Neural Data-to-Text Generation with LM-based Text Augmentation

†Ernie Chang, ⊙ Xiaoyu Shen*, †Dawei Zhu, †Vera Demberg, ⊗ Hui Su
†Dept. of Language Science and Technology, Saarland University
⊙ Amazon Alexa AI, Berlin
⊗ Pattern Recognition Center, Wechat AI, Tencent Inc, China
{cychang}@coli.uni-saarland.de

Abstract

For many new application domains for data-to-text generation, the main obstacle in training neural models consists of a lack of training data. While usually large numbers of instances are available on the data side, often only very few text samples are available. To address this problem, we here propose a novel few-shot approach for this setting. Our approach automatically augments the data available for training by (i) generating new text samples based on replacing specific values by alternative ones from the same category, (ii) generating new text samples based on GPT-2, and (iii) proposing an automatic method for pairing the new text samples with data samples. As the text augmentation can introduce noise to the training data, we use *cycle consistency* as an objective, in order to make sure that a given data sample can be correctly reconstructed after having been formulated as text (and that text samples can be reconstructed from data).

On both the E2E and WebNLG benchmarks, we show that this weakly supervised training paradigm is able to outperform fully supervised seq2seq models with less than 10% annotations. By utilizing all annotated data, our model can boost the performance of a standard seq2seq model by over 5 BLEU points, establishing a new state-of-the-art on both datasets.

1 Introduction

Neural data-to-text generation has been the subject of much recent research. The task aims at transforming *source-side* structured data into *target-side* natural language text (Reiter and Dale, 2000; Barzilay and Lapata, 2005). While neural end-to-end systems afford the advantage of easy adaptability (Lebret et al., 2016; Wiseman et al., 2017), huge amounts of data-text pairs are still necessary to perform on par with their rule-based counterparts (van der Lee et al., 2018). This makes using neural systems less appealing: oftentimes, in-domain text samples are not readily available, and there is a high cost to collecting in-domain texts which fit the data samples, and annotating these texts with the data labels – the cost for collecting this data might hence even outweigh the efforts of designing a rule-based system (Gkatzia, 2016). The goal of this work is to improve the performance of neural data-to-text models in scenarios where only very few text samples exist (we assume that these text samples are paired with corresponding data samples). We aim to answer how we can make the most of the scarce annotations, together with large amounts of unlabelled data, in order to push the limit of the neural data-to-text models. Figure 1 illustrates the scenario.

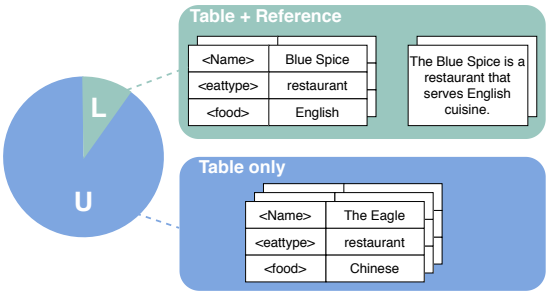

Figure 1: **Few-shot scenario**: The model is expected to learn data-to-text generation with few labeled instances (i.e. table-text pairs). The example is taken from the E2E dataset.

To address the limited-data challenge, we propose a simple yet effective way of augmenting the text side with the pretrained language model (LM) GPT-2 (Radford et al., 2019). Unlike other text augmentation work employed in data-to-text generation systems (Freitag and Roy, 2018; Agarwal et al., 2018), our proposal assumes little to no domain-dependent heuristics. It consists of two steps: (1) information augmentation by slot-value replacement and (2) LM augmentation by GPT-2

*Work done prior to joining Amazon.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 758–768
April 19 - 23, 2021.

generation.

Once we have augmented the set of text samples, we are essentially in a similar setting as previously proposed semi-supervised approaches to data-to-text generation Schmitt and Schütze (2019); Qader et al. (2019); Su et al. (2020), which assume the presence of vast amounts of *unpaired* data and text instances. These approaches exploit a *cycle consistency* objective in order to learn a pairing for the data samples. The cycle consistency objective tries to make sure that data samples can be reconstructed correctly from their textual formulations, and similarly that texts can be reconstructed after having been parsed into a data representation.

As the automatically generated text samples from GPT-2 might be very noisy and not pair well with data samples, we align each augmented text sample with its most similar unlabeled data sample, as defined in their encoded vector space. This idea is inspired by recent work on representation matching in MT (Artetxe and Schwenk, 2019; Ruiter et al., 2019). To ensure good quality of the training data, only pairs above a certain similarity threshold ϵ are retained as pseudo pairs for training. The quality of the pseudo pairs will gradually improve as the encoder improves in the training process. In return, the learning of the encoder will also be facilitated with the improved quality of pseudo pairs as a virtuous cycle.

On two data-to-text benchmarks E2E (Novikova et al., 2017) and WebNLG (Gardent et al., 2017), we show that our LM-augmented weakly supervised model succeeds on outperforming fully supervised seq2seq model, though utilizing less than 10% of the data annotations. It even outperforms previous work which *additionally has access to all unpaired text samples*. When trained with full data annotations, it is able to boost the model performance by up to 5 BLEU points, establishing a new state-of-the-art on both datasets.

In summary, this work makes the following contributions:

1. We study the few-shot data-to-text scenario where, unlike previous works, no further target-side text is available.

2. We present an effective way of automatically augmenting target text by resorting to the pretrained LM GPT-2.

3. We propose utilizing the augmented text by a combination of cycle consistency and representation matching. The resulting model outperforms standard seq2seq model with less than 10% data annotations.

4. The proposed model is shown to be complementary with current seq2seq pretraining techniques, and can offer orthogonal improvements when combining both.

2 Related Work

Building neural data-to-text systems with few paired samples (but a large set of unpaired samples) has been a hot research topic recently. Most works adopt the idea of cycle consistency (Zhu et al., 2017), which has been used in many text generation tasks like machine translation (Artetxe et al., 2017; Lample et al., 2017) and style transfer (Prabhumoye et al., 2018; Subramanian et al., 2018). Schmitt and Schütze (2019); Qader et al. (2019); Su et al. (2020); Chang et al. (2020, 2021a,b) applied this idea to the task of data-to-text generation and reported promising results. Ma et al. (2019) separate the generation process into few-shot content selection and surface realization components and learn them separately. Nonetheless, all of these approaches assume the existence of *huge quantity of unpaired text samples*, which, as we mentioned, is an unrealistic assumption for the task of data-to-text generation. Freitag and Roy (2018) proposes to reconstruct usable sequences re-written from data with rules for unsupervised data-to-text generation. Unfortunately, designing these rules require efforts similar to building a template-based system. (Budzianowski and Vulić, 2019; Chen et al., 2020; Peng et al., 2020) tackle the few-shot challenge by finetuning a pretrained LM to incorporate prior knowledge from general-domain text or data-text pairs. We show that our technique is complementary with them and can offer orthogonal improvements when combining both.

3 Problem Formulation

We represent the data samples as D and the text samples as T. In our work, we do not restrict the format of the data. Each $d \in D$ can be a set of key-value pairs, as in Figure 1, or in form of RDF triples as in Gardent et al. (2017). Each text $t \in T$ consists of a sequence of words. In few-shot settings, we are assumed to have (1) k labeled pairs (D_L, T_L) and (2) large quantities of unlabeled data

D_U where $|D_U| \gg k > 0$[1]. This, we believe, is a more realistic setting as unlabeled data are usually abundant and also can be easily fabricated from predefined schemata. Notably, we assume no access to outside resources containing in-domain text. The k annotations are all we know about the text side.

4 Approach

In this section, we first explain our proposed new method for text sample augmentation, and then discuss methods to remove noise and automatically align the data by elaborating on the ideas of cycle consistency and representation matching. Finally, we summarize the approach and present the detailed algorithm.

4.1 Text Augmentation

To mitigate the paucity of the set of text samples T, we propose a pipeline approach to augment the text samples by (1) information augmentation and (2) LM augmentation.

4.1.1 Information Augmentation

We generate additional text samples by performing slot-value replacements. As many data values are exactly copied to the text samples, these copied information can be easily detected and replaced with other values (for the same slot type) to enrich the information space of the text samples. This can be considered as a simplified version of traditional methods of template mining where key words are extracted to construct templates (Kondadadi et al., 2013; Oya et al., 2014). An example is shown in Figure 2. Each text sample is augmented with 10 more distinct text samples or with all possible values being replaced.

The slot-value replacement is efficient to implement. However, it can only detect identical values and augment text with the same combinatorial patterns as the few-shot annotations. To enrich the linguistic realizations of text sentences and enable new combinations of information, we further propose a LM augmentation approach using GPT-2.

4.1.2 LM Augmentation

GPT-2 (Radford et al., 2019) is a language model pretrained on the collected WebText. It has demonstrated remarkable zero-shot multitask adaptability by simply feeding the input of each task into the LM and continuing to generate words. People have also also shown that GPT-2 is able to improve classification tasks via in-domain text augmentation (Papanikolaou and Pierleoni, 2020; Sun et al., 2020). We use a similar technique by first fine-tuning GPT-2 in the few-shot annotations (Wolf et al., 2019), and then applying it to produce synthetic text through an iterative conditional generation process: With initial seeds being samples of T_L plus new samples from information augmentation, the LM iteratively conditions on the previous output sentence to generate in-domain text[2]. Each synthetic sentence is pruned if it (1) is shorter than 5 words or (2) contains only special tokens. The iterative generation is terminated when all tokens in the initial seeds are covered or if the maximum of 100 runs is reached. All the unpruned synthetic text samples are added into the space of T to benefit the learning direction of $t \to d' \to t$ and $\tilde{t} \to t$. Figure 2 depicts the generation process of GPT-2.

In practice, obtaining clean in-domain text requires extreme efforts of designing heuristic rules. Nonetheless, the synthetic text from GPT-2 makes decent sense and can already provide useful signals to drive the learning process.

4.2 Cycle Consistency

The core idea of encouraging cycle consistency is that starting from one sample in a domain, the model first maps it into the other domain, then maps it back (He et al., 2016). The resulting sample should be identical to the original sample. Specifically, let $p_\theta(t|d)$ be the probability distribution to map a data sample d to its corresponding text t, and $p_\phi(d|t)$ be the probability distribution to map text back to data. Starting from a data sample $d \in D$, its objective is:

$$\max_{\phi} \mathbb{E}_{d \sim p(D)} \log p_\phi(d|t'); \; t' \sim p_\theta(t|d) \qquad (1)$$

which basically ensures the consistency in the direction of $d \to t' \to d$. Note that only p_ϕ is updated in this direction and p_θ serves only as as an auxiliary function to provide pseudo samples t' from d. Though it is also possible to update θ at the same time through tricks like Gumbel-softmax (Jang et al., 2016) or REINFORCE (Williams, 1992), we find it did not lead to better performance, yet complicated the training. Similar observations have

[1] We force $k > 0$ as we believe a reasonable generation system needs a least a few demonstrations of the annotation.

[2] We adopt the Top-k random sampling setting with $k = 2$ to encourage diversity and reduce repetition (Radford et al., 2019)

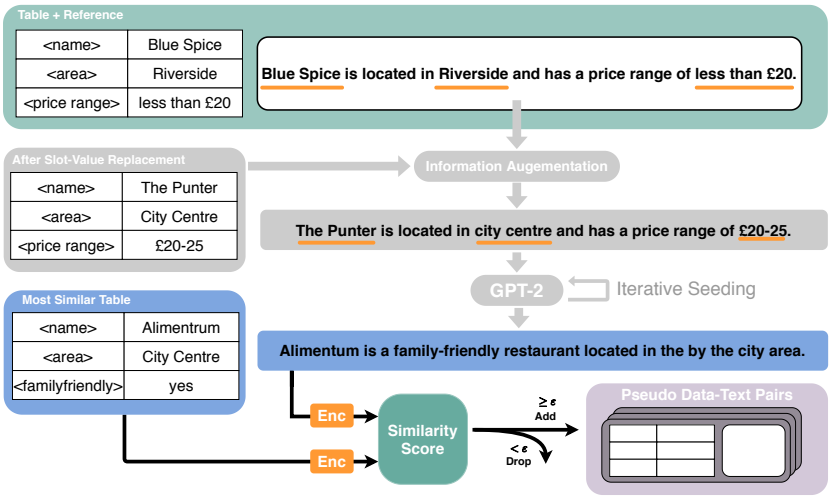

Figure 2: Depiction of *text augmentation* and *representation matching*. Each text sample first goes through information augmentation by slot-value replacement, then passed to GPT-2 with iterative conditional generation. The augmented text samples are paired with the most similar data from the corpus with a threshold cutoff.

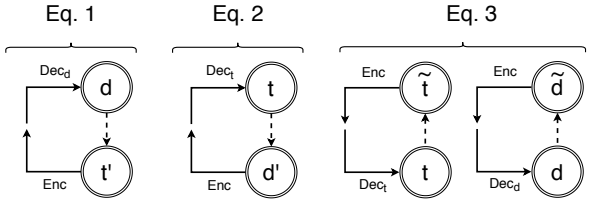

Figure 3: Four directions of cycle consistency. Gradients are backpropagated only through solid lines.

been made in Lample et al. (2018); He et al. (2020); Garcia et al. (2020).

Similarly, starting from a text $t \in T$, the objective is to ensure the consistency in the direction of $t \rightarrow d' \rightarrow t$:[3]

$$\max_{\theta} \mathbb{E}_{t \sim p(T)} \log p_\theta(t|d'); \; d' \sim p_\phi(d|t) \quad (2)$$

Finally, we further add two denoising autoencoding objectives on both the data and text sides:

$$\max_{\theta,\phi} \mathbb{E}_{d \sim p(D), t \sim p(T)} \log p_\phi(d|\tilde{d}) p_\theta(t|\tilde{t}) \quad (3)$$

where $\tilde{d}$ and $\tilde{t}$ are the corrupted versions of d and t. We use the same noise function as in Lample et al. (2018) which randomly permutes and pads a portion of the input. This can encourage the encoder to learn meaningful latent representations by reconstructing the input itself (Currey et al., 2017; Lample et al., 2018).

Figure 3 illustrates all the four directions of the cycle consistency objective.

We use one shared encoder Enc for both the data and text sides. Each data sample is flattened into a sequence by making a list of slot value pairs and fed into the same encoder. Using the same encoder for both types of input gives the model an inductive bias to project similar data/text into surrounding latent space.

We will show later that encoder sharing is essential for a good performance under the few-shot scenario. From the shared encoded space, two separate decoders Dec_d and Dec_t are used to decode d and t respectively[4].

4.3 Representation Matching

Apart from training under the cycle consistency, we further consider matching each synthetic text with its most similar data sample and treating them as supplementary training pairs. Compared with the pseudo d' obtained from back translation (Eq. 2), the matched data samples are extracted from the existing corpus D_U and thereby are guaranteed to be clean. This can provide a much more stable training signal especially at the initial training stage[5]. Previous work has used representation matching to automatically extract pseudo training pairs for machine translation (Artetxe and Schwenk, 2019; Ruiter et al., 2019). Baziotis et al. (2019); Chu and

[3] In the MT community, the equivalent step is usually called *back translation* (Sennrich et al., 2016; Lample et al., 2018).

[4] The shared encoding has also been shown effective in other tasks like machine translation (Lample et al., 2018) and image transition (Zhu et al., 2017). We further tried sharing the decoder as in Johnson et al. (2017) but find no improvement (see Table 2).

[5] In theory, as we can fabricate arbitrary possible data samples from the predefined schema and add to the corpus, we can always find one matched data for a text samples.

Liu (2019) also demonstrate that the representation similarity between input-output pairs can serve as a useful regularization for unsupervised text summarization. We adopt a similar idea to create pseudo pairs based on their cosine similarity in the representation space. To summarize, the process of representation matching can be described as:

$$\max_{\theta,\phi} \mathbb{E}_{t\sim p(T')} \mathbb{1}_{cos(d^*,t)>\varepsilon} (\log p_\theta(t|d^*) + \log p_\phi(d^*|t)); \quad (4)$$
$$d^* = \arg\max_{d\in D} \cos(d, t)$$

where T' is augmented text from the LM and $\mathbb{1}$ is the indicator function. We also perform mean pooling over the encoded representations before matching them. ε is a threshold. Pseudo pairs with a cosine similarity less than ε will be discarded. Ideally, as the encoder improves, the pseudo pairs created by representation matching will make more sense, which can in turn benefit the training of the encoder.

4.4 Summary

Apart from the above unsupervised objective, on the few annotated data-text pairs, we can impose the supervised objective:

$$\max_{\theta,\phi} \mathbb{E}_{d,t\sim p(D_L,T_L)} \log p_\theta(t|d) + \log p_\phi(d|t) \quad (5)$$

where (D_L, T_L) contains the k data annotations. Putting all together, we summarize it in Algorithm 1. In the training stage, we optimize the objectives of cycle consistency, representation matching and supervised learning sequentially to maintain a constant ratio of signals from all sides.

Algorithm 1 Few-shot Data-to-text Framework

Input: $D_U, (D_L, T_L)$
Create (D_a, T_a) by information augmentation ;
$(D_L, T_L) \leftarrow (D_L, T_L) \cup (D_a, T_a)$;
Create T' by LM augmentation ;
$T \leftarrow T_L \cup T'$;
repeat
  Sample batch data from (D_L, T) ;
  Cycle consistency:
  Optimize by Eq. 1 + Eq. 2 + Eq. 3;
  Representation Matching:
  Optimize by Eq. 4;
  Supervised Training:
  Optimize by Eq. 5;
until convergence

5 Experiment Setting

Data We conduct experiments on the E2E (Novikova et al., 2017) and WebNLG (Colin et al., 2016) datasets. E2E is a crowd-sourced dataset containing 50k instances in the restaurant domain. The inputs are dialogue acts consisting of three to eight slot-value pairs. WebNLG contains 25k instances describing entities belonging to fifteen distinct DBpedia categories. The inputs are up to seven RDF triples of the form *(subject, relation, object)*.

Configuration The model is implemented based on fairseq (Ott et al., 2019). We use 600-dimensional token embedding and Adam optimizer with initial learning rate at 0.0002. Batch size is kept at 48 with a dropout rate at 0.3. We employ beam search with size 3 for decoding and select models based on BLEU-4 scores on the development set. The score is averaged over 10 random initialization runs. In this work, the seq2seq models are built upon the long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997). For LSTM cells, both the encoder and decoder have 3 layers, amounting to 18M parameters for the seq2seq model (600-dimension and 1024 hidden units). Maximum sequence length is set as 100 for E2E and 200 for WebNLG (SPM-based). All encoder parameters are shared between data and text samples. All models were trained on 1 Nvidia V100 GPUs (32GB and CUDA Version 10.2) for 4k steps. The total batch size is around 48K tokens per GPU and we use the Adam optimizer ($\epsilon = 1\text{e}{-6}$, $\beta_2 = 0.98$) along with linear learning rate decay scheduling. The total number of updates is set to 8000 for all training and models are selected based on optimal validation BLEU4. At decoding time, sentences are generated using greedy decoding.

6 Results and Analysis

In this section, we present experiment results and analysis. We first compare our model with other baselines on both datasets, then perform a set of ablation studies on the E2E dataset to see the effects of each component. Finally, we analyze how text augmentation helps improves the model, include example outputs and show the human evaluation results in the end.

Model	E2E - 10%				E2E - 100%			
	BLEU	NIST	METEOR	ROUGE-L	BLEU	NIST	METEOR	ROUGE-L
SLUG	-	-	-	-	66.19	8.61	44.54	67.72
Seq2seq	53.38	6.10	38.10	60.53	63.32	6.81	41.25	62.91
Qader et al. (2019)	58.10	6.24	41.32	62.84	64.20	7.14	44.68	65.31
Chen et al. (2020)	59.10	7.49	40.25	63.23	63.72	7.76	40.25	66.23
Proposed (LSTM)	64.24	7.71	43.53	66.81	68.88	8.89	48.53	72.12
Model	WebNLG - 10%				WebNLG - 100%			
	BLEU	NIST	METEOR	ROUGE-L	BLEU	NIST	METEOR	ROUGE-L
Melbourne	-	-	-	-	44.93	8.98	36.58	60.40
Seq2seq	36.54	7.3	35	54.61	44.60	8.49	38.23	59.67
Qader et al. (2019)	38.66	7.81	34.1	56.95	47.19	8.71	37.90	58.61
Chen et al. (2020)	39.40	7.84	37.25	56.23	46.15	8.52	39.1	58.5
Proposed (LSTM)	43.75	8.29	33.58	58.49	50.26	8.86	40.71	61.29

Table 1: Performance on E2E and WebNLG with 10% and 100% data. Qader et al. (2019) utilizes all ground-truth unpaired text samples while our proposed model only gets access to the few-shot data annotations.

Comparison with Other Models In Table 1, we compare our model with (1) seq2seq baseline, (2) cycle consistency model as in Qader et al. (2019)[6] and (3) finetuned GPT-2 model as in Chen et al. (2020)[7]. For all models, we try running with 10% and 100% annotations to see how they perform under different data sizes. Our model is implemented both with LSTM encoder-decoders, same as the seq2seq baseline for a fair comparison. Note that Qader et al. (2019) further *utilized all the ground-truth unpaired text samples*, while the other models run only on the few-shot annotations. We also include the results of **SLUG** (Juraska et al., 2018) and **MELBOURNE** (Gardent et al., 2017), the overall winner on automatic metrics in the E2E and WebNLG challenge respectively(both seq2seq-based). SLUG uses a heuristic slot aligner based on a set of handcrafted rules and combines a complex pipeline of data augmentation, selection, model ensemble and reranker.

The results show that our proposed model significantly improves over the baseline on both the few-shot and fully supervised setting. The improvement is more evident when only 10% annotations are available, with a leap of 11 and 7 BLEU scores on E2E and WebNLG respectively. It also outperforms systems relying on task-dependent heuristics. In comparison, Qader et al. (2019), though with access to all text samples at all percentages, still underperforms our model with tangible margin. On the fully supervised setting, it brings little to no difference compared with the seq2seq baseline as no more extra data is incorporated in the training process. As such, we also observe that the text augmentation from finetuned GPT-2 model helps the proposed model on the few-shot setting, but its advantage also vanishes when all data annotations are available.

[6]The author did not open-source their code. We reproduced their model based on our implementation. The results on 10k annotations matches their reports in the paper.

[7]https://github.com/czyssrs/Few-Shot-NLG

Model/Share	None	Enc	Dec	Both
Supervised	53.20	-	-	-
+ $t \rightarrow d' \rightarrow t$	53.19	53.28	53.17	**53.29**
+ $d \rightarrow t' \rightarrow d$	53.15	**56.12**	53.49	56.07
+ $t \rightarrow t$	53.74	**56.39**	55.29	55.73
+ $d \rightarrow d$	53.37	**56.44**	56.09	56.11
+ Noise	54.13	**57.37**	56.59	57.04

Table 2: Ablation study for cycle consistency (10% annotations). BLEU-4 score is reported. Each line adds one condition on top of the previous one. **Supervised** is a supervised seq2seq baseline.

In Figure 4, we draw the model performance with varying number of data annotations. All models are trained from scratch with 10 different random initializations and the standard deviation of the BLEU-4 score is visualized. We can see our model (LSTM-based), though with a relatively larger standard deviation due to the uncertainty of text augmentation sampling, still consistently outperforms other baselines significantly and even surpasses the fully supervised seq2seq model with less than 10% of data annotations.

Ablation Study on Cycle Consistency In Table 2, we study how the four directions, input noise and parameter sharing affect the performance of cycle-consistency. The experiments are conducted with 10% annotations and *no further unpaired text*

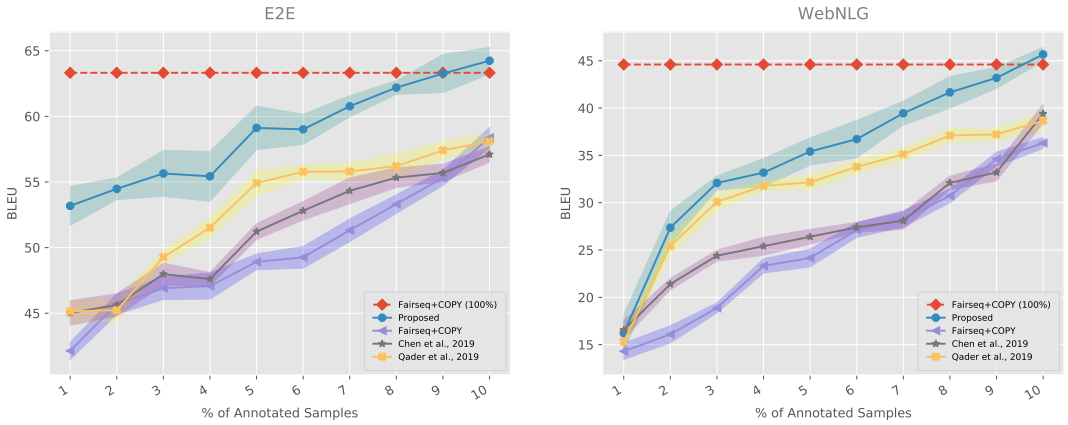

Figure 4: Model performance with varying number of data annotations. Our model with 10% annotations outperforms the seq2seq model trained on 100% pairs (dotted line) on both datasets. Shades are the sample standard deviation based on 10 runs of different model initializations.

samples are available.

As can be observed, adding the training direction $t \rightarrow d' \rightarrow t$ (i.e. back translation) has little effects on top of the supervised seq2seq baseline. This is expected since back translation is naturally designed to incorporate additional unpaired text samples. When run only on the few-shot annotations, its power is very limited. The backward direction $d \rightarrow t' \rightarrow d$ is surprisingly useful when the encoder is shared between the data and text. Though this direction will not affect the text decoder at all, the improvement suggests the model can benefit a lot by simply structuring its encoded space and mapping aligned data-text pairs to similar vector space. The autoencoding directions brings a little improvement. When combined with input noise, the performance further increases. This is similar to previous findings that denoising autoencoding is more helpful in inducing meaningful latent space (Lample et al., 2018) in comparison to simply learning to copy the original input.

The results also suggest encoder sharing is important for the cycle consistency objective to work in our few-shot setting. Decoder sharing, in contrast, makes little or even negative influence. This is kinda similar as in multilingual machine translation where sharing the decoder among languages might negatively interfere with the performance (Johnson et al., 2017).

Ablation Study on Text Augmentation On top of the four-direction cycle consistency training, we study the effects of text augmentation in Table 3. We compare our proposed info + LM augmentation with (1) random augmentation, where a random text from Wikipedia is sampled to the augmented text space, (2) unsupervised data augmentation (Xie et al., 2019) where text samples are augmented with paraphrases of current annotations and (3) ground-truth augmentation with reference obtained from the left training corpus, which can serve as an upper bound of text augmentation techniques. We test the performance with 1%, 5%, 10% and 20% annotations to see the effects with varying number of supervisions.

As can be seen, the random augmentation even

Text Augmentation	1%	5%	10%	20%
None	44.18	50.22	57.37	63.28
Random	41.30	49.62	57.71	62.79
UDA	44.24	50.09	57.66	61.30
Info	45.63	52.22	58.80	63.22
+ LM	48.67	53.53	59.04	**64.78**
Reference	**55.33**	**54.92**	**59.11**	64.26
Random (+RM)	42.32	50.10	58.53	63.29
UDA (+RM)	44.32	52.22	58.80	61.27
Info (+RM)	48.63	56.66	60.80	63.52
+ LM (+RM)	53.18	59.12	64.24	**65.38**
Reference (+RM)	**62.74**	**63.28**	**64.93**	65.27

Table 3: Ablation study for text augmentation with varying number of annotations. Experiments are performed on the E2E dataset. **LM** augmentation outperforms **Random** by a large margin, and even outperforming augmentation with ground-truth references on some occasions. Representation matching (**RM**) boosts the overall performance further.

Model	E2E			WebNLG		
	Fluency	Miss	Wrong	Fluency	Miss	Wrong
Seq2Seq	3.68	49	63	3.95	57	48
Cycle-only	4.08	46	66	4.23	48	44
Finetune GPT-2	4.21	43	57	4.10	39	45
Proposed (LSTM)	**4.33**	**39**	**44**	**4.27**	**31**	**39**

Table 4: Human Evaluation on the sampled outputs (100 instances) for models with 10% annotated data. Cycle-only indicates the approach in Qader et al. (2019); and Finetuned GPT-2 is refers to Chen et al. (2020).

Data:
[name] Blue Spice [eat type] restaurant [food] Chinese [area] city centre [family friendly] no [near] Rainbow Vegetarian Café
Reference: at the [city centre], there is a [restaurant] called the [Blue Spice].

seq2seq/info-aug: Blue Spice restaurant near Rainbow Vegetarian Café has a *5 star rating. prices start at £30.*
+LM-aug: located near Rainbow Vegetarian Café is a Chinese theme eatery and restaurant called Blue Spice. It is in the city centre area.

Figure 5: Generation examples with different text augmentation techniques. Trained on 5 data annotations (see the above toy training set). The attribute combination of input data is unseen in the 5 annotations. Hallucinated contents are *italitized*.

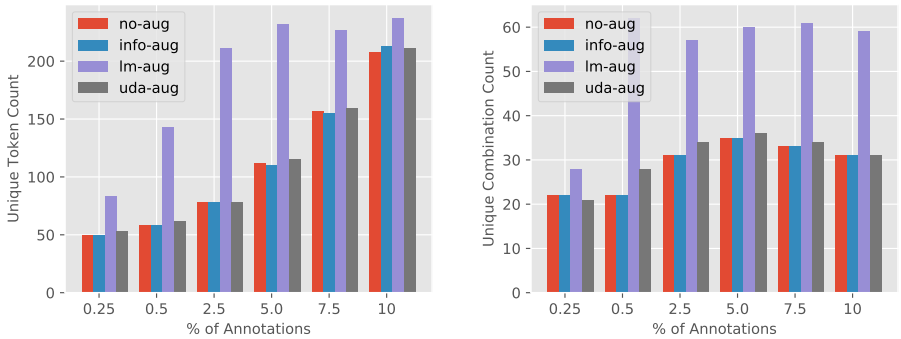

Figure 6: Additional number of decoded unique tokens (non-copied) and unique combinations of information on the testset with varying number of annotations.

harms the model performance, suggesting reasonable in-domain text augmentation are necessary for the model improvement. UDA augmentation also makes rather little difference as it simply paraphrases the current available annotations but cannot bring any new information. The information augmentation by slot-value replacement helps improve a bit. When combined with LM, the performance can be further boosted, especially for lower-resource scenarios. The representation matching always helps lift the performance, with gains of up to 10 BLEU points. As expected, the benefit from text augmentation gradually vanishes as more annotations are collected, especially for datasets with relatively simple patterns as E2E.

How text augmentation helps Intuitively the GPT-2 augmentation is expected to impose new tokens and combination patterns to the few-shot annotations. To investigate whether this is the case, for the decoded text in the test phase, we count the number of unique tokens (excluding copied data values) and unique information combination patterns (attribute combinations in E2E). The results in Fig. 6 show that LM-augmentation indeed greatly enriches the vocabulary space, even doubling the generated unique tokens in low-resource scenarios. The same happens for new combination patterns. In contrast, *info-aug* cannot insert new tokens or combinations at all since all it does is replacing data values based on the same text annotation. UDA can impose new tokens by paraphrasing the annotations, but it hardly helps the model generalize to new combinations of information. Moreover, when trained on a toy dataset, we observe from the generation outputs that Seq2seq and info-aug produce the wrong outputs and overfit to the information in the 5 training instances. With LM augmentation, it adapts to the new combination and connects information correctly. Figure 5 shows a generation example with different text augmentation techniques. We train the systems in a toy setting with only 5 data annotations (Trainset in the Appendix). We pick an input data with an unseen attribute combination to test if models can generalize correctly. Seq2seq and info-aug produce the wrong generation overfit to the information in the 5 training instances. With LM augmentation, it adapts to the new combination and connects information correctly.

Human Evaluation We further run a human evaluation on the model outputs to closely check the generation quality. We compared four types of models: the seq2seq baseline, seq2seq plus cycle-consistency as in Qader et al. (2019), finetuned GPT-2 as in Chen et al. (2020) and our proposed model. All models are LSTM-based apart from

the finetuned GPT-2 one. We sample 100 data instances from the test set and apply all the models to generate corresponding text. The data and generated text are evaluated by 50 crowdworkers on Prolific[8]. For each data-text pair, the annotator is instructed to evaluate (1) if the text is fluent (score 0-5 with 5 being fully fluent), (2) if it misses information contained in the source data and (3) if it includes wrong information. The average fluency scores, count of information miss and wrong information are presented in Table 4. The scores are generally consistent with the automatic evaluation results, our proposed model outperforms other ones by a large margin, even though cycle-only can access all unpaired text and finetuned GPT-2 is significantly larger than our LSTM-based seq2seq. The generated text are more fluent, yet maintaining the information completeness and correctness to a large extent.

7 Conclusion

We study few-shot data-to-text generation with only limited annotated data. We propose text augmentation with slot-value replacement followed by GPT-2 generation. The augmented text, when combined with cycle consistency and representation matching, is shown to help the model to generalize to unseen new tokens and patterns of token combinations. With less than 10% annotations, it outperforms supervised seq2seq model trained on 100% annotations and is extensible enough to be combined with pretraining techniques.

Acknowledgements

This research was funded in part by the German Research Foundation (DFG) as part of SFB 248 "Foundations of Perspicuous Software Systems". We sincerely thank the anonymous reviewers for their insightful comments that helped us to improve this paper.

References

Shubham Agarwal, Marc Dymetman, and Eric Gaussier. 2018. Char2char generation with reranking for the e2e nlg challenge. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 451–456.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2017. Unsupervised neural machine translation. *arXiv preprint arXiv:1710.11041*.

[8] https://www.prolific.co/

Mikel Artetxe and Holger Schwenk. 2019. Margin-based parallel corpus mining with multilingual sentence embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3197–3203.

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 141–148, Ann Arbor, Michigan. Association for Computational Linguistics.

Christos Baziotis, Ion Androutsopoulos, Ioannis Konstas, and Alexandros Potamianos. 2019. Seqˆ3: Differentiable sequence-to-sequence-to-sequence autoencoder for unsupervised abstractive sentence compression. 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 673–681.

Paweł Budzianowski and Ivan Vulić. 2019. Hello, it's gpt-2-how can i help you? towards the use of pretrained language models for task-oriented dialogue systems. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 15–22.

Ernie Chang, Jeriah Caplinger, Alex Marin, Xiaoyu Shen, and Vera Demberg. 2020. Dart: A lightweight quality-suggestive data-to-text annotation tool. In *COLING 2020*, pages 12–17.

Ernie Chang, Vera Demberg, and Alex Marin. 2021a. Jointly improving language understanding and generation with quality-weighted weak supervision of automatic labeling. In *EACL 2021*.

Ernie Chang, Hui-Syuan Yeh, and Vera Demberg. 2021b. Does the order of training samples matter? improving neural data-to-text generation with curriculum learning. In *EACL 2021*.

Zhiyu Chen, Harini Eavani, Yinyin Liu, and William Yang Wang. 2020. Few-shot nlg with pretrained language model. *ACL*.

Eric Chu and Peter Liu. 2019. Meansum: a neural model for unsupervised multi-document abstractive summarization. In *International Conference on Machine Learning*, pages 1223–1232.

Emilie Colin, Claire Gardent, Yassine M'rabet, Shashi Narayan, and Laura Perez-Beltrachini. 2016. The WebNLG challenge: Generating text from DBPedia data. In *Proceedings of the 9th International Natural Language Generation conference*, pages 163–167, Edinburgh, UK. Association for Computational Linguistics.

Anna Currey, Antonio Valerio Miceli-Barone, and Kenneth Heafield. 2017. Copied monolingual data improves low-resource neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 148–156.

Markus Freitag and Scott Roy. 2018. Unsupervised natural language generation with denoising autoencoders. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3922–3929.

Xavier Garcia, Pierre Foret, Thibault Sellam, and Ankur P Parikh. 2020. A multilingual view of unsupervised machine translation. *arXiv preprint arXiv:2002.02955*.

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.

Dimitra Gkatzia. 2016. Content selection in data-to-text systems: A survey. *arXiv preprint arXiv:1610.08375*.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *Advances in neural information processing systems*, pages 820–828.

Junxian He, Xinyi Wang, Graham Neubig, and Taylor Berg-Kirkpatrick. 2020. A probabilistic formulation of unsupervised text style transfer. *ICLR*.

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

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Juraj Juraska, Panagiotis Karagiannis, Kevin Bowden, and Marilyn Walker. 2018. A deep ensemble model with slot alignment for sequence-to-sequence natural language generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 152–162.

Ravi Kondadadi, Blake Howald, and Frank Schilder. 2013. A statistical nlg framework for aggregated planning and realization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1406–1415.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, et al. 2018. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5039–5049.

Rémi 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.

Chris van der Lee, Emiel Krahmer, and Sander Wubben. 2018. Automated learning of templates for data-to-text generation: comparing rule-based, statistical and neural methods. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 35–45.

Shuming Ma, Pengcheng Yang, Tianyu Liu, Peng Li, Jie Zhou, and Xu Sun. 2019. Key fact as pivot: A two-stage model for low resource table-to-text generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2047–2057.

Jekaterina Novikova, Ondřej Dušek, 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.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Tatsuro Oya, Yashar Mehdad, Giuseppe Carenini, and Raymond Ng. 2014. A template-based abstractive meeting summarization: Leveraging summary and source text relationships. In *Proceedings of the 8th International Natural Language Generation Conference (INLG)*, pages 45–53.

Yannis Papanikolaou and Andrea Pierleoni. 2020. Dare: Data augmented relation extraction with gpt-2. *arXiv preprint arXiv:2004.13845*.

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*.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. 2018. Style transfer through back-translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 866–876.

Raheel Qader, François Portet, and Cyril Labbé. 2019. Semi-supervised neural text generation by joint learning of natural language generation and natural language understanding models. In *Proceedings of*

the 12th International Conference on Natural Language Generation, pages 552–562.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge university press.

Dana Ruiter, Cristina España-Bonet, and Josef van Genabith. 2019. Self-supervised neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1828–1834, Florence, Italy. Association for Computational Linguistics.

Martin Schmitt and Hinrich Schütze. 2019. Unsupervised text generation from structured data. *arXiv preprint arXiv:1904.09447*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 86–96.

Shang-Yu Su, Chao-Wei Huang, and Yun-Nung Chen. 2020. Towards unsupervised language understanding and generation by joint dual learning. *ACL*.

Sandeep Subramanian, Guillaume Lample, Eric Michael Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2018. Multiple-attribute text style transfer. *arXiv preprint arXiv:1811.00552*.

Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. 2020. {LAMAL}: {LA}nguage modeling is all you need for lifelong language learning. In *International Conference on Learning Representations*.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

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.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. 2019. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232.