Probabilistic Parsing for German using Sister-Head Dependencies

Amit Dubey
Department of Computational Linguistics
Saarland University
PO Box 15 11 50
66041 Saarbrücken, Germany
adubey@coli.uni-sb.de

Frank Keller
School of Informatics
University of Edinburgh
2 Buccleuch Place
Edinburgh EH8 9LW, UK
keller@inf.ed.ac.uk

Abstract

We present a probabilistic parsing model for German trained on the Negra treebank. We observe that existing lexicalized parsing models using head-head dependencies, while successful for English, fail to outperform an unlexicalized baseline model for German. Learning curves show that this effect is not due to lack of training data. We propose an alternative model that uses sister-head dependencies instead of head-head dependencies. This model outperforms the baseline, achieving a labeled precision and recall of up to 74%. This indicates that sister-head dependencies are more appropriate for treebanks with very flat structures such as Negra.

1 Introduction

Treebank-based probabilistic parsing has been the subject of intensive research over the past few years, resulting in parsing models that achieve both broad coverage and high parsing accuracy (e.g., Collins 1997; Charniak 2000). However, most of the existing models have been developed for English and trained on the Penn Treebank (Marcus et al., 1993), which raises the question whether these models generalize to other languages, and to annotation schemes that differ from the Penn Treebank markup.

The present paper addresses this question by proposing a probabilistic parsing model trained on Negra (Skut et al., 1997), a syntactically annotated corpus for German. German has a number of syntactic properties that set it apart from English, and the Negra annotation scheme differs in important respects from the Penn Treebank markup. While Negra has been used to build probabilistic chunkers (Becker and Frank, 2002; Skut and Brants, 1998), the research reported in this paper is the first attempt to develop a probabilistic full parsing model for German trained on a treebank (to our knowledge).

Lexicalization can increase parsing performance dramatically for English (Carroll and Rooth, 1998; Charniak, 1997, 2000; Collins, 1997), and the lexicalized model proposed by Collins (1997) has been successfully applied to Czech (Collins et al., 1999) and Chinese (Bikel and Chiang, 2000). However, the resulting performance is significantly lower than the performance of the same model for English (see Table 1). Neither Collins et al. (1999) nor Bikel and Chiang (2000) compare the lexicalized model to an unlexicalized baseline model, leaving open the possibility that lexicalization is useful for English, but not for other languages.

This paper is structured as follows. Section 2 reviews the syntactic properties of German, focusing on its semi-flexible wordorder. Section 3 describes two standard lexicalized models (Carroll and Rooth, 1998; Collins, 1997), as well as an unlexicalized baseline model. Section 4 presents a series of experiments that compare the parsing performance of these three models (and several variants) on Negra. The results show that both lexicalized models fail to outperform the unlexicalized baseline. This is at odds with what has been reported for English. Learning curves show that the poor performance of the lexicalized models is not due to lack of training data.

Section 5 presents an error analysis for Collins's (1997) lexicalized model, which shows that the head-head dependencies used in this model fail to cope well with the flat structures in Negra. We propose an alternative model that uses sister-head dependencies instead. This model outperforms the two original lexicalized models, as well as the unlexicalized baseline. Based on this result and on the review of the previous literature (Section 6), we argue (Section 7) that sister-head models are more appropriate for treebanks with very flat structures (such as Negra), typically used to annotate languages with semi-free wordorder (such as German).

2 Parsing German

2.1 Syntactic Properties

German exhibits a number of syntactic properties that distinguish it from English, the language that has been the focus of most research in parsing. Prominent among these properties is the **semi-free**

Language	Size	LR	LP	Source
English	40,000	87.4%	88.1%	(Collins, 1997)
Chinese	3,484	69.0%	74.8%	(Bikel and Chiang, 2000)
Czech	19,000	—	80.0% —	(Collins et al., 1999)

Table 1: Results for the Collins (1997) model for various languages (dependency precision for Czech)

wordorder, i.e., German wordorder is fixed in some respects, but variable in others. Verb order is largely fixed: in subordinate clauses such as (1a), both the finite verb *hat* 'has' and the non-finite verb *komponiert* 'composed' are in sentence final position.

(1) a. Weil er gestern Musik komponiert hat.
because er yesterday music composed has
'Because he has composed music yesterday.'
b. Hat er gestern Musik komponiert?
c. Er hat gestern Musik komponiert.

In yes/no questions such as (1b), the finite verb is sentence initial, while the non-finite verb is sentence final. In declarative main clauses (see (1c)), on the other hand, the finite verb is in second position (i.e., preceded by exactly one constituent), while the non-finite verb is final.

While verb order is fixed in German, the order of complements and adjuncts is variable, and influenced by a variety of syntactic and non-syntactic factors, including pronominalization, information structure, definiteness, and animacy (e.g., Uszkoreit 1987). The first position in a declarative sentence, for example, can be occupied by various constituents, including the subject (*er* 'he' in (1c)), the object (*Musik* 'music' in (2a)), an adjunct (*gestern* 'yesterday' in (2b)), or the non-finite verb (*komponiert* 'composed' in (2c)).

(2) a. Musik hat er gestern komponiert.
b. Gestern hat er Musik komponiert .
c. Komponiert hat er gestern Musik.

The semi-free wordorder in German means that a context-free grammar model has to contain more rules than for a fixed wordorder language. For transitive verbs, for instance, we need the rules S $\rightarrow$ V NP NP, S $\rightarrow$ NP V NP, and S $\rightarrow$ NP NP V to account for verb initial, verb second, and verb final order (assuming a flat S, see Section 2.2).

2.2 Negra Annotation Scheme

The Negra corpus consists of around 350,000 words of German newspaper text (20,602 sentences). The annotation scheme (Skut et al., 1997) is modeled to a certain extent on that of the Penn Treebank (Marcus et al., 1993), with crucial differences. Most importantly, Negra follows the dependency grammar tradition in assuming **flat syntactic representations**:

(a) There is no S $\rightarrow$ NP VP rule. Rather, the subject, the verb, and its objects are all sisters of each other, dominated by an S node. This is a way of accounting for the semi-free wordorder of German (see Section 2.1): the first NP within an S need not be the subject.

(b) There is no SBAR $\rightarrow$ Comp S rule. Main clauses, subordinate clauses, and relative clauses all share the category S in Negra; complementizers and relative pronouns are simply sisters of the verb.

(c) There is no PP $\rightarrow$ P NP rule, i.e., the preposition and the noun it selects (and determiners and adjectives, if present) are sisters, dominated by a PP node. An argument for this representation is that prepositions behave like case markers in German; a preposition and a determiner can merge into a single word (e.g., *in dem* 'in the' becomes *im*).

Another idiosyncrasy of Negra is that it assumes special **coordinate categories**. A coordinated sentence has the category CS, a coordinate NP has the category CNP, etc. While this does not make the annotation more flat, it substantially increases the number of non-terminal labels. Negra also contains **grammatical function labels** that augment phrasal and lexical categories. Example are MO (modifier), HD (head), SB (subject), and OC (clausal object).

3 Probabilistic Parsing Models

3.1 Probabilistic Context-Free Grammars

Lexicalization has been shown to improve parsing performance for the Penn Treebank (e.g., Carroll and Rooth 1998; Charniak 1997, 2000; Collins 1997). The aim of the present paper is to test if this finding carries over to German and to the Negra corpus. We therefore use an unlexicalized model as our baseline against which to test the lexicalized models.

More specifically, we used a standard probabilistic context-free grammar (PCFG; see Charniak 1993). Each context-free rule $RHS \rightarrow LHS$ is annotated with an expansion probability $P(RHS|LHS)$. The probabilities for all rules with the same lefthand side have to sum to one, and the probability of a parse tree T is defined as the product of the probabilities of all rules applied in generating T.

3.2 Carroll and Rooth's Head-Lexicalized Model

The head-lexicalized PCFG model of Carroll and Rooth (1998) is a minimal departure from the standard unlexicalized PCFG model, which makes it ideal for a direct comparison.[1]

A grammar rule $LHS \rightarrow RHS$ can be written as $P \rightarrow C_1 \ldots C_n$, where P is the mother category, and $C_1 \ldots C_n$ are daughters. Let $l(C)$ be the lexical head

[1] Charniak (1997) proposes essentially the same model; we will nevertheless use the label 'Carroll and Rooth model' as we are using their implementation (see Section 4.1).

of the constituent C. The rule probability is then defined as (see also Beil et al. 2002):

$$(3)\quad P(RHS|LHS) = P_{rule}(C_1 \ldots C_n|P,l(P)) \cdot \prod_{i=1}^{n} P_{choice}(l(C_i)|C_i,P,l(P))$$

Here $P_{rule}(C_1 \ldots C_n|P,l(P))$ is the probability that category P with lexical head $l(P)$ is expanded by the rule $P \rightarrow C_1 \ldots C_n$, and $P_{choice}(l(C)|C,P,l(P))$ is the probability that the (non-head) category C has the lexical head $l(C)$ given that its mother is P with lexical head $l(P)$.

3.3 Collins's Head-Lexicalized Model

In contrast to Carroll and Rooth's (1998) approach, the model proposed by Collins (1997) does not compute rule probabilities directly. Rather, they are generated using a Markov process that makes certain independence assumptions. A grammar rule $LHS \rightarrow RHS$ can be written as $P \rightarrow L_m \ldots L_1\ H\ R_1 \ldots R_n$ where P is the mother and H is the head daughter. Let $l(C)$ be the head word of C and $t(C)$ the tag of the head word of C. Then the probability of a rule is defined as:

$$(4)\quad \begin{aligned} & P(RHS|LHS) = P(L_m \ldots L_1\ H\ R_1 \ldots R_n|P) \\ = \ & P_h(H|P)P_l(L_m \ldots L_1|P,H)P_r(R_1 \ldots R_n|P,H) \\ = \ & P_h(H|P)\prod_{i=0}^{m} P_l(L_i|P,H,d(i))\prod_{i=0}^{n} P_r(R_i|P,H,d(i)) \end{aligned}$$

Here, P_h is the probability of generating the head, and P_l and P_r are the probabilities of generating the nonterminals to the left and right of the head, respectively; $d(i)$ is a distance measure. (L_0 and R_0 are stop categories.) At this point, the model is still unlexicalized. To add lexical sensitivity, the P_h, P_r and P_l probability functions also take into account head words and their POS tags:

$$(5)\quad \begin{aligned} & P(RHS|LHS) = P_h(H|P,t(P),l(P)) \\ & \cdot \prod_{i=0}^{m} P_l(L_i,t(L_i),l(L_i)|P,H,t(H),l(H),d(i)) \\ & \cdot \prod_{i=0}^{n} P_r(R_i,t(R_i),l(R_i)|P,H,t(H),l(H),d(i)) \end{aligned}$$

4 Experiment 1

This experiment was designed to compare the performance of the three models introduced in the last section. Our main hypothesis was that the lexicalized models will outperform the unlexicalized baseline model. Another prediction was that adding Negra-specific information to the models will increase parsing performance. We therefore tested a model variant that included grammatical function labels, i.e., the set of categories was augmented by the function tags specified in Negra (see Section 2.2).

Adding grammatical functions is a way of dealing with the wordorder facts of German (see Section 2.1) in the face of Negra's very flat annotation scheme. For instance, subject and object NPs have different wordorder preferences (subjects tend to be preverbal, while objects tend to be postverbal), a fact that is captured if subjects have the label NP-SB, while objects are labeled NP-OA (accusative object), NP-DA (dative object), etc. Also the fact that verb order differs between subordinate and main clauses is captured by the function labels: the former are labeled S, while the latter are labeled S-OC (object clause), S-RC (relative clause), etc.

Another idiosyncrasy of the Negra annotation is that conjoined categories have separate labels (S and CS, NP and CNP, etc.), and that PPs do not contain an NP node. We tested a variant of the Carroll and Rooth (1998) model that takes this into account.

4.1 Method

Data Sets All experiments reported in this paper used the treebank format of Negra. This format, which is included in the Negra distribution, was derived from the native format by replacing crossing branches with traces. We split the corpus into three subsets. The first 18,602 sentences constituted the training set. Of the remaining 2,000 sentences, the first 1,000 served as the test set, and the last 1000 as the development set. To increase parsing efficiency, we removed all sentences with more than 40 words. This resulted in a test set of 968 sentences and a development set of 975 sentences. Early versions of the models were tested on the development set, and the test set remained unseen until all parameters were fixed. The final results reported this paper were obtained on the test set, unless stated otherwise.

Grammar Induction For the unlexicalized PCFG model (henceforth **baseline model**), we used the probabilistic left-corner parser Lopar (Schmid, 2000). When run in unlexicalized mode, Lopar implements the model described in Section 3.1. A grammar and a lexicon for Lopar were read off the Negra training set, after removing all grammatical function labels. As Lopar cannot handle traces, these were also removed from the training data.

The head-lexicalized model of Carroll and Rooth (1998) (henceforth **C&R model**) was again realized using Lopar, which in lexicalized mode implements the model in Section 3.2. Lexicalization requires that each rule in a grammar has one of the categories on its righthand side annotated as the head. For the categories S, VP, AP, and AVP, the head is marked in Negra. For the other categories, we used rules to heuristically determine the head, as is standard practice for the Penn Treebank.

The lexicalized model proposed by Collins (1997) (henceforth **Collins model**) was re-implemented by one of the authors. For training, empty categories were removed from the training data, as the model cannot handle them. The same head finding strategy was applied as for the C&R model.

In this experiment, only head-head statistics were used (see (5)). The original Collins model uses sister-head statistics for non-recursive NPs. This will be discussed in detail in Section 5.

Training and Testing For all three models, the model parameters were estimated using maximum likelihood estimation. Both Lopar and the Collins model use various backoff distributions to smooth the estimates. The reader is referred to Schmid (2000) and Collins (1997) for details. For the C&R model, we used a cutoff of one for rule frequencies P_{rule} and lexical choice frequencies P_{choice} (the cutoff value was optimized on the development set).

We also tested variants of the baseline model and the C&R model that include grammatical function information, as we hypothesized that this information might help the model to handle wordorder variation more adequately, as explained above.

Finally, we tested variant of the C&R model that uses Lopar's parameter pooling feature. This feature makes it possible to collapse the lexical choice distribution P_{choice} for either the daughter or the mother categories of a rule (see Section 3.2). We pooled the estimates for pairs of conjoined and non-conjoined daughter categories (S and CS, NP and CNP, etc.): these categories should be treated as the same daughters; e.g., there should be no difference between S $\rightarrow$ NP V and S $\rightarrow$ CNP V. We also pooled the estimates for the mother categories NPs and PPs. This is a way of dealing with the fact that there is no separate NP node within PPs in Negra.

Lopar and the Collins model differ in their handling of unknown words. In Lopar, a POS tag distribution for unknown words has to be specified, which is then used to tag unknown words in the test data. The Collins model treats any word seen fewer than five times in the training data as unseen and uses an external POS tagger to tag unknown words. In order to make the models comparable, we used a uniform approach to unknown words. All models were run on POS-tagged input; this input was created by tagging the test set with a separate POS tagger, for both known and unknown words. We used TnT (Brants, 2000), trained on the Negra training set. The tagging accuracy was 97.12% on the development set.

In order to obtain an upper bound for the performance of the parsing models, we also ran the parsers on the test set with the correct tags (as specified in Negra), again for both known and unknown words. We will refer to this mode as 'perfect tagging'.

All models were evaluated using standard PARSEVAL measures. We report labeled recall (LR) labeled precision (LP), average crossing brackets (CBs), zero crossing brackets (0CB), and two or less crossing brackets ($\leq$2CB). We also give the coverage (Cov), i.e., the percentage of sentences that the parser was able to parse.

4.2 Results

The results for all three models and their variants are given in Table 2, for both TnT tags and perfect tags. The baseline model achieves 70.56% LR and 66.69% LP with TnT tags. Adding grammatical functions reduces both figures slightly, and coverage drops by about 15%. The C&R model performs worse than the baseline, at 68.04% LR and 60.07% LP (for TnT tags). Adding grammatical function again reduces performance slightly. Parameter pooling increases both LR and LP by about 1%. The Collins models also performs worse than the baseline, at 67.91% LR and 66.07% LP.

Performance using perfect tags (an upper bound of model performance) is 2–3% higher for the baseline and for the C&R model. The Collins model gains only about 1%. Perfect tagging results in a performance increase of over 10% for the models with grammatical functions. This is not surprising, as the perfect tags (but not the TnT tags) include grammatical function labels. However, we also observe a dramatic reduction in coverage (to about 65%).

4.3 Discussion

We added grammatical functions to both the baseline model and the C&R model, as we predicted that this would allow the model to better capture the wordorder facts of German. However, this prediction was not borne out: performance with grammatical functions (on TnT tags) was slightly worse than without, and coverage dropped substantially. A possible reason for this is sparse data: a grammar augmented with grammatical functions contains many additional categories, which means that many more parameters have to be estimated using the same training set. On the other hand, a performance increase occurs if the tagger also provides grammatical function labels (simulated in the perfect tags condition). However, this comes at the price of an unacceptable reduction in coverage.

When training the C&R model, we included a variant that makes use of Lopar's parameter pooling feature. We pooled the estimates for conjoined daughter categories, and for NP and PP mother categories. This is a way of taking the idiosyncrasies of the Negra annotation into account, and resulted in a small improvement in performance.

The most surprising finding is that the best performance was achieved by the unlexicalized PCFG

	TnT tagging						Perfect tagging					
	LR	LP	CBs	0CB	≤2CB	Cov	LR	LP	CBs	0CB	≤2CB	Cov
Baseline	70.56	66.69	1.03	58.21	84.46	94.42	72.99	70.00	0.88	60.30	87.42	95.25
Baseline + GF	70.45	65.49	1.07	58.02	85.01	79.24	81.14	78.37	0.46	74.25	95.26	65.39
C&R	68.04	60.07	1.31	52.08	79.54	94.42	70.79	63.38	1.17	54.99	82.21	95.25
C&R + pool	69.07	61.41	1.28	53.06	80.09	94.42	71.74	64.73	1.11	56.40	83.08	95.25
C&R + GF	67.66	60.33	1.31	55.67	80.18	79.24	81.17	76.83	0.48	73.46	94.15	65.39
Collins	67.91	66.07	0.73	65.67	89.52	95.21	68.63	66.94	0.71	64.97	89.73	96.23

Table 2: Results for Experiment 1: comparison of lexicalized and unlexicalized models (GF: grammatical functions; pool: parameter pooling for NPs/PPs and conjoined categories)

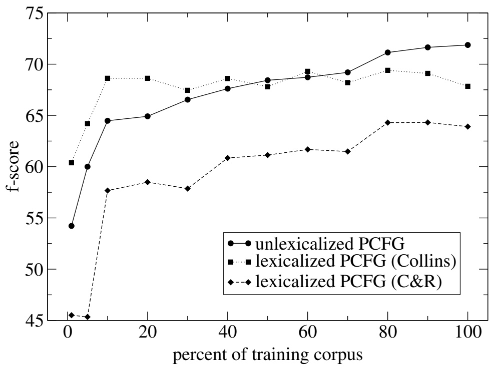

Figure 1: Learning curves for all three models

baseline model. Both lexicalized models (C&R and Collins) performed worse than the baseline. This results is at odds with what has been found for English, where lexicalization is standardly reported to increase performance by about 10%. The poor performance of the lexicalized models could be due to a lack of sufficient training data: our Negra training set contains approximately 18,000 sentences, and is therefore significantly smaller than the Penn Treebank training set (about 40,000 sentences). Negra sentences are also shorter: they contain, on average, 15 words compared to 22 in the Penn Treebank.

We computed learning curves for the unmodified variants (without grammatical functions or parameter pooling) of all three models (on the development set). The result (see Figure 1) shows that there is no evidence for an effect of sparse data. For both the baseline and the C&R model, a fairly high f-score is achieved with only 10% of the training data. A slow increase occurs as more training data is added. The performance of the Collins model is even less affected by training set size. This is probably due to the fact that it does not use rule probabilities directly, but generates rules using a Markov chain.

5 Experiment 2

As we saw in the last section, lack of training data is not a plausible explanation for the sub-baseline performance of the lexicalized models. In this experiment, we therefore investigate an alternative hypothesis, viz., that the lexicalized models do not cope

	Penn	Negra		Penn	Negra
NP	2.20	3.08	VP	2.32	2.59
PP	2.03	2.66	S	2.22	4.22

Table 3: Average number of daughters for the grammatical categories in the Penn Treebank and Negra

well with the fact that Negra rules are so flat (see Section 2.2). We will focus on the Collins model, as it outperformed the C&R model in Experiment 1.

An error analysis revealed that many of the errors of the Collins model in Experiment 1 are chunking errors. For example, the PP *neben den Mitteln des Theaters* should be analyzed as (6a). But instead the parser produces two constituents as in (6b)):

(6) a. [PP neben den Mitteln [NP des Theaters]]
apart the means the theater's
'apart from the means of the theater'.
b. [PP neben den Mitteln] [NP des Theaters]

The reason for this problem is that *neben* is the head of the constituent in (6), and the Collins model uses a crude distance measure together with head-head dependencies to decide if additional constituents should be added to the PP. The distance measure is inadequate for finding PPs with high precision.

The chunking problem is more widespread than PPs. The error analysis shows that other constituents, including Ss and VPs, also have the wrong boundary. This problem is compounded by the fact that the rules in Negra are substantially flatter than the rules in the Penn Treebank, for which the Collins model was developed. Table 3 compares the average number of daughters in both corpora.

The flatness of PPs is easy to reduce. As detailed in Section 2.2, PPs lack an intermediate NP projection, which can be inserted straightforwardly using the following rule:

(7) [PP P ...] → [PP P [NP ...]]

In the present experiment, we investigated if parsing performance improves if we test and train on a version of Negra on which the transformation in (7) has been applied.

In a second series of experiments, we investigated a more general way of dealing with the flatness of

	C&R	Collins	Charniak	Current
Head sister category	X	X	X	
Head sister head word	X	X	X	
Head sister head tag		X	X	
Prev. sister category	X		X	X
Prev. sister head word				X
Prev. sister head tag				X

Table 4: Linguistic features in the current model compared to the models of Carroll and Rooth (1998), Collins (1997), and Charniak (2000)

Negra, based on Collins's (1997) model for non-recursive NPs in the Penn Treebank (which are also flat). For non-recursive NPs, Collins (1997) does not use the probability function in (5), but instead substitutes P_r (and, by analogy, P_l) by:

(8) $P_r(R_i, t(R_i), l(R_i) | P, R_{i-1}, t(R_{i-1}), l(R_{i-1}), d(i))$

Here the head H is substituted by the sister R_{i-1} (and L_{i-1}). In the literature, the version of P_r in (5) is said to capture **head-head relationships**. We will refer to the alternative model in (8) as capturing **sister-head relationships**.

Using sister-head relationships is a way of counteracting the flatness of the grammar productions; it implicitly adds binary branching to the grammar. Our proposal is to extend the use of sister-head relationship from non-recursive NPs (as proposed by Collins) to all categories.

Table 4 shows the linguistic features of the resulting model compared to the models of Carroll and Rooth (1998), Collins (1997), and Charniak (2000). The C&R model effectively includes category information about **all** previous sisters, as it uses context-free rules. The Collins (1997) model does not use context-free rules, but generates the next category using zeroth order Markov chains (see Section 3.3), hence no information about the previous sisters is included. Charniak's (2000) model extends this to higher order Markov chains (first to third order), and therefore includes category information about previous sisters.The current model differs from all these proposals: it does not use any information about the head sister, but instead includes the category, head word, and head tag of the previous sister, effectively treating it as the head.

5.1 Method

We first trained the original Collins model on a modified versions of the training test from Experiment 1 in which the PPs were split by applying rule (7).

In a second series of experiments, we tested a range of models that use sister-head dependencies instead of head-head dependencies for different categories. We first added sister-head dependencies for NPs (following Collins's (1997) original proposal) and then for PPs, which are flat in Negra, and thus similar in structure to NPs (see Section 2.2). Then we tested a model in which sister-head relationships are applied to all categories.

In a third series of experiments, we trained models that use sister-head relationships everywhere except for one category. This makes it possible to determine which sister-head dependencies are crucial for improving performance of the model.

5.2 Results

The results of the PP experiment are listed in Table 5. Again, we give results obtained using TnT tags and using perfect tags. The row 'Split PP' contains the performance figures obtained by including split PPs in both the training and in the testing set. This leads to a substantial increase in LR (6–7%) and LP (around 8%) for both tagging schemes. Note, however, that these figures are not directly comparable to the performance of the unmodified Collins model: it is possible that the additional brackets artificially inflate LR and LP. Presumably, the brackets for split PPs are easy to detect, as they are always adjacent to a preposition. An honest evaluation should therefore train on the modified training set (with split PPs), but collapse the split categories for testing, i.e., test on the unmodified test set. The results for this evaluation are listed in rows 'Collapsed PP'. Now there is no increase in performance compared to the unmodified Collins model; rather, a slight drop in LR and LP is observed.

Table 5 also displays the results of our experiments with the sister-head model. For TnT tags, we observe that using sister-head dependencies for NPs leads to a small decrease in performance compared to the unmodified Collins model, resulting in 67.84% LR and 65.96% LP. Sister-head dependencies for PPs, however, increase performance substantially to 70.27% LR and 68.45% LP. The highest improvement is observed if head-sister dependencies are used for all categories; this results in 71.32% LR and 70.93% LP, which corresponds to an improvement of 3% in LP and 5% in LR compared to the unmodified Collins model. Performance with perfect tags is around 2–4% higher than with TnT tags. For perfect tags, sister-head dependencies lead to an improvement for NPs, PPs, and all categories.

The third series of experiments was designed to determine which categories are crucial for achieving this performance gain. This was done by training models that use sister-head dependencies for all categories but one. Table 6 shows the change in LR and LP that was found for each individual category (again for TnT tags and perfect tags). The highest drop in performance (around 3%) is observed when the PP category is reverted to head-head dependencies. For S and for the coordinated categories (CS,

	TnT tagging						Perfect tagging					
	LR	LP	CBs	0CB	≤2CB	Cov	LR	LP	CBs	0CB	≤2CB	Cov
Unmod. Collins	67.91	66.07	0.73	65.67	89.52	95.21	68.63	66.94	0.71	64.97	89.73	96.23
Split PP	73.84	73.77	0.82	62.89	88.98	95.11	75.93	75.27	0.77	65.36	89.03	93.79
Collapsed PP	66.45	66.07	0.89	66.60	87.04	95.11	68.22	67.32	0.94	66.67	85.88	93.79
Sister-head NP	67.84	65.96	0.75	65.85	88.97	95.11	71.54	70.31	0.60	68.03	93.33	94.60
Sister-head PP	70.27	68.45	0.69	66.27	90.33	94.81	73.20	72.44	0.60	68.53	93.21	94.50
Sister-head all	71.32	70.93	0.61	69.53	91.72	95.92	73.93	74.24	0.54	72.30	93.47	95.21

Table 5: Results for Experiment 2: performance for models using split phrases and sister-head dependencies

CNP, etc.), a drop in performance of around 1% each is observed. A slight drop is observed also for VP (around 0.5%). Only minimal fluctuations in performance are observed when the other categories are removed (AP, AVP, and NP): there is a small effect (around 0.5%) if TnT tags are used, and almost no effect for perfect tags.

5.3 Discussion

We showed that splitting PPs to make Negra less flat does not improve parsing performance if testing is carried out on the collapsed categories. However, we observed that LR and LP are artificially inflated if split PPs are used for testing. This finding goes some way towards explaining why the parsing performance reported for the Penn Treebank is substantially higher than the results for Negra: the Penn Treebank contains split PPs, which means that there are lot of brackets that are easy to get right. The resulting performance figures are not directly comparable to figures obtained on Negra, or other corpora with flat PPs.[2]

We also obtained a positive result: we demonstrated that a sister-head model outperforms the unlexicalized baseline model (unlike the C&R model and the Collins model in Experiment 1). LR was about 1% higher and LP about 4% higher than the baseline if lexical sister-head dependencies are used for all categories. This holds both for TnT tags and for perfect tags (compare Tables 2 and 5). We also found that using lexical sister-head dependencies for all categories leads to a larger improvement than using them only for NPs or PPs (see Table 5). This result was confirmed by a second series of experiments, where we reverted individual categories back to head-head dependencies, which triggered a decrease in performance for all categories, with the exception of NP, AP, and AVP (see Table 6).

On the whole, the results of Experiment 2 are at odds with what is known about parsing for English. The progression in the probabilistic parsing literature has been to start with lexical head-head dependencies (Collins, 1997) and then add non-lexical sister information (Charniak, 2000), as illustrated in Table 4. Lexical sister-head dependencies have only been found useful in a limited way: in the original Collins model, they are used for non-recursive NPs.

	TnT tagging		Perfect tagging	
	ΔLR	ΔLP	ΔLR	ΔLP
PP	−3.45	−1.60	−4.21	−3.35
S	−1.28	0.11	−2.23	−1.22
Coord	−1.87	−0.39	−1.54	−0.80
VP	−0.72	0.18	−0.58	−0.30
AP	−0.57	0.10	0.08	−0.07
AVP	−0.32	0.44	0.10	0.11
NP	0.06	0.78	−0.15	0.02

Table 6: Change in performance when reverting to head-head statistics for individual categories

Our results show, however, that for parsing German, lexical sister-head information is more important than lexical head-head information. Only a model that replaced lexical head-head with lexical sister-head dependencies was able to outperform a baseline model that uses no lexicalization.[3] Based on the error analysis for Experiment 1, we claim that the reason for the success of the sister-head model is the fact that the rules in Negra are so flat; using a sister-head model is a way of binarizing the rules.

6 Comparison with Previous Work

There are currently no probabilistic, treebank-trained parsers available for German (to our knowledge). A number of chunking models have been proposed, however. Skut and Brants (1998) used Negra to train a maximum entropy-based chunker, and report LR and LP of 84.4% for NP and PP chunking. Using cascaded Markov models, Brants (2000) reports an improved performance on the same task (LR 84.4%, LP 88.3%). Becker and Frank (2002) train an unlexicalized PCFG on Negra to perform a different chunking task, viz., the identification of topological fields (sentence-based chunks). They report an LR and LP of 93%.

The head-lexicalized model of Carroll and Rooth (1998) has been applied to German by Beil et al.

[2]This result generalizes to Ss, which are also flat in Negra (see Section 2.2). We conducted an experiment in which we added an SBAR above the S. No increase in performance was obtained if the evaluation was carried using collapsed Ss.

[3]It is unclear what effect *bi*-lexical statistics have on the sister-head model; while Gildea (2001) shows bi-lexical statistics are sparse for some grammars, Hockenmaier and Steedman (2002) found they play a greater role in binarized grammars.

(1999, 2002). However, this approach differs in the number of ways from the results reported here: (a) a hand-written grammar (instead of a treebank grammar) is used; (b) training is carried out on unannotated data; (c) the grammar and the training set cover only subordinate and relative clauses, not unrestricted text. Beil et al. (2002) report an evaluation using an NP chunking task, achieving 92% LR and LP. They also report the results of a task-based evaluation (extraction of sucategorization frames).

There is some research on treebank-based parsing of languages other than English. The work by Collins et al. (1999) and Bikel and Chiang (2000) has demonstrated the applicability of the Collins (1997) model for Czech and Chinese. The performance reported by these authors is substantially lower than the one reported for English, which might be due to the fact that less training data is available for Czech and Chinese (see Table 1). This hypothesis cannot be tested, as the authors do not present learning curves for their models. However, the learning curve for Negra (see Figure 1) indicates that the performance of the Collins (1997) model is stable, even for small training sets. Collins et al. (1999) and Bikel and Chiang (2000) do not compare their models with an unlexicalized baseline; hence it is unclear if lexicalization really improves parsing performance for these languages. As Experiment 1 showed, this cannot be taken for granted.

7 Conclusions

We presented the first probabilistic full parsing model for German trained on Negra, a syntactically annotated corpus. This model uses lexical sister-head dependencies, which makes it particularly suitable for parsing Negra's flat structures. The flatness of the Negra annotation reflects the syntactic properties of German, in particular its semi-free wordorder.

In Experiment 1, we applied three standard parsing models from the literature to Negra: an unlexicalized PCFG model (the baseline), Carroll and Rooth's (1998) head-lexicalized model, and Collins's (1997) model based on head-head dependencies. The results show that the baseline model achieves a performance of up to 73% recall and 70% precision. Both lexicalized models perform substantially worse. This finding is at odds with what has been reported for parsing models trained on the Penn Treebank. As a possible explanation we considered lack of training data: Negra is about half the size of the Penn Treebank. However, the learning curves for the three models failed to produce any evidence that they suffer from sparse data.

In Experiment 2, we therefore investigated an alternative hypothesis: the poor performance of the lexicalized models is due to the fact that the rules in Negra are flatter than in the Penn Treebank, which makes lexical head-head dependencies less useful for correctly determining constituent boundaries. Based on this assumption, we proposed an alternative model hat replaces lexical head-head dependencies with lexical sister-head dependencies. This can the thought of as a way of binarizing the flat rules in Negra. The results show that sister-head dependencies improve parsing performance not only for NPs (which is well-known for English), but also for PPs, VPs, Ss, and coordinate categories. The best performance was obtained for a model that uses sister-head dependencies for all categories. This model achieves up to 74% recall and precision, thus outperforming the unlexicalized baseline model.

It can be hypothesized that this finding carries over to other treebanks that are annotated with flat structures. Such annotation schemes are often used for languages that (unlike English) have a free or semi-free wordorder. Testing our sister-head model on these languages is a topic for future research.

References

Becker, Markus and Anette Frank. 2002. A stochastic topological parser of German. In *Proceedings of the 19th International Conference on Computational Linguistics*. Taipei.

Beil, Franz, 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*. College Park, MA.

Beil, Franz, Detlef Prescher, Helmut Schmid, and Sabine Schulte im Walde. 2002. Evaluation of the Gramotron parser for German. In *Proceedings of the LREC Workshop Beyond Parseval: Towards Improved Evaluation Measures for Parsing Systems*. Las Palmas, Gran Canaria.

Bikel, Daniel M. and David Chiang. 2000. Two statistical parsing models applied to the Chinese treebank. In *Proceedings of the 2nd ACL Workshop on Chinese Language Processing*. Hong Kong.

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

Carroll, Glenn and Mats Rooth. 1998. Valence induction with a head-lexicalized PCFG. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Granada.

Charniak, Eugene. 1993. *Statistical Language Learning*. MIT Press, Cambridge, MA.

Charniak, Eugene. 1997. Statistical parsing with a context-free grammar and word statistics. In *Proceedings of the 14th National Conference on Artificial Intelligence*. AAAI Press, Cambridge, MA.

Charniak, Eugene. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 1st Conference of the North American Chapter of the Association for Computational Linguistics*. Seattle.

Collins, Michael. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and the 8th Conference of the European Chapter of the Association for Computational Linguistics*. Madrid.

Collins, Michael, Jan Hajič, Lance Ramshaw, and Christoph Tillmann. 1999. A statistical parser for Czech. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*. College Park, MA.

Gildea, Daniel. 2001. Corpus variation and parser performance. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Pittsburgh.

Hockenmaier, Julia and Mark Steedman. 2002. Generative models for statistical parsing with combinatory categorial grammar. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia.

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

Schmid, Helmut. 2000. LoPar: Design and implementation. Ms., Institute for Computational Linguistics, University of Stuttgart.

Skut, Wojciech and Thorsten Brants. 1998. A maximum-entropy partial parser for unrestricted text. In *Proceedings of the 6th Workshop on Very Large Corpora*. Montréal.

Skut, Wojciech, 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.

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