Towards Flexible, Small-Domain Surface Generation: Combining Data-Driven and Grammatical Approaches

Andrea Fischer, †Vera Demberg, and Dietrich Klakow
Spoken Language Systems / † MMCI Cluster of Excellence
Saarland University
Saarbrücken, Germany
afischer@lsv.uni-saarland.de / vera@coli.uni-saarland.de
dietrich.klakow@lsv.uni-saarland.de

Abstract

As dialog systems are getting more and more ubiquitous, there is an increasing number of application domains for natural language generation, and generation objectives are getting more diverse (e.g., generating informationally dense vs. less complex utterances, as a function of target user and usage situation). Flexible generation is difficult and labour-intensive with traditional template-based generation systems, while fully data-driven approaches may lead to less grammatical output, particularly if the measures used for generation objectives are correlated with measures of grammaticality. We here explore the combination of a data-driven approach with two very simple automatic grammar induction methods, basing its implementation on OpenCCG.

1 Introduction & Related Work

As language-operated interactive devices become increasingly ubiquitous, there is an increasing need for not only generating utterances that are comprehensible and convey the intended meaning, but language that is adaptive to different users as well as situations (see also (Dethlefs, 2014) for an overview). Adaptation can happen at different levels, concerning content as well as the formulation of generated sentences. We here focus only on sentence formulation with the goal of being able to automatically generate a large variety of different realisations of a given semantic representation. Our study explores the combination of a data-driven approach (Mairesse et al., 2010) with a grammar-based approach using OpenCCG (White et al., 2007).
The use of templates is a common and well-performing approach to natural language generation. Usually, either the generation process consists of selecting appropriate fillers for manually-built patterns, or the semantic specification constrains the allowable surface constructions so strongly that it effectively constitutes a form of template as well. While such approaches do guarantee grammaticality when templates (or grammars, respectively) are well-designed, the amount of formulation variation that can be generated based on templates is either very low, or requires a huge manual effort in template creation.
One relevant objective in adapting to a user and a situation is utterance complexity. (Demberg et al., 2011) show that a dialog system that generates more concise (but also more complex) utterances is preferred in a setting where the user can fully concentrate on the interaction, while a system that generates less complex utterances is preferred in a dual tasking setting while the user has to steer a car (in a simulator) at the same time. But how do we know which utterance is a "complex" one? We can draw on psycholinguistic models of human sentence processing difficulty, such as dependency locality theory (measuring dependency lengths within the sentence; longer dependencies are more difficult), information density (measuring surprisal – the amount of information conveyed in a certain time unit; a higher rate of information per time unit is more difficult) or words-per-concept (how many words are used to convey a concept).
In this paper, we focus on the measure of *information density*, which uses the information-theoretic measure of surprisal (Hale, 2001; Levy, 2008), as well as the ratio of concepts per words. Our aim is to flexibly generate utterances that differ in information density, producing high-density and low-density formulations for the same underlying semantic representation. We evaluate different parametrisations of our approach by evaluating how many different high vs. low density utterances can be generated. We additionally present judgments from human evaluators rating both grammaticality and meaningfulness.
We collect a small corpus of utterances from the target domain and have them annotated by naive participants with a very shallow notion of semantics, inspired by (Mairesse et al., 2010). We then parse the sentences and automatically create typed templates. During generation, these typed templates are then combined into new unseen sentences, covering also previously unobserved semantic combinations. Generation flexibility in this approach depends entirely on the crowd-sourced domain corpus. Our approach is related to (DeVault et al., 2008), who automatically induce a tree-adjoining-grammar for the Doctor Perez domain.
Our system is realised using OpenCCG. Currently, we disallow cross composition and type raising and thus employ Categorial Grammar as the underlying model.

2 Data

Our data consists of 247 German-language utterances informing about movie screenings. Each utterance may inform about the aspects: movie title, director, actor, genre, screen date and time, cinema, ticket price, and the screened version. They were collected from native speakers of German via crowd-sourcing. For this, we generated random semantic requests and elicited realisations for them from native speakers. The obtained surfaces were then annotated by different persons. Annotation follows (Mairesse et al., 2010)'s semantic stack scheme with a slight modification: instead of allowing multiple instances of one semantic value stack, we explicitly mark alternatives as shown.

Am	4.	und am	5. Juli	wird
date inform	07-04-2015 date inform	alternative date inform	07-05-2015 alternative date inform	inform

Titanic	mit	Leonardo DiCaprio	gezeigt .
titanic title inform	actor inform	Leonardo DiCaprio actor inform	inform

Figure 1: Data example from our domain.

We focus on the 117 unique requests with only positive (”inform”) stacks, disregarding negative (”reject”) data for now. We have a total of 158 sentences realising these 117 entries. We use 75% of our data as training set and 25% as development set. As test set, we construct 200 additional requests for which we do not elicit example sentences. All sets contain roughly equal amounts of each semantic aspect.

3 Our Approach

Based on the annotated sentences, our goal is to automatically populate a lexicon of multi-word units such that these units express a specific attribute from our domain and can be combined with other lexicon entries into a grammatically correct structure. We cannot solely rely on shallow language models for grammaticality (as Mairesse does) as the language model scores may be correlated with output from other objective measures. Specifically, one of our measures of grammatical complexity, surprisal, is often estimated based on n-gram models. Hence, when seeking to optimise for short utterances with high surprisal, we might end up only selecting highly ungrammatical utterances. To avoid issues, we decided to explore whether the data-driven approach can be combined with a grammar-based approach. We automatically parse all training data with a dependency parser (we use the dependency parser from the *mate-tools* toolkit, based on (Bohnet, 2010)), and build a categorial grammar based on these parses. The resulting automatically-learned domain-specific lexicon can then be used for generation with OpenCCG. Our approach can hence be thought of as a very naive way of grammar induction.
The dependency parse gives us information about heads and their dependents, which allows us to construct categorial grammar categories. However, we do not know from the automatic parse which dependents are arguments vs. modifiers. We here explore two simple approaches:

In the **all-arguments (arg)** style, we build a CG type that produces exactly the encountered configuration of immediate dominance and linear precedence. This means that we assume all dependents to be arguments of their governing head. We arbitrarily choose to consume the arguments on the *right* of heads first, followed by those on the left.

In the **all-modifiers (mod)** style, we treat all dependents D as modifiers of their head H. Thus, we construct a CG type modifying H's type from each pair (H, d) where $d \in D$.

For both flavours, we use part-of-speech tags as basic types. For now, we forego any additional constraints. Clearly this means that our grammars overgenerate. Our goal here in this paper is to explore the extent to which we are able to generate a large amount of linguistic variants and the extent to which these are considered “good” by human comprehenders.
The modifier-only approach is less constrained than the argument-only variant, which should lead to more variety and lower grammaticality.

3.1 Request Semantics

In our approach, each word is considered to be either semantically informative or semantically void. It is semantically informative if it is a word or placeholder for a certain information type. For instance, “ACTOR” is the placeholder for an actor's name, and the noun “Originalversion” indicates that a movie is shown in its original version. All other words are considered to be semantically void and called *padding*.
In this setting, a request specifies only the semantic stacks to be conveyed plus the amount of *padding* to use. Note that using more *padding* biases the generation process towards more verbose formulations.
Additionally we assign a special semantic representation (”VERB”) to verb types. This is done to focus the search on full sentences instead of accepting arbitrarily complex noun phrases as complete answers to requests.

3.2 Sub-Tree Merging

As our requests are structure-agnostic, the search space always contains all words potentially usable for a request irrespective of compatibility with each other.
In order to alleviate the arising problem of search space size, we merge words that often co-occur into larger entries in the lexicon. We do this as follows: adjacent heads and dependents are merged if they do not both contain semantic information. As an example, a semantically informative adjective (such as “untertitelte”=“subtitled”) cannot merge with a noun head if the latter contains semantic information it-

self (as “Abenteuerfilm”=“adventure movie” does). However, if the head is semantically void (such as “Film”=“movie”), the two words are combined into one lexicon entry “untertitelte_Film” with the semantic assignment “version=subtitled”. This reduces the search space and speeds up search greatly.

We implement two slightly different versions of this. In the first, verbs are exempt from merging. In the second, verbs may be merged with *padding* words, resulting in longer “VERB” chunks. One may expect this to result in slightly increased grammaticality.

4 Experimental Setup

We build four grammars from data: two argument-only (A1, A2) and two modifier-only grammars (M1, M2). In A1 and M1, verbs are exempt from merging, in A2 and M2, verbs are merged with surrounding padding words as described in 3.2.

4.1 Manually-Constructed Grammar

Additionally, we construct a small grammar G manually. In G, we also do not make use of type raising nor cross composition, but we employ features to enforce both congruence on linguistic features and thematic fit between e.g. verbs and nouns (e.g. only a price or a movie may be assigned a cost, but not a director). G models the most common structures used in the original data and contains most of the vocabulary used therein.

4.2 Search Timeout

We determine the search timeout to use in each generation request from the development set. Figure 2 shows achieved development set coverages in dependency of OpenCCG search timeouts. In this experiment, we use a *padding* of 5, as this is the maximal *padding* encountered in data. Our search is thus calibrated on the most complex utterance(s) in the data.

After roughly three hours, most of the grammars have achieved saturation satisfactorily well. We set the timeout to three and a half hours for our main experiments.

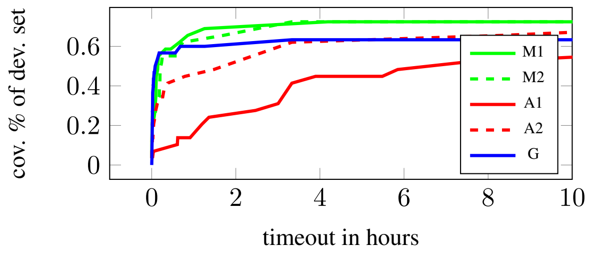

Figure 2: Dev. set coverage vs. search timeout in hours.

4.3 Language Model for Perplexity Evaluation

We train a simple Kneser-Ney smoothed trigram on our training data, which we use in order to pre-select candidates for further evaluation.

4.4 Main Generation Experiment

After training and timeout selection, we automatically generated 200 semantic requests, each consisting of 2 to 8 semantic stacks, and generated realisations for each semantic request by each of our grammars. We do this six times in total, varying the number of padding semantics P between 0 and 5.

We then select one short and one long sentence per semantic request from each grammar’s output. We pick the sentence with the lowest language model perplexity from the 25% longest and 25% shortest sentences, respectively, selecting 1540 sentences.

5 Results

5.1 Test Set Coverage

Table 1 below shows the number of test semantics that each grammar is able to produce results for, grouped by the *padding* they contain (cf. 4.4).

Every other row indicates cumulative coverages, i.e., the number of covered semantics when using up to that many *padding* words, giving an impression of the coverage increments when using more *padding* words.

P	0	1	2	3	4	5
A1	89%	0%	**84%**	71%	51%	18%
$\sum$A1	90%	90%	90%	90%	90%	90%
A2	**89%**	0%	78%	52%	26%	12%
$\sum$A2	89%	89%	89%	89%	89%	89%
M1	44%	13%	73%	67%	52%	14%
$\sum$M1	44%	57%	76%	77%	79%	79%
M2	44%	14%	77%	**75%**	**68%**	**78%**
$\sum$M2	44%	58%	78%	79%	81%	81%
G	11%	**25%**	45%	44%	44%	44%
$\sum$G	11%	36%	45%	45%	45%	45%

Table 1: Test set coverage depending on request *padding* amount P. A1: arg/POS/verbmerge, A2: arg/POS/fullmerge, M1, M2: mod grammars. G: manually-created CCG. Coverages listed with *padding* $= P$ and *padding* $\leq P$ ($\sum$). Best indiv. cov. in bold.

The argument-only grammars achieve highest overall coverage, while the manual grammar achieves the worst coverage. In the arg grammars, using more *padding* deteriorates coverage. This is likely due to search space size increasing. The mod grammars fail to piece together short sentences.

5.2 Grammar Evaluation

In Table 2 we report language model perplexities (PP), parse scores from Stanford Parser, percentages of selected sentences parseable by the German Grammar HPSG, and average human ratings (1=worst, 5=best) of grammaticality and meaningfulness. Annotators agreed exactly in 44%, and differed by no more than 1 in 75.8% of cases. PCFG scores are inconclusive. G performs best except for in perplexity, which we believe is due to G overrepresenting unusual formulations

	misc			grammat.		meaning.	
	PP	PCFG	HP	mean	std	mean	std
A1	**12.69**	-113.57	0.36	3.62	1.24	3.52	1.38
A2	15.00	-128.11	0.45	3.14	1.38	3.11	1.48
M1	19.57	**-111.74**	0.07	1.97	0.94	2.04	**1.03**
M2	25.78	-113.99	0.02	1.80	0.91	2.03	1.13
G	51.79	-124.17	**0.66**	**4.34**	**0.87**	**4.30**	**1.03**

Table 2: Average values rounded to two decimal points. "S": avg. sentence surprisal. "PCFG": mean PCFG parse score. "HP": fraction parseable with HPSG.

as well as the fact that correct use of long-range dependencies leads to local increases in perplexity when the trigram horizon fails to adequately capture the dependency. G has consistently high output quality as evidenced by its small standard deviation of human ratings. The modifier-only grammars consistently perform worst. Both their fraction of HPSG-parseable sentences and human-perceived grammaticality are very low. The argument-only grammars perform fairly well, but do not quite reach up to the manually-written grammar. Their high standard deviation points towards a mix of high-quality and low-quality outputs. Notet that higher HPSG parseability does not necessarily imply higher human ratings. We believe this is due to correct, but confusing or unnatural stacking of attributions.

5.3 Information Density Variation

We plot the distributions of trigram perplexity at sentence level and those of the concepts-per-words ID measure. On both metrics, G is the most variable grammar. We positively note that A1 and A2's CPW range is comparable to that of G. The mod grammars construct more verbose, less informative formulations as evidenced by their lower CPW mean. Perplexity-wise, the arg grammars and mod grammars are very similar. The mod grammars have slightly higher mean perplexities, which – as the CPW plot evidences – does not necessarily indicate a lower ID variability. Rather, we believe this to be a simple reflection of lower local coherence which also diminishes the mod grammars' human ratings. G's extreme perplexity range can be explained by a tendency to overrepresent unlikely formulations. Given the human ratings of the grammars, we interpret the discrepancy between the arg grammars and G to point to a slightly narrower range of correct formulations in A1 and A2.

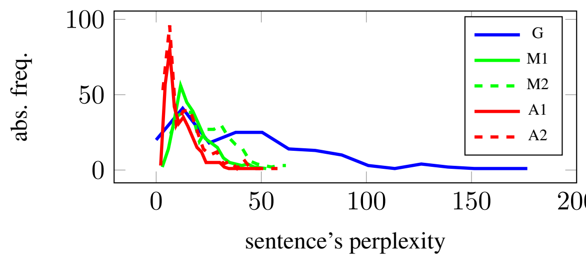

Figure 3: Perplexity distributions for grammars.

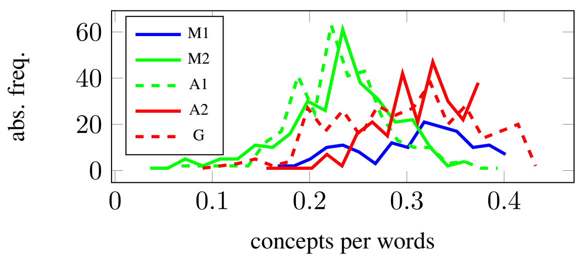

Figure 4: Concepts-per-words distributions for grammars.

6 Conclusion & Future Work

We have presented a simple, effective approach to grammar-based generation using Categorial Grammar as underlying formalism. The argument grammars in particular are able to reproduce the hand-written grammar's range of output variability well while achieving drastically better coverage.
Further work should concentrate on search efficiency, improving the quality of output, and further broadening the coverage of the induced grammars. The first point might be addressed by applying search heuristics which e.g. include the compatibility of elements with each other. We expect coverage, correctness, and variability to greatly benefit from constructing both argument and modifier types within the same grammar.

References

Bernd Bohnet. 2010. Very high accuracy and fast dependency parsing is not a contradiction. In *COLING '10*.

Vera Demberg, Andi Winterboer, and Johanna D. Moore. 2011. A strategy for information presentation in spoken dialog systems.

Nina Dethlefs. 2014. Context-sensitive natural language generation: From knowledge-driven to data-driven techniques.

David DeVault, David Traum, and Ron Artstein. 2008. Making Grammar-Based Generation Easier to Deploy in Dialogue Systems. In *SIGdial 2008*.

John Hale. 2001. A probabilistic earley parser as a psycholinguistic model. In *NAACL HLT 2001*.

Roger Levy. 2008. Expectation-based syntactic comprehension.

François Mairesse, Milica Gašić, Filip Jurčíček, Simon Keizer, Blaise Thomson, Kai Yu, and Steve Young. 2010. Phrase-based statistical language generation using graphical models and active learning. In *ACL 2010*.

Michael White, Rajakrishnan Rajkumar, and Scott Martin. 2007. Towards broad coverage surface realization with ccg. In *UCNLG+MT 2007*.