Constrained Decoding for Neural NLG from Compositional Representations in Task-Oriented Dialogue

Anusha Balakrishnan[*] **Jinfeng Rao**[*] **Kartikeya Upasani**[*] **Michael White**[*†] and **Rajen Subba**[*]

Facebook Conversational AI

{anushabala,raojinfeng,kart,mwhite14850,rasubba}@fb.com

Abstract

Generating fluent natural language responses from structured semantic representations is a critical step in task-oriented conversational systems. Avenues like the E2E NLG Challenge have encouraged the development of neural approaches, particularly sequence-to-sequence (Seq2Seq) models for this problem. The semantic representations used, however, are often underspecified, which places a higher burden on the generation model for sentence planning, and also limits the extent to which generated responses can be controlled in a live system. In this paper, we (1) propose using tree-structured semantic representations, like those used in traditional rule-based NLG systems, for better discourse-level structuring and sentence-level planning; (2) introduce a challenging dataset using this representation for the weather domain; (3) introduce a constrained decoding approach for Seq2Seq models that leverages this representation to improve semantic correctness; and (4) demonstrate promising results on our dataset and the E2E dataset.

1 Introduction

Generating fluent natural language responses from structured semantic representations is a critical step in task-oriented conversational systems. With their end-to-end trainability, neural approaches to natural language generation (NNLG), particularly sequence-to-sequence (Seq2Seq) models, have been promoted with great fanfare in recent years (Wen et al., 2015, 2016; Mei et al., 2016; Kiddon et al., 2016; Dušek and Jurcicek, 2016), and avenues like the recent E2E NLG challenge (Dušek et al., 2018, 2019) have made available large datasets to promote the development of these models. Nevertheless, current NNLG models arguably remain inadequate for most real-world task-oriented dialogue systems, given their inability to (i) reliably perform common sentence planning and discourse structuring operations (Reed et al., 2018), (ii) generalize to complex inputs (Wiseman et al., 2017), and (3) avoid generating texts with semantic errors including hallucinated content (Dušek et al., 2018, 2019).[1]

In this paper, we explore the extent to which these issues can be addressed by incorporating lessons from pre-neural NLG systems into a neural framework. We begin by arguing in favor of enriching the input to neural generators to include discourse relations — long taken to be central in traditional NLG — and underscore the importance of exerting control over these relations when generating text, particularly when using user models to structure responses. In a closely related work, Reed et al. (2018), the authors add control tokens (to indicate contrast and sentence structure) to a flat input MR, and show that these can be effectively used to control structure. However, their methods are only able to control the presence or absence of these relations, without more fine-grained control over their structure. We thus go beyond their approach and propose using full tree structures as inputs, and generating tree-structured outputs as well. This allows us to define a novel method of constrained decoding for standard sequence-to-sequence models for generation, which helps ensure that the generated text contains all and only the specified content, as in classic approaches to surface realization.

On the E2E dataset, our experiments demonstrate much better control over CONTRAST relations than using Reed et al.'s method, and also show improved diversity and expressiveness over standard baselines. We also release a new dataset of responses in the weather domain, which includes the JUSTIFY, JOIN and CONTRAST rela-

[*] Alphabetical by first name

[†] Work done while on leave from Ohio State University

[1] Also see https://ehudreiter.com/2018/11/12/hallucination-in-neural-nlg/.

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 831–844
Florence, Italy, July 28 - August 2, 2019. ©2019 Association for Computational Linguistics

Reference 1	JJ's Pub is not family friendly, but has a high customer rating of 5 out of 5. It is a restaurant near the Crowne Plaza Hotel.
Reference 2	JJ's Pub is not a family friendly restaurant. It has a high customer rating of 5 out of 5. You can find it near the Crowne Plaza Hotel.
E2E MR	name[JJ's Pub] rating[5 out of 5] familyFriendly[no] eatType[restaurant] near[Crowne Plaza Hotel]
Our MR for Reference 1	**CONTRAST** [**INFORM** [name[JJ's Pub] familyFriendly[no]] **INFORM** [rating[5 out of 5]]] **INFORM** [eatType[restaurant] near[Crowne Plaza Hotel]]

Table 1: Sample reference responses, their corresponding meaning representation in the E2E dataset, and its MR according to our proposed ontology.

tions, and where discourse-level structures come into play. On both E2E and weather datasets, we show that constrained decoding over our enriched inputs results in higher semantic correctness as well as better generalizability and data efficiency.

The rest of this paper is organized as follows: Section 2 describes the motivation for using compositional inputs organized around discourse relations. Section 3 explains our data collection approach and dataset.[2] Section 4 shows how to incorporate compositional inputs into NNLG and describes our constrained decoding algorithm. Section 5 presents our experimental setup and results.

2 Towards More Expressive Meaning Representations

2.1 Limitations of Flat MRs

In the E2E dataset, meaning representations (MRs) are a flat list of key-value pairs, where each key is a slot name that needs to be mentioned, and the value is the value of that slot (see Table 1). In Wen et al. (2015), MRs have a similar structure, and additionally contain information about the **dialog act** that needs to be conveyed (REQUEST, INFORM, etc.). These MRs are sufficient to capture basic semantic information, but fail to capture rhetorical (or discourse) relations, like CONTRAST, that have long been taken to be central to generating coherent discourse in traditional NLG (Mann and Thompson, 1988; Moore and Paris, 1993; Reiter and Dale, 2000; Stent et al., 2002). The two references in Table 1 illustrate this problem with the expressiveness of such flat MRs. Critical discourse information, like whether two attributes should be contrasted (or whether to justify a recommendation, etc.), is not captured by the MR. This poses a dual challenge: First, since the MR does not specify these discourse relations, crowdworkers creating the dataset in turn have no instructions on when to use them, and must thus use their own judgment in creating a natural-sounding response. While the E2E organizers tout the resulting response variations as a plus, Reed et al. (2018) find that current neural systems are unable to learn to express discourse relations effectively with this dataset, and explore ways of enriching input MRs to do so. Indeed, now that the E2E system outputs have been released, a search through outputs from all participating systems reveals only 43 outputs (0.4% out of 10080) containing contrastive tokens, on a test set containing about 300 contrastive samples.[3]

Second, going beyond Reed et al., we argue that the **controllability** of these relations through MRs is desirable in live conversational systems, where external knowledge like user models may inform decisions around contrast, grouping, or justifications. While several studies have shown that controlling such discourse behaviors can be critical to user perceptions of quality and naturalness (Lemon et al., 2004; Carenini and Moore, 2006; Walker et al., 2007; White et al., 2010; Demberg et al., 2011), flat MRs provide no means to do so. This leaves it to the neural model to learn general trends in the data, such as contrasting a good attribute like a 5-star rating with a typically dispreferred attribute like not being family friendly or serving English food. However, sometimes people are interested in adult-oriented establishments, and some people may even like English food; for users with these preferences, text generated according to general trends will be incoherent. For example, for a user known to be seeking an adult-oriented locale, Ref. 1 in Table 1 would be incoherent, and less preferable than a

[2] The datasets and implementations can be found at https://github.com/facebookresearch/TreeNLG.

[3] An additional 86 outputs contained these tokens, but were generated by the TR2 template-based system (Smiley et al., 2018). The expected number of contrastive system outputs would be 4,200 if each of the 14 participating systems produced contrastive tokens consistently with the data distribution.

non-contrastive alternative such as *JJ's Pub is a highly-rated restaurant for adults near the Crowne Plaza Hotel*.

2.2 Tree-Structured MRs

In order to overcome these challenges, we propose the use of structured meaning representations like those explored widely in (hybrid) rule-based NLG systems (Rambow et al., 2001; Reiter and Dale, 2000; Walker et al., 2007). Our representation consists of three parts:

1. **Argument** can be any entity or slot mentioned in a response, like the name of a restaurant or the date. Some arguments can be complex and contain sub-arguments (e.g. a `date_time` argument has subfields like `week_day` and `month`).
2. **Dialog act** is an atomic unit that could correspond linguistically to a single clause. A dialog act can contain one or more arguments that need to be expressed. Examples: INFORM, YES, RECOMMEND.
3. **Discourse relation** defines the relationships between dialog acts. A single discourse relation may contain multiple other dialog or discourse relations, allowing for potentially arbitrary degrees of nesting. Examples: JOIN, JUSTIFY, CONTRAST.

A meaning representation that uses this formulation can consist of an arbitrary number and combination of discourse relations and dialog acts, resulting in a nested tree-structured MR with much higher expressiveness and specificity. Table 1, seen earlier, shows an example of an MR structured in this way, as well as the corresponding "flat" MR and its reference in the E2E dataset.

In addition to improved expressiveness, this representation results in more atomic definitions of dialog acts and arguments than in flat MRs. For example, consider the example in the weather domain from Table 2: The response contains multiple dialog acts, a contrast and several instances of ellipsis and grouping (i.e., temperatures are grouped and mentioned separately from wind condition). Additionally, some arguments, like `date_time`, occur multiple times in the response and correspond to different dialog acts, with several different values. A flat MR will struggle to represent 1) the correspondence of arguments to dialog acts; 2) what attributes to group and contrast and 3) semantic equivalence of arguments like `date_time1` and `date_time2`. On the other hand, our MRs ease discourse-level learning and encourage reuse of arguments across multiple dialog acts.

3 Dataset

With this representation in mind, we created an ontology of dialog acts, discourse relations, and arguments, for the weather domain. Our motivation for choosing the weather domain, as explored in (Liang et al., 2009), is that this domain offers significant complexity for NLG. Weather forecast summaries in particular can be very long, and require reasoning over several disjoint pieces of information. In this work, we focused on collecting a dataset that showcases the complexity of weather summaries over date/time ranges. Our weather dataset is also unique in that it was collected in a *conversational* setup (see below).

We collected our dataset in multiple stages:

1. Query collection. We asked crowdworkers to come up with sample queries in the weather domain, like *What's the weather like tomorrow?* and *Do I need an umbrella tonight?*

2. Query annotation. We then wrote rules to automatically parse these queries, and extract key pieces of information, like the location, date, and any attributes that the user specifically requested in the question.

3. MR generation. Our goal was to create MRs that are sufficiently expressive and straightforward to create automatically in a practical system. In the weather domain, it's conceivable that the NLG system has access to a weather API that provides it with detailed weather forecasts for the range requested by the user. To mimic this setting, we generated artificial weather forecasts for every user query based on the arguments (full argument set in Table 3) in the user query. We then created the tree-structured MR by applying a few different types of automatic rules, like adding `CONTRAST` to weather conditions that are in opposition. We add more details of our response generation method and the specific rules for MR creation in Appendix A and B.

4. Response generation and annotation. We presented these tree-structured MRs to trained annotators, and asked them to write responses that expressed the MRs. They were also given the user query and asked to make their responses natural given the query. They were allowed to elide in-

Reference	It'll be sunny throughout this weekend. The high will be in the 60s, but expect temperatures to drop as low as 43 degrees by Sunday evening. There's also a chance of strong winds on Saturday morning.

Flat MR	`condition1[sunny] date_time1[this weekend] avg_high1[60s] low2[43]`
	`date_time2[Sunday evening] chance3[likely] wind_summary3[strong]`
	`date_time3[Saturday morning]`

Our MR	`INFORM [ condition[sunny], date_time_range[ colloquial[this weekend ] ] ]`
	`CONTRAST [`
	`   INFORM [ avg_high[60s] date_time[ [colloquial this weekend ] ] ]`
	`   INFORM [ low[43] date_time[ week_day[Sunday] colloquial[evening] ] ]`
	`]`
	`INFORM [ chance[likely], wind_summary[heavy], date_time[ week_day[Saturday]`
	`colloquial[morning] ] ]`

Table 2: Sample flat MR with reference compared against our proposed tree-structured MR. Nodes in blue are all children of the root node of the tree.

Dialog Acts	INFORM, RECOMMEND, YES, NO, ERROR
Discourse Relations	JOIN, CONTRAST, JUSTIFY
Arguments	date_time*, date_time_range*, location* attire[n], activity[n], condition[n], humidity[n] precip_amount, precip_amount_unit, precip_chance precip_chance_summary, precip_type, sunrise_time, temp, temp_high[s], temp_low[s], temp_unit wind_speed[n], wind_speed_unit, sunset_time, task bad_arg, bad_value, error_reason

Table 3: Ontology for the weather domain dataset that we collected. Arguments marked with * are nested arguments (see Table 4). [n] indicates arguments that have a corresponding _not argument; [s] indicates arguments that have a corresponding _summary.

Argument	**Subfields**
date_time	year, month, day, weekday, colloquial
date_time_range	start_year, start_month, start_day, start_weekday end_year, end_month, end_day, end_weekday, colloquial
location	city, region, country, colloquial

Table 4: Defined subfields for nested arguments.

Frequency	**0**	**1**	**2**	**3**	**4**	**5**
# Dialog Acts	0	6469	12077	9801	4095	685
# Discourse Rels	18137	12494	2393	103	1	0

Table 5: Frequency distribution of number of dialog acts and discourse relations in the weather dataset.

formation when arguments were repeated across dialog acts, and could choose the most appropriate surface forms for any arguments based on contextual clues (e.g. referring to a date as *tomorrow*, rather than *April 24th*, depending on the user's date). Finally, we asked them to label response spans corresponding to each argument, dialog act, and discourse relation in the MR.

5. Quality evaluation. Finally, we presented a different group of annotators with the annotated responses, and asked them to provide evaluations of *fluency*, *correctness*, *naturalness*, and *annotation correctness*.

3.1 Dataset statistics

Our final dataset has 33,493 examples. Each example comprises a user query, the synthetic user context (datetime and location), the tree-structured MR, the response, and a complete tree-structured annotation of the response. Table 6 contains an example from our dataset; as shown, the response annotation structure closely mirrors that of the MR itself. The MRs and responses in the dataset range from very simple (a single dialog act) to very complex (an MR with a depth and width of 4). A distribution of this complexity is shown in Table 5. The vocabulary size is 1485, and the max/average/min lengths of responses are 151/40.6/8. The dataset also poses several challenges in addition to syntactic and semantic complexity. As mentioned before, it has a rich set of referring expressions for dates and date ranges. It also contains user queries on which the written response was based, thus creating the opportunity for studies on improving naturalness or relevance with respect to the user query. These could be useful in particular for learning to express recommendations and justifications, as well as YES and NO dialog acts.

Our final training set contains 25,390 examples, with 11,879 unique MRs. (We consider two MRs to be identical if they have the same **delexicalized** tree structure — see Section 4.1.) The test set contains 3,121 examples, of which 1.1K (35%) have unique MRs *that have never been seen in the training set*.

3.2 Enriched E2E Dataset

We also used heuristic techniques to convert the E2E dataset to use tree-structured MRs. We used the output of Juraska et al.'s (2018) tagger to find a character within each slot in the flat MR, and automatically adjusted these to correspond to a token boundary if they didn't already. We then used the Viterbi segmentations from the model released by Wiseman et al. (2018) to get spans corresponding to each argument. Finally, we used the Berkeley neural parser (Kitaev and Klein, 2018) to identify spans coordinated by *but*, and added CONTRAST relations as parents of the coordinated arguments. We added JOIN based on sentence boundaries. An interesting direction for future research would be

Query	Context	MR	Response
When will it snow next?	Reference date: 29th September 2018	[CONTRAST [INFORM_1 [LOCATION [CITY Parker]] [CONDITION_NOT snow] [DATE_TIME [DAY 29] [MONTH September] [YEAR 2018]]] [INFORM_2 [DATE_TIME [DAY 29] [MONTH September] [YEAR 2018]] [LOCATION [CITY Parker]] [CONDITION heavy rain showers] [CLOUD_COVERAGE partly cloudy]]]	Parker is not expecting any snow, but today there's a very likely chance of heavy rain showers, and it'll be partly cloudy

Annotated Response
[CONTRAST [INFORM_1 [LOCATION [CITY Parker]] is not expecting any [CONDITION_NOT snow]], but [INFORM_2 [DATE_TIME [COLLOQUIAL today]] there's a [PRECIP_CHANCE_SUMMARY very likely chance] of [CONDITION heavy rain showers] and it'll be [CLOUD_COVERAGE partly cloudy]]]

Table 6: Example response, MR, and other metadata from our dataset

to extend Wiseman et al.'s methods to induce tree structures directly. In the final dataset we obtained (~51K examples), ~24K examples (47%) contain JOIN, while 2237 (4.3%) contain CONTRAST.

4 Model

4.1 Seq2Seq with Linearized Trees

In this work, we use a standard Seq2Seq model with attention (Sutskever et al., 2014; Bahdanau et al., 2014), implemented in the `fairseq-py` repository (Gehring et al., 2017). The encoder and decoder are both Long Short-Term Memory (LSTM) -based (Hochreiter and Schmidhuber, 1997) and the decoder uses beam search for generation. The input to the model is a linearized representation of the tree-structured MR, and the output is a linearized tree-structured representation of the annotated response (see Table 6). This means that in addition to predicting tokens for the surface realization of the response, the model must also predict non-terminals (dialog/discourse relations and arguments) to indicate the start or end of each span. One advantage of predicting a tree structure is that the model has supervision on the alignment between the MR and the response. Additionally, this predicted tree structure can be used to help verify the correctness of the predicted response; we leverage this for our constrained decoding approach described next. We also **delexicalized** tokens in the response that correspond to sparse entities, like names in the E2E dataset and temperatures in the weather dataset (see Appendix D).

4.2 Constrained Decoding

As described above, the output structure predicted by the model forms a tree that should correspond neatly to the input MR, barring some instances of ellipses (as with the `date_time` argument in Table 6).[4] Thus, the input MR can be seen as a constraint on the semantic correctness of the prediction; if the predicted structure doesn't match the MR, the prediction is incorrect and can be rejected. Figure 1 illustrates such ideas.

Input MR:
[INFORM [name]]
[CONTRAST [pricerange_expensive] [customerrating_high]]

Outputs:
(1) [INFORM [name name] is] [CONTRAST [pricerange_expensive expensive] but [customerrating_high highly rated] .]

(2) [INFORM [name name] is] [CONTRAST [customerrating_high highly rated] but [pricerange_expensive expensive] .]

(3) [INFORM [name name] is [customerrating_high highly rated] and [pricerange_expensive expensive] .]

Figure 1: Examples of constraint checking. (1) and (2) are valid outputs. (3) fails to meet tree constraints since the CONTRAST node is not present and the INFORM node has illegal children `customerrating` and `pricerange`.

Our beam search algorithm works as follows.[5] First, the input tree is scanned to identify groups of two or more nodes that have the same value, so that ellipsis can be enabled by optionally allowing just one node in each group. Then, as the tree structure is incrementally decoded, non-terminals are checked against the input tree for validity. When an opening bracket token (e.g., `[name`) is generated, it is not accepted if it isn't a child of the current parent node in the input tree, or has already been generated in the current subtree, thereby preventing repetition and hallucination of arguments or acts. When a closing bracket token `]` or an end-of-sentence (EOS) token is generated, it is accepted only if all children of the current parent are covered either directly or through ellipsis, thus ensuring that all children of every node are generated. After each timestep of the beam

[4] A top-level JOIN is automatically added when necessary to create a single-rooted structure.

[5] Pseudocode is given in the supplementary material.

search, the scores of candidates that violate tree constraints are masked so that they do not proceed forward. By removing candidates that violate the constraints early in the beam search, we allow the decoder to explore more hypotheses.

Checking these constraints and tracking coverage requires an alignment between the output and input MRs. While the children of JOIN nodes are required to appear in order, child nodes of other discourse relations and dialogue acts can appear in any order, and thus the corresponding input non-terminal is not always uniquely identifiable when an output non-terminal is opened. For this reason, a set of possible alignments is maintained. In particular, when accepting a non-terminal, all possible nodes in the input that it may correspond to are identified and a state is maintained for each possibility. Open states whose constraints are violated are removed from tracking, and a non-terminal is not accepted when no more open states are left. Though in principle the number of open states could grow large, empirically any alignment non-determinism is quickly resolved.

Note that although the algorithm ensures that the output tree structure is compatible with the input structure, it turns out that the model can still occasionally hallucinate content: since the neural model allows all possible token sequences in principle, it sometimes generates word sequences that express a hallucinated slot by simply skipping over the disallowed slot annotation—thereby bypassing the constraints—especially when given an unusual input. These cases are discussed further below.

5 Experiments

In this section, we first describe our baselines, metrics, and implementation details, followed by experimental results and analyses.

5.1 Experimental Setup

Baselines We consider a few Seq2Seq-based baselines in our experiments (we use the open fairseq implementation (Gehring et al., 2017) for all our experiments). All models use an LSTM-based encoder and decoder, with attention.

S2S-FLAT The input is a flat MR (for the E2E dataset, this is equivalent to the original form of the data; for weather, we remove all discourse relations and treat all dialog acts as a single large MR). The output is the raw delexicalized response.

S2S-TOKEN Following Reed et al. (2018), we add three tokens in the beginning of flat input MR (same as S2S-FLAT) to indicate the number of contrasts, joins and number of sentences (dialog acts) to be generated.[6] The output is the raw delexicalized response.

S2S-TREE Same architecture as S2S-FLAT, but the input and output for this model are the linearized tree-structured MR and the tree-structured response respectively.

S2S-CONSTR Our proposed model. It has the same architecture as S2S-TREE, but decoding during beam search is constrained, as described in Section 4.2.

Data preprocessing In the input MR, all arguments within each dialog act are ordered alphabetically, to ensure a consistent ordering across examples. We also use alignments between the reference and the MR to **filter** information (arguments or dialog acts/discourse relations) that are not expressed in the reference; however, we ensure that any arguments that occur multiple times in the MR, but are elided in the reference for redundancy, are still preserved in the MR. This ensures that the model doesn't have to learn content selection, while still achieving our primary goal of discourse structure control.

The inputs to S2S-FLAT and S2S-TOKEN are prepared by removing all dialog act and discourse information in the linearized MR, and numbering arguments corresponding to the dialog act they belong in. Global order of dialog acts is preserved such that arguments of the first act occur before those arguments in the following acts, but arguments within a dialog act are ordered alphabetically.

Metrics We consider *automatic* and *human evaluation* metrics for our model. Automatic metrics are evaluated on the raw model predictions (which have delexicalized fields, like `temp_low`):

- **Tree accuracy** is a novel metric that we introduce for this problem. It measures whether the tree structure in the prediction matches that of the input MR exactly. We implemented our tree accuracy metric to account for grouping and ellipsis, and will release this implementation along with our dataset.

[6] Reed et al. only report results on controlling CONTRAST using an augmented training set, precluding direct comparison to their results.

- **BLEU-4** (Papineni et al., 2002) is a word-overlap metric commonly used for evaluating NLG systems.

Due to the limitations of automatic metrics for NLG (Novikova et al., 2017; Reiter, 2018), we also performed human evaluation studies by asking annotators to evaluate the quality of responses produced by different models. Annotators provided **binary** ratings on the following dimensions:

- **Grammaticality**: Measures *fluency* of the responses. Our evaluation guidelines included considerations for proper subject-verb agreement, word order, repetition, and grammatical completeness.
- **Correctness**: Measures *semantic correctness* of the responses. Our guidelines included considerations for sentence structure, contrast, hallucinations (incorrectly included attributes), and missing attributes. We asked annotators to evaluate model predictions against the reference (rather than the MR — see Appendix F).

5.2 Constrained Decoding Analysis

We trained each of the models described above on the weather dataset and the E2E dataset, and evaluated automatic metrics on the test set.[7] In the E2E test set, each flat MR has multiple references (and therefore multiple compositional MRs). When computing BLEU scores for the token, tree, and constrained models, we generated one hypothesis for each of the compositional MRs for a single flat MR, and chose the hypothesis with the highest score against all references for that flat MR. We then computed corpus BLEU using these hypotheses. While this isn't an entirely fair way to evaluate these models against the E2E systems, it serves as a sanity check to validate that generation models provided with more semantic information about the references can achieve better BLEU scores against them. For both E2E and weather, we also filtered out, from all model computations, any examples where S2S-CONSTR failed to generate a valid response (5.3).

For human evaluation, we show an overall correctness measure **`Corr`** measured on the full test sets, as well as **`Disc`**, measured on a more challenging subset of the test set that we selected. For the E2E dataset, we chose examples that contained contrasts by identifying references with a *but* (230 total). For the weather dataset, we chose 400 examples where the MR has at least one CONTRAST or JUSTIFY. We also included test examples with argument type combinations previously unseen in the training set (313 total); we expect these to be challenging for all models, and in particular for the flat model, which has to infer the right discourse relation for new combinations of arguments.

5.3 Results

Table 7 shows the results of this experiment. On both the E2E and weather datasets, S2S-CONSTR improves tree accuracy significantly (using McNemar's chi-squared test) over S2S-TREE. Human evaluation metrics also show that models that are aware of the tree-structured MR (S2S-TREE and S2S-CONSTR) perform significantly better on correctness measures than S2S-TOKEN, which is only aware of the presence or absence of discourse relations, and significantly better than S2S-FLAT, which has no awareness of the structure. The gap is larger on **`Disc`**: the flat model gets only 31% of the challenging cases correct on the E2E dataset, while the constrained model's accuracy is more than twice that. A similar gap is evident in the weather dataset. Further, S2S-CONSTR, S2S-TREE, and S2S-TOKEN all show significant improvements in BLEU over the flat baseline. These systems also outperform the E2E baseline TGEN (Dušek and Jurcıcek, 2016) and the challenge winner SLUG (Juraska et al., 2018) on BLEU (0.6519 and 0.6693 respectively, from Dušek et al. (2019)) and diversity metrics (Section 5.4). We note that for the E2E dataset, the BLEU score increases observed with the tree-based models are not statistically significant compared to S2S-TOKEN. We think this may be partly because many discourse patterns are correlated with the flat MR structure in the E2E dataset (e.g. `family-friendly` and `highly rated` are frequently CONTRASTed). By contrast, BLEU score increases are statistically significant for all models on our weather dataset. Also, S2S-CONSTR fails to generate any valid candidates for ~1.5% of the weather test examples. In most of these cases, the model *stutters*, i.e. produces degenerate output like "will be be be . . . ". We suspect that in these cases, the imposed decoding constraints cause the Seq2Seq decoder to get stuck in a pseudoterminal state.

Grammaticality seems to drop slightly for the tree-based models on the weather dataset, but not

[7] We used the scripts provided at `https://github.com/tuetschek/e2e-metrics` by the E2E organizers for evaluating both the E2E and the weather models.

Model	E2E					Weather				
Metric	BLEU	TreeAcc	Gram	Corr	Disc	BLEU	TreeAcc	Gram	Corr	Disc
S2S-Flat	0.6360	-	94.03	63.85	30.87	0.7455	-	98.77	77.09	79.04
S2S-Token	0.7441‡	-	92.29	69.02†	42.29†	0.7493*	-	96.7	81.56†	83.93†
S2S-Tree	0.7458‡	94.86	93.59	83.85†	54.35†	0.7612*	92.5	95.26	87.61†	85.97†
S2S-Constr	**0.7469‡**	**99.25**	**94.33**	**85.89†**	**66.09†**	**0.7660***	**96.92**	95.30	**91.82†**	**93.44†**

Table 7: Automatic and human evaluated metrics on E2E and Weather datasets. All metrics other than BLEU are percentages. `Corr` and `Disc` are the % of examples for which the model prediction was judged by humans as semantically correct; `Disc` is measured on a challenging subset of `Corr`. * indicates BLEU scores that are statistically significant ($p < 0.01$) compared to *all* baselines for that model. ‡indicates statistically significant BLEU scores ($p < 0.01$) compared to S2S-Flat. †indicates human-evaluated correctness scores that are statistically significant ($p < 0.05$), using McNemar's chi-squared test, compared to *all* baselines for that model.

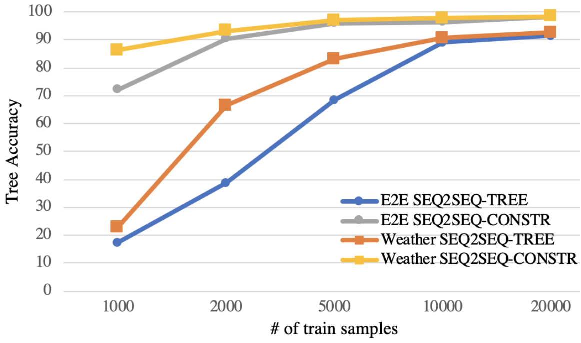

Figure 2: Performance of models on test set for varying number of samples in train set.

on the E2E dataset. One hypothesis from this and the correctness numbers is that the flat models generate more generic (and therefore grammatical), but also incorrect, responses, compared to the tree-based models. We also note that there's a noticeable gap in the E2E dataset between tree accuracy and the correctness numbers from human evaluation. We analyzed 35 examples where our tree accuracy metric disagreed with human evaluation, and found 22 (63%) cases where the compositional MR was missing information in the reference, seemingly due to noise in our automatic annotation process (Section 3.2). We also identified 6 cases (17%) of annotator confusion (for example whether *between £20-30* implies the same meaning as *moderately priced*), sometimes caused by noisy references that contained additional information. The remaining examples all contained legitimate model errors, like content hallucination, or a wrong slot being produced despite a correct non-terminal. One future direction to get more reliable metrics would be to improve the automatic annotation process in Section 3.2 to eliminate noise and flag noisy references. Further experimentation is described in Appendix E.

5.4 Diversity Metrics

We report the diversity metrics used for evaluating E2E challenge submissions in Dušek et al. (2019) (# unique tokens, # unique trigrams, Shannon token entropy (Manning and Schütze, 1999, p.61ff.), conditional bigram entropy (Manning and Schütze, 1999, p.63ff.)). Table 8 shows these numbers, as compared against a few of the E2E participating systems, TGen, SLUG, and ADAPT (Elder et al., 2018). All of the models with enriched semantic representations — S2S-Token, S2S-Tree, and S2S-Constr — show higher diversity than neural baselines without diversity considerations. Combined with our improved BLEU scores, this seems to indicate that adding discourse relation information to input MRs can increase diversity, without incurring losses on automatic metrics (as is the case with the diversity-promoting ADAPT system).

5.5 Data Efficiency and Generalizability

We measured tree accuracy on the full E2E and weather test sets by varying the number of training samples for S2S-Tree and S2S-Constr (Figure 2). S2S-Constr achieves more than 90% tree accuracy with just 2K samples and more than 95% with 5K samples on both datasets, suggesting that constrained decoding can help achieve superior performance with much less data.

Meanwhile, we also investigated the extent to which tree-structured MRs could allow models to generalize to compositional semantics (Figure 3). We first split the complete E2E training set into flat and compositional examples (26896 vs. 24530), where flat examples don't contain any discourse relations. Next, we trained a model on the full weather dataset and flat E2E data, gradually added more compositional E2E samples to the training set, and checked the model's accuracy on a test set

Model	Unique tokens	Unique trigrams	Shannon entropy	Cond. entropy bigrams
TGEN	83	597	5.41	1.32
SLUG	74	507	5.35	1.13
ADAPT	455	3567	6.18	2.09
S2S-TOKEN	137	1147	5.86	1.71
S2S-TREE	134	1030	5.85	1.65
S2S-CONSTR	134	1128	5.86	1.71

Table 8: E2E dataset diversity metrics. Rows in gray correspond to metrics that we cite from Dušek et al. (2019).

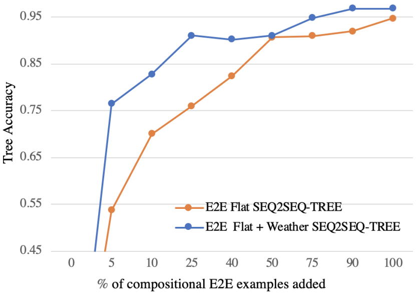

Figure 3: Performance of S2S-TREE models trained on E2E flat data, and flat E2E + full weather dataset, with a fraction of composition E2E.

with only compositional examples. Without any compositional E2E examples, both models fail to produce any valid sequences (not pictured). However, when just 5% of the compositional examples are added to the training data, the E2E-WEATHER model gets a tree accuracy of 76%, while the model trained on E2E only gets 53.72%. The final E2E-WEATHER model also has higher overall accuracy than the E2E-only model. This shows that learned discourse relations can be leveraged for domain adaptation.

6 Related Work

Reed et al.'s (2018) approach to enriching the input, discussed earlier, is the most closely related work to ours. A more recent work by Moryossef et al. (2019) also focuses on exercising more control over input structures through sentence plans; however, their work doesn't touch on discourse relations or constrained decoding. Puduppully et al. (2018) builds a modular end-to-end neural architecture that performs content planning in addition to realization, although they focus on generating text from structured tables, and don't consider discourse structure.

Also related is Kiddon et al.'s (2016) neural checklist model, which tracks the coverage of an input list of ingredients when generating recipes. Our constrained decoding approach goes beyond covering a simple list by enforcing constraints on ordering and grouping of tree structures, but theirs takes coverage into account during model training. A more direct inspiration for our approach is the way coverage has been traditionally tracked in grammar-based surface realization (Shieber, 1988; Kay, 1996; Carroll et al., 1999; Carroll and Oepen, 2005; Nakanishi et al., 2005; White, 2006; White and Rajkumar, 2009). Compared to our approach, grammar-based realizers can prevent hallucination entirely, though at the expense of developing an explicit grammar. Constrained decoding in MT (Post and Vilar, 2018, i.a.) has been used to enforce the use of specific words in the output, rather than constraints on tree structures. Also related are neural generators that take Abstract Meaning Representations (AMRs) as input (Konstas et al., 2017, i.a.) rather than flat inputs; these approaches, however, do not generate trees or use constrained decoding.

7 Conclusions

We show that using rich tree-structured meaning representations can improve expressiveness and semantic correctness in generation. We also propose a constrained decoding technique that leverages tree-structured MRs to exert precise control over the discourse structure and semantic correctness of the generated text. We release a challenging new dataset for the weather domain and an enriched E2E dataset that include tree-structured MRs. Our experiments show that constrained decoding, together with tree-structured MRs, can greatly improve semantic correctness as well as enhance data efficiency and generalizability.

Acknowledgments

We thank Juraj Juraska for supplying the output of their E2E slot tagger. We also thank Michael Auli, Alex Baevski and Mike Lewis for discussion.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Giuseppe Carenini and Johanna D. Moore. 2006. Generating and evaluating evaluative arguments. *Artificial Intelligence*, 170:925–952.

John Carroll, Ann Copestake, Dan Flickinger, and Victor Poznański. 1999. An efficient chart generator for (semi-) lexicalist grammars. In *Proc. EWNLG-99*.

John Carroll and Stefan Oepen. 2005. High efficiency realization for a wide-coverage unification grammar. In *Proc. IJCNLP-05*.

Vera Demberg, Andi Winterboer, and Johanna D Moore. 2011. A strategy for information presentation in spoken dialog systems. *Computational Linguistics*, 37(3):489–539.

Ondřej Dušek and Filip Jurcicek. 2016. Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 45–51. Association for Computational Linguistics.

Ondrej Dušek and Filip Jurcıcek. 2016. Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings. In *The 54th Annual Meeting of the Association for Computational Linguistics*, page 45.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2018. Findings of the E2E NLG challenge. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 322–328. Association for Computational Linguistics.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2019. Evaluating the state-of-the-art of end-to-end natural language generation: The E2E NLG Challenge. *arXiv preprint arXiv:1901.11528*.

Henry Elder, Sebastian Gehrmann, Alexander O'Connor, and Qun Liu. 2018. E2E NLG challenge submission: Towards controllable generation of diverse natural language. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 457–462.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional Sequence to Sequence Learning. *ArXiv e-prints*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Juraj Juraska, Panagiotis Karagiannis, Kevin K. Bowden, and Marilyn A. Walker. 2018. A deep ensemble model with slot alignment for sequence-to-sequence natural language generation. In *NAACL-HLT*.

Martin Kay. 1996. Chart generation. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pages 200–204, Santa Cruz, California, USA. Association for Computational Linguistics.

Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. 2016. Globally coherent text generation with neural checklist models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 329–339. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia. Association for Computational Linguistics.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural AMR: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–157, Vancouver, Canada. Association for Computational Linguistics.

Oliver Lemon, Johanna Moore, Mary Ellen Foster, and Michael White. 2004. Generating tailored, comparative descriptions in spoken dialogue. In *Proc. of FLAIRS*. AAAI.

Percy Liang, Michael I Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 91–99. Association for Computational Linguistics.

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

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA.

Hongyuan Mei, Mohit Bansal, and R. Matthew Walter. 2016. What to talk about and how? selective generation using lstms with coarse-to-fine alignment. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 720–730. Association for Computational Linguistics.

Johanna D. Moore and Cécile Paris. 1993. Planning text for advisory dialogues: Capturing intentional and rhetorical information. *Computational Linguistics*, 19:651–694.

Amit Moryossef, Yoav Goldberg, and Ido Dagan. 2019. Step-by-step: Separating planning from realization in neural data-to-text generation. *arXiv preprint arXiv:1904.03396*.

Hiroko Nakanishi, Yusuke Miyao, and Jun'ichi Tsujii. 2005. Probabilistic methods for disambiguation of an HPSG-based chart generator. In *Proc. IWPT-05*.

Jekaterina Novikova, Ondrej Dusek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for NLG. In *EMNLP*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. ACL-02*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *NAACL-HLT*.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2018. Data-to-text generation with content selection and planning. *arXiv preprint arXiv:1809.00582*.

Owen Rambow, Srinivas Bangalore, and Marilyn Walker. 2001. Natural language generation in dialog systems. In *Proceedings of the First International Conference on Human Language Technology Research*, HLT '01, pages 1–4, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lena Reed, Shereen Oraby, and Marilyn Walker. 2018. Can neural generators for dialogue learn sentence planning and discourse structuring? In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 284–295. Association for Computational Linguistics.

Ehud Reiter. 2018. A structured review of the validity of BLEU. *Computational Linguistics*, 44:393–401.

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

Stuart Shieber. 1988. A uniform architecture for parsing and generation. In *Proc. COLING-88*.

Charese Smiley, Elnaz Davoodi, Dezhao Song, and Frank Schilder. 2018. The E2E NLG challenge: A tale of two systems. In *INLG*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Amanda Stent, Marilyn A. Walker, Steve Whittaker, and Preetam Maloor. 2002. User-tailored generation for spoken dialogue: an experiment. In *INTERSPEECH*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700.

Marilyn Walker, Amanda Stent, Francois Mairesse, and Rashmi Prasad. 2007. Individual and domain adaptation in sentence planning for dialogue. *Journal of Artificial Intelligence Research (JAIR)*, 30:413–456.

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, M. Lina Rojas-Barahona, Pei-Hao Su, David Vandyke, and Steve Young. 2016. Multi-domain neural network language generation for spoken dialogue systems. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 120–129. Association for Computational Linguistics.

Tsung-Hsien Wen, Milica Gasic, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1711–1721. Association for Computational Linguistics.

Michael White. 2006. Efficient realization of coordinate structures in combinatory categorial grammar. *Research on Language and Computation*, 4(1):39–75.

Michael White, Robert A. J. Clark, and Johanna D. Moore. 2010. Generating tailored, comparative descriptions with contextually appropriate intonation. *Computational Linguistics*, 36(2):159–201.

Michael White and Rajakrishnan Rajkumar. 2009. Perceptron reranking for CCG realization. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 410–419, Singapore. Association for Computational Linguistics.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263. Association for Computational Linguistics.

Sam Wiseman, Stuart M. Shieber, and Alexander M. Rush. 2018. Learning neural templates for text generation. In *EMNLP*.

A Weather Forecast Generation

For every example, we extracted the date range requested by the user, and generated artificial weather forecasts for that date range. We generated forecasts of different granularities (hourly or daily) depending on the date requested by the user. If the date that requested was less than 24 hours after the "reference" date in the synthetic user context, we generated hourly forecasts; otherwise, we generated the required number of daily forecasts. To generate forecasts, we selected reasonable mean, standard deviation, min, and max values for temperature and cloud coverage, and used these to sample temperatures for every point in the date range. We also selected random sunrise and sunset times for each day present in the range. We picked values that seemed reasonable, but didn't try too hard to get precise values, since our focus was more on using the forecasts to create complex MRs. After sampling temperatures and cloud coverage amounts for each range, we randomly chose other attributes to include, conditioned on the values of the temperatures and cloud coverage, like precipitation chance, wind speed summary, and other rarer conditions like fog.

B Tree-Structured Weather MR Creation

1. Errors: We added `ERROR` dialog acts whenever the user query contained a weather request for a date too far in the future. We also chose locations to treat as "unknown" randomly, thus adding errors for locations unknown to the system. These ERROR acts are interesting because they capture domain-specific information about the nature and cause of errors, and can potentially be learned across domains. Additionally, including ERROR acts creates scope for interesting responses like "I'm sorry, I don't know where that is. But right now in [user's default location], it's sunny ...".

2. Aggregation: We identified dates that had similar weather attributes (precipitation, cloud coverage, etc.) and created `INFORM` dialog acts that expressed information regarding each date. We then grouped these acts together using a `JOIN` discourse relation.

3. Contrast: We identified attributes that were in opposition ("cloudy" vs. "sunny") and added a parent CONTRAST discourse relation to any such dialog acts. We also contrasted related attributes wherever possible; e.g. the cloud coverage value "sunny" can be contrasted with both "cloudy" and the precipitation type "rain".

4. Yes/no questions: Whenever the user query was a boolean one ("Will it rain tomorrow"), we added `YES` or `NO` dialog acts as appropriate.

5. Justifications/Recommendations: Whenever the user query mentioned an attire or activity ("Should I wear a raincoat tomorrow?"), we assumed that the MR should communicate a recommendation as well as a justification for it ("No, you don't need to wear one tomorrow, it looks like it'll be sunny all day"). In these cases, we added a `RECOMMEND` dialog act, and an `INFORM` dialog act that provides the justification for the recommendation. We added a parent `JUSTIFY` discourse relation to these acts, treating the recommendation as the nucleus and the `INFORM` as the satellite of the justification.

C Dataset Creation Quality

As mentioned in 3, we asked annotators to provide evaluations of collected responses, and used these to filter out noisy references and annotations from our final dataset. The ratings were provided on a 1-5 scale and double annotated, and we filtered out 3,404 examples (out of a total 37,162) that had a score less than 3 on any of the four dimensions: fluency, correctness, naturalness, annotation correctness.

D Data Preprocessing

Infinitely-valued arguments such as names of restaurants, dates, times, and locations such as cities, states are delexicalized (value is replaced by placeholder tokens) in both the input and output of models. This was done following the approach taken by several of the systems in the E2E challenge (Dušek and Jurcıcek, 2016; Juraska et al., 2018; Dušek et al., 2019). The reasoning behind this is that the values of such arguments are often inserted verbatim in the response text, and therefore do not affect the final surface form realization. Replacing these arguments in both the input and output reduces the vocabulary size and prevents sparsity issues. (A copy mechanism, such as the one introduced in Vinyals et al. (2015), can be used to address this, though we did not explore this approach in this work.) The full list of arguments for which we performed delexicalization is:

1. Numerical arguments: temperature-related arguments, precipitation chance, day, month, year (for dates).

2. Named entities: restaurant name (E2E), landmark (E2E), city, region, country, weekday (for dates)

E Additional Experiments

We also experimented with a reranked S2S-TREE in which the beam search candidates are reranked for tree accuracy. This yields a tree accuracy of 97.6% and 95.4% on E2E and weather.

We trained a Recurrent Neural Network Grammar (RNNG) to tag slots in the prediction of S2S-CONSTR in order to filter out hallucinations. The correctness on filtered test sets rose from 85.89% to 87.44% for E2E, and from 91.82% to 93.84% on weather.

F Human Evaluation of Models

When asking annotators to rate the models on correctness, we asked them to rate the response by comparing it against the reference, rather than against the MR. This adds the risk that annotators are confused by noisy references, but we found that it increased annotation speed and agreement rates significantly over evaluating against the MR directly. This is also because our MRs are tree-structured and can be hard to read. We performed double-annotation with a resolution round. **Automatic rejection:** When analyzing evaluation results, we found that it was fairly easy to miss the absence of a contrast or a justification in our weather dataset, especially since our dataset is so large. As a result, annotators were marking several incorrect cases as correct. To address this issue, we automatically marked as incorrect any examples where the MR had a CONTRAST but the response lacked any contrastive tokens, or where the MR has a JUSTIFY but the response lacked any clear markers of a justification. This eliminated noise from 2.8% of all responses.

G Model Training Details

We used the same seq2seq model from the S2S-FLAT baseline for our constrained decoding experiments, which used 300-dimensional GloVe word embeddings (Pennington et al., 2014), a dropout rate of 0.2 (Srivastava et al., 2014), and hidden dimension of 128 in both the encoder and the decoder. We used the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 0.002 to train the seq2seq model. The learning rate is reduced by a factor of 5 if the validation loss stops decreasing. Beam size is set to 10.

H Constrained Decoding Algorithm

```
def build_constraints(MR):
    # nodes in MR are numbered from 0 to n in order of their
    # discovery in depth−first−search.
    # example, for MR: [JOIN [INFORM [A ] [B ] ] [INFORM [B ] [D ]]]
    # ids: JOIN: 0, INFORM: 1, A: 2, B: 3, INFORM: 4, B: 5, D: 6
    for node in MR:
        parent_map[node.id] = node.parent
        children_map[node.id] = node.children
        # map from non−terminal to all node ids of the non−terminal
        # eg: INFORM −> {1, 4} in case of example MR above
        valid_non_terminal_nodes[node.non_terminal].add(node.id)
    # map from node id to nodes that can cover it through ellipsis
    # example, for above MR: {3: {3, 5}, 5: {3, 5}}
    ellipsis_options = compute_ellipsis_options(MR)
    init_state.parent = −1
# current parent
    init_state.coverage = {}
# tracks node ids encountered till now
    # track nodes that have been covered through ellipsis
    init_states.elided_nodes = {}
    states = [init_state]
# list of open states

def children_covered(state, node):
    # returns true if all nodes have covered either
    # directly or through ellipsis
    missing_children = children_map[state.parent] − state.coverage
    for missing_child in missing_children:
        if (ellipsis_options[missing_child]
              − state.elided_nodes) is empty:
            # nodes that have been elided themselves
            # can’t cover other nodes through ellipsis
            return False
    return True

def accept_token(states, next_token):
    # move states one time−step forward by accepting next_token
    # returns False if next_token cannot be accepted by any state
    if not next_token.startswith(”[”) or next_token != ”]”:
        # only non−terminal tokens need to be checked
        return True
    updated_states = []
    for state in states:
        if next_token.startswith(”[”):
            for candidate in valid_non_terminal_nodes[next_token]:
                if candidate in children_map[state.parent]
                      and candidate not in state.coverage:
                    # create a new state for each valid candidate
                    new_state = copy(state)
                    new_state.parent = candidate
                    new_state.coverage.add(candidate)
                    updated_states.append(new_state)
        elif next_token == ”]”
                and children_covered(state, state.parent):
            # accept closing brace for current node and
            # move states up a level in tree
            new_state = copy(state)
            new_state.parent = parent_map[state.parent]
            missing_children =
                  children_map[state.parent] − state.coverage
            # if we’re accepting a closing node with missing children,
            # then all of them must be getting elided
            new_state.elided_nodes.add(missing_children)
            updated_states.append(update(new_state, next_token))
    states = updated_states
    return len(states) > 0

def mask_score(score, states, next_token):
    if accept_token(states, next_token):
        return score
    else:
        return 0
```