Stochastic Realisation Ranking for a Free Word Order Language

Aoife Cahill, Martin Forst and Christian Rohrer
Institute of Natural Language Processing
University of Stuttgart, Germany
{cahillae|forst|rohrer}@ims.uni-stuttgart.de

Abstract

We present a log-linear model that is used for ranking the string realisations produced for given corpus f-structures by a reversible broad-coverage LFG for German and compare its results with the ones achieved by the application of a language model (LM). Like other authors that have developed log-linear models for realisation ranking, we use a hybrid model that uses linguistically motivated learning features *and* a LM (whose score is simply integrated into the log-linear model as an additional feature) for the task of realisation ranking. We carry out a large evaluation of the model, training on over 8,600 structures and testing on 323. We observe that the contribution that the structural features make to the quality of the output is slightly greater in the case of a free word order language like German than it is in the case of English. The exact match metric improves from 27% to 37% when going from the LM-based realisation ranking to the hybrid model, BLEU score improves from 0.7306 to 0.7939.

1 Introduction

Most traditional approaches to stochastic realisation ranking involve applying language model n-gram statistics to rank alternatives (Langkilde, 2000; Bangalore and Rambow, 2000; Langkilde-Geary, 2002). Much work has been carried out into statistical realisation ranking for English. However, n-grams alone (even if they are efficiently implemented) may not be a good enough measure for ranking candidate strings, particularly in free-word order languages.

Belz (2005) moves away from n-gram models of generation and trains a generator on a generation treebank, achieving similar results to a bigram model but at much lower computational cost. Cahill and van Genabith (2006) do not use a language model, but rather rely on treebank-based automatically derived LFG generation grammars to determine the most likely surface order.

Ohkuma (2004) writes an LFG generation grammar for Japanese separate from the Japanese LFG parsing grammar in order to enforce canonical word order by symbolic means. In another purely symbolic approach, Callaway (2003; 2004) describes a wide-coverage system and the author argues that there are several advantages to a symbolic system over a statistical one. We argue that a reversible symbolic system, which is desirable for maintainability and modularity reasons, augmented with a statistical ranking component can produce systematically ranked, high quality surface realisations while maintaining the flexibility associated with hand-crafted systems.

Velldal et al. (2004) and Velldal and Oepen (2005) present discriminative disambiguation models using a hand-crafted HPSG grammar for generation from MRS (Minimal Recursion Semantics) structures. They describe three statistical models for realization ranking: The first is a simple n-gram language model, the second uses structural features in a maximum entropy model for disambiguation and a third uses a combination of the two models. Their results show that the third model where the n-gram language model is combined with the structural features in the maximum entropy disambiguation model performs best. Nakanishi et al. (2005) present similar probabilistic models for a chart generator using a HPSG grammar acquired from the Penn-II Treebank (the Enju HPSG), with the difference that, in their experiments, the model that only uses structural features outperformed the hybrid model. We present a model for realisation ranking similar to the models just mentioned. The main differences between our work and theirs is that we are working within the LFG framework and concentrating on a less configurational language: German.

2 Background

2.1 Lexical Functional Grammar

The work presented in this paper is couched in the framework of Lexical Functional Grammar (LFG). LFG is a grammar formalism which makes use of two representation levels to encode syntactic properties of sentences: constituent structure (c-structure) and functional structure (f-structure). C-structures are context-free trees that encode constituency and linear order. F-structures are attribute-value matrices that encode grammatical relations and morphosyntactic features. While translational equivalents of sentences may vary considerably across languages at the c-structure level, it is assumed that, at the f-structure level, where linear order is abstracted away from, languages behave much more alike. Also, f-structures are taken as the interface from syntax to semantics. From a language-technological perspective, f-structures are the level of representation various (prototypes of) question-answering systems, a sentence condensation system and machine translation systems operate on and generate from.

Figures 1 and 2 illustrate the c-structure and the f-structure that the German broad-coverage LFG presented in the next paragraph produces for (1).

(1)

Verheugen	habe	die	Worte	des
Verheugen	had	the	words	the-GEN
Generalinspekteurs	falsch	interpretiert.		
inspector-general	wrongly	interpreted.		

‘Verheugen had mis-interpreted the words of the inspector-general.’

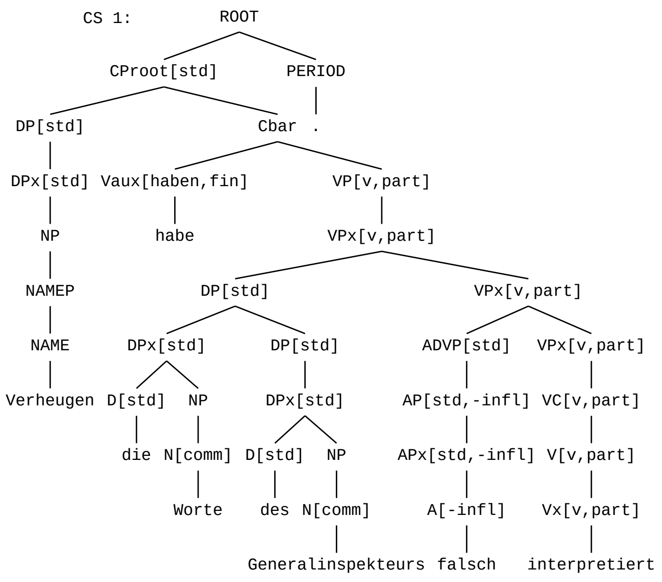

Figure 1: C-structure for (1)

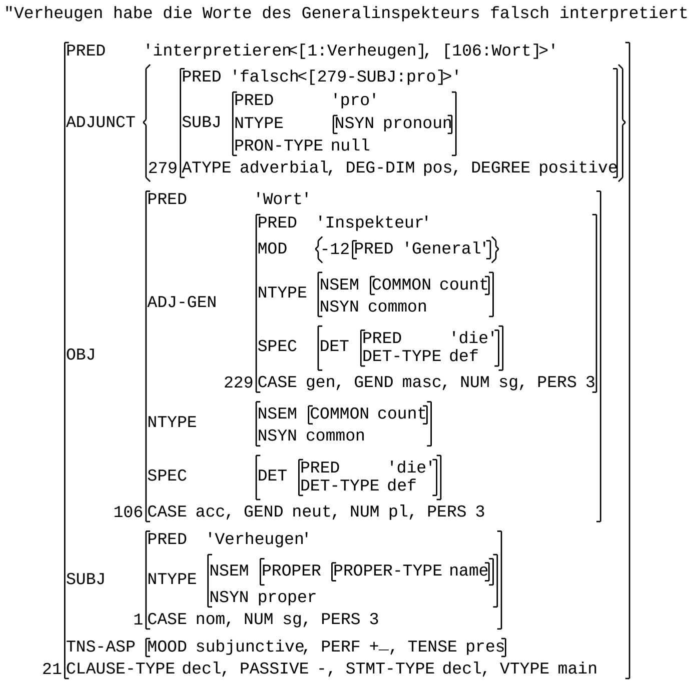
"Verheugen habe die Worte des Generalinspekteurs falsch interpretiert
PRED 'interpretieren<[1:Verheugen], [106:Wort]>'
ADJUNCT
PRED 'falsch<[279-SUBJ:pro]>'
SUBJ
PRED 'pro'
NTYPE NSYN pronoun
PRON-TYPE null
279 ATYPE adverbial, DEG-DIM pos, DEGREE positive
OBJ
PRED 'Wort'
ADJ-GEN
PRED 'Inspekteur'
MOD -12 PRED 'General'
NTYPE NSEM COMMON count
NSYN common
SPEC DET PRED 'die'
DET-TYPE def
229 CASE gen, GEND masc, NUM sg, PERS 3
NTYPE NSEM COMMON count
NSYN common
SPEC DET PRED 'die'
DET-TYPE def
106 CASE acc, GEND neut, NUM pl, PERS 3
SUBJ
PRED 'Verheugen'
NTYPE NSEM PROPER PROPER-TYPE name
NSYN proper
1 CASE nom, NUM sg, PERS 3
TNS-ASP MOOD subjunctive, PERF +_, TENSE pres
21 CLAUSE-TYPE decl, PASSIVE -, STMT-TYPE decl, VTYPE main

Figure 2: F-structure for (1)

2.2 A broad-coverage LFG for German

For the construction of our data, we use the German broad-coverage LFG documented in Dipper (2003) and Rohrer and Forst (2006). It is a hand-crafted grammar developed in and for the LFG grammar development and processing platform XLE (Crouch et al., 2006). It achieves parsing coverage of about 80% in terms of full parses on newspaper text, and for sentences out of coverage, the robustness techniques described in Riezler et al. (2002) (fragment grammar, ‘skimming’) are employed for the construction of partial analyses. The grammar is reversible, which means that the XLE generator can produce surface realisations for well-formed input f-structures.

Recently, the grammar has been complemented with a stochastic disambiguation module along the lines of Riezler et al. (2002), consisting of a log-linear model based on structural features (Forst, 2007). This module makes it possible to determine one c-/f-structure pair as the most probable analysis of any given sentence.

2.3 Surface realisation

As XLE comes with a fully-fledged generator, the grammar can be used both for parsing and for surface realisation. Figure 3 shows an excerpt of the set of strings (and their LM ranking) that are generated from the f-structure in Figure 2. Note that all of these (as well as the the remaining 139 strings in the set) are grammatical; however, some of them are clearly more likely or unmarked than others.

```
1. Falsch  interpretiert habe die Worte
    Wrongly interpreted   had  the words
   des     Generalinspekteurs Verheugen.
   the-GEN inspector-general  Verheugen.

2. Falsch interpretiert habe die Worte
   des Generalinspekteures Verheugen.

3. Die Worte des Generalinspekteurs
   falsch interpretiert habe Verheugen.

5. Die Worte des Generalinspekteurs habe
   Verheugen falsch interpretiert.

7. Verheugen habe die Worte des General-
   inspekteurs falsch interpretiert.
```

Figure 3: Excerpt of the set of 144 strings generated from the f-structure in Figure 2, ordered according to their LM score

Just as hand-crafted grammars, when used for parsing, are only useful for most applications when they have been complemented with a disambiguation module, their usefulness as a means of surface realisation depends on a reliable module for realisation ranking. A long list of arbitrarily ordered output strings is useless for practical applications such as summarisation, QA, MT etc.

Very regular preferences for certain realisation alternatives over others can be implemented by means of so-called optimality marks (Frank et al., 2001), which are implemented in XLE both for the parsing and the generation direction. For ranking string realisations on the basis of 'soft' and potentially contradictory constraints, however, the stochastic approach based on a log-linear model, as it has previously been implemented for English HPSGs (Nakanishi et al., 2005; Velldal and Oepen, 2005), seems more adequate.

3 Experimental setup

3.1 Data

We use the TIGER Treebank (Brants et al., 2002) to train and test our model. It consists of just over 50,000 annotated sentences of German newspaper text. The sentences have been annotated with morphological and syntactic information in the form of functionally labelled graphs that may contain crossing and secondary edges.

We split the data into training and test data using the same data split as in Forst (2007), i.e. sentences 8,001–10,000 of the TIGER Treebank are reserved for evaluation. Within this section, we have 422 TIGER annotation-compatible f-structures, which are further divided into 86 development and 336 test structures. We use the development set to tune the parameters of the log-linear model. Of the 86 heldout sentences and the 336 test sentences, 78 and 323 respectively are of length >3 and are actually used for our final evaluation.

For training, we build a symmetric treebank of 8,609 packed c/f-structure representations in a similar manner to Velldal et al. (2004). We do not include structures for which only one string is generated, since the log-linear model for realisation ranking cannot learn anything from them. The symmetric treebank was established using the following strategy:

1. Parse input sentence from TIGER Treebank.
2. Select all of the analyses that are compatible with the TIGER Treebank annotation.
3. Of all the TIGER-compatible analyses, choose the most likely c/f-structure pair according to log-linear model for parse disambiguation.
4. Generate from the f-structure part of this analysis.

String Realisations	#str.	Avg # words
> 100	1206	18.3
$\geq 50, < 100$	709	14.3
$\geq 10, < 50$	3029	11.8
$> 1, < 10$	3665	7.6
Total	8609	11.3

Table 1: Number of structures and average sentence length according to ambiguity classes in the training set

5. If the input string is contained in the set of output strings, add this sentence and all of its corresponding c/f-structure pairs to training set. The pair(s) that correspond(s) to the original corpus sentence is/are marked as the intended structure(s), while all others are marked as unintended.

Theoretically all strings that can be parsed should be generated by the system, but for reasons of efficiency, punctuation is often not generated in all possible positions, therefore resulting in an input string not being contained in the set of output strings. Whenever this is the case for a given sentence, the c/f-structure pairs associated with it cannot be used for training. Evaluation can be carried out regardless of this problem, but it has to be kept in mind that the original corpus string cannot be generated for all input f-structures. In our test set, it is generated only for 62% of them.

Tables 1 and 2 give information about the ambiguity of the training and test data. For example, in the training data there are 1,206 structures with more than 100 string realisations. Most of the training and test structures have between 2 and 50 possible (and grammatical) string realisations. The average sentence length of the training data is 11.3 and it is 12.8 for the test data.[1] The tables also show that the structures with more potential string realisations correspond to longer sentences than the structures that are less ambiguous when generating.

[1]This is lower than the overall average sentence length of roughly 16 in TIGER because of the restriction that the structure produced by the reversible grammar for any TIGER sentence be compatible with the original TIGER graph. As the grammar develops further, we hope that longer sentences can be included in both training and test data.

String Realisations	#str.	Avg # words
> 100	61	23.7
$\geq 50, < 100$	26	13.5
$\geq$10, < 50	120	11.6
$> 1, < 10$	129	7.8
Total	336	12.8

Table 2: Number of structures and average sentence length according to ambiguity classes in the test set

3.2 Features for realisation ranking

Using the feature templates presented in Riezler et al. (2002), Riezler and Vasserman (2004) and Forst (2007), we construct a list of 186,731 features that can be used for training our log-linear model. Out of these, only 1,471 actually occur in our training data. In the feature selection process of our training regime (see Subsection 3.3), 360 features are chosen as the most discriminating; these are used to rank alternative solutions when the model is applied. The features include c-structure features, features that take both c- and f-structure information into account, sentence length and language model scores. Examples of c-structure features are the number of times a particular category label occurs in a given c-structure, the number of children the nodes of a particular category have or the number of times one particular category label dominates another. Examples of features that take both c- and f-structure information into account are the relative order of functions (e.g. 'Subj precedes Obj'). As in Velldal and Oepen (2005), we incorporate the language model score associated with the string realisation for a particular structure as a feature in our model.

3.3 Training

We train a log-linear model that maximises the conditional probability of the observed corpus sentence given the corresponding f-structure. The model is trained in a (semi-)supervised fashion on the 8,609 (partially) labelled structures of our training set using the `cometc` software provided with the XLE platform. `cometc` performs maximum likelihood estimation on standardised feature values and offers several regularisation and/or feature selection techniques. We apply the combined method of incremental feature selection and l_1 regularisation

Exact Match Upper Bound	62%
Exact Matches	27%
BLEU score	0.7306

Table 3: Results with the language model

Exact Match Upper Bound	62%
Exact Matches	37%
BLEU score	0.7939

Table 4: Results with the log-linear model

presented in Riezler and Vasserman (2004), the corresponding parameters being adjusted on our heldout set.

For technical reasons, the training was carried out on unpacked structures. However, we hope to be able to train and test on packed structures in the future which will greatly increase efficiency.

4 Evaluation

4.1 Evaluating the system's output

We evaluate the most likely string produced by our system in terms of two metrics: **exact match** and **BLEU score** (Papineni et al., 2002). Exact match measures what percentage of the most probable strings are exactly identical to the string from which the input structure was produced. BLEU score is a more relaxed metric which measures the similarity between the selected string realisation and the observed corpus string.

We first rank the generator output with a language model trained on the Huge German Corpus (a collection of 200 million words of newspaper and other text) using the SRILM toolkit. The results are given in Table 3, achieving exact match of 27% and BLEU score of 0.7306. In comparison to the results reported by Velldal and Oepen (2005) for a similar experiment on English, these results are markedly lower, presumably because of the relatively free word order of German.

We then rank the output of the generator with our log-linear model as described above and give the results in Table 4. There is a noticeable improvement in quality. Exact match increases from 27% to 37%, which corresponds to an error reduction of 29%,[2] and BLEU score increases from 0.7306 to 0.7939.

There is very little comparable work on realization ranking for German. Gamon et al. (2002) present work on learning the contexts for a particular subset of linguistic operations; however, no evaluation of the overall system is given.

[2]Remember that the original corpus string is generated only from 62% of the f-structures of our test set, which fixes the upper bound for exact match at 62% rather than 100%.

4.2 Evaluating the ranker

In addition to the exact match and BLEU score metrics, which give us an indication of the quality of the strings being chosen by the statistical ranking system, the quality of the ranking itself is interesting for internal evaluation during development. While exact match tells us how often the correct string is selected, in all other cases, it gives the developers no indication of whether the correct string is close to being selected or not. We propose to evaluate the ranker by calculating the following **ranking score**:

$$s = \begin{cases} \frac{n-r+1}{n} & \text{if } r \text{ is defined,} \\ 0 & \text{if } r \text{ is not defined,} \end{cases}$$

where n is the total number of potential string realisations and r is the rank of the gold standard string. If the gold standard string is not among the list of potentials, i.e. r is undefined, the ranking score is defined as 0.

The ranking scores for the language model and the hybrid log-linear model are given in Table 5. Since the original string is not necessarily in the set of candidate strings, we also provide the upper bound (i.e. if the ranker had chosen the correct string any time the correct string was available). The table shows that in almost 62% of the cases, the original string was generated by the system. The hybrid model achieves a ranking score of 0.5437 and the language model alone achieves 0.4724. The structural features in the hybrid model result in an error reduction of 49% over the baseline language model ranking score. The hybrid model is thus noticeably better at ranking the original string (when available) higher in the list of candidate strings. This error reduction is considerably higher than the error reduction of the much stricter exact match score.

Language Model	0.4724
Hybrid Model	0.5437
Upper Bound	0.6190

Table 5: Evaluating the ranking

5 Error Analysis

We had initially expected the increase in BLEU score to be greater than 0.0633, since German is far less configurational than English and therefore we thought the syntactic features used in the log-linear model would play an even greater role in realisation ranking. However, in our experiments, the improvement was only slighlty greater than the improvement achieved by Velldal and Oepen (2005). In this section, we present some of the more common errors that our system still produces.

Word Choice Often there is more than one surface realisation for a particular group of morphemes. Sometimes the system chooses an incorrect form for the sentence context, and sometimes it chooses a valid, though marked or dispreferred, form. For example, from the structure in Figure 2, the system chooses the following string as the most probable.

Verheugen	habe	die	**Wörter**	des
Verheugen	had	the	**words**	of the
Generalinspekteures	falsch	interpretiert.		
inspector-general	wrongly	interpreted.		

There are two mis-matches in this output string with respect to the original corpus string. In the first case the system has chosen *Wörter* as the surface realisation for the morpheme sequence *Wort+NN.Neut.NGA.Pl* rather than the, in this case, correct form *Worte*. In the second (less critical) case, the system has chosen to mark the genitive case of *Generalinspekteur* with *es* rather than the *s* that is in the original corpus. This is a relatively frequent alternation that is difficult to predict, and there are other similar alternations in the dative case, for example. In the development set, this type of error occurs in 6 of the 78 sentences. In order to correct these errors, the morphological component of the system needs to be improved.

Placement of adjuncts Currently, there is no feature that captures the (relative) location of particular types of adjuncts. In German, there is a strong tendency for temporal adjuncts to appear early in the sentence, for example. Since the system was not provided with data from which it could learn this generalisation, it generated output like the following:

Frauenärzte	haben	die	Einschränkung
Gynaecologists	have	the	restriction
umstrittener	Antibabypillen	wegen	
controversial	birth control pills	because of	
erhöhter	Thrombosegefahr	**am Dienstag**	
increased	risk of thrombosis	**on Tuesday**	
kritisiert.			
critisised.			

'Gynaecologists criticised the restriction on controversial birth control pills due to increased risk of thrombosis on Tuesday.'

where the temporal adjunct *on Tuesday* was generated very late in the sentence, resulting in an awkward utterance. The most obvious solution is to add more features to the model to capture generalisations about adjunct positions.

Discourse Information In many cases, the particular subtleties of an utterance can only be generated using knowledge of the context in which it occurs. For example, the following sentence appears in our development corpus:

Israel	stellt	den	Friedensprozess	nach	Rabins
Israel	puts	the	peace process	after	Rabin's
Tod	nicht	in Frage			
death	not	in question			

'Israel does not challenge the peace process after Rabin's death'

Our system generates the string:

Nach	Rabins	Tod	stellt	Israel	den
After	Rabin's	death	puts	Israel	the
Friedensprozess	nicht	in Frage.			
peace process	not	in question.			

which, taken on its own, gets a BLEU score of 0. The sentence produced by our system is a perfectly valid sentence and captures essentially the same information as the original corpus sentence. However, without knowing anything about the information structure within which this sentence is uttered, we have no way of telling where the emphasis of the sentence is. The work described in this paper is part of a much larger project, and future research is already planned to integrate information structure into the surface realisation process.

6 Discussion and future work

In the work of Callaway (2003; 2004), a purely symbolic system is presented. Our system makes use of a symbolic grammar, but we apply a statistical ranking model to the output. We believe that this gives us the power to restrict the output of the generator without under-generating. The symbolic grammar enforces hard constraints and the statistical ranking models (possibly conflicting) soft constraints to weight the alternatives.

Satisfactory methods for the automatic evaluation of surface realisation (as well as machine translation) have not yet been developed (Belz and Reiter, 2006). The shortcomings of current methods, particularly BLEU score, seem to be even more pronounced for German and other relatively free word-order languages. Given that all of the sentences generated by our system are syntactically valid, one would expect much higher results. A human inspection of the results and investigation of other evaluation metrics such as NIST will be carried out to get a clearer idea of the quality of the ranking.

There is a potential bias in the learning algorithm, in that the average length of the training data sentences is lower than the average length of the test data. In particular, the length of the sentences with more than 100 potential string realisations seems to be considerably longer (cf. Tables 1 and 2). Therefore, it is possible that the system has not learnt enough features from longer sentences to be able to correctly rank the intended string. We plan to increase the size of the training set to also include longer sentences that would hopefully avoid this bias. Moreover, more data will allow us to investigate in detail the effect of more or less training data on the results of a stastical realisation ranking system trained on them.

Finally, more feature design is clearly necessary for the improvement of the system. We already have several features in mind, which have not been implemented for technical reasons. Examples are the distance between a relative clause and its antecedent as well as its weight, which will hopefully allow us to learn which types of relative clauses tend to appear in extraposed position. Another thing that features for realization ranking should capture is the information-structural status of constituents. Information structure is an important factor when generating the correct surface realisation (Kruijff et al., 2001). German is a language in which word order is largely driven by information structure rather than grammatical function, which is often marked morphologically. In future work, we plan to integrate information structure features into the log-linear model and hope that results will improve.

7 Conclusions

In this paper, we have presented a log-linear realisation ranking system for a German LFG system. The reversibility of the grammar ensures that all output strings will be grammatical. The task then becomes to choose the most likely one. We train on over 8,600 partially labelled structures and test on 323 sentences of length >3. To our knowledge, this is the largest statistical realisation experiment carried out for German. The number of structures used is also much greater than the data used in Velldal and Oepen (2005), although the improvement over a baseline language model was only slightly better. We achieved an increase in exact match score from 27% to 37% and an increase in BLEU score from 0.7306 to 0.7939. The fact that our scores are lower than that of Velldal and Oepen (2005) suggests that it may be more difficult to achieve high scores for German data, although this is not necessarily a reflection of the quality of the strings chosen. We also show, using a ranking score, that the log-linear ranking system generally ranks the correct solution considerably higher than our baseline system.

Acknowledgements

The work described in this paper has been carried out as part of the COINS project of the linguistic Collaborative Research Centre (SFB 732) at the University of Stuttgart, which is funded by the German Research Foundation (DFG). Furthermore, we would like to thank John Maxwell of the Palo Alto Research Center for being so responsive to our requests for extensions of the XLE generator functionalities, some of which were crucial for our work.

References

Srinivas Bangalore and Owen Rambow. 2000. Exploiting a probabilistic hierarchical model for generation. In *Proceedings of COLING 2000*, pages 42–48, Saarbrücken, Germany.

Anja Belz and Ehud Reiter. 2006. Comparing automatic and human evaluation of NLG systems. In *Proceedings of EACL 2006*, pages 313–320, Trento, Italy.

Anja Belz. 2005. Statistical generation: Three methods compared and evaluated. In *Proceedings of ENLG 2005*, pages 15–23, Aberdeen, Scotland.

Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2002. The TIGER Treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, Sozopol, Bulgaria.

Aoife Cahill and Josef van Genabith. 2006. Robust PCFG-Based Generation using Automatically Acquired LFG Approximations. In *Proceedings of COLING/ACL 2006*, pages 1033–1040, Sydney, Australia.

Charles B. Callaway. 2003. Evaluating coverage for large symbolic NLG grammars. In *Proceedings of IJCAI 2003*, pages 811–817, Acapulco, Mexico.

Charles B. Callaway. 2004. Wide coverage symbolic surface realization. In *Proceedings of ACL 2004*, pages 125–128, Barcelona, Spain.

Dick Crouch, Mary Dalrymple, Ron Kaplan, Tracy King, John Maxwell, and Paula Newman. 2006. XLE documentation. Technical report, Palo Alto Research Center, CA.

Stefanie Dipper. 2003. *Implementing and Documenting Large-scale Grammars – German LFG*. Ph.D. thesis, IMS, University of Stuttgart.

Martin Forst. 2007. *Disambiguation for a Linguistically Precise German Parser*. Ph.D. thesis, University of Stuttgart.

Anette Frank, Tracy Holloway King, Jonas Kuhn, and John T. Maxwell. 2001. Optimality Theory Style Constraint Ranking in Large-Scale LFG Grammars. In Peter Sells, editor, *Formal and Empirical Issues in Optimality Theoretic Syntax*, pages 367–397. CSLI Publications, Stanford, CA.

Michael Gamon, Eric Ringger, Simon Corston-Oliver, and Robert Moore. 2002. Machine-learned contexts for linguistic operations in German sentence realization. In *Proceedings of ACL 2002*, pages 25–32, Philadelphia, PA.

Geert-Jan M. Kruijff, Ivana Kruijff-Korbayova, John Bateman, and Elke Teich. 2001. Linear Order as Higher-Level Decision: Information Structure in Strategic and Tactical Generation. In *Proceedings of ENLG 2001*, pages 74–83, Toulouse, France.

Irene Langkilde-Geary. 2002. An Empirical Verification of Coverage and Correctness for a General-Purpose Sentence Generator. In *Proceedings of INLG 2002*, NY.

Irene Langkilde. 2000. Forest-based statistical sentence generation. In *Proceedings of NAACL 2000*, pages 170–177, Seattle, WA.

Hiroko Nakanishi, Yusuke Miyao, and Jun'ichi Tsujii. 2005. Probabilistic models for disambiguation of an HPSG-based chart generator. In *Proceedings of IWPT 2005*.

Tomoko Ohkuma. 2004. Japanese generation grammar. Presentation at the ParGram fall meeting, Dublin, Ireland.

Kishore Papineni, Salim Roukos, Todd Ward, and WeiJing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of ACL 2002*, pages 311–318, Philadelphia, PA.

Stefan Riezler and Alexander Vasserman. 2004. Gradient feature testing and l_1 regularization for maximum entropy parsing. In *Proceedings of EMNLP'04*, Barcelona, Spain.

Stefan Riezler, Tracy Holloway King, Ronald M. Kaplan, Richard Crouch, John T. Maxwell, and Mark Johnson. 2002. Parsing the Wall Street Journal using a Lexical-Functional Grammar and Discriminative Estimation Techniques. In *Proceedings of ACL 2002*, Philadelphia, PA.

Christian Rohrer and Martin Forst. 2006. Improving coverage and parsing quality of a large-scale LFG for German. In *Proceedings of LREC-2006*, Genoa, Italy.

Erik Velldal and Stephan Oepen. 2005. Maximum entropy models for realization ranking. In *Proceedings of the 10th MT Summit*, pages 109–116, Thailand.

Erik Velldal, Stephan Oepen, and Dan Flickinger. 2004. Paraphrasing treebanks for stochastic realization ranking. In *Proceedings of TLT Workshop*, pages 149–160, Tübingen, Germany.