Text analysis meets corpus linguistics

Hannah Kermes, Stefan Evert
Institut für Maschinelle Sprachverarbeitung,
Azenbergstr. 12, 70174 Stuttgart, Germany
{kermes,evert}@ims.uni-stuttgart.de

1 Introduction

In recent years, there has been rising interest to using evidence derived from automatic syntactic analysis in large-scale corpus studies. Ideally, of course, corpus linguists would prefer to have access to the wealth of structural and featural information provided by a full parser based on a complex grammar formalism. However, to date such parsers achieve neither the speed nor the robustness needed to process hundreds of millions of words. Beside this practical limitation, there are at least two more fundamental problems with this approach. Firstly, complex grammars tend to produce highly ambiguous output. Without extensive lexical and semantic knowledge, there will often be thousands of different analyses for any given sentence. Secondly, full parser usually embrace a particular theoretical perspective, embodied in the grammar formalism they use. If the researcher's perspective on syntax is different from that of the parser, he or she will find it difficult, if not outright impossible, to apply the parser's analyses to the research question.[1]

Chunk parsers, on the other hand, were expressly designed for the robust processing of large amounts of text, including sentence fragments. Robustness is achieved mostly by avoiding problematic decisions, such as where to attach prepositional phrases. As a consequence, the local structures built by chunk parsers are largely independent from any particular syntactic theory. However, they also provide much less evidence for the syntactic structure of sentences and for other linguistic phenomena. In this paper, we want to show that a chunk parser with an extended chunk definition can provide analyses of sufficient depth and enriched with morpho-syntactic as well as lexical-semantic information to answer such research questions. In order to do so, we need to identify the crucial requirements on text analysis for corpus linguistic research.

2 Requirements on the text analysis

We understand the crucial requirements for a useful tool for corpus linguistic applications to be the following: (i) It has to work on unrestricted text. There should be no limitation to corpus size, i.e., it should be able to deal with small as well as large corpora. It should be able to parse complete sentences as well as fragmentary text. The system should not be domain specific, i.e., it should work basically on any text type. Additional domain specific rules should be easy to incorporate in the grammar. (ii) Lacks in the grammar should not lead to a complete failure to parse. (iii) No manual checking should be required as it is not feasible for large quantities of text. (iv) The system should provide clearly defined and documented interfaces, where the extraction processes can attach. The annotation should make use of linguistic standards. There should be documentation on what is annotated, and how it is annotated. The annotation should be easy to process for further application.

What kind of information should a corpus annotation providing a useful basis for extractions include, beside of the information on token level: (i) the head lemma of annotated structures to determine the lemma of the lexical entry, (ii) morpho-syntactic information, to determine the grammatical function of the structure and to extract additional aspects about potential lexical entries, e.g., singular plural alternations for nouns, (iii) lexical or semantic information, e.g., temporal aspect, (iv) information about certain embeddings, text markers, or construction types, (v) hierarchical representations.

[1] (Kermes and Evert 2002) gives an idea of the problems we had using the supposedly theory-independent syntactic analyses from the German NEGRA treebank to evaluate the noun phrases identified by our chunker. This explains why statistical learning models, which are trained and evaluated on one and the same treebank, often outperform manually developed systems.

3 Full parsers vs. chunkers

3.1 Full parsers

Full parsers are based on a complex grammar that can be formulated in various frameworks (e.g., Lexical Functional Grammar (LFG) or Head Lexicalized Phrase Structure Grammars (HPSG)). These complex grammars are able to model hierarchical structures of a language. They are powerful enough to handle complex constraints about structures, relations, and attachments. Consequently, they are well suited to handle the problem of attachment ambiguities. In general, they make use of detailed knowledge about the function and usage of words to determine the correct analysis of a sentence. As a result a complete hierarchical annotation is delivered providing rich and complex information about structures, relations and functions. The rich and complex annotation provides an excellent base for extraction of linguistic information.

The complexity of the grammars and the use of detailed linguistic and lexicographic knowledge during the parse, however, slow down the parsing speed. Besides, if necessary information is lacking, full parsers fail to deliver an analysis. In most cases, even if only part of a sentence cannot be analyzed, the whole sentence is ignored. The lack of robustness is often caused by a lack of linguistic or lexicographic knowledge in the lexicon used by the parser. Full parsers depend heavily on a rich and detailed prerequisite lexicon that provides them with the necessary information.

The number of grammar rules produces ambiguities. In other words, the parser provides several possible analyses of which only one is correct. Some parsers use heuristic or statistical methods to determine the correct parse or to reduce the number of possible parses. Another possibility to overcome ambiguous or incorrect output is manual correction. As manual correction is time consuming and costly it is almost only used to built a reference corpus, usually a treebank. The limited size of the treebank makes it insufficient for large-scale extractions.

The complex rule system of the underlying grammar can also cause problems. Changes within the grammar can result in unexpected and undesired interactions among rules. It is usually difficult to determine the cause of the interaction because of the complexity of the rules and the rule system. Consequently, it is very complicated and time consuming to modify and adjust the existing grammar to a new text domain or purpose.

3.2 Chunkers

Another approach to text analysis, which has become more and more popular is chunking or chunk parsing. The structures annotated by chunkers are usually non-hierarchical and non-recursive. As the annotated structures are relatively simple, the grammar itself is also relatively simple. In general, it does not consider cases resulting in ambiguities such as, e.g., attachment decisions, or constructions including lexical dependencies. Rich and complex linguistic and lexicographic information is not required. As the name suggests, chunkers do not aim at annotating the structure of a sentence completely, but try to build "chunks" of words. Consequently, the rule system of chunkers is relatively simple, and they are very robust, i.e., they are not apt to fail to parse a sentence because they fail to parse part of the sentence. A chunker analyzes a text as far as it can, and annotates the results.

According to Steve Abney (Abney 1996a), a chunk is:

> A non-recursive core of an intra-clausal constituent, extending from the beginning of the constituent to its head.

Another definition by Abney (Abney 1991) is that:

> The typical chunk consists of a single content word surrounded by a constellation of function words, matching a fixed template.

Thus, the classic notion of a chunk is that of a flat, non-recursive structure. The chunk begins with a function word and ends with the lexical head. This definition excludes all post-head complements and modifiers. Consequently, some chunkers consider a prepositional phrase (PP) to consist only of the preposition itself, as PPs, in general, are head initial structures.

3.3 State-of-the-art systems

There are a variety of chunkers available for German:

CASS parser is a cascaded finite-state parser developed by Steven Abney (Abney 1996b). The German grammar developed within the Verbmobil project produces flat, non-recursive structures. The grammar includes a small lexicon, which is represented using so-called tag-fixes associating lexical classes with PoS-tags. Information about the head lemma of chunks is annotated as an attribute of the chunk. A demo version of the German grammar can be accessed via Internet.[2]

Connexor is a symbolic constraint grammar parser (Voutilainen 1994; Voutilainen and Jaervinen 1995; Voutilainen and Tapanainen 1993). The system was primarily built for English with a full-fledged grammar (ENGCG) (Karlsson *et. al.* 1995; Voutilainen 1997; Voutilainen and Heikkilae 1994). For German there is only a light version of the grammar, which produces simple, non-recursive structures. Lexical information is not available, however, the head lemma of chunks is indicated by a special tag. Unfortunately, the system is not freely available, only a demo version is accessible via Internet.[3]

KaRoParse developed by Frank H. Müller and Tylman Ule (Ule and Müller 2001; Müller and Ule 2001) is a symbolic top-down bottom-up partial parser. The partial analysis includes pre-head but no post-head recursion. The internal structure of a chunk is flat and non-hierarchical. Agreement information as well as lexical information about the chunks is not available. Additionally to lexical phrases they annotate so-called topological fields (Müller and Ule 2002; Veenstra, Müller, and Ule 2002), which divide a German sentence into different sections according to the topological field model of (Höhle 1986).

Chunkie is a partial parser that works with similar techniques as standard PoS taggers (Skut and Brants 1998). It uses the TnT tagger[4] (Brants 2000) to assign tree fragments to sequences of PoS tags. The most likely structure is determined using trigram frequencies. Chunkie produces structures with a maximal depth of three. It includes recursion in pre-head position but no post-head modifiers. Head lemma information is available via the structure. Agreement information and lexical information is not annotated.[5]

Cascaded Markov Models. The partial parser of Brants (Brants 1999) was used to facilitate the syntactic annotation of the NEGRA corpus[6] (Skut *et. al.* 1997). It is based on cascaded Markov models, and operates on several layers. The annotated structures are hierarchical phrases, including complex and recursive embedding. PPs and NP structures are stripped off adverbials at the front, and PPs and relative clauses at the end of the phrase. Head lemma information is available via the structure. Agreement and lexical-semantic information is not available.

4 Problems of the chunking approach

As Kübler and Hinrichs (Kübler and Hinrichs 2001) have pointed out, while chunking approaches have "focused on the recognition of partial constituent structures at the level of individual chunks [...], little or no attention has been paid to the question of how such partial analyses can be combined into larger structures for complete utterances."

In other words, combining chunk structures provided by (most of) the available chunkers often demands complex rules or rules that are neither secure nor theoretically motivated. For German, this is even more so than for English, for which most chunkers are designed. This is due to the fact that German has a propensity to recursive pre-head embedding of complex structures.

If we take the following example of a PP, with its full analysis in (1a) and compare it with the classical chunk analysis in (1b), the problems become obvious.

(1) a. [$_{PP}$ mit [$_{NP}$ [$_{AP}$ kleinen], [$_{AP}$ über [$_{NP}$ die Köpfe [$_{NP}$ der Apostel]] gesetzten] Flammen]]
with small above the heads of the apostles set flames
'with small flames set above the heads of the apostles'

[2] see http://gross.sfs.nphil.uni-tuebingen.de:8080/release/cass.html
[3] see http://www.connexor.com/demos.html
[4] see http://www.coli.uni-sb.de/~thorsten/tnt/
[5] see http://www.coli.uni-sb.de/~skut/chunker/
[6] see http://www.coli.uni-sb.de/sfb378/negra-corpus/negra-corpus.html

b. $[_{PC}$ mit $[_{NC}$ kleinen]], $[_{PC}$ über $[_{NC}$ die Köpfe]] $[_{NC}$ der Apostel] $[_{NC}$ gesetzten Flammen]
with small above the heads of the apostles set flames

The chunk analysis provides four NCs where there should be only one NP. The chunks overlap with the final structure analysis. This is due to the embedding of the complex AP structure in pre-head position of the maximal NP in (1a). In order to fill the gap between the chunk analysis in (1b) and full analysis in (1a), the PCs and NC in (1b) have to be combined.

The simple solution would be a rule as the following:

(2) PP $\rightarrow$ PC (PC | NC) $*$

This rule, however, does not seem theoretically motivated. It might seem logical to assume that a complex PP consists of a PC followed by a number of NCs. It is, however, difficult to find a theoretical background to include following PCs as well. PPs, usually, do not directly embed another PP or PC. For English, which does not involve embedding of complex structures in pre-head position as in (1a), as the translation demonstrates, the problem is not as severe, and a simpler rule involving PP-attachment can be sufficient.

The rule is rather vague and underspecified to an extent that it does not seem very reliable. The rule can easily assemble structures which are too large, and it seems difficult, if not impossible, to formulate restrictions.

Besides, the rule leaves the internal structure mainly opaque. It is, e.g., not obvious what the head of the NP embedded in the complex PP is. The relation between the NC *die Köpfe* (the heads) and the NC *der Apostel* (the apostles) is not indicated. Consequently, the head can equally well be *Apostel* (apostles) or *Flammen* (flames). In the former case, the NC *gesetzten Flammen* (set flames) would be a post-head genitive modifier. In the latter case, the NC *der Apostel* (the apostles) would be a pre-head modifier.

A more complex solution seem necessary, where a number of rules have to be applied:

(3) a. NP $\rightarrow$ NC NC$_{gen}$
b. PP $\rightarrow$ preposition NP
c. AP $\rightarrow$ PP adjective
d. NP $\rightarrow$ (AP)$*$ noun

The NP rule in (3a) will assemble the NC *die Köpfe* with the NC *der Apostel*. The PP rule in (3b) can built the PP *über die Köpfe der Apostel* as well as the complex PP in (1). The AP rule in (3c) can be used to form a complex AP with this PP (*über die Köpfe der Apostel gesetzten*). The NP rule in (3d) can built the complex NP (*die über die Köpfe der Apostel gesetzten Flammen*). Besides of these rules, the NC *gesetzten Flammen* has to be split up, to be able to construct the complex AP *über die Köpfe der Apostel gesetzten* and finally, the complex NP in which it is embedded.

The rules listed in (3) are able to build the hierarchical analysis in (1a). For other examples, one would probably need other or additional rules. The number of rules needed for this single example shows, that a classic chunk analysis leaves much of the parsing to further applications. The classic chunk structures do not seem to be appropriate to form a useful basis for extraction processes. In order to do so, the classic chunk concept has to be extended.

5 YAC – a recursive chunker for unrestricted German text

YAC is a fully automatic recursive chunker designed for unrestricted German text. It is based on a symbolic regular expression grammar written in the CQP query language (Christ *et. al.* 1999), which is evaluated by the query processor component of the IMS Corpus Workbench.[7] The chunker works on a corpus that is tokenized and part-of-speech tagged using the STTS tag set (Schiller *et. al.* 1999). For tokenization and PoS-tagging the TreeTagger[8] (Schmid 1994; Schmid 1995) is used. The German grammar additionally requires lemma and agreement information on token level, which is annotated using the IMSLex morphology (Lezius, Dipper, and Fitschen 2000).

In some respects YAC is a typical chunker: (i) it is robust, i.e., it works on unrestricted text, (ii) it works fully automatically, (iii) it does not provide a full but only a partial analysis of text, (iv) it does

[7] see http://www.ims.uni-stuttgart.de/projekte/CorpusWorkbench/
[8] see http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/DecisionTreeTagger.html

not make highly ambiguous attachment decisions. But YAC is different from other state-of-the-art chunkers in that it extends the classic chunk definition of Abney, and that it provides additional information together with the annotated chunks.

5.1 YAC's chunk definition

The classic chunk definition of Abney as a "non-recursive core of an intra-clausal constituent, extending from the beginning of the constituent to its head" (Abney 1996a) is extended by two main aspects: (i) recursive pre-head embedding, (ii) partial post-head embedding.

The chunks annotated by YAC are still intra-clausal constituents, however, they are no longer non-recursive but include recursive embedding in pre-head (4a) as well as in post-head position (4b). Post-head recursion automatically implies that the chunk does not end with the head but may have modifiers in post-head position. The English *of-* Phrases are often realized as post-head nominal genitive modifiers as is the case in (4b). Thus, it is useful if not necessary to attach them. Post-head modifiers can but do not necessarily have to have the same category as the chunk itself. In (4c), e.g., an adverbial phrase is embedded in an NP in post-head position.

(4) a. [$_{NP}$ die kleinen, über [$_{NP}$ die Köpfe der Apostel] gesetzten Flammen]
the small, above the heads of the apostles set flames
'the small flames set above the heads of the apostles'

b. [$_{NP}$ die Köpfe [$_{NP}$ der Apostel]]
the heads of the apostles

c. [$_{NP}$ Jahre [$_{AdvP}$ später]]
years later

There are, however, certain limitations with respect to post-head modifiers. YAC annotates only non-ambiguous constructions. Consequently, highly ambiguous attachment decisions, such as PP-attachment, are not made. Solving these ambiguity requires comprehensive lexical, linguistic, and context information, and in some cases world knowledge.

The chunk definition of Abney is extended and reformulated as follows:

(5) A chunk is a continuous part of an intra-clausal constituent including recursion and pre-head as well as post-head modifiers but not PP-attachment, or sentential elements.

The chunks are additionally enriched with head lemma and morpho-syntactic information as well as certain lexical-semantic properties.

5.2 Annotated structures

The structures annotated by YAC comprise the following lexical phrase categories:(i) adverbial phrases (AdvP), (ii) adjectival phrases (AP), (iii) noun phrases (NP), (iv) prepositional phrases (PP), (v) verbal complexes (VC), (vi) single verbs (V), (vii) subordinate clauses (CL).

Adverbial phrases are relatively simple constructions consisting of an adverb and an optional particle. In certain contexts, an optional adverbial modifier is allowed. Examples of possible AdvPs are given in (6).

(6) vielleicht (perhaps); zu bald (to soon); "sehr bald" (very soon)

Adjectival phrases can be simple as well as complex phrase structures. The simplest APs consist solely of the adjective head as in (7a). Slightly more complex are APs embedding adverbial structures as in (7b). Other APs can comprise specific constructions triggered by a certain lexical head as in (7c), and complex embeddings as in (7d).

(7) a. möglich (possible)

b. schreiend lila
screamingly purple

c. rund zwei Meter hohe
around two meters high

d. über die Köpfe der Apostel gesetzten
above the heads of the apostles set
'set above the heads of the apostles'

Noun phrases range from simple constructions consisting of a single noun or pronouns as in (8a), specific constructions as in (8b-c) to complex constructions involving recursion in pre-head positions as in (8d).

(8) a. Oktober (October); er (he)

b. 4,9 Milliarden Dollar
4.9 billion dollar

c. "Frankensteins Fluch"
Frankenstein's curse

d. kleine, über die Köpfe der Apostel gesetzte Flammen
small, above the heads of the apostles set flames
'small flames set above the heads of the apostles'

Prepositional phrases comprise pronominal adverbs as in (9a) and more complex structures embedding coordinated NPs as in (9b) as well as complex NPs as in (9b-c).

(9) a. davon (thereof)

b. zwischen Basel und St. Moritz
between Basel and St. Moritz

c. mit kleinen über die Köpfe der Apostel gesetzten Flammen
with small above the heads of the apostles set flames
'with small flames set above the heads of the apostles'

Verbal complexes are simple structures comprising of adjacent verbal elements as in (10).

(10) a. gemunkelt (rumored)

b. muß gerechnet werden
has counted to be
'has to be counted'

c. zu bekommen
to get

Subordinate Clauses are assembled using the annotated lexical phrases. On the one hand, the rules used to build the clauses prove that the chunks annotated by YAC are easily combined to larger clausal constructions. On the other hand, the clauses themselves are useful for the extraction of linguistic evidence.

5.3 Annotated features

Feature attributes specifying certain properties and characteristics of the chunks are annotated as well. The properties are classified and stored in different feature attributes to ease the access. One large disjunctive feature attribute holding all information would be unwieldy. As each chunk category has different characteristics, the annotated feature attributes vary from chunk category to chunk category. In other words, each chunk category has its own annotation scheme. For AdvP, e.g., only head lemma and lexical properties are annotated, while for NPs head lemma, lexical properties, and morpho-syntactic information is annotated. There are three general feature attributes, which are common to most of the chunks: (i) head lemma, (ii) morpho-syntactic information, (iii) lexical-semantic and structural properties.

The head lemma of the chunk is taken from the lemma value the head position. The head position is either specified in the rule using a target, or in the rule processing using a "fixed" position, i.e., a position that can be determined independently of the actual results relative to another position. Normally, the head lemma is a single token, derived from a single position. In some cases, however, the lemmas of several tokens have been subsumed to form the head lemma. Multi-word proper nouns, e.g., have a multi-token head lemma, as a single lemma cannot be filtered out. The head lemma of verbal complexes with separated prefixes is a single-token head, however, it has been taken from two different positions. The head lemma of PPs consists of two separate head lemma items: the lemma of the preposition, and the lemma of the embedded NP.

YAC obtains morpho-syntactic information for chunks from the morpho-syntactic features of relevant elements within the chunk. Invariant elements (e.g., invariant APs such as *lila* (purple)) are not considered. The morpho-syntactic information does not have to be, and in most cases, is not unique. If there is more than one element relevant for agreement, it is possible to reduce the ambiguity. An intersection of the different value-sets is used to determine the morpho-syntactic information of the chunk. In contrast to probabilistic approaches, no guessing is involved, i.e., if the value is still ambiguous, it is left ambiguous. In the case that no value is returned, i.e., the relevant elements do not agree, the chunk is rejected, as agreement is required within a chunk.

Lexical-semantic and structural properties are important for the parsing as well as for further applications. The properties can be triggers for specific internal structures, functions, and usages of chunks. Some of the properties are inherent in the corpus itself, i.e., they can be determined from the information already present in the corpus: (i) PoS-tags, (ii) text markers. Named entities, e.g., can be derived from the PoS-tag `NE` for proper noun (11).

(11) [$_{NP}$ Johann Sebastian Bach]
NE NE NE

Text markers such as quotation marks, parenthesis, and brackets indicate the special character of a chunk (e.g. as named entity or possible modifier, cf. (12)), and can function as a secure context in which the restrictions on the chunks are relaxed.

(12) "Wilhelm Meisters Lehrjahre"

Other properties are determined by external knowledge sources, such as lexicons and ontologies. Local adverbs (13), e.g., are identified according to manually prepared word lists.

(13) hier (here); dort (there)

Another possibility to derive properties is from the chunking process itself. In this case, specific embeddings are indicated as properties of the embedding chunk. Complex AP embedding PPs (14a) and NPs (14b) are marked by a respective feature indicating the embedded structure.

(14) a. [$_{AP}$ [$_{PP}$ über die Köpfe der Apostel] gesetzten]
above the heads of the apostles set
'set above the heads of the apostles'

b. [$_{AP}$ [$_{NP}$ der "Inkatha"-Partei] angehörenden]
to the Inkatha party belonging
'belonging to the Inkatha party'

5.4 The chunking process

The grammar rules of YAC, written in the CQP query language, are evaluated by the CQP query processor and then post-processed with Perl[9] scripts. This approach ensures good performance even on large corpora, provides a modular design of the annotation rules, and enables interactive grammar development. The powerful post-processing step greatly enhances the expressiveness of the grammar. See (Evert and Kermes 2003) for an in-depth discussion of the technical aspects of the YAC chunker.

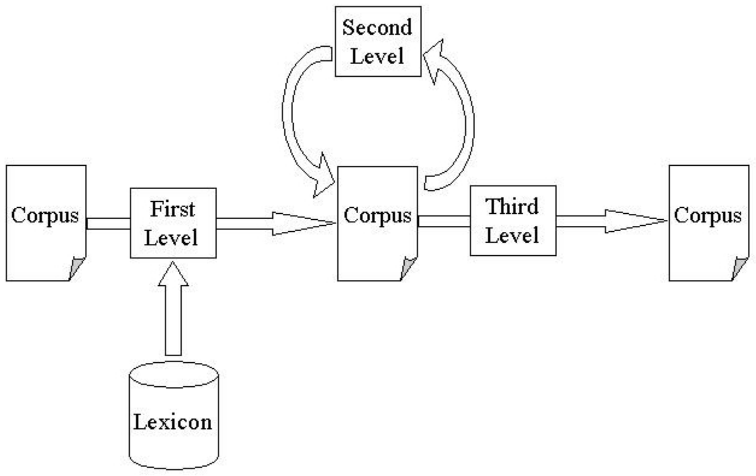
The chunking process is divided intro three levels, which serve different purposes:

- First Level: (i) annotates base-chunks, (ii) annotates chunks with a specific internal structure, (iii) introduces lexical-semantic properties
- Second Level: (i) main parsing level, (ii) iterative application of general phrase structure rules to build recursive chunks
- Third Level: finishing level

Figure 1 illustrates the architecture of the YAC parsing process.

[9] see http://www.perl.com/

Figure 1: Parsing Architecture of YAC



There are several advantages of annotating base-chunks with specific internal structures and introducing lexical and semantic information in the first level: (i) the specific rules do not interact with the main parsing rules, (ii) the rules for chunks which do not involve complex (recursive) embedding have to be applied only once, (iii) the additional rules which are necessary to cover specific phenomena of specialized text domains can be included easily without affecting the main parsing process, (iv) the rules of the main parsing process can be kept relatively simple and general, as most special cases are already covered, (v) only a relatively small number of "general" rules is needed for the main parsing process.

6 Evaluation

As a gold standard for the evaluation of the NP chunks we used the NEGRA treebank (Skut *et. al.* 1998) consisting of 355,096 tokens of German newspaper text with manually corrected part-of-speech tagging and parse trees. A reference set of 99,116 NPs was extracted from a version of the NEGRA treebank encoded in the TigerXML format (Mengel and Lezius 2000) using XSLT stylesheets. Unfortunately, the syntactic annotation scheme of the NEGRA treebank (Skut *et. al.* 1997), which omits all projections that are not strictly necessary to determine the constituent structure of a sentence, is not very well suited for automatic extraction tasks. Thus, the extraction of a gold standard for the evaluation was not a trivial task as has been discussed in (Kermes and Evert 2002).

We then applied YAC to the text of the NEGRA treebank using the manually disambiguated part-of-speech tagging provided in the corpus. Lemma and agreement information was added from the IMSLex morphology database. The 99,116 NPs in the reference set were then compared to the 101,165 NPs identified by YAC. Of those, 89,237 were correct (true positives), corresponding to a precision of 88.21% and a recall of 90.03%.[10] Since the head lemmas of NPs are not explicitly annotated in the NEGRA treebank and would have to be guessed from the part-of-speech tags at token level, we did not evaluate the head lemma annotations provided by YAC.

In real applications manually corrected part-of-speech tagging is not available. For this reason, we also evaluated YAC on a version of the corpus that was automatically part-of-speech tagged with the TreeTagger. Using its standard training corpus and a custom tagger lexicon based on the IMSLex morphology, the TreeTagger achieved a tagging precision of 94.82% (336,692 tokens correct out of 355,096). Most of the tagging errors were proper nouns that the morphology did not recognize. With this fully automatic process, YAC identified 103,484 NPs, of which 85,353 were correct, which gives a precision of 82.48% and a recall of 86.11%.

[10] Note that this evaluation strategy is equivalent to computing labelled precision and recall restricted to NP chunks.

7 Examples of applications

YAC annotates larger structures than classic chunkers. As has been said before, the chunks can easily be combined to larger constructions such as clauses. Thus, evidence for corpus linguistic research can be extracted easily, even if the constructions under investigation are complex. An example is the usage of predicative adjectives subcategorizing finite and infinite clauses. A relatively simple query is sufficient to find the evidence. Example (15) gives a phrase structure model of the query for predicative adjectives subcategorizing a finite clause in extraposed position.

(15) PredAP + finite clause $\rightarrow$ VC (AdvP | $NP_{[temp]}$ | $CL_{[rel]}$)* $AP_{[pred]}$ $CL_{[fin]}$

Maximal constituents of a sentence are combined with a finite clause structure to model the construction. The VC in initial position is followed by an unspecified number of possible adjuncts, which can be AdvPs, temporal NPs, relative clauses, and appositions. Then comes a predicative AP, immediately followed by a finite clause. The annotation of lexical-semantic properties allows us to include NPs with temporal aspect ($NP_{[temp]}$) as adjuncts. Predicative APs are identified by the annotated feature *pred* (for predicative). Annotation of head lemmas gives easy access to the adjective lemma.

The query in (15) is easily reformulated to find topicalized clauses. In this case, the clause structure simply has to be moved to the topic position in front of the VC. Infinite clauses can be searched for with similar queries, where $CL_{[fin]}$ is replaced by $CL_{[inf]}$. A detailed corpus study of this phenomenon is presented in (Kermes and Heid 2003).

The annotations provided by YAC allow extraction of evidence for complex linguistic constructions. Subcategorization information, positional and morpho-syntactic variations (e.g. scrambling (topicalized vs. extraposed) as well as singular/plural alternations), and selectional preferences.

8 References

Abney S 1991. Parsing by chunks. In Berwick R, Abney S, and Tenny C (eds), *Principle-Based Parsing*. Kluwer Academic Publishers.

Abney S 1996a. *Chunk stylebook*. Working draft.

Abney S 1996b. Partial parsing via finite-state cascades. In *Proceedings of the ESSLLI '96 Robust Parsing Workshop*.

Brants T 1999. Cascaded Markov models. In *Proceedings of 9th Conference of the European Chapter of the Association for Computational Linguistics EACL-99*, Bergen, Norway.

Brants T 2000. Inter-annotator agreement for a German newspaper corpus. In *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC-2000)*, Athens, Greece.

Christ O, Schulze B M, Hofmann A, König E 1999. *The IMS Corpus Workbench: Corpus Query Processor (CQP): User's Manual*. Technical report, IMS, University of Stuttgart.

Evert S, Kermes H 2003. Annotation, storage, and retrieval of mildly recursive structures. In *Proceedings of the Workshop on Shallow Processing of Large Corpora*, Lancaster, UK

Höhle T 1986. Der Begriff 'Mittelfeld', Anmerkungen über die Theorie der topologischen Felder. In *Akten des Siebten Internationalen Germanistenkongresses*, Göttingen, pp. 329-340.

Karlsson F, Voutilainen A, Heikkilae J, Anttila A 1995. *Constraint Grammar: A Language-Independent System for Parsing Unrestricted Text*. Berlin, de Gruyter.

Kübler S, Hinrichs E W 2001. TüSBL: a similarity-based chunk parser for robust syntactic processing. In *Proceedings of HLT 2001*, San Diego, California.

Kermes H, Evert S 2002. YAC – a recursive chunker for unrestricted German text. In Rodriguez M G, Araujo C P (eds), *Proceedings of the Third International Conference on Language Resources and Evaluation*, Las Palmas, Spain, pp. 1805-1812.

Kermes H, Heid U 2003. Using chunked corpora for the acquisition of collocations and idiomatic expressions. In *Proceedings of COMPLEX 2003*, to appear.

Lezius W, Dipper S, Fitschen A 2000. IMSLex – representing morphological and syntactical information in a relational database. In Heid U, Evert S, Lehmann E, Rohrer C (eds), *Proceedings of the 9th EURALEX International Congress,* Stuttgart, Germany, pp. 133-139.

Mengel A, Lezius W 2000. An XML-based representation format for syntactically annotated corpora. In *Proceedings of the Second International Conference on Language Resources and Engineering (LREC)*, Volume 1, Athens, Greece, pp. 121-126.

Müller F H, Ule T 2002. Annotating topological fields and chunks – and revising PoS tags at the same time. In *Proceedings of the Nineteenth International Conference on Computational Linguistics (COLING 2002)*, pp. 695-701.

Müller F H, Ule T 2001. Satzklammer annotieren und Tags korrigieren. Ein mehrstufiges 'top-down-bottom-up'-System zur flachen, robusten Annotierung von Sätzen im Deutschen. In *Proceedings of the Jahrestagung der Gesellschaft für linguistische Datenverarbeitung 2001.*

Schiller A, Teufel S, Stöckert C, Thielen C 1999. *Guidelines für das Tagging deutscher Textcorpora mit STTS.* Technical report, University of Stuttgart and University of Tübingen.

Schmid H 1994. Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*, Manchester, UK, pp. 44-49.

Schmid H 1995. Improvements in part-of-speech tagging with an application to German. In *Proceedings of the ACL SIGDAT-Workshop.*

Skut W, Brants T 1998. A maximum-entropy partial parser for unrestricted text. In *Sixth Workshop on Very Large Corpora*, Montreal, Canada, pp. 143-151.

Skut W, Brants T, Krenn B, Uszkoreit H 1998. A linguistically interpreted corpus of German newspaper texts. In *Proceedings of the ESSLLI Workshop on Recent Advances in Corpus Annotation*, Saarbrücken, Germany.

Skut W, Krenn B, Brants T, Uszkoreit H 1997. An annotation scheme for free word order languages. In *Proceedings of the Fifth Conference on Applied Natural Language Processing ANLP-97*, Washington, DC.

Ule T, Müller F H 2001. KaRoPars: Ein System zur linguistischen Annotation großer Textkorpora des Deutschen. In *Proceedings of the Workshop Werkzeuge zur automatischen Analyse und Verarbeitung von Texten: Formate, Tools, Software-Systeme,* Trier, Germany.

Veenstra J, Müller F H, Ule T 2002. Topological fields chunking for German. In *Proceedings of the Sixth Conference on Natural Language Learning (CoNLL 2002),* pp. 56-62.

Voutilainen A 1994. *Three studies of grammar-based surface parsing of unrestricted English text.* Technical Report 24, Department of General Linguistics, University of Helsinki, Finland.

Voutilainen A 1997. *The ENGCG-2 tagger in outline*. http://www.ling.helsinki.fi/~avoutila/cg/doc/index.html.

Voutilainen A, Heikkilae J 1994. An English constraint grammar (ENGCG): a surface-syntactic parser of English. In Fries U, Totti G, Schneider P (eds), *Creating and using English language corpora*. Amsterdam, Rodopi.

Voutilainen A, Jaervinen T 1995. Specifying a shallow grammatical representation for parsing purposes. In *Proceedings of the Seventh Conference of the European Chapter of the Association for Computational Linguistics*, Dublin, pp. 210-214.

Voutilainen A, Tapanainen P 1993. Ambiguity resolution in a reductionist parser. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics (EACL'93)*, Utrecht, pp. 394-403.