Tagging Grammatical Functions

Thorsten Brants, Wojciech Skut, Brigitte Krenn
Universität des Saarlandes
Computational Linguistics
D-66041 Saarbrücken, Germany
{brants,skut,krenn}@coli.uni-sb.de

In Proceedings of EMNLP-2, Providence, RI, 1997

Abstract

This paper addresses issues in automated treebank construction. We show how standard part-of-speech tagging techniques extend to the more general problem of structural annotation, especially for determining grammatical functions and syntactic categories. Annotation is viewed as an interactive process where manual and automatic processing alternate. Efficiency and accuracy results are presented. We also discuss further automation steps.

1 Introduction

The aim of the work reported here is to construct a corpus of German annotated with syntactic structures (treebank). The required size of the treebank and granularity of encoded information make it necessary to ensure high annotation efficiency and accuracy. Annotation automation has thus become one of the central issues of the project.

In this section, we discuss the relation between automatic and manual annotation. Section 2 focuses on the annotation format employed in our treebank. The annotation software is presented in section 3. Sections 4 and 5 deal with automatic assignment of grammatical functions and phrasal categories. Experiments on automating the annotation are presented in section 6.

1.1 Automatic vs. Manual Annotation

A problem for corpus annotation is the trade-off between efficiency, accuracy and coverage. Although accuracy increases significantly as annotators gain expertise, incorrect hand-parses still occur. Their frequency depends on the granularity of the encoded information.

Due to this residual error rate, automatic annotation of frequently occurring phenomena is likely to yield better results than even well-trained human annotators. For infrequently occurring constructions, however, manual annotation is more reliable, as is manual annotation of phenomena involving non-syntactic information (e.g., resolution of attachment ambiguities based on world knowledge).

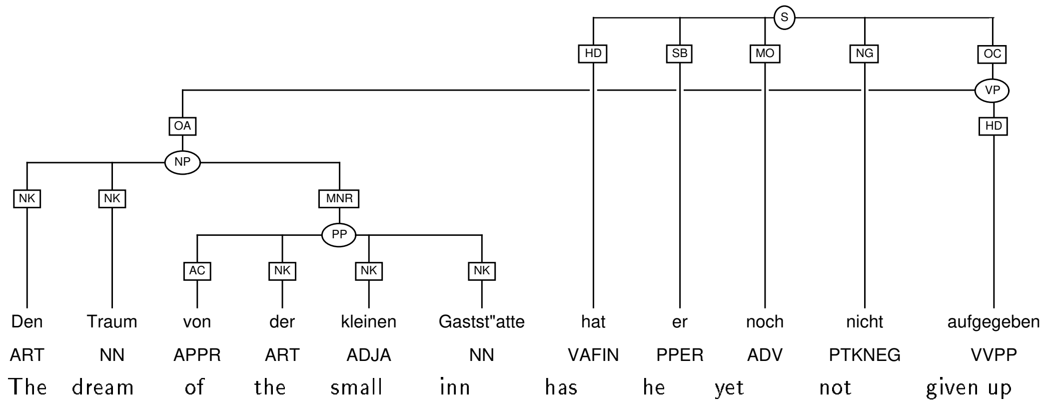
As a consequence, efficiency and reliability of annotation can be significantly increased by combining automatic annotation with human processing skills and supervision, especially if this combination is implemented as an interactive process.

2 Annotation Scheme

Existing treebanks of English ((Marcus et al., 1994), (Sampson, 1995), (Black et al., 1996)) contain conventional phrase-structure trees augmented with annotations for discontinuous constituents. As this encoding strategy is not well-suited to a free word order language like German, we have focussed on a less surface-oriented level of description, most closely related to the LFG f-structure, and representations used in dependency grammar. To avoid confusion with theory-specific constructs, we use the generic term *argument structure* to refer to our annotation format. The main advantages of the model are: it is relatively theory-independent and closely related to semantics. For more details on the linguistic specifications of the annotation scheme see (Skut et al., 1997). A similar approach has been also successfully applied in the TSNLP database, cf. (Lehmann et al., 1996).

In contrast to conventional phrase-structure grammars, argument structure annotations are not influenced by word order. Local and non-local dependencies are represented in the same way, the latter indicated by crossing branches in the hierarchical structure, as shown in figure 1 where in the VP the terminals of the direct object OA (*den Traum von der kleinen Gaststätte*) are not adjacent to the head HD *aufgegeben*[1]. For a related handling

[1] See appendix A for a description of tags used throug-



'He has not yet given up the dream of a small inn.'

Figure 1: Example sentence

of non-projective phenomena see (Tapanainen and Järvinen, 1997).

Such a representation permits clear separation of word order (in the surface string) and syntactic dependencies (in the hierarchical structure). Thus we avoid explicit explanatory statements about the complex interrelation between word order and syntactic structure in free word order languages. Such statements are generally theory-specific and therefore are not appropriate for a descriptive approach to annotation. The relation between syntactic dependencies and surface order can nontheless be inferred from the data. This provides a promising way of handling free word order phenomena.[2].

3 Annotation Tool

Since syntactic annotation of corpora is time-consuming, a partially automated annotation tool has been developed in order to increase efficiency.

3.1 The User Interface

For optimal human-machine interaction, the tool supports immediate graphical representation of the structure being annotated.

Since keyboard input is most efficient for assigning categories to words and phrases, cf. (Lehmann et al., 1996; Marcus et al., 1994), and structural manipulations are executed most efficiently using the mouse, both an elaborate keyboard and optical interface is provided. As suggested by Robert MacIntyre[3], it is most efficient to use one hand for structural commands with the mouse and the other hand for short keyboard input.

By additionally offering online menus for commands and labels, the tool suits beginners as well as experienced users. Commands such as "group words", "group phrases", "ungroup", "change labels", "re-attach nodes", "generate postscript output", etc. are available.

The three tagsets (word, phrase, and edge labels) used by the annotation tool are variable. They are stored together with the corpus, which allows easy modification and exchange of tagsets. In addition, appropriateness checks are performed automatically. Comments can be added to structures.

Figure 2 shows a screen dump of the graphical interface.

3.2 Automating Annotation

Existing treebank annotation tools are characterised by a high degree of automation. The task of the annotator is to correct the output of a parser, i.e., to eliminate wrong readings, complete partial parses, and adjust partially incorrect ones.

Since broad-coverage parsers for German, especially robust parsers that assign predicate-argument structure and allow crossing branches, are not available, or require an annotated traing corpus (cf. (Collins, 1996), (Eisner, 1996)).

As a consequence, we have adopted a bootstrapping approach, and gradually increased the degree of automation using already annotated sentences as training material for a stochastic processing module.

This aspect of the work has led to a new model of human supervision. Here automatic annotation and human supervision are combined interactively whereby annotators are asked to confirm the *local*

hout this paper.

[2]'Free' word order is a function of several interacting parameters such as category, case and topic-focus articulation. Varying the order of words in a sentence yields a continuum of grammaticality judgments rather than a simple right-wrong distinction.

[3]personal communication, Oct. 1996

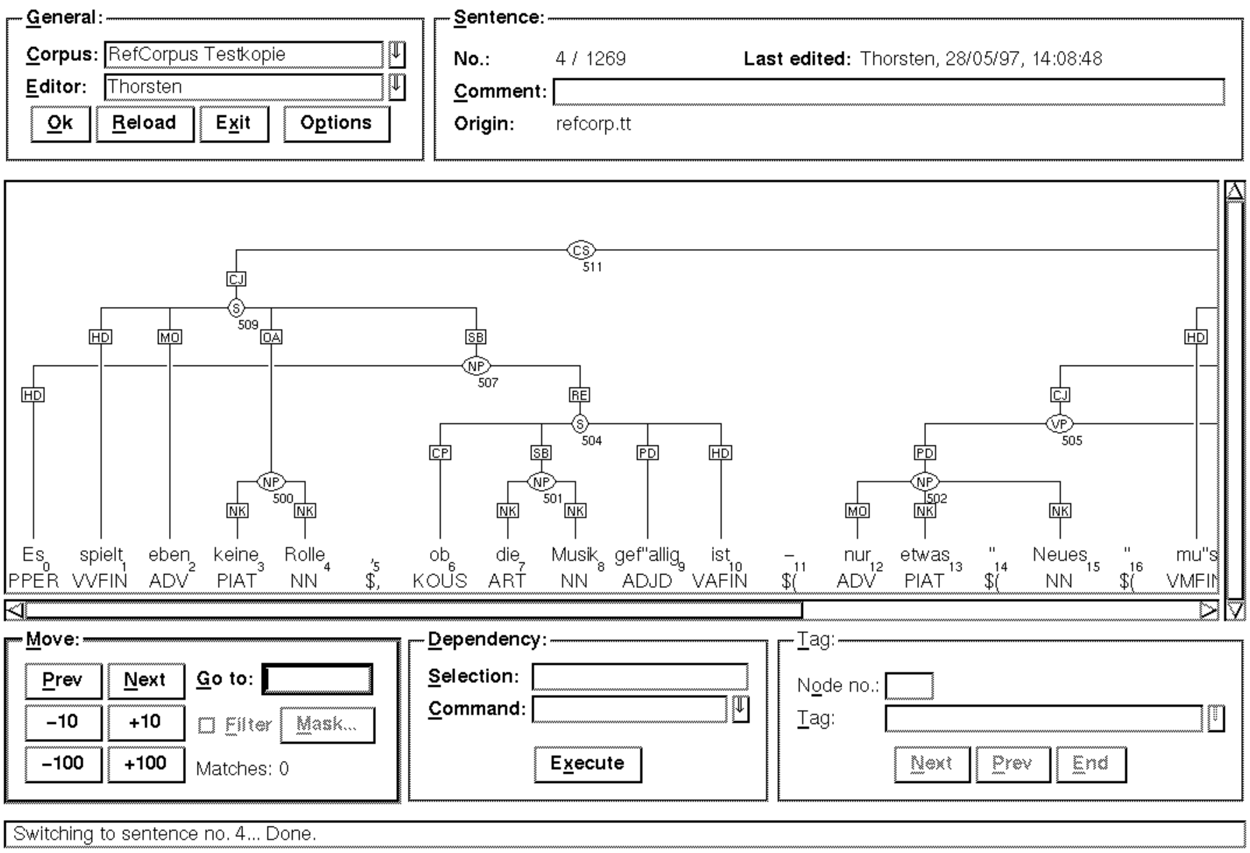


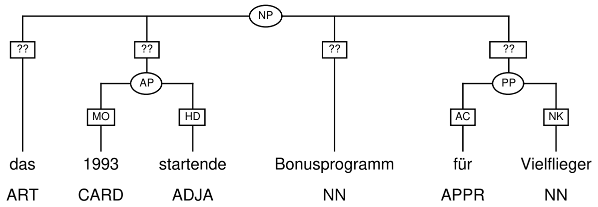
Figure 2: Screen dump of the annotation tool

predictions of the parser. The size of such 'supervision increments' varies from local trees of depth one to larger chunks, depending on the amount of training data available.

We distinguish six degrees of automation:

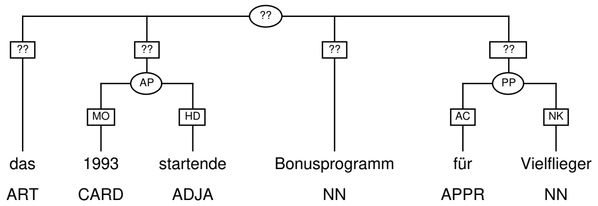
0) Completely manual annotation.
1) The user determines phrase boundaries and syntactic categories (S, NP, VP, ...). The program automatically assigns grammatical functions. The annotator can alter the assigned tags (cf. figure 3).
2) The user only determines the components of a new phrase (local tree of depth 1), while both category and function labels are assigned automatically. Again, the annotator has the option of altering the assigned tags (cf. figure 4).
3) The user selects a substring and a category, whereas the entire structure covering the substring is determined automatically (cf. figure 5).
4) The program performs simple bracketing, i.e., finds 'kernel phrases' without the user having to explicitly mark phrase boundaries. The task can be performed by a chunk parser that is equipped with an appropriate finite state grammar (Abney, 1996).
5) The program suggests partial or complete parses.

A set of 500 manually annotated training sentences (step 0) was sufficient for a statistical tagger to reliably assign grammatical functions, provided the user determines the elements of a phrase and its category (step 1). Approximately 700 additional sentences have been annotated this way. Annotation efficiency increased by 25 %, namely from an average annotation time of 4 minutes to 3 minutes per sentence (300 to 400 words per hour). The 1,200 sentences were used to train the tagger for automation step 2. Together with improvements in the user interface, this increased the efficiency by another 33%, from approximately 3 to 2 minutes (600 words per hour). The fastest annotators cover up to



'the bonus program for frequent fliers starting in 1993'

Figure 3: Example for automation level 1: the user has marked *das*, the AP, *Bonusprogramm*, and the PP as a constituent of category NP, and the tool's task is to determine the new edge labels (marked with question marks), which are, from left to right, NK, NK, NK, MNR.



'the bonus program for frequent fliers starting in 1993'

Figure 4: Example for automation level 2: the user has marked *das*, the AP, *Bonusprogramm* and the PP as a constituent, and the tool's task is to determine the new node and edge labels (marked with question marks).

1000 words per hour.

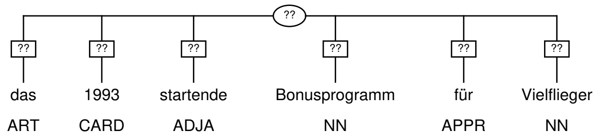
At present, the treebank comprises 3000 sentences, each annotated independently by two annotators. 1,200 of the sentences are compared with the corresponding second annotation and are cleaned, 1,800 are currently cleaned.

In the following sections, the automation steps 1 and 2 are presented in detail.

4 Tagging Grammatical Functions

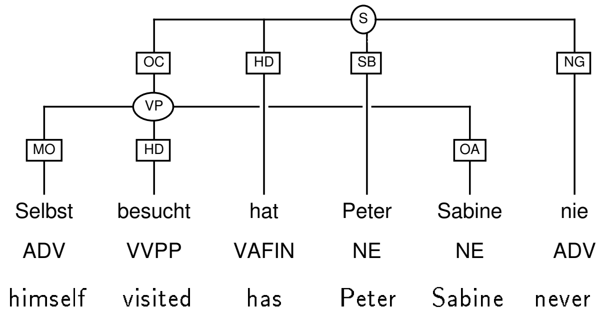
4.1 The Tagger

In contrast to a standard part-of-speech tagger which estimates lexical and contextual probabilities of tags from sequences of word-tag pairs in a corpus, (e.g. (Cutting et al., 1992; Feldweg, 1995)), the tagger for grammatical functions works with lexical and contextual probability measures $P_Q(\cdot)$ depending on the category of the mother node (Q). Each phrasal category (S, VP, NP, PP etc.) is represented by a different Markov model. The categories of the daughter nodes correspond to the outputs of the Markov model, while grammatical functions correspond to states.



'the bonus program for frequent fliers starting in 1993'

Figure 5: Example for automation level 3: the user has marked the words as a constituent, and the tool's task is to determine simple sub-phrases (the AP and PP) as well as the new node and edge labels (cf. previous figures for the resulting structure).



'Peter never visited Sabine himself'

Figure 6: Example sentence

The structure of a sample sentence is shown in figure 6. Figure 7 shows those parts of the Markov models for sentences (S) and verb phrases (VP) that represent the correct paths for the example.[4]

Given a sequence of word and phrase categories $T = T_1 \ldots T_k$ and a parent category Q, we calculate the sequence of grammatical functions $G = G_1 \ldots G_k$ that link T and Q as

$$\operatorname*{argmax}_G P_Q(G|T) \tag{1}$$

$$= \operatorname*{argmax}_G \frac{P_Q(G) \cdot P_Q(T|G)}{P_Q(T)}$$

$$= \operatorname*{argmax}_G P_Q(G) \cdot P_Q(T|G)$$

Assuming the Markov property we have

[4] cf. appendix A for a description of tags used in the example

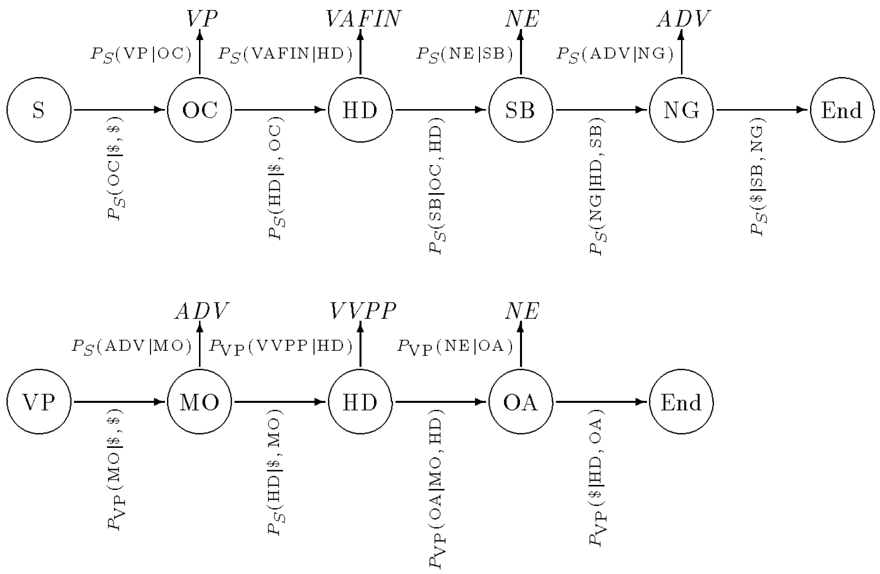


Figure 7: Parts of the Markov models used in *Selbst besucht hat Peter Sabine nie* (cf. figure 6). All unused states, transitions and outputs are omitted.

$$P_Q(T|G) = \prod_{i=1}^{k} P_Q(T_i|G_i) \tag{2}$$

and

$$P_Q(G) = \prod_{i=1}^{k} P_Q(G_i|C_i) \tag{3}$$

The contexts C_i are modeled by a fixed number of surrounding elements. Currently, we use two grammatical functions, which results in a trigram model:

$$P_Q(G) = \prod_{i=1}^{k} P_Q(G_i|G_{i-2}, G_{i-1}) \tag{4}$$

The contexts are smoothed by linear interpolation of unigrams, bigrams, and trigrams. Their weights are calculated by deleted interpolation (Brown et al., 1992).

The predictions of the tagger are correct in approx. 94% of all cases. In section 4.3, we demonstrate how to cope with wrong predictions.

4.2 Serial Order

As the annotation format permits trees with crossing branches, we need a convention for determining the relative position of overlapping sibling phrases in order to assign them a position in a Markov model. For instance, in figure 6 the range of the terminal node positions of VP overlaps with those of the subject SB and the finite verb HD. Thus there is no single a-priori position for the VP node[5].

The position of a phrase depends on the position of its descendants. We define the relative order of two phrases recursively as the order of their *anchors*, i.e., some specified daughter nodes. If the anchors are words, we simply take their linear order.

The exact definition of the anchor is based on linguistic knowledge. We choose the most intuitive alternative and define the anchor as the head of the phrase (or some equivalent function). Noun phrases do not necessarily have a unique head; instead, we use the last element in the noun kernel (elements of the noun kernel are determiners, adjectives, and nouns) to mark the anchor position. Except for NPs, we employ a default rule that takes the leftmost element as the anchor in case the phrase has no (unique) head.

Thus the position of the VP in figure 6 is defined as equal to the string position of *besucht*. The position of the VP node in figure 1 is equal to that of *aufgegeben*, and the position of the NP in figure 3 is equivalent to that of *Bonusprogramm*.

4.3 Reliability

Experience gained from the development of the Penn Treebank (Marcus et al., 1994) has shown that au-

[5]Without crossing edges, the serial order of phrases is trivial: phrase Q_1 precedes phrase Q_2 if and only if all terminal nodes derived from Q_1 precede those of Q_2. This suffices to uniquely determine the order of sibling nodes.

tomatic annotation is useful only if it is absolutely correct, while wrong analyses are often difficult to detect and their correction can be time-consuming.

To prevent the human annotator from missing errors, the tagger for grammatical functions is equipped with a measure for the reliability of its output.

Given a sequence of categories, the tagger calculates the most probable sequence of grammatical functions. In addition, it computes the probabilities of the second-best functions of each daughter node. If some of these probabilities are close to that of the best sequence, the alternatives are regarded as equally suited and the most probable one is not taken to be the sole winner, the prediction is marked as unreliable in the output of the tagger.

These unreliable predictions can be further classified in that we distinguish "unreliable" sequences as opposed to "almost reliable" ones.

The distance between two probabilities for the best and second-best alternative, p_{best} and p_{second}, is measured by their quotient. The classification of reliability is based on thresholds. In the current implementation we employ three degrees of reliability which are separated by two thresholds θ_1 and θ_2. θ_1 separating unreliable decisions from those considered almost reliable. θ_2 marks the difference between almost and fully reliable predictions.

Unreliable:

$$\frac{p_{best}}{p_{second}} < \theta_1$$

The probabilities of alternative assignments are within some small specified distance. In this case, it is the annotator who has to specify the grammatical function.

Almost reliable:

$$\theta_1 \leq \frac{p_{best}}{p_{second}} < \theta_2$$

The probability of an alternative is within some larger distance. In this case, the most probable function is displayed, but the annotator has to confirm it.

Reliable:

$$\frac{p_{best}}{p_{second}} \geq \theta_2$$

The probabilities of all alternatives are much smaller than that of the best assignment, thus the latter is assigned.

For efficiency, an extended Viterbi algorithm is used. Instead of keeping track of the best path only (cf. (Rabiner, 1989)), we keep track of all paths that fall into the range marked by the probability of the best path and θ_2, i.e., we keep track of all alternative paths with probability p_{alt} for which

$$p_{alt} \geq \frac{p_{best}}{\theta_2}.$$

Suitable values for θ_1 and θ_2 were determined empirically (cf. section 6).

5 Tagging Phrase Categories

The second level of automation (cf. section 3) automates the recognition of phrasal categories, and so frees the annotator from typing phrase labels. The task is performed by an extension of the tagger presented in the previous section where different Markov models for each category were introduced. The annotator determines the category of the current phrase, and the tool runs the appropriate model to determine the edge labels.

To assign the phrase label automatically, we run all models in parallel. Each model assigns grammatical functions and, more important for this step, a probability to the phrase. The model assigning the highest probability is assumed to be most adequate, and the corresponding label is assigned to the phrase.

Formally, we calculate the phrase category Q (and at the same time the sequence of grammatical functions $G = G_1 \ldots G_k$) on the basis of the sequence of daughters $T = T_1 \ldots T_k$ with

$$\operatorname*{argmax}_Q \max_G P_Q(G|T).$$

This procedure is equivalent to a different view on the same problem involving one large (combined) Markov model that enables a very efficient calculation of the maximum.

Let $\mathcal{G}_Q$ be the set of all grammatical functions that can occur within a phrase of type Q. Assume that these sets are pairwise disjoint. One can easily achieve this property by indexing all used grammatical functions with their associated phrases and, if necessary, duplicating labels, e.g., instead of using HD, MO, ..., use the indexed labels HD_S, HD_{VP}, MO_{NP}, ... This property makes it possible to determine a phrase category by inspecting the grammatical functions involved.

When applied, the combined model assigns grammatical functions to the elements of a phrase (not knowing its category in advance). If transitions between states representing labels with different indices are forced to zero probability (together with smoothing applied to other transitions), all labels assigned to a phrase get the same index. This uniquely identifies a phrase category.

The two additional conditions

$$G \in \mathcal{G}_{Q1} \Rightarrow G \notin \mathcal{G}_{Q2} \quad (Q_1 \neq Q_2)$$

and

$$G_1 \in \mathcal{G}_Q \wedge G_2 \notin \mathcal{G}_Q \Rightarrow P(G_2|G_1) = 0$$

are sufficient to calculate

$$\underset{G}{\operatorname{argmax}} P(G|T)$$

using the Viterbi algorithm and to identify both the phrase category and the respective grammatical functions.

Again, as described in section 4, we calculate probabilities for alternative candidates in order to get reliability estimates.

The overall accuracy of this approach is approx. 95%, and higher if we only consider the reliable cases. Details about the accuracy are reported in the next section.

6 Experiments

To investigate the possibility of automating annotation, experiments were performed with the cleaned part of the treebank[6] (approx. 1,200 sentences, 24,000 words). The first run of experiments was carried out to test tagging of grammatical functions, the second run to test tagging of phrase categories.

6.1 Grammatical Functions

This experiment tested the reliability of assigning grammatical functions given the category of the phrase and the daughter nodes (supplied by the annotator).

Let us consider the sentence in figure 6: two sequences of grammatical functions are to be determined, namely the grammatical functions of the daughter nodes of S and VP. The information given for *selbst besucht Sabine* is its category (VP) and the daughter categories: adverb (ADV), past participle (VVPP), and proper noun (NE). The task is to assign the functions modifier (MO) to ADV, head (HD) to VVPP and direct (accusative) object (OA) to NE. Similarly, function tags are assigned to the components of the sentence (S).

The tagger described in section 4 was used.

The corpus was divided into two disjoint parts, one for training (90% of the respective corpus), and one for testing (10%). This procedure was repeated 10 times with different partitions. Then the average accuracy was calculated.

The thresholds for search beams were set to $\theta_1 = 5$ and $\theta_2 = 100$, i.e., a decision is classified as reliable if there is no alternative with a probability larger than $\frac{1}{100}$ of the best function tag. The prediction is classified as unreliable if the probability of an alternative is larger than $\frac{1}{5}$ of the most probable tag.

[6]The corpus is part of the German newspaper text provided on the ECI CD-ROM. It has been part-of-speech tagged and manually corrected previously, cf. (Thielen and Schiller, 1995).

Table 1: Levels of reliability and the percentage cases where the tagger assigned a correct grammatical function (or would have assigned if a decision is forced).

	cases	correct
reliable	89%	96.7%
marked	7%	84.3%
unreliable	4%	57.3%
overall	100%	94.2%

If there is an alternative between these two thresholds, the prediction is classified as almost reliable and marked in the output (cf. section 4.3: marked assignments are to be confirmed by the annotator, unreliable assignments are deleted, annotation is left to the annotator).

Table 1 shows tagging accuracy depending on the three different levels of reliability. The results confirm the choice of reliability measures: the lower the reliability, the lower the accuracy.

Table 2 shows tagging accuracy depending on the category of the phrase and the level of reliability. The table contains the following information: the number of all mother-daughter relations (i.e., number of words and phrases which are immediately dominated by a mother node of a particular category), the overall accuracy for that phrasal category and the accuraciees for the three reliability intervals.

6.2 Error Analysis for Function Assignment

The inspection of tagging errors reveals several sources of wrong assignments. Table 3 shows the 10 most frequent errors[7] which constitute 25% of all errors (1509 errors occurred during 10 test runs).

Read the table in the following way: line 2 shows the second-most frequent error. It concerns NPs occurring in a sentence (S); this combination occurred 1477 times during testing. In 286 of these occurrences the NP is assigned the grammatical function OA (accusative object) manually, but of these 286 cases the tagger assigned the function SB (subject) 56 times.

The errors fall into the following classes:

1. There is insufficient information in the node labels to disambiguate the grammatical function.

Line 1 is an example for insufficient information. The tag NP is uninformative about its case and therefore the tagger has to distinguish SB (subject) and

[7]See appendix A for a description of tags used in the table.

Table 2: Tagging accuracy for assigning grammatical functions depending on the category of the mother node. For each category, the first row shows the percentage of branches that occur within this category and the overall accuracy, the following rows show the relative percentage and accuracy for different levels of reliability.

	cases	correct
S	26%	89.1%
decision	85%	92.7%
marked	8%	81.9%
no decision	7%	52.9%
VP	7%	90.9%
decision	97%	92.2%
marked	1%	57.7%
no decision	2%	52.3%
NP	26%	96.4%
decision	86%	98.6%
marked	10%	86.8%
no decision	4%	73.0%
PP	24%	97.9%
decision	92%	99.2%
marked	6%	85.8%
no decision	2%	75.5%
others	18%	94.7%
decision	91%	98.0%
marked	6%	82.8%
no decision	3%	22.1%

Table 3: The 10 most frequent errors in assigning grammatical functions. The table shows a mother and a daughter node category, the frequency of this particular combination (sum over 10 test runs), the grammatical function assigned manually (and its frequency) and the grammatical function assigned by the tagger (and its frequency).

	phrase	elem	f	original		assigned	
1.	S	NP	1477	SB	894	OA	65
2.	S	NP	1477	OA	286	SB	56
3.	NP	PP	470	PG	52	MNR	50
4.	S	VP	613	PD	47	OC	42
5.	PP	PP	252	PG	30	MNR	30
6.	VP	NP	286	DA	32	OA	26
7.	S	NP	1477	PD	72	SB	25
8.	S	NP	1477	MO	33	SB	21
9.	S	S	186	MO	78	PD	21
10.	VP	PP	453	SBP	21	MO	21

OA (accusative object) on the basis of its position, which is not very reliable in German. Missing information in the labels is the main source of errors. Therefore, we currently investigate the benefits of a morphological component and percolation of selected information to parent nodes.

2. Due to the n-gram approach, the tagger only sees a local window of the sentences.

Some linguistic knowledge is inherently global, e.g., there is at most one subject in a sentence and one head in a VP. Errors of this type may be reduced by introducing finite state constraints that restrict the possible sequences of functions within each phrase.

3. The manual annotation is wrong, and a correct tagger prediction is counted as an error.

At earlier stages of annotation, the main source of errors was wrong or missing manual annotation. In some cases, the tagger was able to abstract from these errors during the training phase and subsequently assigned the correct tag for the test data. However, when performing a comparison against the corpus, these differences are marked as errors. Most of these errors were eliminated by comparing two independent annotations and cleaning up the data.

6.3 Phrase Categories

In this experiment, the reliability of assigning phrase categories given the categories of the daughter nodes (they are supplied by the annotator) was tested.

Consider the sentence in figure 6: two phrase categories are to be determined (VP and S). The information given for *selbst besucht Sabine* is the sequence of categories: adverb (ADV), past participle

Table 4: Levels of reliability and the percentage of cases in which the tagger assigned a correct phrase category (or would have assigned if a decision is forced).

	cases	correct
reliable	79%	98.5%
marked	16%	90.4%
unreliable	5%	65.9%
overall	100%	95.4%

(VVPP), and proper noun (NE). The task is to assign category VP. Subsequently, S is to be assigned based on the categories of the daughters VP, VAFIN, NE, and ADV.

The extended tagger using a combined model as described in section 5 was applied.

Again, the corpus is divided into two disjoint parts, one for training (90% of the corpus), and one for testing (10%). The procedure is repeated 10 times with different partitions. Then the average accuracy was calculated.

The same thresholds for search beams as for the first set of experiments were used.

Table 4 shows tagging accuracy depending on the three different levels of reliability.

Table 5 shows tagging accuracy depending on the category of the phrase and the level of reliability. The table contains the following information: the percentage of occurrences of the particular phrase, the overall accuracy for that phrasal category and the accuracy for each of the three reliability intervals.

6.4 Error Analysis for Category Assignment

When forced to make a decision (even in unreliable cases) 435 errors occured during the 10 test runs (4.5% error rate). Table 6 shows the 10 most-frequent errors which constitute 50% of all errors.

The most frequent error was the confusion of S and VP. They differ in that sentences S contain finite verbs and verb phrases VP contain non-finite verbs. But the tagger is trained on data that contain incomplete sentences and therefore sometimes erroneously assumes an incomplete S instead of a VP. To avoid this type of error, the tagger should be able to take the neighborhood of phrases into account. Then, it could detect the finite verb that completes the sentence.

Adjective phrases AP and noun phrases NP are confused by the tagger (line 5 in table 6), since almost all AP's can be NP's. This error could also

Table 5: Tagging accuracy for assigning phrase categories, depending on the manually assigned category. For each category, the first row shows the percentage of phrases belonging to a specific category (according to manual assignment) and the percentage of correct assignments. The following rows show the relative percentage and accuracy for different levels of reliability.

	cases	correct
S	20%	97.5%
decision	96%	99.7%
marked	2%	63.2%
no decision	2%	29.0%
VP	9%	93.2%
decision	71%	96.4%
marked	24%	91.3%
no decision	5%	60.9%
NP	29%	96.1%
decision	81%	99.3%
marked	13%	91.8%
no decision	6%	64.9%
PP	24%	98.7%
decision	94%	99.6%
marked	4%	92.5%
no decision	2%	70.8%
others	18%	89.0%
decision	42%	91.7%
marked	45%	90.6%
no decision	12%	73.2%

Table 6: The 10 most frequent errors in assigning phrase categories (summed over reliability levels). The table shows the phrase category assigned manually (and its frequency) and the category erroneously assigned by the tagger (and its frequency).

	phrase	f	assigned	f
1.	VP	828	S	46
2.	NP	2812	NM	32
3.	NP	2812	PP	31
4.	NP	2812	S	25
5.	AP	419	NP	15
6.	DL	20	CS	15
7.	PP	2298	NP	15
8.	S	1910	NP	15
9.	AP	419	PP	11
10.	MPN	293	NP	11

be fixed by inspecting the context and detecting the associated NP.

As for assigning grammatical functions, insufficient information in the labels is a significant source of errors, cf. the second-most frequent error. A large number of cardinal-noun pairs forms a numerical component (NM), like *7 Millionen, 50 Prozent*, etc (*7 million, 50 percent*). But this combination also occurs in NPs like *20 Leute, 3 Monate*, ...(*20 people, 3 months*), which are mis-tagged since they are less frequent. This can be fixed by introducing an extra tag for nouns denoting numericals.

7 Conclusion

A German newspaper corpus is currently being annotated with a new annotation scheme especially designed for free word order languages.

Two levels of automatic annotation (level 1: assigning grammatical functions and level 2: assigning phrase categories) have been presented and evaluated in this paper.

The overall accuracy for assigning grammatical functions is 94.2%, ranging from 89% to 98%, depending on the type of phrase. The least accuracy is achieved for sentences, the best for prepositional phrases. By suppressing unreliable decisions, precision can be increased to range from 92% to 99%.

The overall accuracy for assigning phrase categories is 95.4%, ranging from 89% to 99%, depending the category. By suppressing unreliable decisions, precision can also be increased to range from 92% to over 99%.

In the error analysis, the following sources of misinterpretation could be identified: insufficient linguistic information in the nodes (e.g., missing case information), and insufficient information about the global structure of phrases (e.g., missing valency information). Morphological information in the tagset, for example, helps to identify the objects and the subject of a sentence. Using a more fine-grained tagset, however, requires methods for adjusting the granularity of the tagset to the size (and coverage) of the corpus, in order to cope with the sparse data problem.

8 Acknowledgements

This work is part of the DFG Sonderforschungsbereich 378 *Resource-Adaptive Cognitive Processes*, Project C3 *Concurrent Grammar Processing.*

We wish to thank the universities of Stuttgart and Tübingen for kindly providing us with a hand-corrected part-of-speech tagged corpus. We also wish to thank Jason Eisner, Robert MacIntyre and Ann Taylor for valuable discussions on dependency parsing and the Penn Treebank annotation. Special thanks go to Oliver Plaehn, who implemented the annotation tool, and to our six fearless annotators.

References

Abney, Steven. 1996. Partial parsing via finite-state cascades. In *Proceedings of the ESSLLI'96 Robust Parsing Workshop*, Prague, Czech Republic.

Black, Ezra, Stephen Eubank, Hideki Kashioka, David Magerman, Roger Garside, and Geoffrey Leech. 1996. Beyond skeleton parsing: Producing a comprehensive large-scale general-english treebank with full grammaticall analysis. In *Proc. of COLING-96*, pages 107–113, Kopenhagen, Denmark.

Brown, P. F., V. J. Della Pietra, Peter V. deSouza, Jenifer C. Lai, and Robert L. Mercer. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

Collins, Michael. 1996. A new statistical parser based on bigram lexical dependencies. In *Proceedings of ACL-96*, Sant Cruz, CA, USA.

Cutting, Doug, Julian Kupiec, Jan Pedersen, and Penelope Sibun. 1992. A practical part-of-speech tagger. In *Proceedings of the 3rd Conference on Applied Natural Language Processing (ACL)*, pages 133–140.

Eisner, Jason M. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of COLING-96*, Kopenhagen, Denmark.

Feldweg, Helmut. 1995. Implementation and evaluation of a german hmm for pos disambiguation.

In *Proceedings of EACL-SIGDAT-95 Workshop*, Dublin, Ireland.

Lehmann, Sabine, Stephan Oepen, Sylvie Regnier-Prost, Klaus Netter, Veronika Lux, Judith Klein, Kirsten Falkedal, Frederik Fouvry, Dominique Estival, Eva Dauphin, Hervé Compagnion, Judith Baur, Lorna Balkan, and Doug Arnold. 1996. TSNLP — Test Suites for Natural Language Processing. In *Proceedings of COLING 1996*, Kopenhagen.

Marcus, Mitchell, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. 1994. The penn treebank: Annotating predicate argument structure. In *Proceedings of the Human Language Technology Workshop*, San Francisco, Morgan Kaufmann.

Rabiner, L. R. 1989. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, volume 77(2), pages 257–285.

Sampson, Geoffrey. 1995. *English for the Computer.* Oxford University Press, Oxford.

Skut, Wojciech, Brigitte Krenn, Thorsten Brants, and Hans Uszkoreit. 1997. An annotation scheme for free word order languages. In *Proceedings of ANLP-97*, Washington, DC.

Tapanainen, Pasi and Timo Järvinen. 1997. A non-projective dependency parser. In *Proceedings of ANLP-97*, Washington, DC.

Thielen, Christine and Anne Schiller. 1995. Ein kleines und erweitertes Tagset fürs Deutsche. In *Tagungsberichte des Arbeitstreffens Lexikon + Text 17./18. Februar 1994, Schloß Hohentübingen. Lexicographica Series Maior*, Tübingen. Niemeyer.

Appendix A: Tagsets

This section contains descriptions of tags used in this paper. These are *not* complete lists.

A.1 Part-of-Speech Tags

We use the Stuttgart-Tübingen-Tagset. The complete set is described in (Thielen and Schiller, 1995).

ADJA	attributive adjective
ADJD	adverbial adjective
ADV	adverb
APPR	preposition
ART	article
CARD	cardinal number
FM	foreign material
KOKOM	comparing conjunction
KOUS	sub-ordinating conjunction
NE	proper noun
NN	common noun
PIAT	indefinite pronoun
PPER	personal pronoun
PTKNEG	negation
VAFIN	finite auxiliary
VMFIN	finite modal verb
VVPP	past participle of main verb

A.2 Phrasal Categories

AP	adjective phrase
CS	coordination of sentences
DL	discurse level
MPN	multi-word proper noun
NM	multi-token numerical
NP	noun phrase
PP	prepositional phrase
S	sentence
VP	verb phrase

A.3 Grammatical Functions

AC	adpositional case marker
CJ	conjunct
DA	dative
HD	head
JU	junctor
MNR	post-nominal modifier
MO	modifier
NG	negation
NK	noun kernel
OA	accusative object
OC	clausal object
PD	predicative
PG	pseudo genitive
PNC	proper noun component
SB	subject
SBP	passivized subject
SP	subject or predicative