Experiments in German Noun Chunking

Michael Schiehlen
Institute for Computational Linguistics, University of Stuttgart,
Azenbergstr. 12, 70174 Stuttgart
mike@adler.ims.uni-stuttgart.de

Abstract

The paper describes a method to process recursive noun phrases with finite-state cascades. It is shown that chunking of recursive noun phrases necessitates a readjustment of the finite-state cascades approach. In particular, the property of monotonicity must be given up. Furthermore, the paper explores the influence of POS tags and online agreement checking on the overall performance.

1 Introduction

Finite-state parsers (Abney, 1997) rank among the fastest parsers to date. Such parsers facilitate the linguistic processing of large amounts of text. Furthermore, it has been shown that finite-state parsers can be used to extract predicate-argument structure at least for complement relations. Since arguments are typically expressed by NPs or PPs, identifying NPs and PPs is an important step in determining predicate-argument structure. In German, grammatical roles are not encoded in relative string positions but rather in case and number information on NPs and PPs. Thus, at least for German, identification of case and number is a prerequisite for the determination of predicate-argument structure, and the computation of agreement features needs to be explicitly addressed. Another characteristic of German is its common[1] usage of center-embedding in noun phrases. The combination of agreement checking and center embedding makes sure that, in contrast to English, determination of (full) noun chunks requires not one, but a *cascade* of finite-state automata. An iterative application of finite-state automata which all serve to match the same type of chunks leads to certain problems, since tokens may be mistakenly assigned to chunks at the lower levels even if they should be inserted only at a higher level. Explicit representation of ambiguities at the individual levels would jeopardize the complexity class.

The paper is organized as follows. Section 2 discusses a variety of definitions that have been proposed for noun chunks in the literature. Section 3 describes the system used to parse full noun chunks and discusses possible strategies. Section 4 presents and discusses the results of the experiments made to evaluate the different strategies and compares the performance of the present system with those described in the literature. Section 5 concludes.

[1] In the test corpus used in the experiments (NEGRA, described below), as much as 6.3% of all maximal noun phrases show center-embedding.

2 Noun chunks

A noun chunk may be roughly defined as the kernel of a noun phrase, which implies that a noun chunk at least includes the head noun. But as often it is the details that cause the problems. Abney (1991) gives the following definition of chunks. A chunk is a substring of the sentence, and it has syntactic structure which comprises a connected subgraph of the sentence's parse tree. Every chunk is centered around a major head. A **major head** is a content word which does not appear between any function word f and the content word selected by f. Thus, in example (1a) *proud* does not qualify as a major head while it does in example (1b).

(1) a. the proud man

b. a man proud of his son

The **root** of a chunk is the highest node in the parse tree which has the chunk's major head as its semantic head. Now we can formally define a chunk. A **chunk** is a maximal string of words which consists of the chunk's major head and words that are dominated by the chunk's root but are not contained in other chunks. Note that a word dominated by the root is not automatically part of the chunk even if it is not contained in other chunks. A case in point is the preposition *in* in example (2) which is dominated by the root of the *house* chunk and not included in any other chunk. Inclusion of *in* in the chunk would destroy its string property.

(2) $[_{PP}$ in $[_{NP}$ John's] house] $\rightarrow$ in {John's} {house}

Abney's definition goes so far. Some amendments are in order.

Note e.g. that example (2) gives a counterexample to the definition of major heads: The name *John* is a major head although it comes between a function word (*in*) and its complement (*house*). Thus, we take it that nouns and names are major heads in all contexts.

Often a determiner is missing (cf. example (3a)) or a head noun is elided (cf. example (3b)). If null determiners and empty nouns are not assumed, the pre-nominal adjectives (e.g. *poor*) will form chunks of their own in this case. Thus, we take it that the syntactic theory underlying Abney's definition comes with null determiners and empty categories.

(3) a. ∅ poor people
b. the poor ∅

A more serious problem is coordination. If major heads are coordinated there is a single projection which has multiple semantic heads. In particular, the chunks corresponding to the coordinated content words all have the same root. They either exclude each other or collapse to a single chunk. The second choice is obviously much preferable and yields the chunk in example (4).

(4) {the old men and women}

If we want to enforce the requirement that every chunk contains no more than one major head, an option is to exclude from chunks all conjunctions and commas coordinating major heads. The conjunction *and* in example (5) is such a case. If conjunctions between major heads are excluded, Abney's definition predicts two separate chunks in example (5), one for *men* and one for *women*. The rest of the paper adopts this second approach.

(5) {the old men} and {women}

The last problem is the definition of noun chunks. **Noun chunks** are defined as those chunks whose major head is a noun. Since prepositions are treated as function words, the content word in a prepositional phrase is the noun. Thus, prepositional phrases also count as noun chunks. Note that there is also an independent reason in German to treat prepositional phrases and nominal phrases simultaneously: Prepositions and definite articles can be contracted, see e.g. *im* in example (7).

The literature on German noun chunking does not agree on a single definition of noun chunks. Abney's definition of noun chunks is just one among many and is followed mainly in finite-state approaches (Abney, 1997) (Neumann et al., 1997). Henceforth, noun chunks in Abney's sense will be called **base noun chunks**. A principal reason for discontent with Abney's definition is the fact that German routinely[2] allows NPs and PPs in prenominal adjective phrases (cf. example (6)). Since this is an instance of center embedding, the usual techniques of underspecification in chunking cannot be applied.

(6) Die vom Bundesgerichtshof
the by the German Federal High Court
und den Wettbewerbshütern
and the guards against unfair competition
als Verstoß gegen das Kartellverbot
as infringement of anti-cartel legislation
gegeißelte zentrale TV-Vermarktung
censured central television marketing
ist gängige Praxis.
is common practice.
Central television marketing, censured by the German Federal High Court and the guards against unfair competition as an infringement of anti-cartel legislation, is common practice.

Furthermore, base noun chunks may correspond to ungrammatical constituents. So the base noun chunk *nachlassenden Kräfte* cannot be used on its own, i.e. without an opening determiner.

(7) die {im Alter} {nachlassenden Kräfte}
the in the age diminishing strength
the strength diminishing in old age

Let us have a look at alternative ways to define noun chunks. Schmid and Schulte im Walde (2000) define a noun chunk as the part of a noun phrase between determiner and (first) head noun. They go on to give an explicit definition of the noun chunks they consider: A noun chunk may be a combination of a non-obligatory determiner, optional adjectives or cardinals and the noun itself; a nominalized adjective; a pronoun, a proper name, or a cardinal indicating a year; a noun chunk refined by a proper name (see example (8)). Coordinated noun phrases and appositions are explicitly excluded.

(8) der Eroberer Christoph Kolumbus
the conqueror Christopher Columbus

Skut and Brants (1998a) (1998b) and Brants (1999) give a negative definition of noun chunks: A noun chunk is a NP or PP stripped of prenominal adverbials and postnominal PPs and relative clauses. Thus they implicitly allow coordinated NPs and appositions, but also prenominal and postnominal genitives (9a), prenominal measure phrases (9b), and conjunctions of prepositions, determiners, adjectives, nouns.

(9) a. Marias Version der Geschichte
Mary's version of the story

b. zwei Millionen Dollar Strafe
two millions dollar penalty
a penalty of two million dollars

[2] In the NEGRA treebank (340,000 tokens of newspaper text) 28.2% of all adjective phrases contain an NP or PP.

Noun chunks in this sense will be called **full noun chunks**. Note that coordination may lead to attachment ambiguities which cannot be resolved in the syntax (see example (10)).

(10) {?parts {?of Scotland and Northern Ireland}

Thus full noun chunks incorporate an aspect of syntactically irresolvable ambiguity.

3 System Description

3.1 Finite-State Cascades

For the implementation of the noun chunker described in this paper, Abney's (1997) method of finite-state cascades is adopted. In this approach the parse tree is built deterministically from the leaf nodes to root. The nodes in the parse tree are partitioned into a number of levels (see Figure 1 for an example). At each level, a finite-state automaton is applied which specializes in recognizing a certain type of chunks (e.g. noun chunks or clause chunks). Identified chunks are replaced by single tokens, and the resulting input stream is presented to the next finite-state automaton.

Ambiguities are dealt with in three ways: 1. Lexical ambiguities are resolved by a POS tagger. 2. Attachment ambiguities are kept underspecified as far as possible. Modifiers in ambiguous positions are inserted only at clause level. 3. All other ambiguities are resolved using the longest-match criterion, which predicts that chunks should be chosen so as to be as long as possible. See Abney (1993) for a defense of this heuristic.

3.2 Agreement Checking

German poses a particular challenge in that grammatical roles are not determined by syntactic positions but rather by case. Hence it is essential to compute case as accurately as possible. Since case varies with other agreement features like gender, number, and adjective declination, full computation of agreement ensures better results[3]. The matter is complicated by the fact that the agreement features of most case-bearing elements (determiners, appositive adjectives, nouns) are extremely ambiguous; only in interaction these features put constraints on case selection. There are several ways to integrate agreement checking into finite-state parsing:

1. The POS tags serving as input to the the finite-state automaton are equipped with disjunctive agreement information. The finite-state grammar is updated so as to handle such POS tags. Care is taken to exclude sequences of POS tags that lead to agreement failures. The result of agreement checking is expressed in the final states. An advantage of this approach is that the finite-state techniques ensure that agreement is checked as soon as possible. Furthermore the resulting finite-state automata can be composed and reversed at will. The main drawback is an explosion of the transition table by a factor in the order of magnitude of 500.

2. Another approach leaves the grammars and automata as they were and evaluates chunks for agreement only in a post-processing step (Abney, 1997). This method has the drawback that agreement failure has no immediate effect on the determination of noun chunks. For the sentence of Figure 1, a finite-state parser ignoring agreement and only using the longest-match criterion would produce *die Anfang der Rechnungsperiode* as noun chunk.

3. Finally it is possible to interleave agreement checking and chunk recognition (Neumann et al., 2000), taking into account agreement failures as soon as possible. Often such a procedure is straightforward: Every preposition, determiner, adjective, noun just contribute their agreement constraints. There are, however, some problems if the agreement constraints cannot be assigned on the basis of one word alone, as is the case with circumpositions (in (11) *um–willen* takes genitive, although *um* only allows accusative)

(11) um Gottes willen
for God's sake

and words that are ambiguous between determiners and pronouns (in the garden path sentence (12) *eines* is genitive as a determiner, but nominative or accusative as a pronoun).

(12) weil diese Erfahrung eines
because this experience a/one
Bauern klarmacht
farmer makes clear
because this experience makes clear one thing to farmers

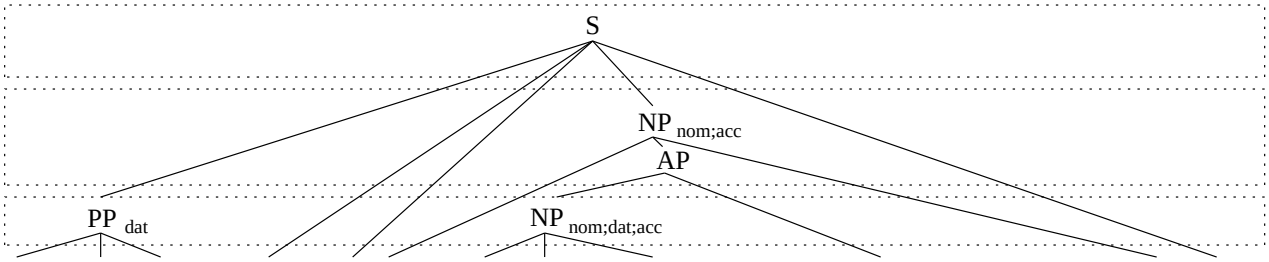
For building the finite-state parser, the second and third approach were tested and compared (see Section 4). Agreement features were implemented in bit vectors to facilitate unification.

3.3 Parsing Full Noun Chunks

None of the German finite-state parsers discussed in the literature (Abney, 1997) (Neumann et al., 1997) attempts to treat full noun chunks. Indeed there are major problems in using finite-state cascades for full noun chunks. Figure 2 shows the level of base noun chunks for the sentence in Figure 1. Two noun chunks are recognized that do not show up in the final analysis. Also in example (13), repeated from (2), a noun chunk is recognized (*{house}*) without appearing in the final analysis.

(13) in {John's} {house}

[3] Schmid and Schulte im Walde (2000) only compute case to keep the number of parameters down.



The price rises implemented at the beginning of the account period had lain in this range, too.

Figure 1: Levels in a Syntax Tree

Figure 2: Level of Base Noun Chunks

Two ways to cope with the dilemma come to mind.
1. Either the idea of getting by with finite-state techniques is given up, and a pushdown automaton is employed. There is, however, no general algorithm to render pushdown automata deterministic and minimal. Thus, some effort has to go into identifying disambiguation factors for the decision when to start and end an embedded noun chunk.
2. The other approach introduces non-monotonicity into the cascade of finite-state automata in the sense that base noun chunks recognized at one level are discarded at another. To implement this idea, base noun chunk are classified as to whether they cover material that could form the start, a middle part, or the end of a full noun chunk. (E.g., a base noun chunk without determiner and preposition might be the end of a full noun chunk.) A special finite-state automaton is built which puts together base noun phrases and other material to full noun phrases. (E.g., a full noun chunk might consist of a preposition (*in* in example (13)), a genitive noun chunk (*{John's}*), and a base noun chunk classified as potential end of a full noun chunk (*{house}*).) If the automaton finds a match, the relevant base noun chunks (e.g. *house* in the example discussed) are broken up into their components. Now the noun chunk recognizer is again applied. If the full noun chunk is morphologically well-formed, it is recognized in this step. Otherwise the chunker reassembles the original base noun chunks.

Chunking approaches that are not based on finite-state technology have no problems with full noun chunks. Skut and Brants (1998a) (1998b) use a POS tagger to infer bracketed full noun chunks, while Schmid and Schulte im Walde (2000) make use of a stochastic context-free parser. Brants (1999) also employs a cascade (of Markov Models). He circumvents the problems mentioned by allowing *ambiguous* layers and thus effectively availing himself of a chart. The approach of Kermes and Evert (2002) is rule-based, works in cascades and explicitly represents ambiguities at intermediate levels.

4 Experiments

In order to find grounds to empirically choose between strategies and to compare the performance of the present system with other chunkers described in the literature, several experiments were performed. The gold standard was extracted from the NEGRA treebank, a collection of syntactically annotated German newspaper articles (Skut et al., 1997). The current version of this tree bank provides syntactic analyses for 321,000 tokens. Skut and Brants (1998a) and Brants (1999) evaluated their chunkers on the same corpus, which contained less data at that time (210,000 tokens in 1998 and 300,000 in 1999). From this corpus 100,974 base noun chunks and 78,942 full noun chunks were extracted. Only 19.9% of the full noun chunks are recursive, 6.3% are center embedding structures. Noun chunks were extracted using a quite sophisticated Perl script; they were not checked for correctness by hand, however. The internal structure of full noun chunks was not taken into account. Agreement information is not

given in the tree bank and it was not attempted to extract it e.g. via evaluation of grammatical roles.

As a baseline the machine-learning approach of Ramshaw and Marcus (1995) was used which does not require any manual effort in grammar writing. In this approach a tagger is used to divide the tokens into three classes: `I` inside a noun chunk; `O` outside a noun chunk; `B` beginning of a new noun chunk if previous token was inside a noun chunk. Ramshaw and Marcus (1995) used the Brill tagger, while in the experiments described here the tree tagger of Schmid (1994) was employed which was at hand. On the training and test data of Ramshaw and Marcus (1995) (sections 15–18 of the Wall Street Journal for training, section 20 for testing), the tree tagger reaches a precision value of 90.7% and a recall of 91.2%. Ramshaw and Marcus (1995) get 91.8% precision and 92.3% recall with the Brill tagger. Thus, the tree tagger performs a bit worse, presumably because it only takes into account the current word and POS tag and the two POS tags to the left. The Brill tagger additionally inspects the two words to the left and the two words and POS tags to the right. The baseline values reported below were determined using 10-fold cross validation.

The finite-state grammar recognizing noun chunks was written so as to match noun chunks in a left-to-right parse. The same grammar can be automatically reversed and then used in a right-to-left parse. Note that the longest-match criterion may lead to different results if the automaton reads the string from right to left (cf. example (14)).

(14) der 14 Jahre alte Junge
the 14 years old boy
the 14-year-old boy
→ {der 14} {Jahre} {alte Junge}
← {der} {14 Jahre} {alte Junge}

Since in German heads (e.g. verbs and adjectives) usually follow their complements, it is imaginable that parsing right-to-left leads to improvements even if a right-to-left parse is psychologically implausible. To test this hypothesis, the performance of the finite-state parser in a right-to-left traversal was determined in the experiments as well.

As discussed in section 3.2, it might also improve performance to check agreement during recognition. Thus in a separate battery of tests, the performance gain involved in checking agreement online was measured.

4.1 Quality of POS Tagging

Many approaches to noun chunking rely on a POS tagger to disambiguate lexical ambiguities (e.g. Abney's finite-state cascades, Ramshaw and Marcus's tagging approach, and machine learning approaches in general (Tjong Kim Sang et al., 2000)). Only a small number of approaches (Brants, 1999) (Schmid and Schulte im Walde, 2000) determines lexical categories and chunks in one go. Presumably the overall performance depends on the quality of the tagging results used. Four grades of quality can be distinguished.

1. The POS tags used are "ideal", i.e. supplied by the tree bank. In this case, tagging accuracy is 100%.
2. The POS tags are determined by a tagger, which is trained on the tree bank, so it has an ideal lexicon.
3. The POS tags are determined by a tagger, but the tagger is trained on some independent corpus.
4. Disjunctions of POS tags are used. The noun chunker selects the tags on its own. In a finite-state approach, the longest-match criterion controls this selection.

After training on the NEGRA tree bank, the tree tagger determined 98.77% of all POS tags correctly. Without training, it only reached an accuracy of 94.73% on the NEGRA tree bank[4], mostly due to unknown proper names. In the tables given below, option 1 (ideal tags) is abbreviated by POS-I, option 2 (tags with ideal lexicon) by POS-L, option 3 (tags by a POS tagger) is written as POS-T, and option 4 (tags determined by chunker) as POS-C.

4.2 Base Noun Chunks

First the results obtained for the base noun chunking task will be discussed (see Figure 3).

The same method ((Ramshaw and Marcus, 1995) with the tree-tagger) achieves better results for German (94.5% precision and 92.9% recall) than for English[5] (90.7% precision and 91.2% recall). Obviously the base noun chunking task is easier in German. A reason may be that German nouns are on average less ambiguous than their English counterparts: On average a potential noun has 1.02 readings in German (in the NEGRA tree bank) but 1.20 readings in English (in the WSJ sections).

The modes of POS tagging performed as predicted: The tree-bank tags came out best, but were followed closely by the tags of the tagger trained on the tree bank. Third were the tags determined without lexical information. The performance of the fourth strategy (the chunker selects the tags according to the longest-match criterion) was disappointing and even below the baseline. The moral is that at least for finite-state parsers, using a POS tagger to filter out lexical readings beforehand is a good idea.

The direction of processing (left to right → or right to left ←) did not make much difference.

[4] Kermes and Evert (2002) report a similar figure (94.82%).

[5] The experiment done on the Wall Street Journal used the POS tags of the treebank.

	baseline		FS →		FS ←		FS+agrm →		FS+agrm ←	
	prec	recall	prec	recall	prec	recall	prec	recall	prec	recall
POS-I	94.55	92.99	99.06	98.99	99.05	99.01	99.16	99.16	99.19	99.22
POS-L	90.65	91.61	98.33	97.58	98.31	97.59	98.43	97.81	98.46	97.88
POS-T	91.30	89.48	95.35	93.66	95.33	93.67	95.39	93.88	95.45	93.96
POS-C			90.67	86.42	88.93	86.21	90.54	88.91	88.96	87.05

Figure 3: Results for Base Noun Chunking

	baseline		FS →		FS ←		FS+agrm →		FS+agrm ←	
	prec	recall	prec	recall	prec	recall	prec	recall	prec	recall
POS-I	88.63	87.50	97.36	90.16	97.41	90.06	97.66	91.44	97.65	94.06
POS-L	86.05	84.73	96.37	88.42	96.39	88.32	96.57	89.76	96.61	92.40
POS-T	86.00	84.88	90.83	83.61	90.87	83.53	91.00	84.79	91.11	87.10

Figure 4: Results for Full Noun Chunking

Checking agreement produced a small improvement. In checking agreement for base noun chunks, adjective declination was neglected (see example (7)).

4.3 Full Noun Chunks

Figure 4 shows the results for the full noun chunk task. As can be seen in the baseline results, the task is much more demanding, in part because it involves a degree of forced guessing (see example (10)).

For the full noun chunk task, more results have been published. Skut and Brants (1998b) report a precision value of 89.0% on ideal POS tags. They used our baseline technique and got similar results. Skut and Brants (1998a), who use a maximum entropy model, reach 93.4% precision and 94.1% recall on ideal POS tags. Brants (1999) uses a cascade of Markov models, but he gives no results for the task examined here (determining the outer boundaries of a full noun chunk). Schmid and Schulte im Walde (2000) report results of up to 93.29% precision and 92.19% recall. They also give figures for the task of additionally identifying case and category. However, their parser has access to information like case constraints of governing verbs and lexicalized probability distributions for grammatical relations, which it uses to determine a *unique* case. The system described here lacks such information and thus outputs a *disjunction* of case and number values. Kermes and Evert (2002) also evaluated the performance of their system using the NEGRA treebank, but got considerably worse results (84.74% precision, 82.36% recall) for unclear reasons.

While the precision values in Figure 4 approximately stay the same for all configurations, a certain tendency can be seen in the recall values: Recall improves with agreement checking and even more with right-to-left traversal. The most frequent errors made without but not with agreement checking are the following:

- genitives after prepositions (see example (13), repeated here as (15)): The recognizer finds a noun chunk which is morphologically impossible.

(15) {in John's} {house}

- conjunction attachment, see example (16): Sometimes agreement excludes a wide attachment which is favoured by the longest-match criterion.

(16) {das Leben {von Schauspielern}
the life of actors
und Zirkusleuten}
and circus people

- adjoining noun phrases: In the example presented (17) adjective declination is used to determine the chunk boundary.

(17) {diese beiden} {ähnliche Erfolge}
these two similar successes

There are, however, some errors that only occur if agreement is checked. The set-up of the system only allows for NP coordination but not N′ coordination. In some cases adjective declination distinguishes N′ from NP with null determiner (*nachlassenden Kräfte* versus *nachlassende Kräfte* in example (18), cf. (7)). These cases are lost only if agreement checking is switched on.

(18) die [Verletzungen und nachlassenden Kräfte]$_N$
the injuries and diminishing strength

The gain in recall observed when parsing right-to-left is mainly due to a heuristic that could only be incorporated into the right-to-left chunker: In case of conjunction attachment, shortest match is preferred. An illustration can be seen in example (19).

(19) → I saw {a man with a telescope in the garden and in the house}.
← I saw a man with a telescope {in the garden and in the house}.

5 Conclusion

The paper has presented a method to recognize and morphologically interpret noun chunks. A distinction has been made between *base noun chunks* (non-recursive) and *full noun chunks* (potentially recursive). Two chunkers have been constructed, one recognizes base noun chunks, the other identifies full noun chunks. On a SUN Ultra-250, the base noun chunker can process 12,500 words/second, while the full noun chunker achieves 5,200 words/second. In contrast to other full noun chunkers proposed in the literature (Brants, 1999) (Schmid and Schulte im Walde, 2000), the chunker described here is completely deterministic and does not rely on an explicit representation of ambiguities (like a chart). In the experiments, it has been shown that the labour of writing a grammar, which is needed in the finite-state approach, indeed leads to improvements in performance. It has further been shown that the longest-match criterion is a bad substitute for a POS tagger and that a POS tagger is useful as a preprocessing tool for finite-state parsers. Online agreement checking improves performance, more so for the full noun chunking task than for base noun chunking. The direction of processing (left-to-right or right-to-left) did not lead to clear distinction, although right-to-left performed consistently better if combined with agreement checking.

The best treatment of coordination (and apposition for that matter) is still an open question. Since coordination attachment depends on factors that are not accessible to a parser, it should be underspecified like pre- or post-nominal modifiers. There is, however, a major difference between coordination attachment and modifier attachment in that coordination constrains agreement. Thus the choice of attachment point for coordination may have consequences for the selection of grammatical roles and hence for the predicate-argument structure even if it is reduced to head-complement-relations.

The noun chunker described in the paper is intended to be used as a submodule of a finite-state parser for German newspaper text. It is planned to convert the output of this parser into underspecified semantic representations on which simple inferences can be drawn as needed e.g. for tasks like Information Extraction.

References

Steven Abney. 1991. Parsing by Chunks. In Robert C. Berwick, Steven P. Abney, and Carol Tenny, editors, *Principle-based Parsing: computation and psycholinguistics*, pages 257–278. Kluwer Academic Publishers, Dordrecht, Holland.

Steven Abney. 1993. Reliability. In *Abstracts, Deutsche Gesellschaft für Sprachwissenschaft.*

Steven Abney. 1997. Partial Parsing via Finite-State Cascades. *Journal of Natural Language Engineering*, 2(4):337–344.

Thorsten Brants. 1999. Cascaded Markov Models. In *Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics (EACL'99)*, Bergen, Norway.

Hannah Kermes and Stefan Evert. 2002. YAC – A Recursive Chunker for Unrestricted German Text. In *Proceedings of LREC*, pages 1805–1812.

Günter Neumann, Rolf Backofen, Judith Baur, Markus Becker, and Christian Braun. 1997. An Information Extraction Core System for Real World German Text Processing. In *Proceedings of the 5th International Conference of Applied Natural Language Processing (ANLP'97)*, pages 208–215, Washington, DC.

Günter Neumann, Christian Braun, and Jakub Piskorski. 2000. A Divide-and-Conquer Strategy for Shallow Parsing of German Free Text. In *Proceedings of the 6th International Conference of Applied Natural Language Processing (ANLP'00)*, pages 239–246, Seattle, WA.

Lance A. Ramshaw and Mitchell P. Marcus. 1995. Text Chunking Using Transformation-Based Learning. In *Proceedings of the 3rd ACL Workshop on Very Large Corpora.*

Helmut Schmid and Sabine Schulte im Walde. 2000. Robust German Noun Chunking With a Probabilistic Context-Free Grammar. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING '00)*, August.

Helmut Schmid. 1994. Probabilistic Part-Of-Speech Tagging Using Decision Trees. Technical report, Institut für maschinelle Sprachverarbeitung, Universität Stuttgart, Germany.

Wojciech Skut and Thorsten Brants. 1998a. A Maximum-Entropy Partial Parser for Unrestricted Text. In *Proceedings of the 6th Workshop on Very Large Corpora*, Montréal, Québec.

Wojciech Skut and Thorsten Brants. 1998b. Chunk Tagger - Statistical Recognition of Noun Phrases. In *Proceedings of the ESSLLI-98 Workshop on Automated Acquisition of Syntax and Parsing*, Saarbrücken.

Wojciech Skut, Brigitte Krenn, Thorsten Brants, and Hans Uszkoreit. 1997. An Annotation Scheme for Free Word Order Languages. In *Proceedings of the ANLP-97*, Washington, DC.

Erik F. Tjong Kim Sang, Walter Daelemans, Hervé Déjean, Rob Koeling, Yuval Krymolowski, Vasin Punyakanok, and Dan Roth. 2000. System Combination to Base Noun Phrase Identification. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING '00)*, Saarbrücken, Germany.