NPs, we copy these GFs down into the POS tags of articles and pronouns. For example, a rule like NP-OA → ART-NK NN-NK would be replaced by NP-OA →ART-OA NN-NK. A similar improvement has been independently noted by Schiehlen (2004).

PP case Prepositions determine the case of the NP they govern. While the case is often unambiguous (i.e. *für* 'for' always takes an accusative NP), at times the case may be ambiguous. For instance, *in* 'in' may take either an accusative or dative NP. We use the labels -OA, -OD, etc. for unambiguous prepositions, and introduce new categories AD (accusative/dative ambiguous) and DG (dative/genitive ambiguous) for the ambiguous categories. For example, a rule such as PP → P ART-NK NN-NK is replaced with PP →P-AD ART-AD NN-NK if it is headed by the preposition *in*.

SBAR marking German subordinate clauses have a different word order than main clauses. While subordinate clauses can usually be distinguished from main clauses by their GF, there are some GFs which are used in both cases. This transformation adds an SBAR category to explicitly disambiguate these

	No suffix	With suffix
	F-score	F-score
Normal rules	66.3	66.2
LP/ID rules	66.5	66.6
Markov rules	**69.4**	69.1

Table 1: Effect of rule type and suffix analysis.

cases. The transformation does not add any extra nonterminals, rather it replaces rules such as S → KOUS NP V NP (where KOUS is a complementizer POS tag) with SBAR → KOUS NP V NP.

S GF One may argue that, as far as syntactic disambiguation is concerned, GFs on S categories primarily serve to distinguish main clauses from subordinate clauses. As we have explicitly done this in the previous transformation, it stands to reason that the GF tags on S nodes may therefore be removed without penalty. If the tags are necessary for semantic interpretation, presumably they could be re-inserted using a strategy such as that of Blaheta and Charniak (2000) The last transformation therefore removes the GF of S nodes.

3.3 Method

To allow comparisons with earlier work on NEGRA parsing, we use the same split of training, development and testing data as used in Dubey and Keller (2003). The first 18,602 sentences are used as training data, the following 1,000 form the development set, and the last 1,000 are used as the test set. We remove long-distance dependencies from all sets, and only consider sentences of length 40 or less for efficiency and memory concerns. The parser is given untagged words as input to simulate a realistic parsing task. A probabilistic CYK parsing algorithm is used to compute the Viterbi parse.

We perform two sets of experiments. In the first set, we vary the rule type, and in the second, we report the additive results of the treebank reannotations described in Section 3.2. The three rule types used in the first set of experiments are standard CFG rules, our version of LP/ID rules, and 2^{nd} order Markov CFG rules. The second battery of experiments was performed on the model with Markov rules.

In both cases, we report PARSEVAL labeled bracket scores (Magerman, 1995), with the brackets labeled by syntactic categories but not grammatical functions. Rather than reporting precision and recall of labelled brackets, we report only the F-score, i.e. the harmonic mean of precision and recall.

	No suffix	With suffix
	F-score	F-score
GF Baseline	69.4	69.1
+Coord GF	70.2	71.5
+NP case	71.1	72.4
+PP case	71.0	72.7
+SBAR	70.9	72.6
+S GF	71.3	**73.1**

Table 2: Effect of re-annotation and suffix analysis with Markov rules.

3.4 Results

Table 1 shows the effect of rule type choice, and Table 2 lists the effect of the GF re-annotations. From Table 1, we see that Markov rules achieve the best performance, ahead of both standard rules as well as our formulation of probabilistic LP/ID rules.

In the first group of experiments, suffix analysis marginally lowers performance. However, a different pattern emerges in the second set of experiments. Suffix analysis consistently does better than the simpler word generation probability model.

Looking at the treebank transformations with suffix analysis enabled, we find the coordination reannotation provides the greatest benefit, boosting performance by 2.4 to 71.5. The NP and PP case re-annotations together raise performance by 1.2 to 72.7. While the SBAR annotation slightly lowers performance, removing the GF labels from S nodes increased performance to 73.1.

3.5 Discussion

There are two primary results: first, although LP/ID rules have been suggested as suitable for German's flexible word order, it appears that Markov rules actually perform better. Second, adding suffix analysis provides a clear benefit, but only after the inclusion of the Coord GF transformation.

While the SBAR transformation slightly reduces performance, recall that we argued the S GF transformation only made sense if the SBAR transforma-

tion is already in place. To test if this was indeed the case, we re-ran the final experiment, but excluded the SBAR transformation. We did indeed find that applying S GF without the SBAR transformation reduced performance.

4 Smoothing & Search

With the exception of DOP models (Bod, 1995), it is uncommon to smooth unlexicalized grammars. This is in part for the sake of simplicity: unlexicalized grammars are interesting because they are simple to estimate and parse, and adding smoothing makes both estimation and parsing nearly as complex as with fully lexicalized models. However, because lexicalization adds little to the performance of German parsing models, it is therefore interesting to investigate the impact of smoothing on unlexicalized parsing models for German.

Parsing an unsmoothed unlexicalized grammar is relatively efficient because the grammar constraints the search space. As a smoothed grammar does not have a constrained search space, it is necessary to find other means to make parsing faster. Although it is possible to efficiently compute the Viterbi parse (Klein and Manning, 2002) using a smoothed grammar, the most common approach to increase parsing speed is to use some form of beam search (cf. Goodman (1998)), a strategy we follow here.

4.1 Models

We experiment with three different smoothing models: the modified Witten-Bell algorithm employed by Collins (1999), the modified Kneser-Ney algorithm of Chen and Goodman (1998) the smoothing algorithm used in the POS tagger of Brants (2000). All are variants of linear interpolation, and are used with 2^{nd} order Markovization. Under this regime, the probability of adding the i^{th} child to $\mathrm{A} \rightarrow \mathrm{B}_1 \ldots \mathrm{B}_n$ is estimated as

$$\begin{aligned} & P(B_i|A,B_{i-1},B_{i-2}) \\ = \; & \lambda_1 P(B_i|A,B_{i-1},B_{i-2}) + \\ & \lambda_2 P(B_i|A,B_{i-1}) + \lambda_3 P(B_i|A) + \lambda_4 P(B_i) \end{aligned}$$

The models differ in how the λ's are estimated. For both the Witten-Bell and Kneser-Ney algorithms, the λ's are a function of the context A, B_{i-2}, B_{i-1}. By contrast, in Brants' algorithm the λ's are constant for all possible contexts. As both the Witten-Bell and Kneser-Ney variants are fairly well known, we do not describe them further. However, as Brants' approach (to our knowledge) has not been used elsewhere, and because it needs to be modified for our purposes, we show the version of the algorithm we use in Figure 1.

$\lambda_1, \lambda_2, \lambda_3 \leftarrow 0$
for each trigram x_1, x_2, x_3 with $c(x_1, x_2, x_3) > 0$
  $d_3 \leftarrow \begin{cases} \frac{c(x_i, x_{i-1}, x_{i-2})-1}{c(x_{i-1}, x_{i-2})-1} & \text{if } c(x_{i-1}, x_{i-2}) > 1 \\ 0 & \text{if } c(x_{i-1}, x_{i-2}) = 1 \end{cases}$
  $d_2 \leftarrow \begin{cases} \frac{c(x_i, x_{i-1})-1}{c(x_{i-1})-1} & \text{if } c(x_{i-1}) > 1 \\ 0 & \text{if } c(x_{i-1}) = 1 \end{cases}$
  $d_1 \leftarrow \frac{c(x_i)-1}{N-1}$
  if $d_3 = \max d_1, d_2, d_3$ then
    $\lambda_3 \leftarrow \lambda_3 + c(x_i, x_{i-1}, x_{i-2})$
  elseif $d_2 = \max d_1, d_2, d_3$ then
    $\lambda_2 \leftarrow \lambda_2 + c(x_i, x_{i-1}, x_{i-2})$
  else
    $\lambda_1 \leftarrow \lambda_1 + c(x_i, x_{i-1}, x_{i-2})$
end
$\lambda_1 \leftarrow \frac{\lambda_1}{\lambda_1+\lambda_2+\lambda+3}$
$\lambda_2 \leftarrow \frac{\lambda_2}{\lambda_1+\lambda_2+\lambda+3}$
$\lambda_3 \leftarrow \frac{\lambda_3}{\lambda_1+\lambda_2+\lambda+3}$

Figure 1: Smoothing estimation based on the Brants (2000) approach for POS tagging.

4.2 Method

The purpose of this is experiment is not only to improve parsing results, but also to investigate the overall effect of smoothing on parse accuracy. Therefore, we do not simply report results with the best model from Section 3. Rather, we re-do each modification in Section 3 with both search strategies (Viterbi and beam) in the unsmoothed case, and with all three smoothing algorithms with beam search. The beam has a variable width, which means an arbitrary number of edges may be considered, as long as their probability is within 4×10^{-3} of the best edge in a given span.

4.3 Results

Table 3 summarizes the results. The best result in each column is italicized, and the overall best result

	No Smoothing Viterbi	No Smoothing Beam	Brants Beam	Kneser-Ney Beam	Witten-Bell Beam
GF Baseline	69.1	70.3	72.3	72.6	72.3
+Coord GF	71.5	72.7	75.2	75.4	74.5
+NP case	72.4	*73.3*	76.0	76.1	75.6
+PP case	72.7	73.2	76.1	*76.2*	*75.7*
+SBAR	72.6	73.1	**76.3**	76.0	75.3
+S GF Removal	*73.1*	72.6	75.7	75.3	75.1

Table 3: Effect of various smoothing algorithms.

in shown in bold. The column titled Viterbi reproduces the second column of Table 2 whereas the column titled Beam shows the result of re-annotation using beam search, but no smoothing. The best result with beam search is 73.3, slightly higher than without beam search.

Among smoothing algorithms, the Brants approach yields the highest results, of 76.3, with the modified Kneser-Ney algorithm close behind, at 76.2. The modified Witten-Bell algorithm achieved an F-score of 75.7.

4.4 Discussion

Overall, the best-performing model, using Brants smoothing, achieves a labelled bracketing F-score of 76.2, higher than earlier results reported by Dubey and Keller (2003) and Schiehlen (2004).

It is surprisingly that the Brants algorithm performs favourably compared to the better-known modified Kneser-Ney algorithm. This might be due to the heritage of the two algorithms. Kneser-Ney smoothing was designed for language modelling, where there are tens of thousands or hundreds of thousands of tokens having a Zipfian distribution. With all transformations included, the nonterminals of our grammar did have a Zipfian marginal distribution, but there were only several hundred tokens. The Brants algorithm was specifically designed for distributions with fewer tokens.

Also surprising is the fact that each smoothing algorithm reacted differently to the various treebank transformations. It is obvious that the choice of search and smoothing algorithm add *bias* to the final result. However, our results indicate that the choice of search and smoothing algorithm also add a degree of *variance* as improvements are added to the parser. This is worrying: at times in the literature, details of search or smoothing are left out (e.g. Charniak (2000)). Given the degree of variance due to search and smoothing, it raises the question if it is in fact possible to reproduce such results without the necessary details.[2]

5 Error Analysis

While it is uncommon to offer an error analysis for probabilistic parsing, Levy and Manning (2003) argue that a careful error classification can reveal possible improvements. Although we leave the implementation of any improvements to future research, we do discuss several common errors. Because the parser with Brants smoothing performed best, we use that as the basis of our error analysis.

First, we found that POS tagging errors had a strong effect on parsing results. This is surprising, given that the parser is able to assign POS tags with a high degree of accuracy. POS tagging results are comparable to the best stand-alone POS taggers, achieving results of 97.1% on the test set, matching the performance of the POS tagger described by Brants (2000) When GF labels are included (e.g. considering ART-SB instead of just ART), tagging accuracy falls to 90.1%. To quantify the effect of POS tagging errors, we re-parsed with correct POS tags (rather than letting the parser guess the tags), and found that labelled bracket F-scores increase from 76.3 to 85.2. A manual inspection of 100 sentences found that GF mislabelling can accounts for at most two-thirds of the mistakes due to POS tags. Over one third was due to genuine POS tagging errors. The most common problem was verb mistagging: they are either confused with adjectives (both

[2] As an anonymous reviewer pointed out, it is not always straightforward to reproduce statistical parsing results even when the implementation details *are* given (Bikel, 2004).

Model	LB F-score
This paper	**76.3**
Dubey and Keller (2003)	74.1
Schiehlen (2004)	71.1

Table 4: Comparison with previous work.

take the common -en suffix), or the tense was incorrect. Mistagged verb are a serious problem: it entails an entire clause is parsed incorrectly. Verb mistagging is also a problem for other languages: Levy and Manning (2003) describe a similar problem in Chinese for noun/verb ambiguity. This problem might be alleviated by using a more detailed model of morphology than our suffix analyzer provides.

To investigate pure parsing errors, we manually examined 100 sentences which were incorrectly parsed, but which nevertheless were assigned the correct POS tags. Incorrect modifier attachment accounted for for 39% of all parsing errors (of which 77% are due to PP attachment alone). Misparsed coordination was the second most common problem, accounting for 15% of all mistakes. Another class of error appears to be due to Markovization. The boundaries of VPs are sometimes incorrect, with the parser attaching dependents directly to the S node rather than the VP. In the most extreme cases, the VP had no verb, with the main verb heading a subordinate clause.

6 Comparison with Previous Work

Table 4 lists the result of the best model presented here against the earlier work on NEGRA parsing described in Dubey and Keller (2003) and Schiehlen (2004). Dubey and Keller use a variant of the lexicalized Collins (1999) model to achieve a labelled bracketing F-score of 74.1%. Schiehlen presents a number of unlexicalized models. The best model on labelled bracketing achieves an F-score of 71.8%.

The work of Schiehlen is particularly interesting as he also considers a number of transformations to improve the performance of an unlexicalized parser. Unlike the work presented here, Schiehlen does not attempt to perform any suffix or morphological analysis of the input text. However, he does suggest a number of treebank transformations. One such transformation is similar to one we prosed here, the NP case transformation. His implementation is different from ours: he annotates the case of pronouns and common nouns, whereas we focus on articles and pronouns (articles are pronouns are more strongly marked for case than common nouns). The remaining transformations we present are different from those Schiehlen describes; it is possible that an even better parser may result if all the transformations were combined.

Schiehlen also makes use of a morphological analyzer tool. While this includes more complete information about German morphology, our suffix analysis model allows us to integrate morphological ambiguities into the parsing system by means of lexical generation probabilities.

Levy and Manning (2004) also present work on the NEGRA treebank, but are primarily interested in long-distance dependencies, and therefore do not report results on local dependencies, as we do here.

7 Conclusions

In this paper, we presented the best-performing parser for German, as measured by labelled bracket scores. The high performance was due to three factors: (i) treebank transformations (ii) an integrated model of morphology in the form of a suffix analyzer and (iii) the use of smoothing in an unlexicalized grammar. Moreover, there are possible paths for improvement: lexicalization could be added to the model, as could some of the treebank transformations suggested by Schiehlen (2004). Indeed, the suffix analyzer could well be of value in a lexicalized model.

While we only presented results on the German NEGRA corpus, there is reason to believe that the techniques we presented here are also important to other languages where lexicalization provides little benefit: smoothing is a broadly-applicable technique, and if difficulties with lexicalization are due to sparse lexical data, then suffix analysis provides a useful way to get more information from lexical elements which were unseen while training.

In addition to our primary results, we also provided a detailed error analysis which shows that PP attachment and co-ordination are problematic for our parser. Furthermore, while POS tagging is highly accurate, the error analysis also shows it does

have surprisingly large effect on parsing errors. Because of the strong impact of POS tagging on parsing results, we conjecture that increasing POS tagging accuracy may be another fruitful area for future parsing research.

References

Franz Beil, Glenn Carroll, Detlef Prescher, Stefan Riezler, and Mats Rooth. 1999. Inside-Outside Estimation of a Lexicalized PCFG for German. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, University of Maryland, College Park.

Daniel M. Bikel. 2004. Intricacies of Collins' Parsing Model. *Computational Linguistics*, 30(4).

Don Blaheta and Eugene Charniak. 2000. Assigning function tags to parsed text. In *Proceedings of the 1st Conference of the North American Chapter of the ACL (NAACL), Seattle, Washington.*, pages 234–240.

Rens Bod. 1995. *Enriching Linguistics with Statistics: Performance Models of Natural Language*. Ph.D. thesis, University of Amsterdam.

Taylor L. Booth. 1969. Probabilistic Representation of Formal Languages. In *Tenth Annual IEEE Symposium on Switching and Automata Theory*, pages 74–81.

Thorsten Brants. 2000. TnT: A statistical part-of-speech tagger. In *Proceedings of the 6th Conference on Applied Natural Language Processing*, Seattle.

Eugene Charniak. 2000. A Maximum-Entropy-Inspired Parser. In *Proceedings of the 1st Conference of North American Chapter of the Association for Computational Linguistics*, pages 132–139, Seattle, WA.

Stanley F. Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Center for Research in Computing Technology, Harvard University.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

Amit Dubey and Frank Keller. 2003. Parsing German with Sister-head Dependencies. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 96–103, Sapporo, Japan.

Gerald Gazdar, Ewan Klein, Geoffrey Pullum, and Ivan Sag. 1985. *Generalized Phase Structure Grammar*. Basil Blackwell, Oxford, England.

Joshua Goodman. 1998. *Parsing inside-out*. Ph.D. thesis, Harvard University.

Mark Johnson. 1998. PCFG models of linguistic tree representations. *Computational Linguistics*, 24(4):613–632.

Dan Klein and Christopher D. Manning. 2002. A* Parsing: Fast Exact Viterbi Parse Selection. Technical Report dbpubs/2002-16, Stanford University.

Dan Klein and Christopher D. Manning. 2003. Accurate Unlexicalized Parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan.

Roger Levy and Christopher D. Manning. 2003. Is it Harder to Parse Chinese, or the Chinese Treebank? In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*.

Roger Levy and Christopher D. Manning. 2004. Deep Dependencies from Context-Free Statistical Parsers: Correcting the Surface Dependency Approximation. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*.

David M. Magerman. 1995. Statistical Decision-Tree Models for Parsing. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 276–283, Cambridge, MA.

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

Micheal Schiehlen. 2004. Annotation Strategies for Probabilistic Parsing in German. In *Proceedings of the 20th International Conference on Computational Linguistics*.

Wojciech Skut, Brigitte Krenn, Thorsten Brants, and Hans Uszkoreit. 1997. An annotation scheme for free word order languages. In *Proceedings of the 5th Conference on Applied Natural Language Processing*, Washington, DC.

Hans Uszkoreit. 1987. *Word Order and Constituent Structure in German*. CSLI Publications, Stanford, CA.