Building Production-Quality NLG Models with Minimal Labelled Data

Anonymous EMNLP submission

Abstract

Natural language generation (NLG) plays an important role in task-oriented dialog systems to provide meaningful and natural responses to user's requests. However, training a NLG model that could surface production-ready quality responses usually requires a large amount of training data. In this paper, we propose two novel data-efficient approaches to bootstrap the model. We first propose a template-based approach that leverages a scenario generation framework to create full coverage of possible scenarios and their corresponding synthetic annotations. Secondly, we leverage the pretrained BART model with a bucketing method that groups scenarios based on their dialog act structures. Extensive experiments on three datasets show our approaches achieve production-quality with 10 times less labelled data than a standard NLG dataset.

1 Introduction

Natural language generation (NLG) plays an important role in task-oriented dialog system to provide meaningful and natural responses to user's requests. NLG components of dialog systems have often relied on handcrafted templates (Deemter et al., 2005) to produce system responses. Although template-based approaches provide good control and easy domain bootstrapping, the number of templates required for full coverage has an impractical scaling behavior as the complexity of a domain increases. Additionally, it is not trivial to condition a response on the user's request or other available context, substantially limiting the naturalness of the response.

Recently, model-based NLG has attracted widespread attention (Dušek et al., 2019; Balakrishnan et al., 2019) for its ability to generate more contextual and cohesive responses and generalize across domains and languages. However, data-driven NLG approaches require a large number of annotated training utterances, and creating annotated responses with structure information (e.g., dialog acts) is very time consuming in general.

In this work, we aim to combine the merits of both worlds. We first propose a scenario generation framework which uses a template-like approach to create an unlimited number of synthetically annotated <scenario, response> pairs. The framework is designed to be user-friendly and domain-agnostic so that creating grammatically-correct and natural-sounding responses is straightforward. Experiments show that models trained on such synthetic data produce fully grammatical and semantically correct responses.

As a second approach, we leverage pre-trained language models for further domain bootstrapping. We consider the BART model (Lewis et al., 2019) given its superior results on sequence generation tasks. Unlike previous approaches that fine-tune on a random selection of samples, we propose a bucketing idea that groups scenarios by their tree structures and find it to be more data-efficient. We fine-tune the pre-trained BART model on bucketing data, and perform data-augmentation by using sequence level knowledge distillation (Kim, 2016), to auto-annotate the unlabelled scenarios. Combining these ideas, we find we can achieve production-ready quality for a domain with minimal data labeling.

We view our contributions as follows:

- We propose a novel domain-agnostic framework for synthetic annotation creation for NLG.
- We propose a new bucketing and data augmentation idea with BART to improve data efficiency.
- We release two new datasets on Alarm and Time domains for use in data-efficient NLG modeling.
- Extensive experiments show our approach can achieve production-ready quality with as little as 10% of the labelled data of a standard NLG dataset.

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.
Flat MR	`condition1[sunny] date_time1[this weekend] avg_high1[60s] low2[43] date_time2[Sunday evening]`
Our MR (Scenario)	`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] ] ] ]`
Annotated Reference	[INFORM It'll be [condition sunny] throughout [date_time_range colloquial[this weekend]]. [CONTRAST [INFORM The high will be in the [avg_high 60s]]] , [INFORM but expect temperatures to drop as low as [avg_low 43 degrees] by [date_time [week_day Sunday] [colloquial evening]]].
Bucket Hash	[inform[condition,date_time_range[colloquial]],contrast[inform[avg_high, date_time[colloquial]], inform[low, date_time[colloquial,week_day]]]]

Table 1: Sample flat MR with reference compared against tree-structured MR. The last second row shows an annotated reference with the tree-structured MR. Nodes in blue are all children of the root node of the tree.

2 Related Work

NLG from structured data has been an active research area for decades, facilitated by datasets like the E2E Challenge (Novikova et al., 2017), MultiWoz (Budzianowski et al., 2018) and Weather (Balakrishnan et al., 2019). Early NLG system (Reiter and Dale, 2000) divide generation into content selection, macro/micro planning and surface realization. Recently, data-driven approaches (Wen et al., 2015), especially Seq2Seq methods (Balakrishnan et al., 2019; Rao et al., 2019), have become popular for their superior naturalness and simplicity.

Our work is closest to (Chen et al., 2020; Peng et al., 2020), where they leverage pretrained GPT models (Radford et al., 2019) and fine-tune on a small amount of domain examples. However, there are a few key differences: First, we adopt the tree-structured meaning representations (MRs) proposed in (Balakrishnan et al., 2019), which lead to increased response naturalness and modeling complexity compared to flat MRs. Second, we propose novel methods of combining synthetic data creation and data augmentation with BART (Lewis et al., 2019) model. Lastly, we achieve the same level of accuracy attained by the large pretrained models by distilling substantially smaller models that can be easily deployed in a real production setting.

3 Approach

Inspired by Balakrishnan et al. (2019), we reuse their tree-structured meaning representation (MR) which provides a better control of the discourse structure and content in generated utterances. Their tree-structured MRs consist of three sets of non-terminal tokens: `argument`, `dialog act` and `discourse act`. A dialog act is a minimum atomic unit that contains arguments to be expressed in an utterance, while discourse acts define the relationship between dialog acts. A tree-structured MR example for the weather domain is provided in Table 1, along with a flat MR and human reference. Experiments in Balakrishnan et al. suggest that tree-structured MRs lead to better controllability and naturalness of model responses. For clarity, we use MR and scenario interchangeably in the paper.

3.1 Scenario Generation Framework

A major challenge for data-driven NLG models is that the creation of annotated responses is time-consuming. Additionally, a live NLG system needs to have full coverage of different MR (scenario) structures to make sure almost all user requests can be fulfilled. Our scenario generation framework is exactly designed to provide an easy way to generate all possible scenario structure variations and create synthetic annotated responses. The framework contains three major components:

- **Entity**: Each entity is an argument type. Example entities include `date_time`, `location`, `duration`, `person`, etc. Note that arguments can be nested, e.g., `location` can have a single sub-argument `city` or both `street_address` and `zipcode`. At the generation time, the entity structure and value are randomly generated from pre-defined ranges.
- **Operator**: Defines the relationship between entities in a scenario. Such relationship are important to create semantically correct scenarios. For example, we would like to have `date_time` of alarms chronologically ordered if a user is querying for multiple alarms. Example operators include `min`, `max`, `compare`, `contrast`, etc.
- **Config**: A triple of the form <user request, scenario, annotation>. Each config includes a predefined list of user requests, annotations, and corre-

sponding scenarios. A config can have reference to multiple entities to fill in the entity structure and value. We provide a simple example config in Table 1 in the Appendix section.

With this framework, we can create unlimited annotations that cover all possible MR structures. This approach provides several benefits to bootstrap an NLG model: 1) While ensuring coverage, the created configs are also carefully reviewed to be grammatical and semantically correct; 2) The framework can be easily extended to different domains and languages, making the synthetic data creation simple to finish in a few hours; 3) Though it's a rule-based approach, we could still make annotations more natural by conditioning on user's requests, e.g., we could add `YES/NO` to annotations when a user request is a binary question.

3.2 Data Augmentation with BART

In this work, we adopt the BART model to investigate how pre-training can help in the NLG context. After fine-tuning BART on a small set of data, we run it on the unlabelled scenarios, as in self-training approaches (Kedzie and McKeown, 2019). The model predictions that match the unlabelled scenarios in tree structures are considered as "correct" annotations and selected for augmenting the training data, along the lines of sequence-level knowledge distillation (Kim, 2016). This simple idea offers us unlimited "free" annotations and allows us to train small models for both effectiveness and efficiency considerations.

Bucketing. As our ontology is fully tree-structured, modeling structure biases can be essential. Rao et al. (2019) suggested a naive model that doesn't consider structure is difficult to generalize to unseen structures. Therefore, to further improve data efficiency, we propose a bucketing strategy that groups scenarios into buckets based on their *bucket hashes*. A bucket hash is generated by an in-order traversal of ascenario tree, while ignoring argument values. An example of bucket hash is shown in the last row in Table 1.

4 Experiments

Datasets: We conduct experiments on both the public Weather dataset from (Balakrishnan et al., 2019) and two internal datasets for Alarm and Time domains. The Alarm and Time datasets are created by following the same process in Balakrishnan et al. (2019), which we detail in the Appendix section.

Dataset	Train(HUMAN)		Train(SYN)		Val	Test
	#Sample	#Buck	#Sample	#Buck		
Weather	25390	2180	86401	6141	3078	3121
Alarm	7163	126	39079	1354	2024	1024
Time	6237	273	48039	218	1717	891

Table 2: Dataset Statistics.

We will release the two internal datasets and our synthetic datasets upon acceptance. The dataset statistics are shown in Table 2. For the human-annotated and synthetic training sets, we report both number of samples and buckets. As we can see, weather is a more complicated domain due to its large bucket size.

Models: We consider both a standard attention-based Seq2Seq (Bahdanau et al., 2014) (**S2S**) and BART model (Lewis et al., 2019) in our experimentation. We leave the hyper-parameter details in the Appendix section. For each model, we experiment with different data for model training:

- **BASE**: the original full training set was used.
- **SYN**: the synthetic data generated from Sec. 3.1.
- **1B**: we randomly selected 1 example from each bucket in the full training set and used these for model training.

For all these data settings, the validation and the test sets remain the same. Additionally, for S2S model, we experiment with following:

- **1B-AUG**: We first used the BART model trained under **1B** setting to annotate the remaining scenarios in the original training set. Then we select annotations that pass tree accuracy check, and combine them with **1B** data for model training.
- **SYN-1B-AUG**: combines **SYN** and **1B-AUG**.

Metrics: We consider both automatic metrics and human evaluation results. For automatic metrics, we report both BLEU-4 (Papineni et al., 2002) and Tree Accuracy (`TreeAcc`). Tree accuracy is a binary metric which measures whether the ontology tree structure in model prediction matches that in scenario input. For human evaluation, we asked annotators to rate model responses on a *binary* scale on the following two dimensions:

- **Grammaticality** (`Gram`): Our evaluation guidelines included considerations for proper subject-verb agreement, word order, completeness, etc.
- **Correctness** (`Corr`): Measures *semantic correctness* of the responses. Our guidelines included considerations for sentence structure, contrast, hallucinations (incorrectly added attributes), and missing attributes. We asked annotators to evaluate model predictions against the MR.

Data	Model	Weather				Time			
Metric		BLEU	TREEACC	CORR	GRAM	BLEU	TREEACC	CORR	GRAM
BASE	S2S	91.5	91.4	99	99	96.1	99.7	99	100
	BART	91.5	93.4	100	99	95.4	99.7	99	100
SYN	S2S	65.7	44.8	79	99	89.8	90.1	100	99
	BART	76.2	44.0	97	99	93.1	91.1	99	100
1B	S2S	68.3	55.3	97	98	84.8	67.9	87	90
	BART	90.0	89.8	100	99	93.2	89.5	93	91
1B-AUG	S2S	90.3	91.2	99	99	94.1	99.8	96	92
SYN-1B-AUG	S2S	90.1	90.7	100	99	93.3	97.9	97	100

Table 3: Results on Weather and Time datasets. All metrics are percentages.

Data	Model	Alarm			
	Metric	BLEU	ACC	CORR	GRAM
BASE	S2S	92.9	99.4	100	100
	BART	92.9	99.8	100	100
SYN	S2S	89.2	98.2	100	99
	BART	89.8	99.5	100	99
1B	S2S	58.9	24.5	80	97
	BART	92.2	88.9	100	99
1B-AUG	S2S	81.4	91.3	100	98
SYN-1B-AUG	S2S	90.5	99.7	100	100

Table 4: Results on Alarm dataset.

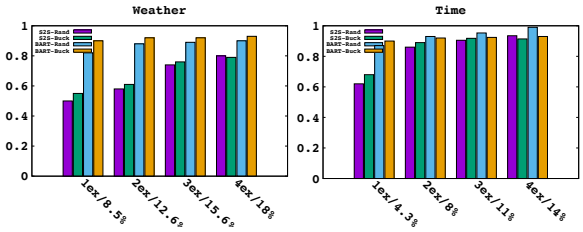

Figure 1: Bucketing Ablation Study on Weather/Time

Due to the shortage of annotator bandwidth by Covid-19, part of the human evaluations were conducted by the authors. However, our extensive prior experience has indicated that our evaluations are highly correlated to the third-party annotators'. Also, `Gram` and `Corr` are fairly objective, and thus unlikely to be biased by the authors' involvement.

5 Results

We show our results on weather and time in Table 3, and alarm in Table 4. For each experiment, we randomly sampled 100 model responses that pass tree accuracy for human evaluations.[1] In the 1B setting, the data percentages for weather/time/alarm domain are 8.5%/4.3%/1.7%, respectively.

As we can see, first, on all three datasets, S2S and BART are roughly comparable in BASE setting for all metrics, suggesting the pretraining benefits are limited when we have enough human annotations. However, the differences between S2S and BART are huge under 1B setting when data is scarce. Using the synthetic data (SYN), we can see all three domains achieve close to 100% grammaticality and correctness. Tree accuracy on Time and Alarm are fairly high, while Weather is much lower, which is due to a large number of buckets missing in the weather synthetic training data. Combining BART augmentation with bucketing (1B-AUG), we can see a significant boost on all metrics with a small S2S model for all three domains. Moreover, combing synthetic data with 1B-AUG (SYN-1B-AUG), we can see a further boost on grammaticality and correctness, while still being roughly comparable in tree accuracy and BLEU score. To conclude, combining all these ideas enables us to train a small S2S model that achieves production-ready quality with less than 10% human annotations.[2]

Bucketing: To further study how bucketing helps, we performed ablation studies to compare: 1) selecting different number of examples per bucket, i.e., [1,2,3,4], to the training set; and 2) selecting same amount of examples randomly. Our results on weather and time domain are shown in Figure 1. As alarm domain shows similar trend to time, we put its figure in the appendix. The x-axis shows the number of examples per bucket and the percentage of full training set, and y-axis shows tree accuracy. Clearly we can see that bucketing leads to better data efficiency for both S2S and BART model. A minor exception is on the Time domain, where the BART model with random sampling surpassed bucketing with >2 examples per bucket. Overall, we see bucketing is more effective for small models and data-scarce settings.

6 Conclusion

In this work, we have introduced a novel synthetic data creation approach and a data augmentation method with pretrained language models. Experiments show combining these ideas enables production-quality NLG models to be trained with minimum annotations.

[1] In a production system, a template-based back-off strategy can be used for responses that fail the tree accuracy filter, so only the ones that pass the filter are relevant for human evaluation.

[2] In fact, our models for all three domains have been deployed in production at the submission time, and currently serve thousands of user requests daily.

References

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

Anusha Balakrishnan, Jinfeng Rao, Kartikeya Upasani, Michael White, and Rajen Subba. 2019. Constrained decoding for neural NLG from compositional representations in task-oriented dialogue. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. To appear.

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.

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

Kees Van Deemter, Mariët Theune, and Emiel Krahmer. 2005. Real versus template-based natural language generation: A false opposition? *Computational linguistics*, 31(1):15–24.

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*.

Chris Kedzie and Kathleen McKeown. 2019. A good sample is hard to find: Noise injection sampling and self-training for neural language generation models. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 584–593, Tokyo, Japan. Association for Computational Linguistics.

Rush Kim. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The e2e dataset: New challenges for end-to-end generation. *arXiv preprint arXiv:1706.09254*.

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

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.

Jinfeng Rao, Kartikeya Upasani, Anusha Balakrishnan, Michael White, Anuj Kumar, and Rajen Subba. 2019. A tree-to-sequence model for neural nlg in task-oriented dialog. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 95–100.

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

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.