Construction of a German HPSG grammar from a detailed treebank

Bart Cramer[†] **and Yi Zhang**[†‡]
Department of Computational Linguistics & Phonetics, Saarland University, Germany[†]
LT-Lab, German Research Center for Artificial Intelligence, Germany[‡]
{bcramer,yzhang}@coli.uni-saarland.de

Abstract

Grammar extraction in deep formalisms has received remarkable attention in recent years. We recognise its value, but try to create a more precision-oriented grammar, by hand-crafting a core grammar, and learning lexical types and lexical items from a treebank. The study we performed focused on German, and we used the Tiger treebank as our resource. A completely hand-written grammar in the framework of HPSG forms the inspiration for our core grammar, and is also our frame of reference for evaluation. [1]

1 Introduction

Previous studies have shown that treebanks can be helpful when constructing grammars. The most well-known example is PCFG-based statistical parsing (Charniak and Johnson, 2005), where a PCFG is induced from, for instance, the Penn Treebank. The underlying statistical techniques have been refined in the last decade, and previous work indicates that the labelled f-score of this method converges to around 91%.

An alternative to PCFGs, with more linguistic relevance, is formed by deeper formalisms, such as TAG (Joshi and Schabes, 1997), CCG (Steedman, 1996), LFG (Kaplan and Bresnan, 1995) and HPSG (Pollard and Sag, 1994). For LFG (Butt et al., 2002) and HPSG (Flickinger, 2000; Müller, 2002), large hand-written grammars have been developed. In the case of HPSG, the grammar writers found the small number of principles too restrictive, and created more rules (approximately 50 to 300) to accommodate for phenomena that vanilla HPSG cannot describe correctly. The increased linguistic preciseness comes at a cost, though: such grammars have a lower out-of-the-box coverage, i.e. they will not give an analysis on a certain portion of the corpus.

Experiments have been conducted, where a lexicalised grammar is learnt from treebanks, a methodology for which we coin the name *deep grammar extraction*. The basic architecture of such an experiment is to convert the treebank to a format that is compatible with the chosen linguistic formalism, and read off the lexicon from that converted treebank. Because all these formalisms are heavily lexicalised, the core grammars only consist of a small number of principles or operators. In the case of CCG (Hockenmaier and Steedman, 2002), the core grammar consists of the operators that CCG stipulates: function application, composition and type-raising. Standard HPSG defines a few schemata, but these are usually adapted for a large-scale grammar. Miyao et al. (2004) tailor their core grammar for optimal use with the Penn Treebank and the English language, for example by adding a new schema for relative clauses.

Hockenmaier and Steedman (2002), Miyao et al. (2004) and Cahill et al. (2004) show fairly good results on the Penn Treebank (for CCG, HPSG and LFG, respectively): these parsers achieve accuracies on predicate-argument relations between 80% and 87%, which show the feasibility and scalability of this approach. However, while this is a simple method for a highly configurational language like English, it is more difficult to extend to languages with more complex morphology or with word orders that display more freedom. Hockenmaier (2006) is the only study known to the authors that applies this method to German, a language that displays these properties.

This article reports on experiments where the advantages of hand-written and derived grammars

[1] The research reported in this paper has been carried out with financial support from the Deutsche Forschungsgemeinschaft and the German Excellence Cluster of Multimodal Computing & Interaction.

Proceedings of the 2009 Workshop on Grammar Engineering Across Frameworks, ACL-IJCNLP 2009, pages 37–45,
Suntec, Singapore, 6 August 2009. ©2009 ACL and AFNLP

are combined. Compared to previous deep grammar extraction approaches, a more sophisticated core grammar (in the framework of HPSG) is created. Also, more detailed syntactic features are learnt from the resource treebank, which leads to a more precise lexicon. Parsing results are compared with GG (German Grammar), a previously hand-written German HPSG grammar (Müller, 2002; Crysmann, 2003; Crysmann, 2005).

2 Core grammar

2.1 Head-driven phrase structure grammar

This study has been entirely embedded in the HPSG framework (Pollard and Sag, 1994). This is a heavily lexicalised, constraint-based theory of syntax, and it uses typed feature structures as its representation. HPSG introduces a small number of principles (most notably, the Head Feature Principle) that guide the construction of a few Immediate Dominance schemata. These schemata are meant to be the sole basis to combine words and phrases. Examples of schemata are head-complement, head-subject, head-specifier, head-filler (for long-distance dependencies) and head-modifier.

In this study, the core grammar is an extension of the off-the-shelf version of HPSG. The type hierarchy is organised by a typed feature structure hierarchy (Carpenter, 1992), and can be read by the LKB system (Copestake, 2002) and the PET parser (Callmeier, 2000). The output is given in Minimal Recursion Semantics (Copestake et al., 2005) format, which can be minimally described as a way to include scope information in dependency output.

2.2 The German language

Not unlike English, German uses verb position to distinguish between different clause types. In declarative sentences, verbs are positioned in the second position, while subordinate classes are verb-final. Questions and imperatives are verb-initial. However, German displays some more freedom with respect to the location of subjects, complements and adjuncts: they can be scrambled rather freely. The following sentences are all grammatical, and have approximately the same meaning:

(1) a. Der Präsident hat gestern das Buch gelesen.
The.NOM President.NOM has yesterday the.ACC book.ACC read.PERF.
'The president read the book yesterday'

b. Gestern hat der Präsident das Buch gelesen.

c. Das Buch hat der Präsident gestern gelesen.

As can be seen, the main verb is placed at second position (the so-called 'left bracket'), but all other verbs remain at the end of the sentence, in the 'right bracket'. Most linguistic theories about German recognise the existence of topological fields: the *Vorfeld* before the left bracket, the *Mittelfeld* between both brackets, and the *Nachfeld* after the right bracket. The first two are mainly used for adjuncts and arguments, whereas the *Nachfeld* is typically, but not necessarily, used for extraposed material (e.g. relative clauses or comparative phrases) and some VPs. Again, the following examples mean roughly the same:

(2) a. Er hat das Buch, das sie empfohlen hat, gelesen.
He has the.ACC Book.ACC, that she recommended has, read.PERF.
He has read the book that she recommended.

b. Er hat das Buch gelesen, das sie empfohlen hat.

c. Das Buch hat er gelesen, das sie empfohlen hat.

Another distinctive feature of German is its relatively rich morphology. Nominals are marked with case, gender and number, and verbs with number, person, tense and mood. Adjectives and nouns have to agree with respect to gender, number and declension type, the latter being determined by the (non-)existence and type of determiner used in the noun phrase. Verbs and subjects have to agree with respect to number and person. German also displays highly productive noun compounding, which amplifies the need for effective unknown word handling. Verb particles can either be separated from or concatenated to the verb: compare 'Er schläft aus' ('He sleeps in') and 'Er

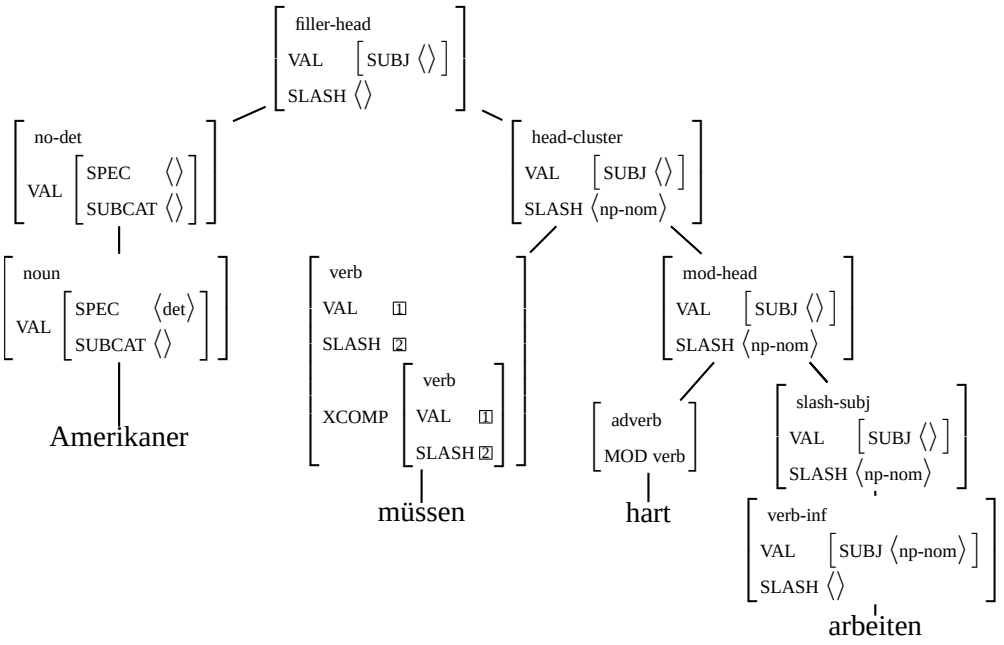

Figure 1: This figure shows a (simplified) parse tree of the sentence 'Amerikaner müssen hart arbeiten' ('Americans have to work hard').

wird ausschlafen' ('He will sleep in'). In such verbs, the word 'zu' (which translates to the English 'to' in 'to sleep') can be infixed as well: 'er versucht auszuschlafen' ('He tries to sleep in').

These characteristics make German a comparatively complex language to parse with CFGs: more variants of the same lemma have to be memorised, and the expansion of production rules will be more diverse, with a less peaked statistical distribution. Efforts have been made to adapt existing CFG models to German (Dubey and Keller, 2003), but the results still don't compare to state-of-the-art parsing of English.

2.3 Structure of the core grammar

The grammar uses the main tenets from Head-driven Phrase Structure Grammar (Pollard and Sag, 1994). However, different from earlier deep grammar extraction studies, more sophisticated structures are added. Müller (2002) proposes a new schema (head-cluster) to account for verb clusters in the right bracket, which includes the possibility to merge subcategorisation frames of e.g. object-control verbs and its dependent verb. Separate rules for determinerless NPs, genitive modification, coordination of common phrases, relative phrases and direct speech are also created.

The free word order of German is accounted for by scrambling arguments with lexical rules, and by allowing adjuncts to be a modifier of unsaturated verb phrases. All declarative phrases are considered to be head-initial, with an adjunct or argument fronted using the SLASH feature, which is then discharged using the head-filler schema. The idea put forward by, among others, (Kiss and Wesche, 1991) that all sentences should be right-branching is linguistically pleasing, but turns out be computationally very expensive (Crysmann, 2003), and the right-branching reading should be replaced by a left-branching reading when the right bracket is empty (i.e. when there is no auxiliary verb present).

An example of a sentence is presented in figure 1. It receives a right-branching analysis, because the infinitive 'arbeiten' resides in the right bracket. The unary rule slash-subj moves the required subject towards the SLASH value, so that it can be discharged in the *Vorfeld* by the head-filler schema. 'müssen' is an example of an argument attraction verb, because it pulls the valence feature (containing SUBJ, SUBCAT etc; not visible in the diagram) to itself. The head-cluster rule assures that the VAL value then percolates upwards. Because 'Amerikaner' does not have a specifier, a separate unary rule (no-det) takes care of discharging the SPEC feature, before it can be combined with the filler-head rule.

As opposed to (Hockenmaier, 2006), this study

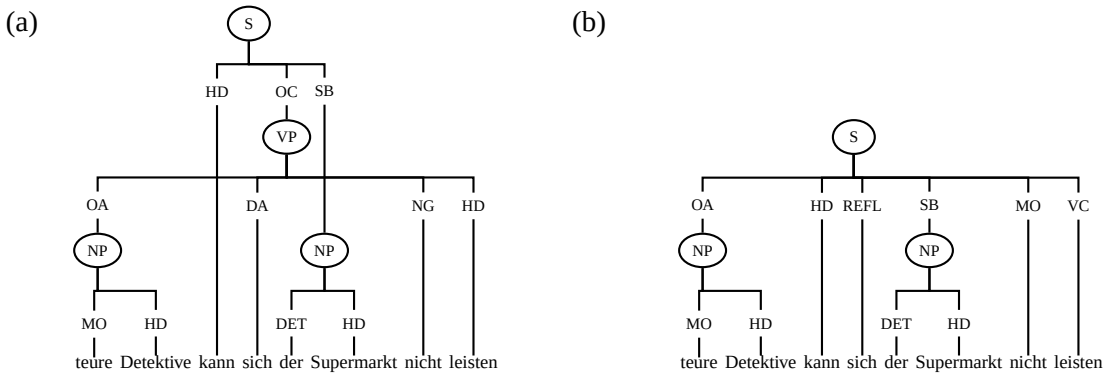

Figure 2: (a) shows the original sentence, whereas (b) shows the sentence after preprocessing. Note that NP is now headed, that the VP node is deleted, and that the verbal cluster is explicitly marked in (b). The glossary of this sentence is 'Expensive.ACC detectives.ACC can REFL the.NOM supermarket.NOM not afford'

employs a core lexicon for words that have marked semantic behaviour. These are usually closed word classes, and include items such as raising and auxiliary verbs, possessives, reflexives, articles, complementisers etc. The size of this core lexicon is around 550 words. Note that, because the core lexicon only contains function words, its coverage is negligible without additional entries.

3 Derivation of the lexicon

3.1 The Tiger treebank

The Tiger treebank (Brants et al., 2002) is a treebank that embraces the concept of constituency, but can have crossing branches, i.e. the tree might be non-projective. This allowed the annotators to capture the German free word order. Around one-third of the sentences received a non-projective analysis. An example can be found in figure 2. Additionally, it annotates each branch with a syntactic function.

The text comes from a German newspaper (Frankfurter Rundschau). It was annotated semi-automatically, using a cascaded HMM model. After each phase of the HMM model, the output was corrected by human annotators. The corpus consists of over 50,000 sentences, with an average sentence length of 17.6 tokens (including punctuation). The treebank employs 26 phrase categories, 56 PoS tags and 48 edge labels. It also encodes number, case and gender at the noun terminals, and tense, person, number and mood at verbs. Whether a verb is finite, an infinitive or a participle is encoded in the PoS tag. A peculiarity in the annotation of noun phrases is the lack of headedness, which was meant to keep the annotation as theory-independent as reasonably possible.

3.2 Preprocessing

A number of changes had to be applied to the treebank to facilitate the read-off procedure:

- A heuristic head-finding procedure is applied in the spirit of (Magerman, 1995). We use priority lists to find the NP's head, determiner, appositions and modifiers. PPs and CPs are also split into a head and its dependent.
- If a verb has a separated verb particle, this particle is attached to the lemma of the verb. For instance, if the verb 'schlafen' has a particle 'aus', the lemma will be turned into 'ausschlafen' ('sleep in'). If this is not done, subcategorisation frames will be attributed to the wrong lemma.
- Sentences with auxiliaries are non-projective, if the adjunct of the embedded VP is in the *Vorfeld*. This can be solved by flattening the tree (removing the VP node), and marking the verbal cluster (VC) explicitly. See figure 2 for an example. 67.6% of the original Tiger treebank is projective, and with this procedure, this is lifted to 80.1%.
- The Tiger treebank annotates reflexive pronouns with the PoS tag PRF, but does not distinguish the syntactic role. Therefore, if a verb has an object that has PRF as its part-of-speech, the label of that edge is changed into REFL, so that reflexive verbs can be found.

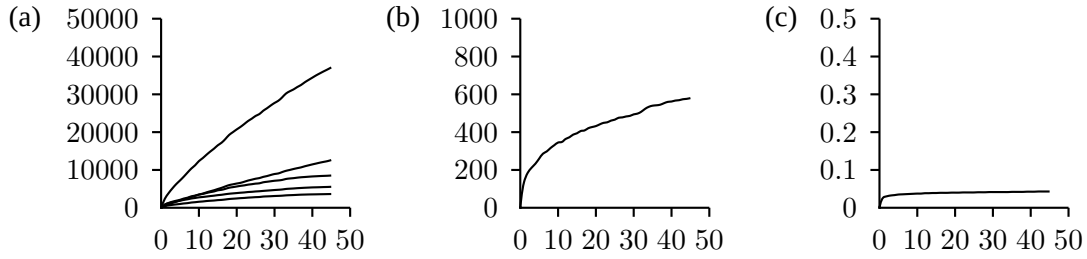

Figure 3: These graphs show learning curves of the algorithm on the first 45,000 sentences of the Tiger treebank. Graph (a) indicates the amount of lemmas learnt (from top to bottom: nouns, names, adjectives, verbs and adverbs). The graph in (b) shows the number of distinct lexical types for verbs that are learnt. Graph (c) shows the average proportion of morphological forms that is observed per verb lemma, assuming that each verb has 28 different forms: infinitive, zu (to) + infinitive, participle, imperative and 24 finite forms (3 (person) * 2 (number) * 2 (tense) * 2 (mood)).

The preprocessing stage failed in 1.1% of the instances.

3.3 Previous work

The method described in Hockenmaier (2006) first converts the Tiger analysis to a tree, after which the lexical types were derived. Because it was the author's goal to convert all sentences, some rather crude actions had to be taken to render non-projective trees projective: whenever a certain node introduces non-projectivity, some of its daughters are moved to the parent tree, until that node is projective. Below, we give two examples where this will lead to incorrect semantic composition, with the consequence of flawed lexicon entries. We argue that it is questionable whether the impressive conversion scores actually represent a high conversion quality. It would be interesting to see how this grammar performs in a real parsing task, but no such study has been carried out so far.

The first case deals with extraposed material in the *Nachfeld*. Typical examples include relative phrases, comparatives and PH/RE constructions[2]. The consequence is that the head of the extraposed material will be connected to the verb, instead of to the genuine head.

Another example where Hockenmaier's algorithm will create incorrect lexical entries is when the edge label is PAR (for 'parentheses') or in the case of appositions. Consider the following sentence:

(3) mit 160 Planstellen (etliche sind
with 160 permanent posts (several are
allerdings noch unbesetzt)
however still unoccupied)

The conclusion that will be drawn from this sentence is that 'sind' can modify nouns, which is only true due to the parentheses, and has no relation with the specific characteristics of 'sind'. Similarly, appositions will act as modifiers of nouns. Although one might argue that this is the canonical CCG derivation for these phenomena, it is not in the spirit of the HPSG grammars, and we believe that these constructions are better handled in rules than in the lexicon.

3.4 Procedure

In our approach, we will be more conservative, and the algorithm will only add facts to its knowledge base if the evidence is convincing. That means that less Tiger graphs will get projective analyses, but that doesn't have to be a curse: we can derive lexical types from non-projective analyses just as well, and leave the responsibility for solving the more complex grammatical phenomena to the core grammar. For example, lexical rules will deal with fronting and *Mittelfeld* scrambling, as we have stated before. This step of the

[2]NPs, AVPs and PPs can, instead of their usual headed structure, be divided in two parts: a 'placeholder' and a 'repeated element'. These nodes often introduce non-projectivity, and it is not straightforward to create a valid linguistic analysis for these phenomena. Example sentences of these categories (NPs, AVPs and PPs, respectively) are:

(1) [PH Es] ist wirklich schwer zu sagen, [RE welche Positionen er einnimmt]

(2) Man muß sie also [PH so] behandeln , [RE wie man eine Weltanschauungsbewegung behandelt]

(3) Alles deutet [PH darauf] hin [RE daß sie es nicht schaffen wird]

These examples all have the RE in the *Nachfeld*, but their placement actually has a large variety.

procedure has indeed strong affinity with deep lexical acquisition, except for the fact that in DLA all lexical types are known, and this is not the case in this study: the hand-written lexical type hierarchy is still extended with new types that are derived from the resource treebank, mostly for verbs.

The basic procedure is as follows:

- Traverse the graph top-down.
- For each node:
  - Identify the node's head (or the deepest verb in the verb cluster[3]);
  - For each complement of this node, add this complement to the head's subcategorisation frame.
  - For each modifier, add this head to the possible MOD values of the modifier's head.
- For each lexical item, a mapping of (lemma, morphology) → word form is created.

After this procedure, the following information is recorded for the verb lemma 'leisten' from figure 2:

- It has a subcategorisation frame 'npnom-reflnpacc'.
- Its infinitive form is 'leisten'.

The core grammar defines that possible subjects are nominative NPs, expletive 'es' and CPs. Expletives are considered to be entirely syntactic (and not semantic), so they will not receive a dependency relation. Complements may include predicative APs, predicative NPs, genitive, dative and accusative NPs, prepositional complements, CPs, reflexives, separable particles (also purely syntactic), and any combination of these. For non-verbs, the complements are ordered (*i.e.* it is a list, and not a verb). Verb complementation patterns are sets, which means that duplicate complements are not allowed. For verbs, it is also recorded whether the auxiliary verb to mark the perfect tense should be either 'haben' (default) or 'sein' (mostly verbs that have to do with movement). Nouns are annotated with whether they can have appositions or not.

[3]That means that the head of a S/VP-node is assumed to be contained in the lexicon, as it must be some sort of auxiliary.

Results from the derivation procedure are graphed in figure 3. The number of nouns and names is still growing after 45,000 sentences, which is an expected result, given the infinite nature of names and frequent noun compounding. However, it appears that verbs, adjectives and adverbs are converging to a stable level. On the other hand, lexical types are still learnt, and this shows a downside of our approach: the deeper the extraction procedure is, the more data is needed to reach the same level of learning.

The core grammar contains a little less than 100 lexical types, and on top of that, 636 lexical types are learnt, of which 579 are for verbs. It is interesting to see that the number of lexical types is considerably lower than in (Hockenmaier, 2006), where around 2,500 lexical types are learnt. This shows that our approach has a higher level of generalisation, and is presumably a consequence of the fact that the German CCG grammar needs distinct lexical types for verb-initial and verb-final constructions, and for different argument scramblings in the *Mittelfeld*, whereas in our approach, hand-written lexical rules are used to do the scrambling.

The last graph shows that the number of word forms is still insufficient. We assume that each verb can have 28 different word forms. As can be seen, it is clear that only a small part of this area is learnt. One direction for future research might be to find ways to automatically expand the lexicon after the derivation procedure, or to hand-code morphological rules in the core grammar.

4 Parsing

4.1 Methodology

All experiments in this article use the first 45,000 sentences as training data, and the consecutive 5,000 sentences as test data. The remaining 472 sentences are not used. We used the PET parser (Callmeier, 2000) to do all parsing experiments. The parser was instructed to yield a parse error after 50,000 passive edges were used. Ambiguity packing (Oepen and Carroll, 2000) and selective unpacking (Zhang et al., 2007) were used to reduce memory footprint and speed up the selection of the top-1000 analyses. The maximum entropy model, used for selective unpacking, was based on 200 treebanked sentences of up to 20 words from the training set. Part-of-speech tags delivered by the stock version of the TnT tagger (Brants,) were

	Tiger	T.+TnT	GG
Out of vocabulary	71.9 %	5.2 %	55.6 %
Parse error	0.2 %	1.5 %	0.2 %
Unparsed	7.9 %	37.7 %	28.2 %
Parsed	20.0 %	55.6 %	16.0 %
Total	100.0 %	100.0 %	100.0 %
Avg. length	8.6	12.8	8.0
Avg. nr. of parses	399.0	573.1	19.2
Avg. time (s)	9.3	15.8	11.6

Table 1: This table shows coverage results on the held-out test set. The first column denotes how the extracted grammar performs without unknown word guessing. The second column uses PoS tags and generic types to guide the grammar when an unknown word is encountered. The third column is the performance of the fully hand-written HPSG German grammar by (Müller, 2002; Crysmann, 2003). OOV stands for out-of-vocabulary. A parse error is recorded when the passive edge limit (set to 50,000) has been reached. The bottom three rows only gives information about the sentences where the grammar actually returns at least one parse.

	Training set	Test set
All	100.0 %	100.0 %
Avg. length	14.2	13.5
Coverage	79.0 %	69.0 %
Avg. length	13.2	12.8
Correct (top-1000)	52.0%	33.5 %
Avg. length	10.4	8.5

Table 2: Shown are the treebanking results, giving an impression of the quality of the parses. The 'training set' and 'test set' are subsets of 200 sentences from the training and test set, respectively. 'Coverage' means that at least one analysis is found, and 'correct' indicates that the perfect solution was found in the top-1000 parses.

used when unknown word handling was turned on. These tags were connected to generic lexical types by a hand-written mapping. The version of GG that was employed (Müller, 2002; Crysmann, 2003) was dated October 2008[4].

4.2 Results

Table 1 shows coverage figures in three different settings. It is clear that the resulting grammar has a higher coverage than the GG, but this comes at a cost: more ambiguity, and possibly unnecessary ambiguity. Remarkably, the average processing time is lower, even when the sentence lengths and ambiguity rates are higher. We attribute this to the smaller feature structure geometry that is introduced by the core grammar (compared to the GG). Using unknown word handling immediately improved the coverage, by a large margin. Larger ambiguity rates were recorded, and the number of parser errors slightly increased.

Because coverage does not imply quality, we wanted to look at the results in a qualitative fashion. We took a sample of 200 sentences from both the training and the test set, where the ones from the training set did not overlap with the set used to train the MaxEnt model, so that both settings were equally influenced by the rudimentary MaxEnt model. We evaluated for how many sentences the exactly correct parse tree could be found among the top-1000 parses (see table 2). The difference between the performance on the training and test set give an idea of how well the grammar performs on unknown data: if the difference is small, the grammar extends well to unseen data. Compared to evaluating on lexical coverage, we believe this is a more empirical estimation of how close the acquisition process is to convergence.

Based on the kind of parse trees we observed, the impression was that on both sets, performance was reduced due to the limited predictive power of the disambiguation model. There were quite a few sentences for which good parses could be expected, because all lexical entries were present. This experiment also showed that there were systematic ambiguities that were introduced by inconsistent annotation in the Tiger treebank. For in-

[4]It should be noted that little work has gone in to providing unknown word handling mechanisms, and that is why we didn't include it in our results. However, in a CoNLL-2009 shared task paper (Zhang et al., 2009), a coverage of 28.6% was reported when rudimentary methods were used.

stance, the word 'ein' was learnt as both a number (the English 'one') and as an article ('a'), leading to spurious ambiguities for each noun phrase containing the word 'ein', or one of its morphological variants. These two factors reinforced each other: if there is spurious ambiguity, it is even harder for a sparsely trained disambiguation model to pull the correct parse inside the top-1000.

The difference between the two 'correct' numbers in table 2 is rather large, meaning that the 'real' coverage might seem disappointingly low. Not unexpectedly, we found that the generic lexical types for verbs (transitive verb, third person singular) and nouns (any gender, no appositions allowed) was not always correct, harming the results considerably.

A quantitative comparison between deep grammars is always hard. Between DELPH-IN grammars, coverage has been the main method of evaluation. However, this score does not reward richness of the semantic output. Recent evidence from the ERG (Ytrestøl et al., 2009) suggests that the ERG reaches a top-500 coverage of around 70% on an unseen domain, a result that this experiment did not approximate. The goal of GG is not computational, but it serves as a testing ground for linguistic hypotheses. Therefore, the developers have never aimed at high coverage figures, and crafted the GG to give more detailed analyses and to be suited for both parsing and generation. We are happy to observe that the coverage figures in this study are higher than GG's (Zhang et al., 2009), but we realise the limited value of this evaluation method. Future studies will certainly include a more granular evaluation of the grammar's performance.

5 Conclusion and discussion

We showed how a precise, wide-coverage HPSG grammar for German can be created successfully, by constructing a core grammar by hand, and appending it with linguistic information from the Tiger treebank. Although this extracted grammar suffers considerably more from overgeneration than the hand-written GG, we argue that our conservative derivation procedure delivers a more detailed, compact and correct compared to previous deep grammar extraction efforts. The use of the core lexicon allows us to have more linguistically motivated analyses of German than approaches where the core lexicon only comprises the textbook principles/operators. We compared our lexicon extraction results to those from (Hockenmaier, 2006). Also, preliminary parsing experiments are reported, in which we show that this grammar produces reasonable coverage on unseen text.

Although we feel confident about the successful acquisition of the grammar, there still remain some limiting factors in the performance of the grammar when actually parsing. Compared to coverage figures of around 80%, reported by (Riezler et al., 2001), the proportion of parse forests containing the correct parse in this study is rather low. The first limit is the constructional coverage, meaning that the core grammar is not able to construct the correct analysis, even though all lexical entries have been derived correctly before. The most frequent phenomena that are not captured yet are PH/RE constructions and extraposed clauses, and we plan to do an efficient implementation (Crysmann, 2005) of these in a next version of the grammar. Second, as shown in figure 3, data scarcity in the learning of the surface forms of lemmas negatively influences the parser's performance on unseen text.

In this paper, we focused mostly on the correctness of the derivation procedure. We would like to address the real performance of the grammar/parser combination in future work, which can only be done when parses are evaluated according to a more granular method than we have done in this study. Furthermore, we ran into the issue that there is no straightforward way to train larger statistical models automatically, which is due to the fact that our approach does not convert the source treebank to the target formalism's format (in our case HPSG), but instead reads off lexical types and lexical entries directly. We plan to investigate possibilities to have the annotation be guided automatically by the Tiger treebank, so that the disambiguation model can be trained on a much larger amount of training data.

Acknowledgements

We would like to thank Rebecca Dridan, Antske Fokkens, Stephan Oepen and the anonymous reviewers for their valuable contributions to this paper.

References

T. Brants. TnT: a statistical part-of-speech tagger. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*.

S. Brants, S. Dipper, S. Hansen, W. Lezius, and G. Smith. 2002. The TIGER Treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, pages 24–41.

M. Butt, H. Dyvik, T.H. King, H. Masuichi, and C. Rohrer. 2002. The parallel grammar project. In *International Conference On Computational Linguistics*, pages 1–7.

A. Cahill, M. Burke, R. ODonovan, J. Van Genabith, and A. Way. 2004. Long-distance dependency resolution in automatically acquired wide-coverage PCFG-based LFG approximations. In *Proceedings of ACL-2004*, pages 320–327.

U. Callmeier. 2000. PET–a platform for experimentation with efficient HPSG processing techniques. *Natural Language Engineering*, 6(01):99–107.

B. Carpenter. 1992. *The Logic of Typed Feature Structures: With Applications to Unification Grammars, Logic Programs, and Constraint Resolution*. Cambridge University Press, Cambridge, UK.

E. Charniak and M. Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of ACL-2005*, pages 173–180.

A. Copestake, D. Flickinger, C. Pollard, and I. Sag. 2005. Minimal Recursion Semantics: An Introduction. *Research on Language & Computation*, 3(4):281–332.

A. Copestake. 2002. *Implementing Typed Feature Structure Grammars*. CSLI Publications, Stanford, CA, USA.

B. Crysmann. 2003. On the efficient implementation of German verb placement in HPSG. In *Proceedings of RANLP-2003*, pages 112–116.

B. Crysmann. 2005. Relative Clause Extraposition in German: An Efficient and Portable Implementation. *Research on Language & Computation*, 3(1):61–82.

A. Dubey and F. Keller. 2003. Probabilistic parsing for German using sister-head dependencies. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 96–103.

D. Flickinger. 2000. On building a more effcient grammar by exploiting types. *Natural Language Engineering*, 6(1):15–28.

J. Hockenmaier and M. Steedman. 2002. Acquiring compact lexicalized grammars from a cleaner treebank. In *Proceedings of LREC-2002*, pages 1974–1981.

J. Hockenmaier. 2006. Creating a CCGbank and a Wide-Coverage CCG Lexicon for German. In *Proceedings of ACL-2006*, pages 505–512.

A.K. Joshi and Y. Schabes. 1997. Tree-adjoining grammars. *Handbook of formal languages*, 3:69–124.

R.M. Kaplan and J. Bresnan. 1995. Lexical-Functional Grammar: A formal system for grammatical representation. *Formal Issues in Lexical-Functional Grammar*, pages 29–130.

T. Kiss and B. Wesche. 1991. Verb order and head movement. *Text Understanding in LILOG, Lecture Notes in Artificial Intelligence*, 546:216–242.

D. Magerman. 1995. Statistical decision-tree models for parsing. In *Proceedings of ACL-1995*, pages 276–283.

Y. Miyao, T. Ninomiya, and J. Tsujii. 2004. Corpus-oriented grammar development for acquiring a Head-driven Phrase Structure Grammar from the Penn Treebank. In *Proceedings of IJCNLP-2004*.

S. Müller. 2002. *Complex Predicates: Verbal Complexes, Resultative Constructions, and Particle Verbs in German*. CSLI Publications, Stanford, CA, USA.

S. Oepen and J. Carroll. 2000. Ambiguity packing in constraint-based parsing: practical results. In *Proceedings of NAACL-2000*, pages 162–169.

C.J. Pollard and I.A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. University Of Chicago Press, Chicago, IL, USA.

S. Riezler, T.H. King, R.M. Kaplan, R. Crouch, J.T. Maxwell III, and M. Johnson. 2001. Parsing the Wall Street Journal using a Lexical-Functional Grammar and discriminative estimation techniques. In *Proceedings of ACL-2001*, pages 271–278.

M. Steedman. 1996. *Surface structure and interpretation*. MIT Press, Cambridge, MA, USA.

G. Ytrestøl, D. Flickinger, and S. Oepen. 2009. Extracting and Annotating Wikipedia Sub-Domains. In *Proceedings of the Seventh International Workshop on Treebanks and Linguistic Theories*, pages 185–197.

Y. Zhang, S. Oepen, and J. Carroll. 2007. Efficiency in Unification-Based N-Best Parsing. In *Proceedings of the Tenth International Conference on Parsing Technologies*, pages 48–59.

Y. Zhang, R. Wang, and S.. Oepen. 2009. Hybrid Multilingual Parsing with HPSG for SRL. In *Proceedings of CoNLL-2009, to appear*.