Template Guided Text Generation for Task-Oriented Dialogue

Mihir Kale
Google
Mountain View
mihirkale@google.com

Abhinav Rastogi
Google Research
Mountain View
abhirast@google.com

Abstract

Virtual assistants such as Google Assistant, Amazon Alexa, and Apple Siri enable users to interact with a large number of services and APIs on the web using natural language. In this work, we investigate two methods for Natural Language Generation (NLG) using a single domain-independent model across a large number of APIs. First, we propose a schema-guided approach which conditions the generation on a schema describing the API in natural language. Our second method investigates the use of a small number of templates, growing linearly in number of slots, to convey the semantics of the API. To generate utterances for an arbitrary slot combination, a few simple templates are first concatenated to give a semantically correct, but possibly incoherent and ungrammatical utterance. A pre-trained language model is subsequently employed to rewrite it into coherent, natural sounding text. Through automatic metrics and human evaluation, we show that our method improves over strong baselines, is robust to out-of-domain inputs and shows improved sample efficiency. [1]

1 Introduction

Virtual assistants have become popular in recent years and task-completion is one of their most important aspects. These assistants help users in accomplishing tasks such as finding restaurants, buying sports tickets, finding the weather etc., by providing a natural language interface to many services or APIs available on the web. Most systems include a natural language understanding and dialogue state tracking module for semantic parsing of the dialogue history. This is followed by a policy module which interacts with the APIs, whenever required, and generates the actions to be taken by the system to continue the dialog. In the end, the Natural Language Generation (NLG) module converts these actions into an utterance, which is surfaced to the user. Being the user-facing interface of the dialogue system, NLG is one of the most important components impacting user experience.

Traditional NLG systems heavily utilize a set of templates to produce system utterances. Although the use of templates gives good control over the outputs generated by the system, defining templates becomes increasingly tedious as more APIs are added. Supporting multi-domain conversations spanning multiple APIs quickly grows out of hand, requiring expert linguists and rigorous testing to ensure the grammatical correctness and appropriateness of generated utterances. Consequently, data-driven generative approaches have gained prominence. Such systems require much less effort and can generate utterances containing novel patterns. Meanwhile, with the rapid proliferation of personal assistants, supporting large number of APIs across multiple domains has become increasingly important, resulting in research on supporting new APIs with few labelled examples (few-shot learning). To this end, generative models pre-trained on large amounts of unannotated text have been increasingly successful.

In this work, we address the challenges of joint modeling across a large number of domains, and data efficient generalization to new domains and APIs for NLG. Our contributions are the following:

1. We propose two methods for zero-shot and few-shot NLG. Our first method, the Schema-Guided NLG, represents slots using their natural language descriptions. Our second method - Template Guided Text Generation (T2G2) employs a simple template-based representation of system actions and formulates NLG as an utterance rewriting task (Figure 1).

[1] Our code and data is available at github.com/google-research/schema-guided-dialogue

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 6505–6520,
November 16–20, 2020. ©2020 Association for Computational Linguistics

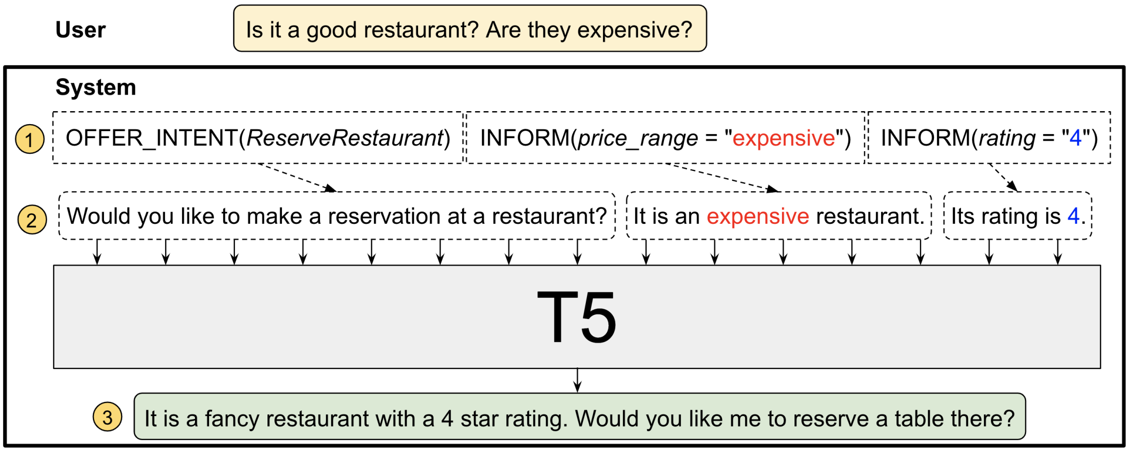

User
Is it a good restaurant? Are they expensive?
System
1
OFFER_INTENT(*ReserveRestaurant*)
INFORM(*price_range* = "expensive")
INFORM(*rating* = "4")
2
Would you like to make a reservation at a restaurant?
It is an expensive restaurant.
Its rating is 4.
T5
3
It is a fancy restaurant with a 4 star rating. Would you like me to reserve a table there?

Figure 1: Overall architecture of our proposed template guided approach. 1. The policy module outputs a set of actions in response to the user utterance. 2. Simple templates convert each action into a natural language utterance. 3. Template-generated utterances are concatenated and fed to a T5 encoder-decoder model(Raffel et al., 2020). The model rewrites it to a conversational response surfaced to the user.

2. We present the first NLG results on the Schema-Guided dialogue dataset (Rastogi et al., 2019), which exceeds all other datasets in scale, providing a total of 45 APIs over 20 domains. While the current state-of-the-art pre-training based methods struggle to generalize to unseen (zero-shot) APIs, our proposed methods are robust to out-of-domain inputs and display improved sample efficiency.

3. We conduct an extensive set of experiments to investigate the role of dialogue history context, cross-domain transfer learning and few-shot learning. We share our findings to guide the design choices in future research.

2 Related Work

Natural language generation from structured input (NLG) has been an active area of research, facilitated by creation of datasets like WikiBio (Lebret et al., 2016), E2E challenge (Novikova et al., 2017), WebNLG (Gardent et al., 2017) and MultiWOZ (Budzianowski et al., 2018). Neural sequence models have been extensively used in a variety of configurations for NLG in dialogue systems. Wen et al. (2017) proposed a two-step approach: first generating a delexicalized utterance with placeholders for slots and then post-processing it to replace placeholders with values from API results, whereas Nayak et al. (2017) highlighted the importance of conditioning responses on slot values.

Sequence to sequence architectures directly converting a sequential representation of system actions to a system response are also very common (Wen et al., 2015; Du sek and Jurcicek, 2016b; Zhu et al., 2019; Chen et al., 2019). Domain-adaptation and transfer learning in low resource settings has also been an extensively studied problem (Tran and Le Nguyen, 2018; Chen et al., 2020; Peng et al., 2020; Mi et al., 2019), with recently released datasets like SGD (Rastogi et al., 2019) and FewShotWOZ (Peng et al., 2020) providing a good benchmark. Meanwhile, language models pre-trained on large amount of unannotated text corpus have achieved state-of-the-art performance across several natural language processing tasks (Devlin et al., 2019; Yang et al., 2019; Liu et al., 2019; Radford et al., 2019; Keskar et al., 2019), including natural language generation (Peng et al., 2020; Kale and Roy, 2020).

Our template based approach bears similarities to sentence fusion (Barzilay and McKeown, 2005), and prototype based text editing (Hossain et al., 2020; Cao et al., 2018; Guu et al., 2018; Wu et al., 2019). However, none of these works tackle text generation from structured data.

3 Model

For a given system dialogue turn, let $\mathcal{A} = \{d_i(s_i = v_i)\}_{i=1}^{A}$ be the set of actions which are produced by the system, where A is the total number of actions for this turn. Each action consists of a single dialogue act d_i representing the semantics of the action, along with optional slot and value parameters - s_i and v_i respectively. For example, *inform*, *req_more* and *request* are some of the dialogue acts

Approach	Representation of System Actions
Naive	inform (restaurant = Opa!) inform (cuisine = greek)
Schema Guided	inform (name of restaurant = Opa!) inform (type of food served = greek)
Template Guided	How about the restaurant Opa!. The restaurant serves greek food.
Ground Truth	Opa! is a nice greek restaurant. How does it sound?

Figure 2: An example showing the representation of system actions utilized by the three schemes. The template representation is generated by concatenating sentences obtained from two templates, which are "*inform(restaurant = $x) → How about the restaurant $x.*" and "*inform(cuisine = $x) → The restaurant serves $x food.*".

defined in the SGD dataset (Rastogi et al., 2019), which are used for informing the value of a slot to the user, asking if the user needs some other help, and requesting the value of a slot from the user respectively. Some acts like *inform* require both the slot and value parameters, whereas acts like *request* require the slot parameter only and acts like *req_more* require none. Some datasets allow multiple slot-value arguments for a single act, but such actions can generally be converted to the above representation by decomposing them into multiple actions with the same act, each containing exactly one slot-value pair.

The goal of NLG is to translate $\mathcal{A}$ to a natural language response with the same semantic content. To this end, we first convert the set $\mathcal{A}$ into a sequence. Then, we finetune a Text-to-Text Transfer Transformer (T5) (Raffel et al., 2020) model, which is a pre-trained sequence to sequence transformer, to generate the natural language response using this sequence as input. Now, we present three different methods for converting $\mathcal{A}$ into a sequence, the last two being our contributions. They are also summarized in Figure 2.

3.1 Naive Representation

This approach uses the most basic representation of actions, similar to that used in many prior works (Novikova et al., 2017; Zhu et al., 2019; Peng et al., 2020). Canonical representations of each action - $a_i, a_i(s_i)$ or $a_i(s_i = v_i)$, depending on the parameters present in the action, are concatenated together to obtain a sequence representation of $\mathcal{A}$. Although this representation is simple to obtain and gives state of the art results for several data-to-text benchmarks (Kale and Rastogi, 2020), it suffers from two drawbacks -

(i) **Semantics -** This representation doesn't convey much information about the semantics of a slot. Consequently, the model may need a larger number of training examples to identify the semantics of a slot from its usage in the system utterances in the training data.

(ii) **Representation Bias -** This representation is very different from what the encoder has seen during pre-training phase, which is natural language text. As a result, the representations learnt during pre-training may not transfer well. Peng et al. (2020) mitigate this by conducting additional pre-training using large scale annotated dialogue datasets. While this method is effective, a large in-domain corpus may not always be available.

3.2 Schema Guided Representation

Recent work on low-resource natural language understanding tasks have used natural language descriptions of slots. These descriptions are easy to obtain, directly encode the semantics of the slot and have been shown to help when in-domain training data is sparse. While description based representations have become popular for tasks like spoken language understanding (Bapna et al., 2017) and dialogue state tracking (Rastogi et al., 2019), they have not yet been applied to the language generation task. We propose an extension of the Naive representation by replacing the slot names with their natural language descriptions. The action representations, as illustrated in Figure 2, are $a_i, a_i(desc(s_i))$ and $a_i(desc(s_i) = v_i)$, where $desc(s)$ represents a natural language description of slot s. This solves the first drawback of the Naive representation mentioned above.

3.3 Template Guided Representation

We solve the representation bias problem by converting the set of actions output by the system into a natural language utterance. We employ a technique similar to that used in Rastogi et al. (2019), where simple utterances are generated using a minimal set of manually defined templates. Specifically, as shown in Figure 3, we define one template for each

Action	Template
notify_success	Your ride is booked and the cab is on its way.
goodbye	Have a safe ride!
request(dest)	Where are you riding to?
request(shared)	Are you comfortable sharing the ride?
confirm(dest=\$x*)*	You are going to \$x.
inform(fare=\$x*)*	Your ride costs \$x dollars.
inform(seats=\$x*)*	The cab is for \$x riders.

Figure 3: Example templates for a ride-sharing API. Parameterized templates are defined for actions which contain a slot value.

Statistic	E2E	MWoz	SGD
Domains	1	7	20
Unseen domains	0	0	4
System acts	1	7	10
Slots	8	23	184
Unseen slots	0	0	41
Train size	33k	57k	160k
Dev size	4.3k	7.3k	24k
Test size	4.7k	7.3k	42k

Table 1: Comparison of NLG datasets. MWoz is short for MultiWOZ. Train/Dev/Test sizes represent the number of system turns. Unseen domains refers the test set.

system action. The representation of $\mathcal{A}$ is obtained by concatenating the corresponding templatized representation of each action in $\mathcal{A}$. See Figure 2 for a complete example.

Note that, our focus here is not to generate conversational and grammatically correct utterances, but to have a simple representation of the actions, which can be rewritten by the model into a natural and fluent response. Hence, we do not need to cover all edge cases typically required in template based methods - handling of plurals, subject-verb agreement, morphological inflection etc. - and only need to define a small number of templates. For most APIs, this amounts to around 15-30 templates, which can easily be written by the API developer. The actual number varies depending on the number of slots and intents supported by the API [2]. Some special slots like date, time and price are formatted using special rules, which can be reused across APIs. For instance, we convert the date “2019-03-06” to “6th March”, the time “18:40” to “6:40 pm”, and price “60” to “\$60”. We call this step *value paraphrasing*. Since this method relies on a combination of templates and transfer learning from language models, we name it **Template Guided Text Generation (T2G2)**.

4 Experimental Setup

We conduct a series of experiments to compare the three system action representations presented above. We also evaluate NLG in few-shot settings and investigate a few other aspects of the SGD dataset. In each of the experiments reported in this paper, we start with a pre-trained T5-small model[3]. It has 6 layers each in the encoder and decoder, with a total of around 60 million parameters. The model is then fine-tuned on the corresponding dataset using a constant learning rate of 0.001 and batch size of 256 for 5000 steps. The checkpoint yielding the highest BLEU score on the development set is picked for reporting test set results. During inference, we use beam search with a width of 4 and length penalty $\alpha = 0.6$.

5 Action Representations

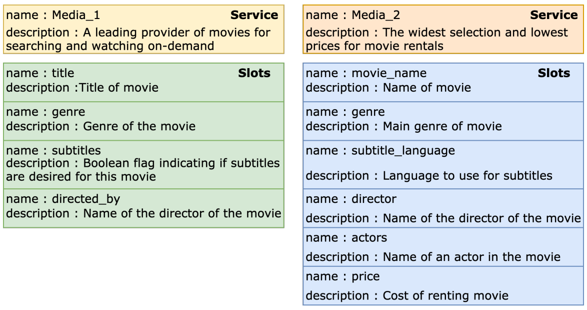
We compare the different methods of action representation on MultiWOZ 2.1 (Budzianowski et al., 2018), the cleaned version of the E2E restaurant corpus (Novikova et al., 2017; Du sek et al., 2019) and the Schema-Guided Dialogue (SGD) (Rastogi et al., 2019) dataset. The SGD dataset features a larger number of domains and slots, and the presence of multiple APIs per domain (Figure 4) makes it representative of practical scale-related challenges faced by today’s virtual assistants. Furthermore, as opposed to the other two datasets, its evaluation sets contain many domains, and consequently slots, which are not present in the training set. Even for domains shared between the training and evaluation sets, the evaluation sets contain additional slots in some cases. This focus on zero-shot generalization to new domains and APIs makes SGD more challenging than existing NLG benchmarks. Table 1 compares these datasets.

5.1 Automatic Evaluation

Following prior work (Wen et al., 2015), we use BLEU (Papineni et al., 2002) and Slot Error Rate

[2] Please see Appendix D for more examples of templates.

[3] github.com/google-research/text-to-text-transfer-transformer



name : Media_1 **Service**
description : A leading provider of movies for searching and watching on-demand

name : title **Slots**
description :Title of movie

name : genre
description : Genre of the movie

name : subtitles
description : Boolean flag indicating if subtitles are desired for this movie

name : directed_by
description : Name of the director of the movie

name : Media_2 **Service**
description : The widest selection and lowest prices for movie rentals

name : movie_name **Slots**
description : Name of movie

name : genre
description : Main genre of movie

name : subtitle_language
description : Language to use for subtitles

name : director
description : Name of the director of the movie

name : actors
description : Name of an actor in the movie

name : price
description : Cost of renting movie

Figure 4: Schemas of two APIs from the Media domain present in the SGD dataset.

Model	BLEU	SER
HDSA (Chen et al., 2019)	26.5	12.14
SC-GPT (Peng et al., 2020)	30.8	**0.53**
Naive	**34.6**	1.27
Schema	33.3	1.89
T2G2	34.4	1.85

Table 2: Performance of models on MultiWOZ.

(SER) (Dušek and Jurcicek, 2019) as automatic metrics. SER represents the fraction of generated texts where at least one slot was not correctly copied from the structured data. Since this metric relies on string matching, we cannot use it to evaluate binary slots like *has_live_music*. Its exact match nature also prevents it from identifying paraphrases of slot values, e.g. *expensive* and *costly*. For E2E we use additional metrics used in prior work for this benchmark - NIST (Doddington, 2002), ROUGE-L (Lin, 2004), METEOR (Lavie and Agarwal, 2007), CIDEr (Vedantam et al., 2015), and BLEU.

MultiWOZ and E2E Table 2 lists results on the MultiWOZ and Table 3 on E2E. We train separate models for each dataset. On both datasets, T2G2 and Schema are comparable to the state-of-the-art Naive approach. We note that the SER score on MultiWOZ is slightly worse in comparison with SC-GPT. SC-GPT generates 5 predictions for each input and then ranks them based on the SER score itself. On the other hand, we generate a single output, on which SER is evaluated. Overall, the results indicate that with enough annotated data, the Naive approach is enough to attain good performance. Both datasets are large and feature limited variety (MultiWOZ has 57K utterances spread over just 5 domains, while E2E has 33k utterances spread over just 8 slots). Zero-shot and few-shot settings offer a greater and more realistic challenge, and we explore these settings next. The SGD dataset, which spans 20 domains, enables us to study these settings.

Model	BLEU	N	M	R	C
SC-LSTM	23.7	4.0	32.9	39.3	0.4
TGen	40.7	6.2	37.8	56.1	**1.9**
Naive	42.1	**6.4**	38.5	56.2	**1.9**
Schema	**43.1**	**6.4**	**38.7**	56.8	**1.9**
T2G2	42.5	**6.4**	**38.7**	**56.9**	**1.9**

Table 3: Performance of models on E2E. Results for SC-LSTM (Wen et al., 2015) and TGen (Novikova et al., 2017) have been taken from Du sek et al. (2019). N,M,R,C stand for NIST, METEOR, ROUGE and CIDEr respectively.

BLEU	**Naive**	**Schema**	**T2G2**	**Copy**
Unseen	14.9	15.8	**22.2**	16.1
Seen	27.7	27.5	**29.4**	19.2
Overall	26.2	26.2	**28.6**	18.8
SER	**Naive**	**Schema**	**T2G2**	**Copy**
Unseen	0.7	0.4	**0.0**	-
Seen	1.1	0.8	**0.4**	-
Overall	1.0	0.8	**0.4**	-

Table 4: BLEU and SER metrics on SGD dataset. *Copy* refers to a trivial baseline comprising of the template based input representation and has 0 SER by definition.

Adaptation to New Domains The ideal NLG model should be able to handle domains it was not exposed to during training. The SGD dataset, which features unseen domains in the evaluation sets, lets us us assess the zero-shot capability of NLG systems. We report results in Table 4 on two test sets - the *seen* set consists of domains that were seen during training, while the *unseen* set consists of brand new domains aka the zero-shot setting. Firstly, all models exhibit low SER scores in both seen and unseen domains, with the template approach being the lowest. This suggests that pre-trained language models are adept at copying and this skill also generalizes to out-of-domain examples.

The Schema-Guided representation performs at par with Naive representation on seen domains. At the same time, the slot descriptions do improve performance on the unseen domains (+0.9 BLEU), albeit to a limited degree. More effective ways of incorporating descriptions is a promising area for future work. For the seen domains, T2G2 outperforms Naive by 1.7 BLEU. The results on the

Model	Input or Generated sequence
Actions	Flights offer(airlines=American Airlines, outbound_departure_time=2:40 pm, is_nonstop=True, price=$78)
Template	Would you like to fly with American Airlines? The onward flight takes off at 2:40 pm. It is a direct flight. The ticket costs $78.
Reference	Do you wish to fly on American Airlines? The onward direct flight leaves at 2:40 pm, and the ticket price is $78. The ticket costs $78
Naive	There is an American Airlines flight that leaves at 2:40 pm and is not stop. The ticket costs $78.
Schema	There is an American Airlines flight that leaves at 2:40 pm and is not stop. The total cost is $78.
T2G2	How about an American Airlines flight that leaves at 2:40 pm? It's a direct flight and costs $78.
Actions	RentalCars confirm(has_insurance=True, car_type=hatchback)
Template	Please confirm the following details: It is a Hatchback car. It comes with insurance.
Reference	Please confirm: You want me to book a Hatchback car with insurance.
Naive	Please confirm: You want me to reserve a Hatchback car with a reassurance.
Schema	Please confirm: You want to rent a Hatchback.
T2G2	Okay, please confirm that you'd like to have a Hatchback car with an insurance.
Actions	Media inform(starring=Andrea Martin)
Template	Antonio Bustorff acted in it.
Reference	The actor is Antonio Bustorff.
Naive	The movie is directed by Antonio Bustorff.
Schema	The movie is directed by Antonio Bustorff.
T2G2	Antonio Bustorff acted in the movie.

Figure 5: A few examples of utterances generated from different models, along with the system actions, their template based representation (as shown in Figure 1) and the reference output. The errors are underlined.

unseen domains are more striking with an improvement of 7.3 points. This confirms the hypothesis that our simple template based input scheme offers superior generalization capabilities with a low overhead. The template model learns to "fuse" sentences and is able to successfully extend this skill to unseen domains.

5.2 Qualitative Analysis

In Figure 5 we list a few examples of model predictions. The first example illustrates a case where the model has to deal with a seen domain *Flights* but an unseen slot *is_nonstop*. Such a case would be common when new functionality needs to be added to an existing domain. Both Naive and Schema are unable to verbalize the slot correctly. While the template input contains all the information, it sounds very robotic. T2G2, on the other hand, takes the 4 template sentences as input and rewrites them into a fully accurate but much more natural sounding response.

The next example is from *RentalCars*, and features an unseen slot *has_insurance*. Schema fails to mention this slot. Naive attempts to verbalize it, but uses the wrong word (*reassurance*). T2G2, however, is able to paraphrase the template input into grammatical text without dropping any information.

The final example features an unseen slot *starring* from the *Movies* domain. Naive and Schema treat Antonio Bustroff as a director, since the slot *directed_by* appears during training. However, T2G2 simply relies on the template input and copies the phrase *acted in*. We refer the reader to Appendix F for more qualitative examples.

5.3 Human Evaluation

We conduct a human evaluation study via crowd sourcing [4]. Each human rater is shown the responses generated by different models and the ground truth response in a random order. Following (Peng et al., 2020), they are asked to rate each response on a scale of 1 (bad) to 3 (good) along two axes - *informativeness* and *naturalness*. Informativeness quantifies whether the response contains all the information contained in the dialogue acts, whereas naturalness evaluates whether the response sounds coherent, grammatical and natural. Each example is rated by 3 different workers. The final metric is an average of all the ratings.

A total of 500 randomly chosen examples are rated - 250 each from seen and unseen domains - across the 3 models discussed above and the ground truth response (*human*). With 3 ratings per example, this leads to a total of 6,000 ratings. Results are shown in Table 5.

Naturalness On the overall test set, all models outperform the human authored ground truth. This showcases the strength of pre-trained language models in generating natural sounding utterances, echoing findings from prior works. (Radford et al.,

[4] Examples of the rating UI can be found in Appendix E.

Naturalness				
	Naive	Schema	T2G2	GT
Unseen	2.43^{4}	2.41	$\mathbf{2.46}^{2,4}$	2.37
Seen	$\mathbf{2.48}^{4}$	2.45	2.47^{4}	2.40
Overall	2.45^{4}	2.43^{4}	$\mathbf{2.46}^{2,4}$	2.38

Informativeness				
	Naive	Schema	T2G2	GT
Unseen	2.36	2.49^{1}	$\mathbf{2.55}^{1,2}$	2.51^{1}
Seen	2.57	$\mathbf{2.59}^{4}$	2.56	2.54
Overall	2.46	2.54^{1}	$\mathbf{2.56}^{1}$	2.53^{1}

Table 5: Human evaluation results comparing different models and the ground truth. The superscripts 1 to 4 indicate that the model is significantly better than Naive, Schema, T2G2 and ground truth respectively, as determined by a one-tailed paired t-test with $p < 0.05$.

2019; Peng et al., 2020).

Informativeness Simply generating a fluent response is not enough. Its paramount for the responses to be factually grounded in the structured data, so that the wrong information is not conveyed to the user. For informativeness, we notice that all models perform well on the seen domains. However, on unseen domains, the Naive approach fares poorly. Schema outperforms Naive by a large margin on unseen domains. T2G2 further improves upon Schema. These results suggest Schema and T2G2 offer promising avenues to improve the zero-shot generalization capability of NLG systems. Moreover, both Naive and Schema see large drops on unseen domains, while T2G2 performs equally well on both seen and unseen domains.

Recall that Naive representation demonstrated strong scores on the SER metric for unseen domains. However, the low human scores on informativeness suggest that getting perfect scores on metrics like SER may not be a reliable way to judge factual accuracy. As models become stronger, better evaluation metrics need to be developed to accurately measure the improvements.

6 Few-Shot NLG

Virtual assistants need to support a constantly increasing number of domains and APIs. In order to keep labelled data costs under control, improving few-shot learning methods is important. In this section, we study the trade-off between the number of annotated training examples and performance of NLG.

6.1 Dataset

K	Dialogues	Examples
5	70	558
10	140	1,075
20	280	2,140
40	560	4,312
80	1,120	8,624
All	16,141	164,978

Table 6: Data statistics of FewShotSGD training splits.

Prior work (Mi et al., 2019; Tran and Le Nguyen, 2018; Wen et al., 2016) has studied few-shot learning and domain adaptation in a simulated setting by creating small subsets. However, lack of knowledge of the exact data splits makes it difficult to make comparisons to other methods. To remedy this, we create a new canonical split of the SGD dataset as described below.

- We make K-shot subsets for varying values of K $[5, 10, 20, 40, 80]$. In this setting each of the 14 domains from the training set have K dialogs.

- For all the few-shot splits we make sure that they contain examples for every dialogue act and slot type present in the full training set. For every domain, we make sure that each dialog act (inform, request etc.) and slot (name, time, price etc.) is represented at least once. However, all combinations of dialog acts and slots may not exist.

- The dev and test sets are left untouched.

This benchmark is referred to as FewShotSGD and we make the exact splits publicly available. The exact number of examples in each split is given in Table 6.

6.2 Results

In few shot experiments, we examine the performance of different models as a function of the amount of labelled data. The training setup remains the same, as described in section 4. Results are reported in Figure 6, where we can clearly see the performance improving as more training data becomes available. In all the K-shot settings, T2G2 gives consistent improvements of 4-5 BLEU while reducing the SER by a large margin. Even in the extreme 5-shot setting, the SER is just 3.6%.

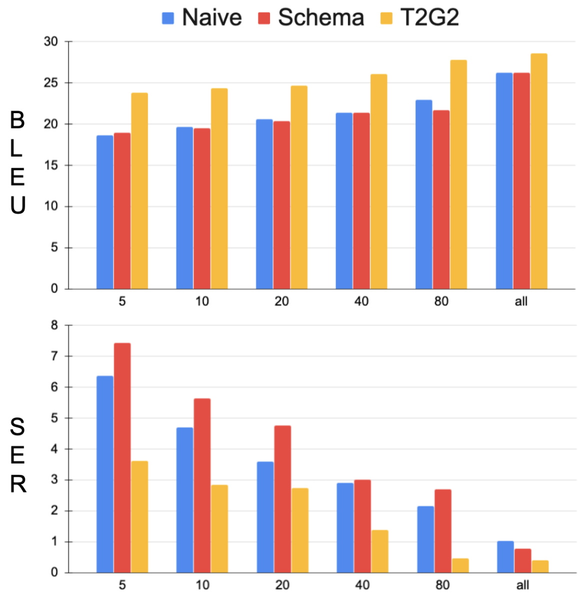


Figure 6: Performance in few-shot settings. The x-axis indicates the number of dialogues per domain in the training set. For exact scores, please refer to Appendix A.

Remarkably, T2G2 in the 80-shot setting outperforms the Naive model trained on the *entire* dataset, which is 20x larger. In the 5-shot setting, T2G2 performs on par with 80-shot Naive. We take this as evidence that our template guided input representation can lead to significant reduction in labelled data requirements.

7 Other Experiments

In this section, we conduct experiments to explore a few other aspects of our setup on the SGD dataset. For these experiments we use the Naive representation, since it is more widely adopted in prior work. We hope that these experiments will guide design choices in the future NLG models.

7.1 Joint Modeling

Joint modeling, instead of domain specific models, could be beneficial in low resource settings if there is some similarity between the underlying structure. Furthermore, having a single model for all domains also reduces the maintenance workload and is resource efficient. For NLG systems, it could also help in maintaining consistent styles across domains and APIs.

Because of these merits, we investigate the effect of joint modeling on SGD dataset. We focus on the 12 domains that are present in all 3 splits - train, dev and test. We train a single model on all these domains and compare it with individual models trained for each domain separately. As shown in Table 7, joint modeling leads to a win-win situation by improving BLEU by 3.4 points and reducing SER from 4.7% to just 1%, while requiring fewer parameters and resources. For further analysis of transfer learning across domains, we refer the reader to Appendix C.

Domain	**Separate**		**Joint**	
	BLEU	**SER**	**BLEU**	**SER**
Homes	22.9	1.6	26.3	0.2
Buses	18.6	4.0	23.4	0.0
Media	28.9	8.4	29.9	4.6
RideShare	20.3	2.1	26.0	0.0
Movies	21.4	23.0	29.3	4.9
Flights	19.7	1.0	20.5	0.0
Music	25.3	0.6	28.5	0.0
Services	25.3	0.6	29.2	0.0
RentalCars	17.6	9.2	22.1	2.0
Restaurants	25.8	4.0	27.5	0.1
Events	30.7	0.5	31.9	0.0
Hotels	26.6	1.6	29.7	0.2
Average	23.6	4.7	**27.0**	**1.0**

Table 7: Joint vs domain-specific (separate) NLG.

k	0	1	3	5	7
BLEU	26.2	29.0	31.5	32.4	32.6
SER	1.0	1.0	0.8	0.9	0.7

Table 8: Changing the size of the context. k represents the number of previous utterances used.

7.2 Role of Context

Dialogue acts represent the semantic content of the system response, but they don't contain any information about the lexical and syntactic content. The previous utterances in the dialogue history or context are important for generating good responses because they can help model conversational phenomena such as co-reference, elision, entrainment (lexical and syntactic alignment of responses) and avoid repetition (Du sek and Jurcicek, 2016a). Context also helps add variations to the responses generated across different conversations for the same system actions.

Table 8 shows the performance of NLG as more utterances from the dialogue context are given as input. In these experiments, we concatenate the last k utterances to the system action representation obtained from the Naive method. The model benefits from the additional context, showing an improvement of upto 6 BLEU. Just a single context utterance - the previous user utterance - results in an improvement of nearly 3 BLEU.

The evaluation for $k >= 2$ is not completely realistic, because we used the ground truth system utterances in the context during evaluation as opposed to the utterances generated by the NLG model itself. Regardless, the improvements clearly point to effectiveness of the added context at the cost of more resources. We hope these results inspire more work in this exciting direction.

8 Conclusion and Future Work

In this work, we proposed schema guided and template guided input representation schemes for task oriented response generation. Coupled with pre-trained language models, the template guided approach enables zero-shot generalization to new domains with little effort. Moreover, we show that it can lead to drastic reduction in annotation costs. We also present the first set of results on the multi-domain SGD dataset, which we hope will pave the way for further research in few-shot, zero-shot and multi-domain language generation.

While in this paper we use standard pre-trained models, designing pre-training tasks tailored to *sentence fusion* is an interesting line of future work. We also hope to apply T2G2 to languages other than English. Obtaining annotated data in non-English languages is an even bigger challenge, making the sample efficiency of our template rewriting approach especially suited to this setting. Another interesting line of future work is to investigate the use of T2G2 for generating user utterances, which could be useful for dialogue data augmentation and user simulation. This requires adding the ability to generate utterances with stylistic variations to capture different user personalities while maintaining consistency in style and vocabulary over a single dialogue.

References

Ankur Bapna, Gokhan Tür, Dilek Hakkani-Tür, and Larry Heck. 2017. Towards Zero-Shot Frame Semantic Parsing for Domain Scaling. *Proc. Interspeech 2017*, pages 2476–2480.

Regina Barzilay and Kathleen R McKeown. 2005. Sentence Fusion for Multidocument News Summarization. *Computational Linguistics*, 31(3):297–328.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. MultiWOZ-A Large-Scale Multi-Domain Wizard-of-Oz Dataset for Task-Oriented Dialogue Modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026.

Ziqiang Cao, Wenjie Li, Sujian Li, and Furu Wei. 2018. Retrieve, Rerank and Rewrite: Soft Template Based Neural Summarization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 152–161.

Wenhu Chen, Jianshu Chen, Pengda Qin, Xifeng Yan, and William Yang Wang. 2019. Semantically conditioned dialog response generation via hierarchical disentangled self-attention. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3696–3709, Florence, Italy. Association for Computational Linguistics.

Zhiyu Chen, Harini Eavani, Wenhu Chen, Yinyin Liu, and William Yang Wang. 2020. Few-shot NLG with pre-trained language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 183–190, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145. Morgan Kaufmann Publishers Inc.

Ond rej Du sek, David M Howcroft, and Verena Rieser. 2019. Semantic Noise Matters for Neural Natural Language Generation. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 421–426.

Ond rej Du sek and Filip Jurcicek. 2016a. A Context-aware Natural Language Generator for Dialogue Systems. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 185–190.

Ond rej Du sek and Filip Jurcicek. 2016b. 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.

Ondřej Dušek and Filip Jurcicek. 2019. Neural Generation for Czech: Data and Baselines. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 563–574.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The WebNLG Challenge: Generating Text from RDF Data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133.

Kelvin Guu, Tatsunori B Hashimoto, Yonatan Oren, and Percy Liang. 2018. Generating Sentences by Editing Prototypes. *Transactions of the Association for Computational Linguistics*, 6:437–450.

Nabil Hossain, Marjan Ghazvininejad, and Luke Zettlemoyer. 2020. Simple and Effective Retrieve-Edit-Rerank Text Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2532–2538.

Mihir Kale and Abhinav Rastogi. 2020. Text-to-Text Pre-Training for Data-to-Text Tasks. *arXiv*, pages arXiv–2005.

Mihir Kale and Scott Roy. 2020. Machine Translation Pre-training for Data-to-Text Generation–A Case Study in Czech. *arXiv preprint arXiv:2004.02077*.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL: A Conditional Transformer Language Model for Controllable Generation. *arXiv preprint arXiv:1909.05858*.

Alon Lavie and Abhaya Agarwal. 2007. METEOR: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231. Association for Computational Linguistics.

R'emi Lebret, David Grangier, and Michael Auli. 2016. Neural Text Generation from Structured Data with Application to the Biography Domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text summarization branches out*, pages 74–81.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*.

Fei Mi, Minlie Huang, Jiyong Zhang, and Boi Faltings. 2019. Meta-Learning for Low-resource Natural Language Generation in Task-oriented Dialogue Systems. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 3151–3157. AAAI Press.

Neha Nayak, Dilek Hakkani-Tür, Marilyn Walker, and Larry Heck. 2017. To Plan or not to Plan? Discourse Planning in Slot-Value Informed Sequence to Sequence Models for Language Generation. *Proc. Interspeech 2017*, pages 3339–3343.

Jekaterina Novikova, Ond rej Du sek, and Verena Rieser. 2017. The E2E Dataset: New Challenges For End-to-End Generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Baolin Peng, Chenguang Zhu, Chunyuan Li, Xiujun Li, Jinchao Li, Michael Zeng, and Jianfeng Gao. 2020. Few-shot Natural Language Generation for Task-Oriented Dialog. *arXiv preprint arXiv:2002.12328*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. *OpenAI Blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2019. Towards Scalable Multi-domain Conversational Agents: The Schema-Guided Dialogue Dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Van-Khanh Tran and Minh Le Nguyen. 2018. Adversarial Domain Adaptation for Variational Neural Language Generation in Dialogue Systems. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1205–1217.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. CIDEr: Consensus-based Image Description Evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Tsung-Hsien Wen, Milica Gasic, Nikola Mrk si'c, Lina M 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.

Tsung-Hsien Wen, Milica Gasic, Nikola Mrk si'c, 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.

Tsung-Hsien Wen, David Vandyke, Nikola Mrk si'c, Milica Gasic, Lina M Rojas Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A Network-based End-to-End Trainable Task-oriented Dialogue System. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449.

Yu Wu, Furu Wei, Shaohan Huang, Yunli Wang, Zhoujun Li, and Ming Zhou. 2019. Response Generation by Context-Aware Prototype Editing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7281–7288.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *Advances in neural information processing systems*, pages 5754–5764.

Chenguang Zhu, Michael Zeng, and Xuedong Huang. 2019. Multi-task Learning for Natural Language Generation in Task-Oriented Dialogue. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1261–1266.

A Additional Experiment Details

All models are trained on a 4x4 TPU slice, each taking 1-3 hours to finish training for 5000 steps. We provide development set BLEU scores in Tables 9 and 10. These scores are computed on the entire development set which includes both seen and unseen domains. In Table 11, we list the exact performance numbers for the few-shot NLG experiments.

B Automatic Metrics

Prior work has used different metrics for different benchmarks. Moreover, for the same metric (e.g. BLEU), different implementations are used. For fair comparison, for each dataset, we report the results using the implementation used in prior work. For E2E, we use the implementation from the e2e-metrics [5] suite. For computing BLEU on MultiWOZ, we use code made available in the SC-GPT codebase [6]. For model development i.e checking the best checkpoint based on the validation set, we rely on *sacrebleu* [7] across all experiments, since it has become the standard implementation in machine translation literature. We urge the NLG community to also converge upon a single implementation of BLEU. Taking inspiration from MT, the BLEU scores on experiments involving the SGD dataset are computed using *sacrebleu.*

model	BLEU
Naive	28.8
SG	29.9
T2G2	30.3

Table 9: Development set performance on the SGD dataset.

K	Naive	Schema	T2G2
5	19.8	20.0	22.0
10	21.3	22.0	24.0
20	23.4	22.4	24.5
40	23.1	25.3	25.6
80	26.1	24.9	27.8
All	28.8	29.9	27.5

Table 10: Development set BLEU scores in few-shot settings. K-shot denotes K dialogs for an API in the training set.

C Transfer Learning Across Domains

To measure the amount of transfer learning from one domain to another, we evaluate each domain specific model trained in Section 7.1 on all the domains and observe domain specific metrics. Results can be found in Table 12 and 13.

D Templates

In Tables 14, 15 and 16, we provide templates used for a few different APIs. The full set of templates is available with the code. Note that the linguistic quality of the templates does not need to be very

K	Naive		Schema		T2G2	
	BLEU	SER	BLEU	SER	BLEU	SER
5	18.7	6.4	18.9	7.4	23.8	3.6
10	19.7	4.7	19.5	5.6	24.4	2.9
20	20.6	3.6	20.4	4.7	24.7	2.8
40	21.4	2.9	21.4	3.0	26.0	1.4
80	23.0	2.2	21.7	2.7	27.8	0.5
All	26.3	1.0	26.2	0.8	28.6	0.4

Table 11: Test set performance in few-shot settings. K-shot denotes K dialogs for an API in the training set.

[5] https://github.com/tuetschek/e2e-metrics

[6] https://github.com/pengbaolin/SC-GPT

[7] https://github.com/mjpost/sacreBLEU

	homes	buses	media	rides	movies	flights	music	services	rental	restaurants	events	hotels
homes	1.6	14.7	7.7	6.2	11.7	17.1	28.3	20	17.1	18	27.9	12.6
buses	13.8	4	19.8	2.9	26.4	19.2	32.4	24.5	22.9	21.2	30.1	19
media	38.6	42.4	8.4	20.2	33.6	48.7	26.9	44.6	38.7	36.7	40.8	37.4
rides	34.9	31.8	19.4	2.1	37.3	43.9	41.4	37.6	34.1	28.8	35.5	31.1
movies	24.5	32.8	11	7.1	23	35.7	22.8	24.8	24.4	22.6	30.2	20.2
flights	9.7	5.1	17	2.2	22.3	1	25.9	18.1	8.2	14.4	21.3	19.1
music	36.6	38.5	3.9	20.1	24	48.4	0.6	26.3	28.4	23.9	38.3	33.6
services	4.8	19.1	3.7	5.8	10.4	29.6	20.8	0.6	20.6	5.8	16.4	11.6
rental	17.8	7	15.5	5.6	21.2	15.4	28.7	19.7	9.2	16.6	22.8	19.7
restaurants	9.8	21.9	10.9	5.2	21.9	33.1	24.7	6.9	18.4	4	15.7	19.2
events	1.4	30.4	3.7	1.3	10	32.2	14.4	8.3	20.7	10	0.5	13.4
hotels	5.2	10.1	6.3	1.3	8.8	19.8	18.6	5.2	6.7	6.3	8.5	1.6

Table 12: SER scores for domain specific models, when evaluated on all domains. The column denotes the domain on which the model was trained, while the row represents the domain used for evaluation.

	homes	buses	media	ridesg	movies	flights	music	services	rental	restaurants	events	hotels
homes	22.9	7.4	17.5	11.6	18.4	7.6	6.3	15.8	10.5	12.7	17	15.8
buses	12.6	18.6	11.2	11.2	11.3	9.7	4.6	13	12	12	17.5	12.9
media	6.1	5.6	28.9	9.1	16.2	3.8	10.6	9.5	4.9	8.7	8.4	11.5
rides	6.8	4.7	11.6	20.3	9.2	3.1	5.1	7.6	6.1	8.5	7.6	12.3
movies	9.6	7.5	21	9.3	21.4	7.3	9.9	14	9.5	11.5	15.1	15.7
flights	11.5	13.1	12.6	10.7	13.5	19.7	6	13	12.9	11.2	16.1	11.8
music	8.5	5.3	21.7	8.3	17.9	3.9	25.3	11.2	5.2	9.6	10.9	12.1
services	14.8	10.7	18.7	9.9	21	7.5	9.5	25.3	13.7	20.5	20.9	18.8
rental	11.7	11.9	12	9.6	14.1	7.9	4.5	15	17.6	14.2	16.8	14.3
restaurants	15.4	10	17.4	10.5	17.1	8	9.5	21.2	12	25.8	19.3	17.9
events	17.4	11.4	19.3	12.6	23.5	10.2	10.9	19.8	14	19.4	30.7	19.1
hotels	12.1	9.1	15.6	8.7	18.9	8	6.9	17.2	10.5	16.8	17.3	26.6

Table 13: BLEU scores for domain specific models, when evaluated on all domains. The column denotes the domain on which the model was trained, while the row represents the domain used for evaluation.

high, as long as the semantics of the dialog act are captured. This makes it easy for the API developers themselves to quickly create the simple templates.

E Human Evaluation Tasks

Figures 7 and 8 show examples of rater tasks for naturalness and informativeness respectively.

F Example Outputs

Sample utterances generated using the different models for various domains are shown in the examples below. The system actions, its template based representation used by the T2G2 model as input, and the reference response are also provided. The predictions are from models trained on the full SGD dataset and without any dialogue history context. The unseen domains have been marked with an asterisk.

Dialog Act	Template
GOODBYE	Enjoy!
INFORM!!free_entry!!False	No, entry to this place is not free.
INFORM!!free_entry!!True	Yes, entry is free.
INFORM!!good_for_kids!!False	No, this place isn't exactly good for kids.
INFORM!!good_for_kids!!True	Yes, the place is pretty nice to take your kids to.
INFORM!!phone_number!!@	Their phone number is @.
INFORM_COUNT!!FindAttractions!!count!!@	There are @ attractions that could interest you.
NOTIFY_FAILURE!!FindAttractions	Sorry, can't find anything for your constraints.
OFFER!!attraction_name!!@	You should check out @.
OFFER!!category!!@	This is a @.
REQUEST!!location	Where should I search for attractions in?
REQ_MORE	Anything else I could assist with?

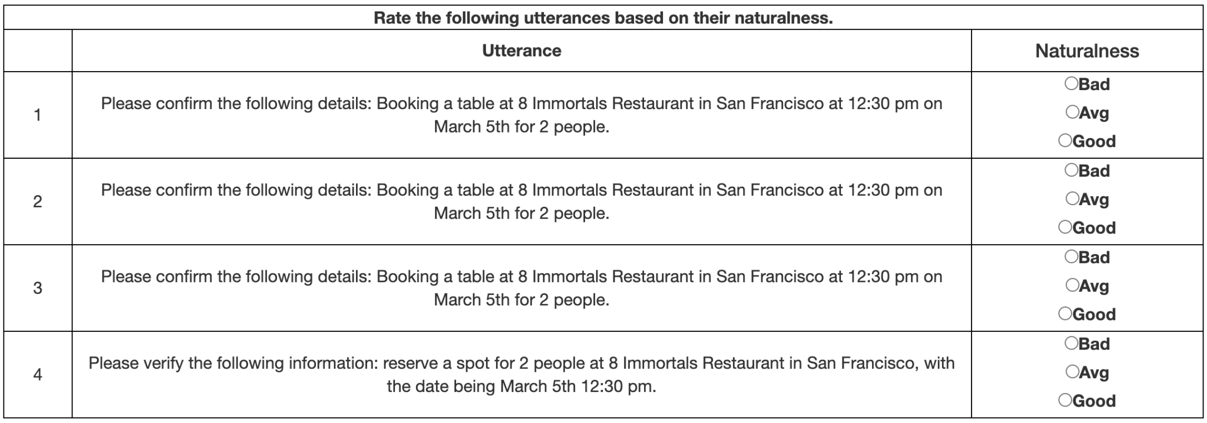
Table 14: Templates from a Travel API.

Dialog Act	Template
CONFIRM!!destination!!@	You want a ride to @.
CONFIRM!!number_of_seats!!@	You need @ seats.
CONFIRM!!ride_type!!@	You prefer a @ ride.
GOODBYE	Have a wonderful day.
INFORM!!ride_fare!!@	The ride costs @.
INFORM!!wait_time!!@	The cab is expected in @ minutes.
NOTIFY_FAILURE!!GetRide	I'm sorry, I could not find a ride for you at this time.
NOTIFY_SUCCESS!!GetRide	I booked your ride and the cab is on its way.
REQUEST!!destination	Where do you want to go to?
REQUEST!!destination!!@	Are you going to @?
REQUEST!!destination!!@@	Are you going to @ or @?
REQUEST!!number_of_seats	How many seats do you need?
REQUEST!!ride_type	Do you have a preferred type of ride?
REQ_MORE	Can I help you with anything else?

Table 15: Templates from a RideSharing API.

Dialog Act	Template
CONFIRM!!subtitle_language!!@	with subtitles in @.
CONFIRM!!title!!@	playing @
GOODBYE	Have a good day.
INFORM!!genre!!@	It is a @ movie.
INFORM!!starring!!@	@ acted in it.
INFORM_COUNT!!FindMovies!!count!!@	There're @ movies you may like.
NOTIFY_FAILURE!!FindMovies	I failed to find any movies matching your preference.
NOTIFY_FAILURE!!PlayMovie	Failed to play the movie.
NOTIFY_SUCCESS!!PlayMovie	Started playing the movie.
OFFER!!title!!@	What about @?
OFFER!!title!!@@	What about @ or @?
OFFER!!title!!@@@	Do you like @, @ or @?
OFFER_INTENT!!PlayMovie	Do you want to play the movie?
REQUEST!!genre	What kind of movies do you like?
REQUEST!!title	Which movie do you want to watch?
REQ_MORE	What else can I help?

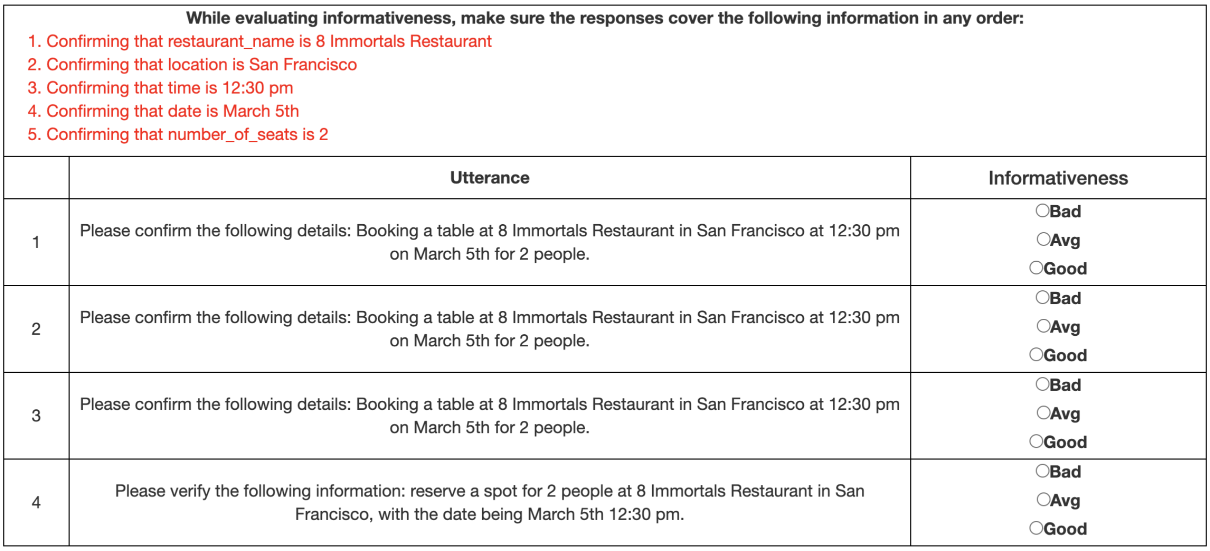
Table 16: Templates from a Movies API.



Rate the following utterances based on their naturalness.

	Utterance	Naturalness
1	Please confirm the following details: Booking a table at 8 Immortals Restaurant in San Francisco at 12:30 pm on March 5th for 2 people.	○Bad ○Avg ○Good
2	Please confirm the following details: Booking a table at 8 Immortals Restaurant in San Francisco at 12:30 pm on March 5th for 2 people.	○Bad ○Avg ○Good
3	Please confirm the following details: Booking a table at 8 Immortals Restaurant in San Francisco at 12:30 pm on March 5th for 2 people.	○Bad ○Avg ○Good
4	Please verify the following information: reserve a spot for 2 people at 8 Immortals Restaurant in San Francisco, with the date being March 5th 12:30 pm.	○Bad ○Avg ○Good

Figure 7: Example of a human rater task to evaluate naturalness. Each row represents the output from one of Naive, Schema, T2G2 and Ground Truth. The order of rows is shuffled across different tasks.



While evaluating informativeness, make sure the responses cover the following information in any order:

1. Confirming that restaurant_name is 8 Immortals Restaurant
2. Confirming that location is San Francisco
3. Confirming that time is 12:30 pm
4. Confirming that date is March 5th
5. Confirming that number_of_seats is 2

	Utterance	Informativeness
1	Please confirm the following details: Booking a table at 8 Immortals Restaurant in San Francisco at 12:30 pm on March 5th for 2 people.	○Bad ○Avg ○Good
2	Please confirm the following details: Booking a table at 8 Immortals Restaurant in San Francisco at 12:30 pm on March 5th for 2 people.	○Bad ○Avg ○Good
3	Please confirm the following details: Booking a table at 8 Immortals Restaurant in San Francisco at 12:30 pm on March 5th for 2 people.	○Bad ○Avg ○Good
4	Please verify the following information: reserve a spot for 2 people at 8 Immortals Restaurant in San Francisco, with the date being March 5th 12:30 pm.	○Bad ○Avg ○Good

Figure 8: Example of a human rater task to evaluate informativeness. Each row represents the output from one of Naive, Schema, T2G2 and Ground Truth. The order of rows is shuffled across different tasks.

Domain	**Model**	**Input or Generated sequence**
RentalCars	*Actions*	request(start_date=March 12th) request(pickup_location)
	Template	Do you want to have the car from March 12th? Where do you want to pick up the car?
	Reference	Will you be renting the car from March 12th? Please provide the pick-up location as well.
	Naive	When do you want to start on March 12th and where do you want to pick it up?
	SlotDesc	When do you want it from March 12th to pick it up?
	T2G2	Do you want to pick up the car from March 12th and where do you want to pick it up?
Hotels	*Actions*	inform(smoking_allowed=False) notify_success
	Template	Successfully booked the hotel. No, smoking is not allowed on the property.
	Reference	Your reservation is booked. No, smoking is not allowed on the property at this hotel.
	Naive	Your reservation has been made. Unfortunately, the hotel does not allow you to make the reservation.
	SlotDesc	Your reservation has been made. Unfortunately, they do not allow smoking.
	T2G2	I have booked the hotel. No, smoking is not allowed on the property.
Homes	*Actions*	confirm(property_name=Almaden lake Apartments) confirm(visit_date=March 13th)
	Template	Please confirm the following details: You are scheduling a visit to Almaden Lake Apartments and you want to visit the property on March 13th.
	Reference	Please confirm details to schedule a visit to Almaden lake Apartments on March 13th.
	Naive	Please confirm the following details: You are scheduling a visit to Almaden lake Apartments on March 13th.
	SlotDesc	Please confirm: You want to visit Almaden lake Apartments on March 13th.
	T2G2	Please confirm the following details: You are scheduling a visit to Almaden Lake Apartments on March 13th.
Services	*Actions*	request(type)
	Template	What kind of therapist do you need?
	Reference	Let me know your required type of Therapist to search for.
	Naive	What kind of doctor are you looking for?
	SlotDesc	What kind of therapist are you looking for?
	T2G2	What kind of therapist do you need?
Restaurants	*Actions*	confirm(restaurant_name=Nizza La Bella) confirm(location=Albany) confirm(time=6:15 pm) confirm(number_of_seats=2) confirm(date=March 7th)
	Template	Please confirm the following details: Booking a table at Nizza La Bella. The city is Albany. The reservation is at 6:15 pm. The reservation is for 2 people. The date is March 7th.
	Reference	Please confirm: You want me to book a table for 2 people at Nizza La Bella located in Albany for 6:15 pm on March 7th.
	Naive	Please confirm the following details: Booking a table at Nizza La Bella in Albany at 6:15 pm for 2 people on March 7th.
	SlotDesc	Please confirm the following details: Booking a table at Nizza La Bella in Albany at 6:15 pm for 2 people on March 7th.
	T2G2	Please confirm the following details: Booking a table at Nizza La Bella in Albany at 6:15 pm on March 7th for 2 people.
Media	*Actions*	inform(starring=Antonio Bustorff)
	Template	Antonio Bustorff acted in it.
	Reference	The actor is Antonio Bustorff.
	Naive	The movie is directed by Antonio Bustorff.
	SlotDesc	The movie is directed by Antonio Bustorff.
	T2G2	Antonio Bustorff acted in the movie.
Music	*Actions*	confirm(track=strangers) confirm(device=Patio)
	Template	Please confirm the following details: playing the song Strangers in Patio
	Reference	Please confirm: play strangers in patio?
	Naive	Please confirm the following details: Play strangers on the Patio.
	SlotDesc	Please confirm: play strangers on Patio.
	T2G2	Please confirm the following details: playing Strangers in Patio.
Buses	*Actions*	request(from_city) request(departure_date)
	Template	Where are you leaving from? When do you plan to leave?
	Reference	What is your departure date and city?
	Naive	Where are you leaving from and when do you plan to leave?
	SlotDesc	Where are you leaving from and when do you plan to leave?
	T2G2	Where are you leaving from and when do you plan to leave?

Figure 9: A few examples of utterances generated from different models, along with the system actions, their template based representation (as shown in Figure 1) and the reference output. The unseen domains have been marked with an asterisk.

Domain	Model	Input or Generated sequence
Events	*Actions*	confirm(event_name=Acting For Teens) confirm(number_of_tickets=4) confirm(date=March 11th) confirm(city=New York)
	Template	Please confirm the following details: Booking tickets for Acting For Teens. You have 4 tickets in your reservation. The date of it is March 11th. The event is taking place in New York.
	Reference	Please confirm: You want me to purchase 4 tickets for the event called Acting For Teens taking place in New York on March 11th.
	Naive	Please confirm: 4 tickets to Acting For Teens on March 11th in New York.
	SlotDesc	Please confirm: 4 tickets to Acting For Teens on March 11th in New York.
	T2G2	Please confirm: Booking 4 tickets for Acting For Teens on March 11th in New York.
Events	*Actions*	confirm(event_name=Mxmtoon) confirm(number_of_tickets=2) confirm(date=March 10th) confirm(city=New York)
	Template	Please confirm the following details: Booking tickets for Mxmtoon. You have 2 tickets in your reservation. The date of it is March 10th. The event is taking place in New York.
	Reference	Can you please confirm that you want to buy 2 tickets for the Mxmtoon event which happens on March 10th in New York?
	Naive	Please confirm: 2 tickets for Mxmtoon on March 10th in New York.
	SlotDesc	Please confirm: 2 tickets to Mxmtoon on March 10th in New York.
	T2G2	Please confirm: 2 tickets for Mxmtoon on March 10th in New York.
Alarm*	*Actions*	offer(alarm_time=4 pm) offer(alarm_name=Event)
	Template	There is an alarm for 4 pm with name Event.
	Reference	A 4 pm alarm with name Event.
	Naive	There is an event at 4 pm.
	SlotDesc	There is an event at 4 pm.
	T2G2	There is an alarm for 4 pm with event name.
Alarm*	*Actions*	offer(alarm_time=10:30 am) offer(alarm_name=commute) inform_count(count=1)
	Template	You have 1 alarms currently set. There is an alarm for 10:30 am with name Commute.
	Reference	You have 1 alarm at 10:30 am. with name commute
	Naive	I found 1 suitable for you. It's at 10:30 am and it's a commute.
	SlotDesc	I found 1 alarm at 10:30 am. It's a commute.
	T2G2	You have 1 alarm for 10:30 am with the name Commute.
Weather	*Actions*	inform(humidity=30)
	Template	The humidity is around 30 percent.
	Reference	The humidity is about 30 percent.
	Naive	It's 30 minutes.
	SlotDesc	The humidity is 30.
	T2G2	The humidity is around 30 percent.
Weather	*Actions*	inform(humidity=75) inform(wind=4)
	Template	The humidity is around 75 percent. The average wind speed should be 4 miles per hour.
	Reference	The humidity for today is around 75 percent. The average wind is 4 miles an hour.
	Naive	It's a 7 star, and it's a 4 star.
	SlotDesc	The humidity is 75 and the wind speed is 4 miles per hour.
	T2G2	The average wind speed should be 4 miles per hour and the humidity is around 75 percent.
Messaging*	*Actions*	notify_success
	Template	You've successfully shared the location.
	Reference	The location was successfully shared.
	Naive	Your reservation has been made.
	SlotDesc	Your contact has been confirmed.
	T2G2	Your location has been shared.
Messaging*	*Actions*	confirm(location=2190 Bancroft Way) confirm(contact_name=Peter)
	Template	Please confirm the following details: You're sharing the location 2190 Bancroft Way with Peter.
	Reference	So share the location 2190 Bancroft Way with Peter?
	Naive	Please confirm the following details: You are scheduling a visit to Peter at 2190 Bancroft Way.
	SlotDesc	Please confirm the following details: You would like to contact Peter at 2190 Bancroft Way.
	T2G2	Please confirm the following details: You're sharing the location 2190 Bancroft Way with Peter.
Trains*	*Actions*	offer(journey_start_time=7:10 am) offer(total=$91)
	Template	What about the train departing at 7:10 am? It costs $91 in total.
	Reference	There is a train leaving at 7:10 am and costs $91.
	Naive	There is a bus that departs at 7:10 am and costs $91.
	SlotDesc	There is a 7:10 am train that costs $91.
	T2G2	How about the 7:10 am train? It costs $91 in total.
Travel	*Actions*	offer(attraction_name=BODY WORLDS London) offer(category=Museum)
	Template	You should check out BODY WORLDS London. This is a Museum.
	Reference	I suggest a museum called BODY WORLDS London.
	Naive	BODY WORLDS London is a Museum.
	SlotDesc	BODY WORLDS London is a museum.
	T2G2	BODY WORLDS London is a museum.

Figure 10: Continuation of examples in Figure 9.