Toward Bayesian Synchronous Tree Substitution Grammars for Sentence Planning

David M. Howcroft and **Dietrich Klakow** and **Vera Demberg**
Department of Language Science and Technology
Saarland Informatics Campus, Saarland University, Germany
{howcroft,vera}@coli.uni-saarland.de
dietrich.klakow@lsv.uni-saarland.de

Abstract

Developing conventional natural language generation systems requires extensive attention from human experts in order to craft complex sets of sentence planning rules. We propose a Bayesian nonparametric approach to learn sentence planning rules by inducing synchronous tree substitution grammars for pairs of text plans and morphosyntactically-specified dependency trees. Our system is able to learn rules which can be used to generate novel texts after training on small datasets.

1 Introduction

Developing and adapting natural language generation (NLG) systems for new domains requires substantial human effort and attention, even when using off-the-shelf systems for surface realization. This observation has spurred recent interest in automatically learning end-to-end generation systems (Mairesse et al., 2010; Konstas and Lapata, 2012; Wen et al., 2015; Dušek and Jurčíček, 2016); however, these approaches tend to use shallow meaning representations (Howcroft et al., 2017) and do not make effective use of prior work on surface realization to constrain the learning problem or to ensure grammaticality in the resulting texts.

Based on these observations, we propose a Bayesian nonparametric approach to learning sentence planning rules for a conventional NLG system. Making use of existing systems for surface realization along with more sophisticated meaning representations allows us to cast the problem as a grammar induction task. Our system induces synchronous tree substitution grammars for pairs of text plans and morphosyntactically-specified dependency trees. Manual inspection of the rules and texts currently produced by our system indicates that they are generally of good quality, encouraging further evaluation.

2 Overview

Whether using hand-crafted or end-to-end generation systems, the common starting point is collecting a corpus with semantic annotations in the target domain. Such a corpus should exhibit the range of linguistic variation that developers hope to achieve in their NLG system, while the semantic annotations should be aligned with the target input for the system, be that database records, flat 'dialogue act' meaning representations, or hierarchical discourse structures.

For our system (outlined in Figure 1) we focus on generating short paragraphs of text containing one or more discourse relations in addition to propositional content. To this end we use as input a *text plan* representation based on that used in the SPaRKy Restaurant Corpus (Walker et al., 2007). These text plans connect individual propositions under nodes representing relations drawn from Rhetorical Structure Theory (Mann and Thompson, 1988).

Rather than using a fully end-to-end approach to learn a tree-to-string mapping from our text plans to paragraphs of text, we constrain the learning problem by situating our work in the context of a conventional NLG pipeline (Reiter and Dale, 2000). In the pipeline approach, NLG is decomposed into three stages: document planning, sentence planning, and surface realization. Our approach assumes that the text plans we are working with are the product of document planning, and we use an existing parser-realizer for surface realization. This allows us to constrain the learning problem by limiting our search to the set of tree-to-tree mappings which produce valid input for the sur-

Proceedings of The 11th International Natural Language Generation Conference, pages 391–396,
Tilburg, The Netherlands, November 5-8, 2018.

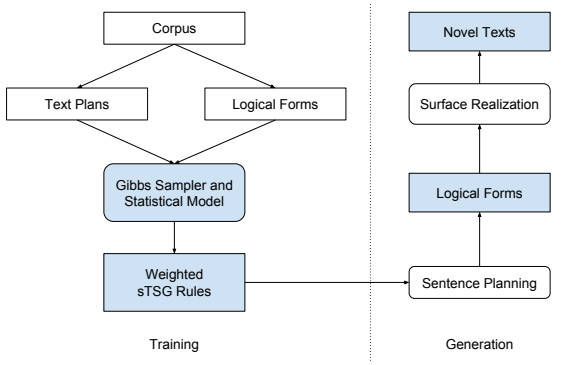

Figure 1: Overview of our pipeline. Square boxes represent data; rounded boxes represent programs. Blue boxes represent our system and the outputs dependent on it, while boxes with white background represent existing resources used by our system.

face realizer, leveraging the linguistic knowledge encoded in this system. Restricting the problem to sentence planning also means that our system needs to learn lexicalization, aggregation, and referring expression generation rules but not rules for content selection, linearization, or morphosyntactic agreement.

The input to our statistical model and sampling algorithm consists of pairs of text plans (TPs) and surface realizer input trees, here called *logical forms* (LFs). At a high level, our system uses heuristic alignments between individual nodes of these trees to initialize the model and then iteratively samples possible alternative and novel alignments to determine the best set of synchronous derivations for TP and LF trees. The synchronous tree substitution grammar rules induced in this way are then used for sentence planning as part of our NLG pipeline.

3 Synchronous TSGs

Synchronous tree substitution grammars (TSGs) are a subset of synchronous tree adjoining grammars, both of which represent the relationships between pairs of trees (Shieber and Schabes, 1990; Eisner, 2003). A tree substitution grammar consists of a set of *elementary trees* which can be used to expand non-terminal nodes into a complete tree.

Consider the example in Figure 2, which shows the text plan and logical form trees for the sentence, *Sonia Rose has very good food quality, but Bienvenue has excellent food quality*.

The logical form in this figure could be derived

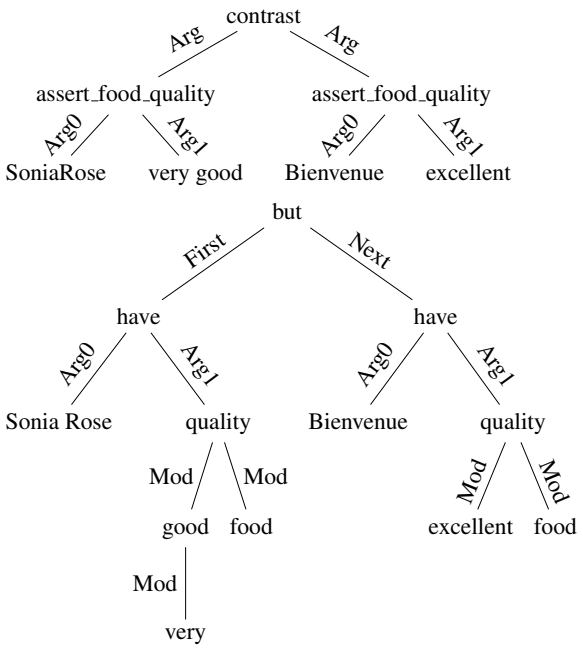

Figure 2: Text plan (top) & logical form (bottom) for the text *Sonia Rose has very good food quality but Bienvenue has excellent food quality*.

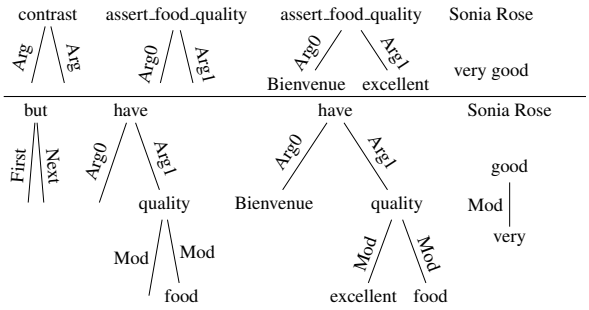

Figure 3: Possible elementary trees for the TP (top row) and LF (bottom row) in Figure 2, omitting some detail for simplicity.

in one of three ways. First, we could simply have this entire tree memorized in our grammar as an elementary tree. This would make the derivation trivial but would also result in a totally ungeneralizable rule. On the other hand, we could have the equivalent of a CFG derivation for the tree consisting of rules like *but* $\rightarrow$ *First Next*, *First* $\rightarrow$ *have*, *have* $\rightarrow$ *Arg0 Arg1*, and so on. These rules would be very general, but the derivation then requires many more steps. The third option, illustrating the appeal of using a tree substitution grammar, involves elementary trees of intermediate size, like those in Figure 3.

The rules in Figure 3 represent a combination

of small, CFG-like rules (e.g. the elementary tree rooted at *but*), larger trees representing memorized chunks (i.e. the rule involving *Bienvenue*), and intermediate trees, like the one including *have* $\rightarrow$ *quality* $\rightarrow$ *food*. In these elementary trees, the empty node sites at the end of an arc represent *substitution sites*, where another elementary tree must be expanded for a complete derivation. In typical applications of TSGs over phrase structure grammars, these substitution sites would be labeled with non-terminal categories which then correspond to the root node of the elementary tree to be expanded. In our (synchronous) TSGs over trees with labeled arcs, we consider the 'nonterminal label' at each substitution site to be the *tree location*, which we define as the label of the parent node paired with the label of the incoming arc.

A *synchronous* tree substitution grammar, then, consists of pairs of elementary trees along with an alignment between their substitution sites. For example, we can combine the TP elementary tree rooted at *contrast* with the LF elementary tree rooted at *but*, aligning each $(\textit{contrast}, \textit{Arg})$ substitution site in the TP to the $(\textit{but}, First)$ and $(\textit{but}, \textit{Next})$ sites in the LF.

4 Dirichlet Processes

The Dirichlet process (DP) provides a natural way to trade off between prior expectations and observations. For our purposes, this allows us to define prior distributions over the infinite, discrete space of all possible pairs of TP and LF elementary trees and to balance these priors against the full trees we observe in the corpus.

We follow the Chinese Restaurant Process formulation of DPs, with concentration parameter $\alpha = 1$.[1] Here, the probability of a particular elementary tree e being observed is given by:

$$P(e) = \frac{\textit{freq}(e)}{\#\textit{obs} + \alpha} + \frac{\alpha}{\#\textit{obs} + \alpha} P_{\textit{prior}}(e), \quad (1)$$

where $\textit{freq}(e)$ is the number of times we have observed the elementary tree e, $\#\textit{obs}$ is the total number of observations, and $P_{\textit{prior}}$ is our prior.

It is clear that when we have no observations, we estimate the probability of e entirely based on our prior expectations. As we observe more data, however, we rely less on our priors in general.

[1] Other concentration parameters are possible, but $\alpha = 1$ is the standard default value, and we do not perform any search for a more optimal value at this time.

5 Statistical Model

Our model uses Dirichlet processes with other DPs as priors (i.e. Hierarchical Dirichlet Processes, or HDPs). This allows us to learn more informative prior distributions (the lower-level DPs) to improve the quality of our predictions for higher-level DPs. Section 5.1 describes the HDPs used to model elementary trees for text plans and logical forms, which rely on prior distributions over possible node and arc labels. This model in turn serves as the prior for the synchronous TSG's pairs of elementary trees, as described in Section 5.2 along with the HDP over possible alignments between the frontier nodes of these pairs of elementary trees. A plate diagram of the model is presented in Figure 4.

5.1 HDP for TSG Derivations

We begin by defining TSG base distributions for text plans and logical forms independently. Our generative story begins with sampling an elementary tree for the root of the tree and then repeating this sampling procedure for each frontier node in the expanded tree.

Since the tree locations l corresponding to frontier nodes are completely determined by the current expansion of the tree, we only need to define a distribution over possible elementary trees conditioned on the tree location:

$$T|l \sim \textit{DP}(1.0, P(e|l)) \quad (2)$$

$$\begin{aligned} P(e|l) =& N(n(\textit{root}(e))|l) \\ & \Pi_{a \in a(\textit{root}(e))} A(a|n(\textit{root}(e))) \\ & \Pi_{\textit{child} \in \textit{children}(\textit{root}(e))} P(\textit{child}|l(\textit{child})), \end{aligned} \quad (3)$$

where N and A are Dirichlet processes over possible $n$ode labels and $a$rc labels, we use $N(n|l)$ for the probability of node label n at tree location l according to DP N, and similarly for A. We further overload our notation to use $n(\textit{node})$ to indicate the node label for a given node, $a(\textit{node})$ to indicate the outward-going arc labels from *node*, and $l(e)$ or $l(\textit{node})$ to indicate the location of a given subtree or node within the tree as an (n, l) pair. $\textit{root}(e)$ is a function selecting the root node of an elementary tree e and $\textit{children}(\textit{node})$ indicates the child subtrees of a given node.

The distributions over node labels given tree locations $N|l$ and arc labels given source node labels $A|n$ are DPs over simple uniform priors:

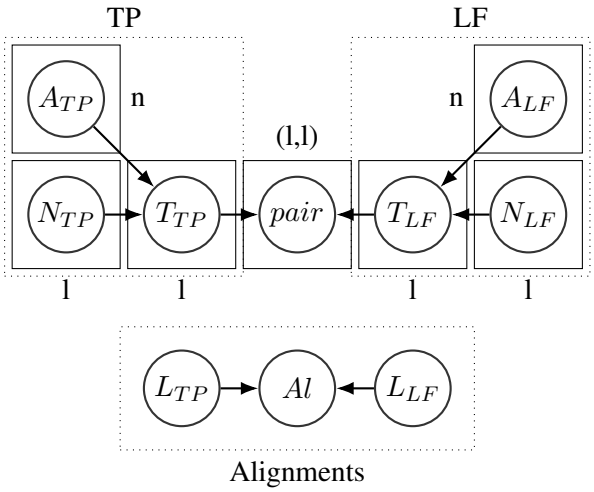

Figure 4: Dependencies in our statistical model, omitting parameters for clarity. Each node represents a Dirichlet process over base distributions (see Sec. 5) with $\alpha = 1$. n here indexes node labels for TPs or LFs as appropriate, while l similarly represents tree locations.

$$N|l \sim DP(1.0, Uniform(\{n \in corpus\})) \quad (4)$$

$$A|n \sim DP(1.0, Uniform(\{a \in corpus\})) \quad (5)$$

5.2 HDP for sTSG Derivations

Our synchronous TSG model has two additional distributions: (1) a distribution over pairs of TP and LF elementary trees; and (2) a distribution over pairs of tree locations representing the probability of those locations being aligned to each other.

Similarly to the generative story for a single TSG, we begin by sampling a pair of TP & LF elementary trees, a *TreePair*, for the root of the derivation. We then sample alignments for the frontier nodes of the TP to the frontier nodes of the LF. For each of these alignments, we then sample the next TreePair in the derivation and repeat this sampling procedure until no unfilled frontier nodes remain.

The distribution over TreePairs for a given pair of tree locations is given by a Dirichlet process with a simple prior which multiplies the probability of a given TP elementary tree by the probability of a given LF elementary tree:

$$pair|l_{TP}, l_{LF} \sim DP(1.0, P(e_{TP}, e_{LF}|l_{TP}, l_{LF})) \quad (6)$$

$$P(e_{TP}, e_{LF}|l_{TP}, l_{LF}) = T_{TP}(e_{TP}|l_{TP})T_{LF}(e_{LF}|l_{LF}) \quad (7)$$

The distribution over possible alignments is given by an DP whose prior is the product of the probabilities of pair of (TP and LF) tree locations in question. These probabilities are each modeled as a DP with a uniform prior over possible tree locations.

$$Al \sim DP(1.0, P(l_{TP}, l_{LF})) \quad (8)$$

$$P(l_{TP}, l_{LF}) = P(l_{TP})P(l_{LF}) \quad (9)$$

$$P(l.) \sim DP(1.0, Uniform(\{l.\})) \quad (10)$$

5.3 Sampling

Our Gibbs sampler adapts the blocked sampling approach of (Cohn et al., 2010) to synchronous grammars. For each text in the corpus, we resample a synchronous derivation for the entire text before updating the associated model parameters.

6 Generation

While our pipeline can in principle work with any reversible parser-realizer, our current implementation uses OpenCCG[2] (White, 2006; White and Rajkumar, 2012). We use the broad-coverage grammar for English based on CCGbank (Hockenmaier, 2006). The 'logical forms' associated with this grammar are more or less syntactic in nature, encoding the lemmas to be used, the dependencies among them, and morphosyntactic annotations in a dependency semantics. Parsing the corpus with OpenCCG provides the LFs we use for training.

After training the model, we have a collection of synchronous TSG rules which can be applied to (unseen) text plans to produce new LFs. For this rule application we use Alto[3] (Koller and Kuhlmann, 2012) because of its efficient implementation of parsing for synchronous grammars. The final stage in the generation pipeline is to realize these LFs using OpenCCG, optionally performing reranking on the resulting texts. Some examples of the resulting texts are provided in the next section.

[2] https://github.com/OpenCCG/openccg
[3] https://bitbucket.org/tclup/alto

7 Example output

As a testbed during development we used the SPaRKy Restaurant Corpus (2007), a corpus of restaurant recommendations and comparisons generated by a hand-crafted NLG system. While the controlled nature of this corpus is ideal for testing during development, our future evaluations will also use the more varied Extended SRC (Howcroft et al., 2017).[4]

After training on about 700 TP-LF pairs for 5k epochs, our system produces texts such as:

1. Chanpen Thai has the best overall quality among the selected restaurants. Its price is 24 dollars and it has good service. **This Thai restaurant** has good food quality, with decent decor.
2. **Since** Komodo's price is 29 dollars and it has good decor, it has the best overall quality among the selected restaurants.
3. Azuri Cafe, **which** is a **Vegetarian** restaurant has very good food quality. Its price is 14 dollars. It has the best overall quality among the selected restaurants.
4. Komodo has very good service. It has **food food quality, with very good food quality, it has very good food quality** and its price is 29 dollars.

Here we see examples of pronominalization throughout, as well as the deictic referring expression *this Thai restaurant* (in 1), which avoids repeating either the pronoun 'it' or the name of the restaurant again. The system also makes good use of discourse connectives (like 'since' in 2) as well as non-restrictive relative clauses (as in 3). However, the system does not always handle punctuation correctly (as in 3) and sometimes learns poor semantic alignments, aligning but ommitting part of the meaning in saying 'Vegetarian' for 'Kosher, Vegetarian' in 3 and completely misaligning 'good' to 'food' in (4) due to the frequent co-occurrence of these words in the corpus. Moreover, example 4 also demonstrates that some combinations of rules based on poor alignments can lead to repetition.

While there is clearly still room for improvement, the quality of the texts overall is encouraging, and we are currently preparing a systematic human evaluation of the system.

8 Related Work

While the present work aims to learn sentence planning rules in general, White and Howcroft (2015) focused on learning clause-combining rules, using a set of templates of possible rule types to extract a set of clause-combining operations based on pattern matching. The resulting rules were, like ours, tree-to-tree mappings; however, our rules proceed directly from text plans to final logical forms, while their approach assumed lexicalized text plans (i.e. logical forms without any aggregation operations applied) paired with logical forms as training input. In learning a synchronous TSG, the model presented here aims to avoid using hand-crafted rule templates, which are more dependent on the specific representation chosen for surface realizer input.

As mentioned in the introduction, there have been a number of attempts in recent years to learn end-to-end generation systems which produce text directly from database records (Konstas and Lapata, 2012), dialogue acts with slot-value pairs (Mairesse et al., 2010; Wen et al., 2015; Dušek and Jurčíček, 2016), or semantic triples like those used in the recent WebNLG challenge (Gardent et al., 2017). In contrast, we assume that content selection and discourse structuring are handled before sentence planning. In principle, however, our methods can be applied to any generation subtask involving tree-to-tree mappings.

9 Discussion and Conclusion

We have presented a Bayesian nonparametric approach to learning synchronous tree substitution grammars for sentence planning. This approach is designed to address specific weaknesses of end-to-end approaches with respect to discourse structure as well as grammaticality. Our preliminary analysis suggests that our approach can learn useful sentence planning rules from smaller datasets than those typically used for training neural models. We are currently preparing to launch an extensive human evaluation of our model compared to current neural approaches to text generation.

Acknowledgments

We would like to thank Leon Bergen for advice on Bayesian induction of TSGs, Jonas Groschwitz for assistance with Alto, and Sacha Beniamine & Meaghan Fowlie for proofreading and revision advice. This work was supported by DFG collaborative research center SFB 1102 'Information Density and Linguistic Encoding'.

[4] For details about differences between these two corpora, we refer the interested reader to Howcroft et al. (2017).

References

Trevor Cohn, Phil Blunsom, and Sharon Goldwater. 2010. Inducing Tree-Substitution Grammars. *Journal of Machine Learning Research* 11:3053–3096.

Ondřej Dušek and Filip Jurčíček. 2016. Sequence-to-Sequence Generation for Spoken Dialogue via Deep Syntax Trees and Strings. In *Proc. of the 54th Annual Meeting of the Association for Computational Linguistics (ACL; Volume 2: Short Papers)*. Association for Computational Linguistics, Berlin, Germany, pages 45–51. https://doi.org/10.18653/v1/P16-2008.

Jason Eisner. 2003. Learning non-isomorphic tree mappings for machine translation. In *Proc. of the 41st Annual Meeting on Association for Computational Linguistics (ACL): Short Papers*. Association for Computational Linguistics, volume 2, pages 205–208.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The WebNLG Challenge: Generating Text from RDF Data. In *Proc. of the 10th International Conference on Natural Language Generation (INLG)*. Association for Computational Linguistics, Santiago de Compostela, Spain, pages 124–133. https://doi.org/10.18653/v1/W17-3518.

Julia Hockenmaier. 2006. Creating a CCGbank and a wide-coverage CCG lexicon for German. In *Proc. of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL)*. Association for Computational Linguistics, Sydney, Australia, pages 505–512. https://doi.org/10.3115/1220175.1220239.

David M. Howcroft, Dietrich Klakow, and Vera Demberg. 2017. The Extended SPaRKy Restaurant Corpus: Designing a Corpus with Variable Information Density. In *Proc. of Interspeech 2017*. ISCA, Stockholm, Sweden, pages 3757–3761. https://doi.org/10.21437/Interspeech.2017-1555.

Alexander Koller and Marco Kuhlmann. 2012. Decomposing TAG algorithms using simple algebraizations. In *Proc. of the 11th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+)*. Association for Computational Linguistics, Paris, France, pages 135–143.

Ioannis Konstas and Mirella Lapata. 2012. Unsupervised concept-to-text generation with hypergraphs. In *Proc. of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*. Association for Computational Linguistics, Montréal, Canada, pages 752–761.

François Mairesse, Milica Gasic, Filip Jurčíček, Simon Keizer, Blaise Thomson, Kai Yu, and Steve Young. 2010. Phrase-based Statistical Language Generation using Graphical Models and Active Learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics, Uppsala, Sweden.

William C Mann and Sandra A Thompson. 1988. Rhetorical Structure Theory: Towards a functional theory of text organization. *TEXT* 8(3):243–281.

Ehud Reiter and Robert Dale. 2000. *Building Natural Language Generation Systems*. Cambridge University Press.

Stuart M. Shieber and Yves Schabes. 1990. Synchronous tree-adjoining grammars. In *Papers Presented to the 13th International Conference on Computational Linguistics*. Helsinki, Finland, volume 3, pages 253–258. https://doi.org/10.3115/991146.991191.

Marilyn A. Walker, Amanda Stent, François Mairesse, and Rashmi Prasad. 2007. Individual and domain adaptation in sentence planning for dialogue. *Journal of Artificial Intelligence Research* 30:413–456. https://doi.org/10.1613/jair.2329.

Tsung-Hsien "Shawn" Wen, Pei-hao Su, David Vandyke, Steve Young, and Trumpington Street. 2015. Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems. In *Proc. of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Lisbon, Portugal, pages 1711–1721. https://doi.org/10.18653/v1/D15-1199.

Michael White. 2006. CCG Chart Realization from Disjunctive Inputs. In *Proc. of the Fourth International Natural Language Generation Conference (INLG)*. Association for Computational Linguistics, Sydney, Australia, pages 12–19.

Michael White and David M. Howcroft. 2015. Inducing Clause-Combining Rules: A Case Study with the SPaRKy Restaurant Corpus. In *Proc. of the 15th European Workshop on Natural Language Generation (ENLG)*. Association for Computational Linguistics, Brighton, United Kingdom, pages 28–37. https://doi.org/10.18653/v1/W15-4704.

Michael White and Rajakrishnan Rajkumar. 2012. Minimal Dependency Length in Realization Ranking. In *Proc. of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP)*. Association for Computational Linguistics, Jeju Island, South Korea, pages 244–255.