

Building Surface Realizers Automatically from Corpora *

Huayan Zhong and Amanda J. Stent

Department of Computer Science

Stony Brook University

Stony Brook, NY 11794-4400 USA

{huayan, stent}@cs.sunysb.edu

Abstract

In this paper, we evaluate the feasibility of automatically acquiring surface realizers from corpora using general-purpose parsing tools and lexicons. We present a basic architecture for acquiring a generation grammar, describe a surface realizer that uses grammars developed in this way, and present a set of experiments on different corpora that highlight possible improvements in our approach.

1 Introduction

The purpose of surface realization (SR) is to generate sentences or phrases from input semantic or syntactic representations that: a) preserve the input meaning; and b) are grammatical. To this end, it may select content words, insert function words, arrange words in order, and perform morphological inflection.

We are interested in modeling aspects of lexical and syntactic variation and lexical/syntactic adaptation directly in the surface realizer (cf. [Creswell, 2003; Purver and Kempson, 2004]). It will be easier to do this if our surface realizer is directly acquired from data. In this paper, we explore the possibilities of automatically acquiring a domain-specific surface realizer from unlabeled data using general-purpose parsing tools and lexicons. We use our surface realizer in spoken and multimodal dialog systems with a variety of architectures, so we chose as our input format logical forms in which content words can, but need not be, specified. For this type of application, we are also concerned about speed of generation, so we looked at the possibility of eliminating a second-stage ranker.

In Section 2 we discuss previous approaches to surface realization. We describe our data in Section 3. In Section 4 we describe our approach to acquiring a generation grammar and our surface realizer. In Sections 6 and 5 we present the results of evaluations of several automatically-acquired generation grammars, and highlight areas for improvement.

*We would like to thank the blind reviewers for their helpful comments, and Gregory Aist and colleagues for collecting and transcribing part of the CALO corpus. This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA), through the Department of the Interior, NBC, Acquisition Services Division, under Contract No. NBCHD030010.

2 Related Work

Most existing surface realizers use hand-written grammars or templates. FUF/SURGE [Elhadad, 1993; Elhadad and Robin, 1997] uses a complex unification grammar to generate sentences. The sentences it generates are of high quality. However, it can take quite a long time to generate a sentence [Callaway, 2003]. Genesis-II [Baptist and Seneff, 2000] and YAG [Channarukul, 1999] use hand-written templates in the air travel domain. Template-based surface realizers work faster than grammar-based surface realizers. However, they are domain-specific so must be changed for each new domain.

Two-stage surface realizers, e.g. [Langkilde, 2002; Oh and Rudnicky, 2002; Ratnaparkhi, 2000] use a hand-written grammar to generate potential outputs, and then a language model to rank these outputs. These surface realizers can produce output of quite high quality without the coding effort required to write a complex grammar by hand [Callaway, 2003].

Most statistical surface realizers are trained on annotated data (e.g. [Rambow and Bangalore, 2000; Marciniak and Strube, 2004; Pan and Shaw, 2004; S. Corston-Oliver and Moore, 2002]). FERGUS [Rambow and Bangalore, 2000] is a two-stage statistical surface realizer. It takes a dependency tree with content words specified as input, and realizes it by selection and combination of TAG derivation trees automatically acquired from an annotated corpus. It then generates a word lattice for the trees, and uses a language model to walk a highly probable path through the lattice, producing an output sentence. Marciniak and Strube report on a statistical surface realizer that generates from logical forms to output sentences [Marciniak and Strube, 2004]. This surface realizer consists of a set of eight classifiers trained on a set of 70 manually annotated texts.

Amalgam is a surface realizer trained on automatically parsed data and takes a logical form as input, so is perhaps closest to ours in style [S. Corston-Oliver and Moore, 2002]. It uses a classification-based approach. We are interested, however, in a direct representation of the mappings between semantics, syntax and realization so that we can manipulate these mappings on-the-fly to produce variation, so we use a probabilistic grammar instead. Our goal is to look at whether it is possible to create a framework in which domain-specific surface realizers can be acquired automatically from unannotated data using general-purpose tools and resources and then adapted during interaction.

Corpus	total # sents (unique sents)	total # words (unique words)
EXPL	5435 (5403)	86584 (9432)
CALO	4255 (2977)	65886 (4317)
WSJ	43411 (43162)	1044265 (40876)

Table 1: Size of corpora

3 Data

We used three corpora for these experiments: WSJ, EXPL and CALO (Table 1). The first, the Penn Wall Street Journal Treebank [Marcus *et al.*, 1993], we selected to allow us to compare performance with systems like FERGUS [Rambow and Bangalore, 2000] and HALogen [Langkilde, 2002], as well as to identify performance gains if there is "perfect" parsing. The second, EXPL, is a corpus of text explanations for children [howstuffworks, 2003], that we selected because of the relative simplicity of its sentence structures. Finally, we selected the CALO corpus, a collection of corpora of spoken dialog in purchasing domains, in order to allow us to look at performance gains if there are domain-specific lexicons and ontologies. Because of the small size of this corpus, we also report results for a grammar acquired from a combination of this corpus with the Penn Treebank. We randomly assigned 90% of each corpus as training data, and 10% as testing data.

4 Grammar Acquisition

Our grammar acquisition pipeline is shown in Figure 1. We use the following general-purpose tools and resources: the LT TTT tokenizer [Grover *et al.*, 2000], the Brill part-of-speech tagger [Brill, 1992], the Collins parser [Collins, 1999], WordNet [Fellbaum, 1998], and VerbNet [Kipper *et al.*, 2000]. Domain-specific replacements for any of these tools or resources can be made at any time. For example, in some of our experiments, we added the ontology and lexicon created for the TRIPS system [Dzikovska *et al.*, 2003].

Input text is automatically regularized, split into sentences and tokenized. Sentences containing quotations are removed. Then, the text is part-of-speech tagged and parsed, giving a set of parse trees, some with errors. We retain only the most likely parse for each sentence. Sentences that cannot be parsed are dropped from further processing.

A set of transformations are applied to the parse trees to convert all sentences to active, declarative form with one main verb only (clauses joined by a coordinating or subordinating conjunction are split).

We then try to identify the semantic roles in the sentence. Considerable research has been done on semantic role labeling (e.g. [Gildea and Jurasky, 2002; Pradhan *et al.*, 2004]); for this initial work, we simply look up the main verb of the sentence in VerbNet [Kipper *et al.*, 2000] and the hypernyms of the verb's arguments in WordNet [Fellbaum, 1998]. If the restrictions of the roles in a verb frame for the verb match the hypernyms of the verb's arguments in the sentence, we add semantic features to the parse tree, annotating the main verb with the verb frame and the verb's arguments with the semantic roles and hypernyms. A sentence can lead to multiple annotated trees if multiple verb frames match. Figure

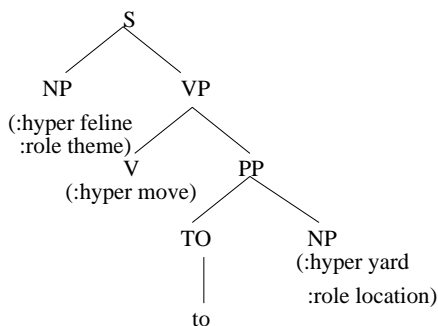


Figure 2: An initial tree for "the cat went to the garden"

2 shows a sample sentence tree extracted from the sentence "The cat went to the garden".

Finally, we extract initial and auxiliary trees from the annotated parse trees. Initial tree types include S, NP, VP and PP; auxiliary tree types include NP, VP, PP, ADJP, ADVP and QP. We cluster these trees by their semantics and assign relative frequencies to each tree. Table 3 shows the number of initial trees we collected for each corpus. The grammars we obtain are probabilistic lexicalized tree-adjoining grammars with semantic features [Abeille and Rambow, 2000].

Table 2 shows the number of failures in our grammar acquisition process after parsing, after verb lookup and after verb frame matching. Note that the fact that the parser found a parse for 100% of the sentences in our corpora does not mean that it found a correct parse. We examined the 330 sentences from our CALO corpus for which there were no verb frames, and labeled them by hand as primarily spelling/tokenization (25), fragmentary (150), conventional (33), containing idiomatic language (16), containing highly unsyntactic language use (70), containing named entities that were problematic (15), or "other" (39). (The total sums to more than 330 because some sentences exhibited multiple phenomena.) We concluded that for this dialog corpus, some sentences just should not be included in the generation grammar. A similar examination of our EXPL corpus (469 'failure' sentences) gave 4 spelling/tokenization, 194 fragmentary, 6 idiomatic, 36 highly unsyntactic, 13 problematic named entities, and 230 other irregularities (mostly unusual verbs or unusual subject/verb combinations like 'react', 'typify' and 'integrate'). In this corpus, most of the fragments are titles or advertising.

The steep drop in data retention that results from an inability to find a matching verb frame is sometimes due to the lack of a suitable verb frame in our knowledge resources, sometimes due to the fact that the verb in question is a stative and sometimes due to parsing errors leading to misidentification of the verb. However, the majority of these errors are due to mismatches between the WordNet hypernyms of words filling sentential roles, and the semantic role type information in VerbNet. We were initially surprised to see such a steep rise in performance for our non-CALO corpora when the TRIPS lexicon and ontology are added, but with this lexicon and ontology these mismatches are not present. This type of mismatch is one of the largest problems with the use of general-purpose knowledge resources created by different researchers

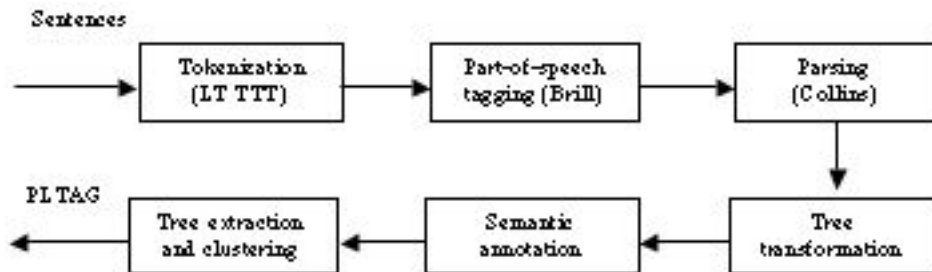


Figure 1: Architecture of grammar acquisition process⁷

Corpus	% parsed	% has verb frames	% into grammar
Specialized			
EXPL	100.00%	88.88%	53.02%
CALO	100.00%	89.80%	62.61%
WSJ	100.00%	64.35%	42.32%
NoParse	100.00%	71.87%	50.91%
General			
EXPL	100.00%	57.22%	41.95%
CALO	100.00%	57.24%	43.87%
WSJ	100.00%	40.06%	30.99%
NoParse	100.00%	48.94%	37.32%

Table 2: Grammar acquisition: Sources of data loss

for different purposes. In the future, we will use statistical semantic role labeling as described by [Gildea and Jurasky, 2002] to overcome the problem of type mismatches, and the smaller problem of missing verbs or missing verb frames.

5 Surface Realizer

We implemented a surface realizer for these grammars that uses a chart to efficiently compute all possible realizations of an input semantic form. The input is a logical form. Content words may be specified in the input; any lexical selection that needs to be done during surface realization is done using WordNet. We find all the initial or auxiliary trees covering each part of the input semantics and place them in the chart. We then fill in the chart with possible combinations of these trees using adjunction and substitution. Finally, we read off those trees that cover all or most of the input logical form and are rooted at S.

We use the same transformation rules used during grammar extraction to transform the resulting syntax tree if the input semantics specify a non-declarative sentence type. These transformation rules, and (if used for dialog) a small number of templates covering conventional utterances (greetings, thanks, acknowledgments) are the only hand-written rules used in any of our surface realizers [Stent, 1999]; for the experiments reported below, we excluded templates covering conventional forms from the surface realizer.

Our surface realizer has a second stage that assigns morphological inflection. We can rank sentences directly using the relative frequencies assigned to trees in the acquired

grammars. Using an independence assumption, we simply multiply the probabilities of all the trees that go into producing the sentence. However, this method of computing the probability of a sentence ignores word likelihoods. If the surface realizer is doing lexical selection using a lexicon that is not domain-specific, the use of a third-stage language model can also be used to rerank sentences.

6 Experiments

We performed three sets of experiments. Our first experiment (General) was designed to look at performance when using only general-purpose tools and resources. We used only VerbNet and WordNet as our knowledge resources and we ignored the parse trees in the Penn Treebank, instead re-parsing those sentences. Our second experiment (NoParse) was designed to look at performance gains when using a treebank with general-purpose resources. We used the Penn Treebank only for this experiment. In our third experiment (Specialized), we added to our knowledge resources a specialized lexicon and ontology from the TRIPS dialog system [Dzikovska *et al.*, 2003]. For each set of experiments we report results when running our surface realizer both with and without a second-stage language model. For these experiments, we used a trigram language model trained on the Brown corpus to avoid artificially boosting performance on the Wall Street Journal with a Wall Street Journal-trained language model.

We used the same test input for each experiment. We constructed test input semi-automatically from the test data from each corpus. We used the same procedure used to acquire

Corpus	# unique initial trees	# np	# vp	#pp	# adjp	# advp	# qp
CALO	3206	133	57	1643	28	18	12
WSJ	25024	175	199	17705	95	63	47
EXPL	6345	101	89	4224	35	21	24

Table 3: Number of initial trees and auxiliary trees extracted from each corpus

Corpus	General	Specialized
EXPL	80.60%	89.69%
CALO	78.61%	88.72%
WSJ	77.39%	86.67%
NoParse	75.41%	86.97%
CALO + WSJ	78.61%	88.89%

Table 4: Coverage of generation grammars (number of sentences generated over number of inputs to generators)

our grammar (tokenize, part-of-speech tag, parse, annotate with semantic information), and then stripped off the logical form from the resulting tree. We retained content words (stemmed), but permitted the surface realizer to perform lexical selection of function words.

For each experiment, we evaluated using automatic and human evaluation. For comparison with previous work ([Callaway, 2003]), we report simple string accuracy (SSA) [Bangalore *et al.*, 2000]. We also report performance using Melamed’s F measure [Turian *et al.*, 2003], which we have found to be somewhat more highly correlated with human judgments than the BLEU score [Stent *et al.*, 2005]. However, neither of these metrics works very well in the absence of multiple reference sentences. Therefore, we also conducted a human evaluation, for which we used the approach described in [Stent *et al.*, 2005].

6.1 Performance and Coverage

Our system was competitive in terms of performance. The average time taken to generate a single sentence for the one stage generation system is 48.41 milliseconds, while the average time for the two stage generation system is 58.72 milliseconds (cf. [Callaway, 2003]). Increasing the size of the grammar does slow this down somewhat, but we use very efficient storage of initial and auxiliary trees and can also resort to chart pruning if the chart becomes too large.

Data losses during grammar acquisition are reported earlier in this paper. These same loss rates apply to our testing data. We show the coverage of our acquired generation grammars, excluding these loss rates, in Table 4. These coverage results again reflect the potential gains from using targeted or specialized lexicons and ontologies.

6.2 Automatic evaluation

We compare the generated sentences with their reference sentences using SSA and Melamed’s F-measure. We show these results in Table 5 for the General experiment, and in Table 6 for the Specialized experiment.

This performance is significantly lower than that of HALOGEN run with a similar setup (.81) and SURGE (.89) [Call-

Corpus	1 stage	2 stages
EXPL	.47 / .55	.52 / .59
CALO	.48 / .60	.52 / .63
WSJ	.45 / .55	.50 / .58
NoParse	.45 / .49	.49 / .52
CALO + WSJ	.48 / .60	.52 / .62

Table 5: General: SSA / Melamed’s F for different generation grammars

Corpus	1 stage	2 stages
EXPL	.73 / .76	.75 / .77
CALO	.74 / .80	.76 / .81
WSJ	.71 / .76	.72 / .77
NoParse	.68 / .72	.70 / .73
CALO + WSJ	.74 / .80	.76 / .80

Table 6: Specialized: SSA / Melamed’s F for different generation grammars

away, 2003; Langkilde, 2002]; however, most errors were due to gaps and inconsistencies in our lexicons and vocabulary gaps between testing and training data.

Overall, the generator could not find a matching tree in the grammar for 10.3% of constituents in the testing sentences. We looked more closely at the CALO+WSJ case; without the WSJ, 9.6% of constituents in the CALO data did not have a matching tree in the grammar, while with it, 9.3% did not. We conclude that the use of this extra data does not help; as long as the training data covers most phenomena that might occur, increases in data size will only help refine the relative frequencies on trees in the grammar.

6.3 Human Evaluation

We selected 156 sentences from our testing data at random. We ran them through our one-stage and two-stage generators, with and without the TRIPS lexicon and ontology, obtaining four output sentences for each testing sentence. An evaluator rated the sentences for fluency and adequacy using the method described in [Stent *et al.*, 2005]; the evaluator was blind to the source of the sentence. The evaluator also marked several types of error that could occur in each sentence: missing verb, incorrect modal verb, absent negation, incorrect preposition, morphology error. Finally, the evaluator also rated the reference sentence for fluency only.

Tables 7 and 8 show the results of our human evaluation. The average score for fluency for the reference sentences was 4.6 (18 had a fluency score of 1). For the candidate sentences, the overall average score for accuracy was 2.5 and for fluency was 2.3 (61 had 5 for both scores; 262 had 1 for both scores).

Corpus	1 stage	2 stages
EXPL	1.6 / 1.7	1.8 / 1.8
CALO	3.1 / 3.2	3.1 / 3.4
WSJ	1.3 / 1.4	1.3 / 1.4
NoParse	1.1 / 1.2	1.4 / 1.5
CALO + WSJ	1.5 / 1.6	1.6 / 1.9

Table 7: General: Average fluency and accuracy for different generation grammars

Corpus	1 stage	2 stages
EXPL	3.4 / 4	3.4 / 4
CALO	3 / 3.4	3 / 3.4
WSJ	3.0 / 3.4	3.3 / 3.8
NoParse	2.5 / 2.6	2.9 / 2.9
CALO + WSJ	1.5 / 1.3	1.5 / 1.4

Table 8: Specialized: Average fluency and accuracy for different generation grammars

Where the reference sentence was length 20 or more (28 sentences), the average candidate sentence scores were 2.1 for adequacy and 1.9 for fluency. Where the reference sentence had length 19 or less (128 sentences), the average candidate sentence scores were 2.5 for adequacy and 2.3 for fluency.

There were 87 instances of a missing verb, 43 of an incorrect preposition, 8 of a missing negation, 36 of an incorrect modal (usually 'can' being replaced by 'will'), and 24 morphology errors (some duplicates). Other issues noticed by the evaluator included dropped passive forms, questions written as declarations, missing phrasal modifiers and relative clauses, and a few reference sentences that were near-quotations. However, because the test input was constructed using our imperfect generator acquisition pipeline, it is not clear for some of these errors (missing negation, incorrect modals, missing verbs, missing modifiers) whether the semantic content was included in the input logical form.

We conclude from this evaluation that in addition to improving our generator acquisition methodology, we should work on our transformation rules and on negation.

6.4 Discussion

We were surprised to find no performance improvement from using treebanked data. We think this result can be explained by the fact that these sentences were longer than those in the other corpus. We noticed during human evaluation that in long sentences, there were phrases that were coherent, but these phrases were arranged in incoherent ways.

We were also surprised to find no performance gains from adding WSJ data to our CALO surface realizer. We had hoped that this additional data would take care of phenomena unseen in the much smaller CALO corpus. Looking at our human evaluation, we now think that it served indeed to augment the number of trees, but not to augment the number of helpful trees.

From both the automatic and the human evaluation, we can see that there are substantial performance gains when using a grammar acquired with targeted lexicons and ontologies.

Also, there are performance gains from using a 2-stage surface realizer, although the size of these gains decreases dramatically in the Specialized case. We think that with some more effort, we can acquire domain-specific grammars whose performance is not significantly different without a second stage sentence ranker.

7 Conclusions and Future Work

In this paper, we evaluated the possibility of using general-purpose tools and resources to acquire probabilistic grammars for generation from unannotated data. We found that mismatches between semantic category labels restricts the utility of general-purpose resources like WordNet and VerbNet, but that when a little domain-specific information is added performance improves dramatically.

In future work, we plan to improve our pipeline by the addition of statistical semantic role labeling and improved parsing. We will also correct logical form representation failings highlighted by our human evaluation. We are currently exploring knowledge-free approaches to this same task, including a classifier-based approach and a graph-based approach, that will also let us meet our goals of: high-quality domain-specific surface realization without treebanks; modeling of genre- and domain-specific variation; and on-line adaptation in dialog.

References

- [Abeille and Rambow, 2000] Anne Abeille and Owen Rambow. *Tree Adjoining Grammars: Formalisms, Linguistic Analysis and Processing*. Center for the Study of Language and Information, Stanford, California, 2000.
- [Bangalore *et al.*, 2000] S. Bangalore, O. Rambow, and S. Whittaker. Evaluation metrics for generation. In *Proceedings of INLG 2000*, 2000.
- [Baptist and Seneff, 2000] Lauren Baptist and Stephanie Seneff. Genesis-II: A Versatile System for Language Generation in Conversational System Applications. In *ICSLP-2000*, volume 3, pages 271–274, Beijing, China, 2000.
- [Brill, 1992] E. Brill. A simple rule-based part-of-speech tagger. In *Proceedings of ANLP 1992*, pages 152–155, Trento, IT, 1992.
- [Callaway, 2003] C. Callaway. Evaluating coverage for large symbolic NLG grammars. In *Proceedings of IJCAI 2003*, 2003.
- [Channarukul, 1999] Songsak Channarukul. Yag: A Template-Based Natural Language Generator For Real-Time Systems. Master's thesis, University of Wisconsin at Milwaukee, 1999.
- [Collins, 1999] M. Collins. *Head-Driven Statistical Models for Natural Language Parsing*. PhD thesis, University of Pennsylvania, 1999.
- [Creswell, 2003] C. Creswell. *Syntactic form and discourse function in natural language generation*. PhD thesis, University of Pennsylvania, 2003.

- [Dzikovska *et al.*, 2003] M. Dzikovska, J. Allen, and M. Swift. Integrating linguistic and domain knowledge for spoken dialogue systems in multiple domains. In *Proceedings of the Workshop on Knowledge and Reasoning in Practical Dialog Systems, IJCAI 2003*, 2003.
- [Elhadad and Robin, 1997] M. Elhadad and J. Robin. SURGE: a Comprehensive Plug-in Syntactic Realization Component for Text Generation, July 1997.
- [Elhadad, 1993] M. Elhadad. FUF: the Universal Unifier User Manual v5.2. Technical report, Department of Computer Science, Ben Gurion University of the Negev, 1993.
- [Fellbaum, 1998] Christiane Fellbaum. *WordNet, an Electronic Lexical Database*. MIT Press, 1998.
- [Gildea and Jurasky, 2002] D. Gildea and D. Jurasky. Automatic labeling of semantic roles. *Computational Linguistics*, 2002.
- [Grover *et al.*, 2000] C. Grover, C. Matheson, A. Mikheev, and M. Moens. LT TTT – a flexible tokenization tool. In *Proceedings of LREC 2000*, 2000.
- [howstuffworks, 2003] How Stuff Works, 2003.
- [Kipper *et al.*, 2000] K. Kipper, H. Dang, and M. Palmer. Class-Based Construction of a Verb Lexicon. In *AAAI-2000 Seventeenth National Conference on Artificial Intelligence*, Austin, TX, 2000.
- [Langkilde, 2002] I. Langkilde. *A Foundation for General-Purpose Natural Language Generation: Sentence Realization using Probabilistic Models of Language*. PhD thesis, Information Science Institute, USC school of Engineering, 2002.
- [Marciniak and Strube, 2004] T. Marciniak and M. Strube. Classification-based generation using TAG. In *Proceedings of INLG 2004*, 2004.
- [Marcus *et al.*, 1993] M. Marcus, M. Marcinkiewicz, and B. Santorini. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2), 1993.
- [Oh and Rudnicky, 2002] A. Oh and A. Rudnicky. Stochastic natural language generation for spoken dialog systems. *Computer Speech and Language*, 16, October 2002.
- [Pan and Shaw, 2004] S. Pan and J. Shaw. SEGUE: A hybrid case-based surface natural language generator. In *Proceedings of INLG 2004*, 2004.
- [Pradhan *et al.*, 2004] S. Pradhan, W. Ward, K. Hacioglu, J. Martin, and D. Jurafsky. Shallow semantic parsing support vector machines. In *Proceedings of of the Human Language Technology Conference/North American chapter of the Association for Computational Linguistics annual meeting (HLT/NAACL-2004)*, Boston, USA, May 2-7 2004.
- [Purver and Kempson, 2004] M. Purver and R. Kempson. Incremental context-based generation for dialogue. In *Proceedings of INLG 2004*, 2004.
- [Rambow and Bangalore, 2000] O. Rambow and S. Bangalore. Using TAGs, a Tree Model, and a Language Model for Generation. In *Fifth International Workshop on Tree-Adjoining Grammars (TAG+)*, Paris, France, 2000. TAG+5.
- [Ratnaparkhi, 2000] Adwait Ratnaparkhi. Trainable methods for surface natural language generation. In *the 1st Meeting of the North American Chapter of the Association of Computational Linguistics*, pages 194–201, Seattle, WA, 2000.
- [S. Corston-Oliver and Moore, 2002] E. Ringger S. Corston-Oliver, M. Gamon and R. Moore. An overview of Amalgam: A machine-learned generation module. In *Proceedings of INLG 2002*, 2002.
- [Stent *et al.*, 2005] A. Stent, M. Marge, and M. Singhai. Evaluating evaluation methods for generation in the presence of variation. In *Proceedings of CICLing 2005*, 2005.
- [Stent, 1999] A. Stent. Content planning in continuous-speech spoken dialog systems. In *Proceedings of the KI'99 workshop "May I Speak Freely?"*, 1999.
- [Turian *et al.*, 2003] J. Turian, L. Shen, and I. D. Melamed. Evaluation of machine translation and its evaluation. In *Proceedings of MT Summit IX*, 2003.