ASDOT: Any-Shot Data-to-Text Generation with Pretrained Language Models

Anonymous EMNLP submission

Abstract

Data-to-text generation is challenging due to the great variety of the input data in terms of domains (e.g., finance vs sports) or schemata (e.g., diverse predicates). Recent end-to-end neural methods thus require substantial training examples to learn to disambiguate and describe the data. Yet, real-world data-to-text problems often suffer from various data-scarce issues: one may have access to only a handful of or no training examples, and/or have to rely on examples in a different domain or schema. To fill this gap, we propose *Any-Shot Data-to-Text* (ASDOT), a new approach flexibly applicable to diverse settings by making efficient use of any given (or no) examples. ASDOT consists of two steps, *data disambiguation* and *sentence fusion*, both of which are amenable to be solved with off-the-shelf pretrained language models (LMs) with optional finetuning. In the *data disambiguation* stage, we employ the prompted GPT-3 model to understand possibly ambiguous triples from the input data and convert each into a short sentence with reduced ambiguity. The *sentence fusion* stage then uses an LM like T5 to fuse all the resulting sentences into a coherent paragraph as the final description. We evaluate extensively on various datasets in different scenarios, including the zero-/few-/full-shot settings, and generalization to unseen predicates and out-of-domain data. Experimental results show that ASDOT consistently achieves significant improvement over baselines, e.g., a 30.81 BLEU gain on the DART dataset under the zero-shot setting.

1 Introduction

Data-to-text generation (Kukich, 1983a; Reiter and Dale, 1997) aims at generating natural language text conditioned on structured data content such as tables and graphs. The task has a broad range of applications such as task-oriented dialog (Wen et al., 2015), weather forecasting (Goldberg et al., 1994; Sripada et al., 2003), sports news reporting (Wiseman et al., 2017), and biography generation (Lebret et al., 2016a; Wang et al., 2018).

The problem is challenging in practice due to the vast diversity of the input data in terms of the domains (e.g., finance vs sports), schemata (e.g., the set of predicates, table structures), etc. The inherent ambiguity makes it particularly difficult to learn to understand and describe the data. For instance, in the tuple `<Fearless, time, 2008>` from a music domain, the predicate word `time` means the release time of an album, while in `<100 metres, time, 9.58>` from sports it expresses the world record time. Recent approaches based on end-to-end neural models, e.g., by finetuning pretrained language models (LMs) (Puduppully et al., 2019a; Koncel-Kedziorski et al., 2019; Zhao et al., 2020), typically require massive training instances to resolve the ambiguity and are not applicable to many data-scarce scenarios.

In practice, a data-to-text problem of interest may have a varying number of training examples, ranging from a (small) set to only a few shots, or even no examples at all, and sometimes may rely on available examples out of the current domain to facilitate the generation. We refer to the diverse practical scenarios as the *any-shot* data-to-text problems. Recent work has studied data-to-text solutions when limited examples are available, but is often restricted to single specific settings. For instance, Chen et al. (2020b) and Su et al. (2021) focused on few-shot problems but fail to apply when no examples are accessible, while the zero-shot neural pipeline by Kasner and Dusek (2022) is not capable of using training examples for further improvement, nor could it handle out-of-domain data due to the reliance on human-crafted templates.

In this paper, we develop *Any-Shot Data-to-Text* (ASDOT), a new flexible approach that makes efficient use of any given (or no) examples and achieves stronger generation quality compared to the prior specific methods. ASDOT draws inspira-

tion from how humans describe data, namely by first disambiguating and understanding the data content, and then fusing and organizing the information together into text paragraphs. As a result, given input data (e.g., a table or graph), ASDOT consists of two intuitive steps, i.e., *data disambiguation* and *sentence fusion*. Importantly, each of the two steps is amenable to be solved with the appropriate off-the-shelf pretrained LMs with optional finetuning, enabling the unique flexibility of ASDOT in the presence of any-shot training examples. More specifically, in data disambiguation aiming to understand each data entry (e.g., triple `<Fearless, time, 2008>`), we use the prompted GPT-3 model (Radford et al., 2019), which has encoded rich commonsense and world knowledge, to convert the triple into a short sentence (`Fearless was released in 2008`) with greatly reduced ambiguity. The subsequent sentence fusion stage then uses another LM, such as T5 (Raffel et al., 2020), to combine all the resulting sentences into a coherent paragraph as the final description. The sentence fusion as a sub-task allows us to incorporate any available in-/out-of-domain training examples as well as existing large weakly supervised corpus (Kasner and Dusek, 2022) to finetune the LM and boost the performance.

We evaluate the proposed approach in a wide range of practical any-shot scenarios, including (1) the *zero-/few-/full-shot* setting where we have access to a varying number of training examples, (2) the *unseen-predicates* setting where we describe the data of new predicates that are never seen in the training examples, and (3) the *out-of-domain* setting where we are presented only with examples from other domains. Extensive experiments show that our approach consistently achieves significant gains over the diverse previous methods specifically designed for each of the different scenarios.

2 Related Work

Data-to-text (D2T) generation is a long-standing problem in natural language processing with broad applications in practice. Early research on this task focused on rule-based and pipeline approaches (Kukich, 1983b; Reiter and Dale, 1997), decomposing the task into text planning, sentence planning, and linguistic realisation (Reiter and Dale, 1997). Recent work has developed various neural approaches. Lebret et al. (2016b) used a neural encoder-decoder for the task, followed by attention (Bahdanau et al., 2015), content selection (Puduppully et al., 2019a), and entity modeling (Puduppully et al., 2019b) for further improved performance. Recent studies have also incorporated pretrained LMs (Kale and Rastogi, 2020b; Ribeiro et al., 2021; Clive et al., 2021). Although previous fully-supervised methods have achieved remarkable performances, most of them require a large amount of in-domain training examples, leading to limited applicability to the common low-data scenarios in practice.

Recent interests are aroused in zero-/few-shot data-to-text generation problems. Chen et al. (2020b) first formulated the few-shot setting and incorporated a pretrained model with a pointer generator as a solution. Chen et al. (2020a) developed a knowledge-grounded pretrained LM for both zero- and few-shot data-to-text generation. Gong et al. (2020) and Chen et al. (2020b) proposed to solve the few-shot task with content matching and prototype memory, respectively. There are also studies on combining templates and pretrained LM for zero-/few-shot generation. For example, Kale and Rastogi (2020a) trained a neural model to rewrite templates for few-shot task-oriented dialogue. Heidari et al. (2021) applied the idea of template rewriting to build a practical few-shot data-to-text system. Kasner and Dusek (2022) proposed a neural pipeline for zero-shot data-to-text generation, which rephrases templates with general-domain text-based operations. Most of the previous methods have each focused on a specific setting (e.g., either zero- or few-shot). In comparison, our work studies a wide spectrum of any-shot scenarios with a varying number of training examples from current or different domains. We develop a new data-to-text approach that is generally applicable and excels in the various settings.

3 Any-Shot Data-to-Text Generation

We propose ASDOT for any-shot data-to-text generation. §3.1 describes the any-shot problems. We then provide an overview of our method (§3.2) and give details of each of the components (§3.3, 3.4). Figure 1 illustrates our method.

3.1 The Any-Shot Data-to-Text Problems

In the data-to-text generation task, we are given structured data (e.g., a table or graph) as input, which can be represented as a set of triples $\{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n\}$. Each triple $\boldsymbol{x}_i = \langle s_i, p_i, o_i \rangle$,

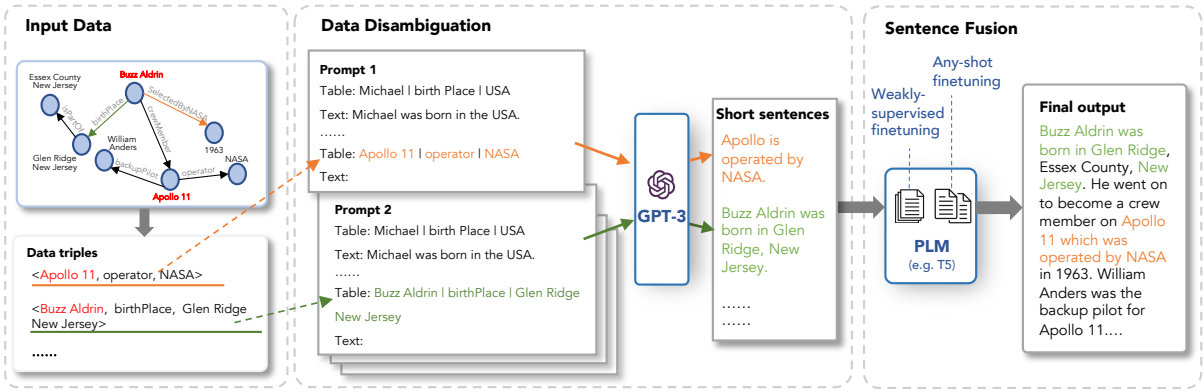


Figure 1: An overview of our method. Our approach consists of two core steps, i.e., *data disambiguation* (§3.3) and *sentence fusion* (§3.4). The approach first leverages a prompted GPT-3 to convert each data triple into short sentences with reduced ambiguity. The resulting sentences are then fused by a pretrained LM with optional finetuning using public weakly-supervised corpus or available training examples.

such as <Apollo 11, operator, NASA> as in Figure 1, consists of a subject s_i, a predicate p_i, and an object o_i, which expresses a relation between the subject and the object. The goal of the task is to generate a paragraph consisting of a sequence of words $\boldsymbol{y} = \{y_1, y_2, ..., y_m\}$ that can describe the input data faithfully and fluently.

Due to the vast diversity of the content domains, data structures, and predicate sets, etc., building a data-to-text solution often suffers from insufficient training examples for learning to understand/describe the target data. In practice, most often we are presented with a varying number of labeled examples, directly or remotely related to the target data. For instance, we may need to describe a table from a financial report on a new website, where we have no access to any labeled examples (i.e., zero-shot) or have access to only a few description examples (i.e., few-shot). Besides, the available examples may not even be in the financial domain (out of domain), or uses different table structures (different schemata) and different table headers (different predicates). We refer to the data-to-text training in the various practical scenarios as the *any-shot* problem. It is highly desirable to develop a general approach that is widely applicable to the different settings.

3.2 Method Overview

Intuitively, a data-to-text generation process consists of two core steps, namely, (1) disambiguating and understanding the data triples, and (2) producing the text description. Previous neural approaches typically model the task in an end-to-end manner and require a large number of training examples to learn the data-to-text mapping. In contrast, we take advantage of the task structure by formulating the two stages and solving each with appropriate resources (e.g., pretrained LMs) that are readily available. Figure 1 offers an overview of the approach. Specifically, since each data triple is inherently ambiguous given the compact predicate words, rich commonsense and world knowledge is required to correctly understand the content. For instance, in <Apollo 11, operator, NASA>, a model would need knowledge to determine that NASA operates Apollo 11 rather than the other way around. Therefore, in the data disambiguation stage, we leverage a powerful LM—GPT-3 in our case—that contains massive implicit knowledge in the parameters, to convert each triple into short sentences with reduced ambiguity (e.g., Apollo is operated by NASA). Once we collect a set of short sentences, in the sentence fusion stage, we use another pretrained LM with optional finetuning to compose the sentences into a well-formed paragraph. The stage offers the flexibility to make use of any available training example to boost performance.

3.3 Data Disambiguation

In this stage, the goal is to generate a short sentence to describe each data triple precisely. As above, a triple can be highly abstract and ambiguous as it compresses complex relational information into the compact format $\mathbf{x} = \langle s, p, o \rangle$, where the predicate p is often a concise word or phrase (e.g., the predicate time in triple <Fearless, time, 2008>). To reduce the ambiguity, we want to "recover" the missing information in the triple by augmenting it into a complete sentence (e.g., Fearless was

released in 2008). Another advantage of converting the structured triples into the free-form text is that a text sequence is more amenable to the LMs used in the subsequent sentence fusion stage (§3.4) as described shortly.

As the above examples show, augmenting a triple into a sentence naturally requires relevant external knowledge (e.g., Fearless is an album). Training a model specifically for the task could be expensive and could easily overfit to the training domain. Instead, we resort to the general GPT-3 model. Specifically, as shown in Figure 1 (middle panel), we provide GPT-3 with a few demonstrations of converting triples into short sentences, and then feed the target triple to elicit the desired sentence. Appendix A shows the complete demonstrations. We found that the same set of four demonstrations is sufficient to be used for target data in any domain. We thus use the same prompt consisting of those demonstrations throughout our experiments.

Querying the GPT-3 API can be slow and expensive. Given a set of target data in a domain, we reduce the number of queries by generating *templates*. More concretely, for each predicate in the set, we sample one triple containing the predicate, and generate a sentence for the triple with GPT-3. Then we replace the subject and object in the sentence with placeholders <subject> and <object> to get a template. For instance, the template for the predicate birthPlace in Figure 1 is "<subject> was born in <object>". We then use the template to generate the sentences for all triples with the same predicate.

It is worth noting that many existing data-to-text approaches, ranging from the classical pipeline solutions (Reiter and Dale, 1997) to the recent neural methods (Kale and Rastogi, 2020a; Kasner and Dusek, 2022), have also included similar template components, while their templates are typically crafted by human annotators, making the approaches hard to apply to the diverse new domains. In contrast, our ASDOT is fully automated with the pretrained LMs, without the need of human efforts nor training examples.

3.4 Sentence Fusion

In the second stage, we aim to fuse the sentences from the last step and produce a final coherent and fluent paragraph as the output data description. We naturally formulate the sentence fusion as a sequence-to-sequence problem, and use the pretrained LMs, particularly T5 (Raffel et al., 2020), as the backbone for solution. Specifically, we simply concatenate the short sentences, prepended with a prefix word "summarize:", and feed them into the T5 model to obtain the output text. We pick "summarize:" as the prefix for T5 to mimic its pretraining configuration, since the sentence fusion task is similar to the summarization task on which T5 was pretrained.

A key advantage of the sentence fusion stage is that the component permits easy finetuning with diverse available resources. On one hand, there are automatically constructed weak supervision datasets publicly available, such as WikiSplit (Botha et al., 2018) mined from Wikipedia's edit history and DiscoFuse (Geva et al., 2019) constructed by rules. In our zero-/few-shot experiments (§4), we finetune the sentence fusion model with the public WikiFluent dataset (Kasner and Dusek, 2022) which was constructed by applying a sentence splitting model on the Wikipedia sentences. On the other hand, one can also use any labeled data-to-text examples (by first converting with the data disambiguation stage), even if the examples are from different domains. This is because the general sentence fusion task tends to be domain-agnostic, since the operations to fuse sentences are usually similar across domains, e.g., by inserting connective words or subsuming one sentence as the clause of another. We evaluate in our experiments the out-of-domain generalization ability of our approach.

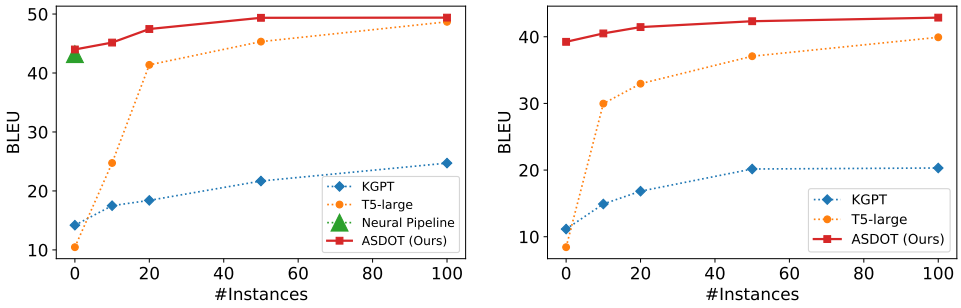
4 Experiments

We conduct extensive experiments on the various any-shot settings. All code and data will be released upon acceptance.

4.1 Datasets

We experiment on three widely-used data-to-text benchmarks based on which we study various any-shot settings.

WebNLG (Gardent et al., 2017) consists of data-text pairs where each data is a set of triples extracted from DBpedia and the text is written by human to describe the data. The dataset is split into training, validation, and test set, with 18,102/872/1,862 examples, respectively. The test set is further split into the test-seen and test-unseen subsets. The instances in the test-unseen set are from Wikipedia categories not seen in the training



(a) WebNLG (b) DART

Figure 2: Results of zero-/few-shot learning on WebNLG (left) and DART (right), respectively. The x-axis is the number of training examples, and the y-axis is the BLEU score. We report results of other metrics in Appendix C. Neural Pipeline (Kasner and Dusek, 2022) is applicable only to the zero-shot setting and the specific WebNLG data due to the need of human-written templates on the dataset. Our method show superior performances under any-shot settings. Our approach shows consistent improvement over the baselines, especially when the training size is small.

set, which is used in our "unseen predicates" experiments (§4.4). WebNLG contains 354 types of predicates in total.

DART (Novikova et al., 2017) is a large open-domain data-to-text corpus, constructed from WikiSQL (Zhong et al., 2017), WikiTableQuestions (Pasupat and Liang, 2015), as well as the WebNLG and E2E datasets. It contains 62,659/2,768/5,097 examples in the training/validation/test sets, respectively, and has 4,299 different predicates in total. To evaluate model generalization to unseen predicates, we extract a subset of 2,71 test examples whose predicates are completely unseen in the training/validation sets, leading to a more difficult test-unseen set compared to that of WebNLG.

E2E (Novikova et al., 2017) is a data-to-text corpus in the restaurant domain annotated by human. The dataset has 42,061/547/629 examples in the training/validation/test sets, respectively. The dataset is relatively easy since it only contains 7 types of predicates and has limited patterns.

4.2 Experimental Setup

For ASDOT, the data disambiguation stage (§3.3) uses the GPT-3 Davinci API provided by OpenAI, with greedy decoding, maximum generation length 256 and the stop token "\n". Please refer to Appendix A for the full prompt we use. For the sentence fusion stage (§3.4), we use T5 models of varying sizes as the sentence fusion LM. In the zero-/few-shot settings (§4.3), we finetune the T5 with the large weakly-supervised data WikiFluent (Kasner and Dusek, 2022) as mentioned in §3.4. We use the Adam optimizer (Kingma and Ba, 2015) with an initial learning rate of 3×10^{-5}, and use a batch size of 64, for 1 epoch. When any shot of labeled data-to-text examples are available, we further finetune the sentence fusion T5 with those examples. For the generation, we use beam search decoding with a beam width of 5. We provide more details of the experimental setup in the appendix A.

Evaluation Metrics Following previous studies, we report the performance in terms of BLEU (Papineni et al., 2002) and METEOR (Banerjee and Lavie, 2005) with gold references, as well as the recent PARENT-F1 metric (Dhingra et al., 2019) which measures the alignment between generated text with both the references and input data. We also perform human evaluation in the few-shot setting as detailed later.

4.3 Zero-, Few-, to Full-Shot Learning

We evaluate ASDOT in the presence of a varying number of training examples, ranging from 0, 10, 20, 50, 100 to the size of the full training set. We experiment on the WebNLG and DART datasets, respectively. In the zero-/few-shot settings, we use the T5-large model for our sentence fusion LM. In the full-shot setting, we test three T5 models of different sizes (small - 60M parameters, base - 220M, and large - 770M) for sentence fusion. Besides, the recent Prefix-Tuning method (Li and Liang, 2021) shows competitive performances on the data-to-text generation task. We thus also incorporate it with the T5-large architecture and report the results.

Baselines In the zero-/few-shot settings, we compare with **KGPT** (Chen et al., 2020a), a knowledge-grounded LM pretrained on large-scale automatically constructed data-to-text corpus, as it is one of the few methods applicable to both zero-/few-

Model	BLEU	METEOR	P-F1
BestPlan	47.24	39.00	-
Pipeline-Trans	51.68	32.00	-
PlanEnc	52.78	41.00	-
DataTuner_FC	52.40	42.40	-
T5-small	56.90	43.05	65.20
ASDOT-small	58.64 (+1.74)	43.47 (+0.42)	66.63 (+1.33)
T5-base	58.53	43.89	66.82
ASDOT-base	60.34 (+1.81)	44.37 (+0.48)	68.17 (+1.35)
T5-large	60.38	44.49	68.49
ASDOT-large	61.32 (+0.94)	**44.79** (+0.30)	**69.69** (+1.20)
Prefix-Tuning	61.03	44.37	69.17
ASDOT-Prefix	**61.38** (+0.35)	44.52 (+0.15)	69.39 (+0.22)

Model	BLEU	METEOR	P-F1
LSTM w attention	29.66	27.00	35.00
E2E Transformer	27.24	25.00	28.00
BART-base	47.11	38.00	55.00
BART-large	48.56	39.00	57.00
T5-small	47.53	39.00	59.33
ASDOT-small	49.32 (+1.79)	39.57 (+0.57)	60.95 (+1.62)
T5-base	49.62	39.69	61.11
ASDOT-base	49.85 (+0.23)	39.91 (+0.22)	61.64 (+0.53)
T5-large	50.17	40.00	61.72
ASDOT-large	**50.79** (+0.62)	**40.36** (+0.36)	**62.52** (+0.80)
Prefix-tuning	50.39	40.13	61.60
ASDOT-Prefix	50.56 (+0.17)	40.22 (+0.09)	62.27 (+0.67)

Table 1: Full-shot learning results on WebNLG (**Left**) and DART (**Right**). ASDOT-*X* denotes our approach with T5-*X* as the sentence fusion model. The best scores are in **bold**. We also show the performance gains against respective baseline models in green

shot data-to-text generation. We also compare with the end-to-end model based on **T5-large**, which has shown remarkable performance on data-to-text tasks with sufficient training examples (Ribeiro et al., 2020). Following Ribeiro et al. (2021), for the T5 baseline, we prepend <H>, <R> and <T> before the subjects, predicates, and objects, respectively, and add a prefix "`translate Graph to English:`" to the input. We finetune the T5 model with available shots of training examples. On the WebNLG dataset, we report another baseline **Neural Pipeline** (Kasner and Dusek, 2022), which is a template-based pipeline method also trained on the WikiFluent dataset and is applicable only to the zero-shot setting. However, the method cannot be used on the DART dataset since its templates are specifically written for WebNLG by human.

In the full-shot setting, we further compare with a wide range of previous full-shot state-of-the-art data-to-text systems, including BestPlan (Moryossef et al., 2019), Pipeline-Trans (Castro Ferreira et al., 2019), PlanEnc (Zhao et al., 2020), DataTuner_FC (Harkous et al., 2020) on WebNLG, and LSTM-with-attention, End-to-End Transformers, and BART-base/large (Nan et al., 2020) on DART.

Automatic Evaluation The zero-/few-shot results are shown in Figure 2. Our method consistently outperforms baseline models on both datasets, demonstrating its strong zero-/few-shot learning ability. In particular, with fewer training examples, our ASDOT tends to outperform other methods by a larger margin. For instance, we achieve 16.06 higher BLEU than T5-large on 10-shot WebNLG, and 10.53 higher on 10-shot DART. This is because the two-stage ASDOT is designed to excel in the low-data contexts by augmenting the generation process with rich external knowledge in pretrained LMs. Neural Pipeline is competitive with ours, but is restricted only to the zero-shot setting on WebNLG. DART contains more diverse types of predicates and thus is arguably more challenging than WebNLG. Our approach tends to achieve stronger performance gains on the difficult dataset.

We report the results of the full-shot setting in Table 1. The performance gain tends to be less significant compared to the zero-/few-shot settings as all methods are presented with a large number of training examples. However, our method still achieves consistently stronger performance over the large diversity of baselines, thanks to ASDOT's proper modeling of the generation process and the incorporation of rich external implicit knowledge.

Human Evaluation We conduct a human evaluation to further assess our ASDOT against other baselines under the 50-shot setting on WebNLG. After training, we sample 50 test instances and ask three proficient English speakers to score the model outputs. Following Chen et al. (2020b), each generated result is evaluated on three aspects: the number of the facts that are consistent with the input table

Model	Faithfulness ↑	Contradict ↓	Fluency ↑
KGPT	0.64	2.34	1.00
T5-large	2.22	0.72	1.58
ASDOT	**2.37**	**0.67**	**1.82**

Table 2: Human evaluation results. ↑ means the higher the better and ↓ means the lower the better. ASDOT outperforms the baselines with $p < 0.05$ in Tukey's HSD test for all the measures.

(*Faithfulness*) and contradicted to the table (*Contradict*), and the language fluency, on a 3-Likert scale (0,1,2). The results are shown in Table 2. The Krippendorff alphas (Krippendorff, 2011) for Faithfulness, Contradict, and language fluency are 0.49, 0.42 and 0.36, respectively, indicating a fair inner-annotator agreement. Consistent with the automatic evaluation results, we observe that ASDOT is substantially better than the baselines on all the three aspects, suggesting that our approach generates more faithful and fluent descriptions.

Ablation Studies We conduct ablation studies to investigate the effects of both the data disambiguation and sentence fusion stages. Table 3 shows the results. Specifically, for the sentence fusion stage, we study the effect of the weakly-supervised finetuning on the WikiFluent corpus (§3.4). From the table, we can see that the performance drops sharply without weakly-supervised finetuning, i.e., by 8.86 BLEU points for the zero-shot setting. However, ASDOT without weak supervision still outperforms the baselines in most cases, validating the strong advantage of our approach under low-data settings. For the data disambiguation stage, we investigate the impact of the automatic templates produced by GPT-3. More concretely, we replace the GPT-3 templates with the human-written templates from Kasner and Dusek (2022). The performance is similar or decreases slightly, demonstrating that the short sentences or templates automatically generated in the data disambiguation stage are of competitive or slightly higher quality than the manually created ones (perhaps due to human errors when writing the hundreds of templates).

4.4 Generating for Unseen Predicates

We now assess the model's capability of describing new predicates that are never seen during training. As mentioned in §4.1, WebNLG provides such an official test-unseen set for the evaluation and we

Model	0	10	20	50	100
KGPT	14.19	17.50	18.40	21.68	24.72
T5-large	10.46	29.10	41.38	46.24	48.68
ASDOT	**43.33**	**45.16**	**47.46**	**49.36**	**49.39**
- *w/o weak-sup*	34.47	39.38	43.67	47.56	48.16
- *w/ manual templ.*	42.02	43.37	46.12	48.28	48.32

Table 3: Ablation results (BLEU) for zero-/few-shot learning on WebNLG. The *w/o weak-sup* row shows the results of ASDOT without weakly supervised finetuning, and *w/ manual templ.* shows the results of using handcrafted templates in the data disambiguation stage.

Model	BLEU	METEOR	P-F1
BestPlan	34.41	37.00	-
Pipeline-Trans	38.92	21.00	-
PlanEnc	38.23	37.00	-
T5-small	47.34	39.95	57.99
ASDOT-small	50.75 (+3.41)	40.63 (+0.68)	61.20 (+3.21)
T5-base	51.11	41.42	60.94
ASDOT-base	54.51 (+3.40)	42.30 (+0.88)	64.36 (+3.42)
T5-large	53.97	42.37	63.81
ASDOT-large	55.74 (+1.77)	**42.94** (+0.57)	**65.90** (+2.09)
Prefix-Tuning	55.26	42.42	65.24
ASDOT-Prefix	**55.86** (+0.60)	42.73 (+0.31)	65.68 (+0.44)

Table 4: Results on WebNLG test-unseen set.

construct a similar (but more difficult) test set on DART where all the test predicates are not included in training. We train the models on WebNLG and DART, and evaluate on the corresponding test-unseen sets, respectively. As in §4.3, we compare ASDOT with the respective end-to-end T5 models (small, base, large, prefix-tuning). We also include the previously reported baseline results on the WebNLG test-unseen set, including BestPlan (Moryossef et al., 2019), Pipeline-Trans (Castro Ferreira et al., 2019) and PlanEnc (Zhao et al., 2020). The experimental results are shown in Table 4 and Table 5, respectively. As can be seen, our method achieves consistent improvements over all the baseline methods, showing the robustness of our method to unseen predicates given the rich commonsense and world knowledge introduced through the pretrained LMs in both stages. The superior performance of ASDOT over the corresponding end-to-end T5 again demonstrates the advantage of our modularization that applies to and improves various pretrained LMs. Similar as in the zero-/few-shot experiments, here we observe that on the more difficult DART test-unseen set

Model	BLEU	METEOR	P-F1
T5-small	37.65	33.27	43.79
ASDOT-small	46.60 (+8.95)	36.91 (+3.64)	52.17 (+8.38)
T5-base	46.13	36.97	49.79
ASDOT-base	50.90 (+4.77)	37.72 (+0.75)	54.98 (+5.19)
T5-large	46.37	36.49	50.32
ASDOT-large	50.70 (+4.33)	37.25 (+0.76)	55.49 (+5.17)
Prefix-tuning	47.07	36.69	49.67
ASDOT-Prefix	**51.99** (+4.92)	**38.11** (+1.42)	**57.26** (+7.59)

Table 5: Results on DART test-unseen set.

Test set	Model	B	M	P
E2E	T5-large	33.23	35.40	**60.18**
	ASDOT	**35.51** (+2.28)	**35.98** (+0.58)	60.06 (–0.12)
DART	T5-large	25.94	33.64	33.50
	ASDOT	**30.42** (+4.48)	**35.30** (+1.66)	**36.60** (+3.10)

Table 6: Out-of-Domain results. **B**, **M** and **P** represent BLEU, METEOR and PARENT-F1, respectively.

with more unseen predicates, our method achieves more significant gains than on WebNLG, which further shows the advantage of our method when generalizing to unseen predicates.

4.5 Learning with Out-of-Domain Examples

At last, we quantitatively measure the generalization ability of our approach across domains. To simulate the out-of-domain setting, we train our model on the WebNLG dataset and evaluate it on the test sets of DART and E2E, respectively. The DART test set includes the instances from the WebNLG and E2E test sets. We remove those instances to avoid any in-domain test examples (w.r.t the WebNLG training examples) and any overlap with E2E evaluation. We compare our method with the end-to-end finetuned T5-large model. The experimental results in Table 6 show that our method outperforms the baseline models on both out-of-domain test sets, echoing the conclusions in previous experiments that our approach with the two-stage design and integration of pretrained LMs has a superior generalization ability to handle data-to-text generation in any-shot scenarios.

4.6 Case Study

Table 7 shows the outputs of our ASDOT (based on T5-large) after the data disambiguation stage and the sentence fusion stage, on two data in the out-of-domain and unseen-predicates settings, respectively. The generated words corresponding to different data triples are highlighted in different colors (as in Figure 1). We also provide the results of the T5-large baseline and the human-written references. As can be seen, ASDOT develops a strong generalization ability to out-of-domain data and unseen predicates. In the first example, ASDOT successfully disambiguates the triple `<Zolder, fastest Lap, Liverpool F.C.>` into `"Liverpool F.C. set the fastest lap in the Zolder"` while the T5 baseline fails to do so and simply generates `"Zolder's faster lap in Liverpool F.C."`. Also, in the second example, the baseline directly copies `"associated Band/associated Musical Artist"` in the output while ASDOT correctly converts it into `"is associated with"`.

Source	<Zolder, fastest Lap, Liverpool F.C.> ; <Zolder, Date, October 5>
Disambig	Liverpool F.C. set the fastest lap in the Zolder. Zolder was on October 5.
Fusion	Liverpool F.C. set the fastest lap in the Zolder on October 5.
Baseline	Zolder's fastest lap is Liverpool F.C. and the date is October 5.
Human	On October 5, 2008, Liverpool F.C. got the fastest lap at a Zolder race.
Source	<Aleksandra Kovac, associated Band/associated Musical Artist, Bebi Dol> ; <Aleksandra Kovac, associated Band/associated Musical, ArtistK2 Kovac sisters duo>
Disambig	Aleksandra Kovac is associated with Bebi Dol. Aleksandra Kovac is associated with K2 Kovac sisters duo.
Fusion	Aleksandra Kovac is associated with Bebi Dol and the K2 Kovac sisters duo.
Baseline	Aleksandra Kovac is an associated band/associated musical artist with Bebi Dol and the K2 Kovac sisters duo.
Human	Aleksandra Kovac is associated with the musical artist Bebi Dol and is part of the band K2 Kovac sisters duo.

Table 7: Qualitative examples in the out-of-domain (top) and unseen-predicates (bottom) settings.

5 Conclusion

We have proposed ASDOT to deal with the diverse any-shot problems for data-to-text generation. ASDOT is composed of two stages, *data disambiguation* that uses prompted GPT-3 to disambiguate input data triples into short sentences, and *sentence fusion* using state-of-the-art pretrained LMs to fuse these sentences into the desired paragraphs. In the process, ASDOT integrates rich external implicit knowledge from the large LMs, which ensures strong generalization capability and broad applicability to zero-/few-/full-shot, unseen-predicates, and out-of-domain training scenarios. Extensive experiments show our approach consistently achieves significant improvements over diverse baselines.

Limitations

One limitation of our approach is that the data disambiguation stage is done by the GPT-3 model locally, i.e., the GPT-3 model only observes one triple and does not utilize the full-table information. In some difficult cases, the full-table context may be needed for disambiguation. Besides, in this work we directly use the output from GPT-3's as the final disambiguation results, which may be problematic since GPT-3 may not always provide the correct templates, especially when working with highly-specialized domains. In addition, our current approach can only be applied to languages that have access to large LMs.

Ethics Statement

We are aware of the ACL Code of Ethics and the ACM Code of Ethics and Professional Conduct and strictly adhere to the rules throughout the course of this research.

Our research does not present any new datasets but introduces a new algorithm for data-to-text generation, which generates text descriptions for a given graph or table. The intended usage of the work may potentially provide benefits to people with difficulties in reading graphs or tables, such as people with visual impairment. We do not anticipate direct harm with the intended usage.

Similar to most generation systems, if harmful input, such as unethical text or input designed for adversarial attacks, exists, our approach is likely to generate unintended output. Therefore, we do not recommend usages of our approach outside controlled research environment before these risks are mitigated. We would also like to point out that a naive deployment of our method may allow malicious exploitation of the backbone Large LMs, thus precautions such as a filtering mechanism need to be implemented.

Our model makes use of the common sense reasoning ability of large LMs, which may reinforce existing social stereotypes, hence care must be taken when applying this approach to materials (e.g. tables and graphs) that are sensitive to populations that already experience marginalization.

Computation-wise, our finetuning procedure takes around 1836 GPU/Hours on NVIDIA GeForce RTX 3090 Ti GPUs. Throughout the study, our prompting module makes about 4600 API calls to Open-AI's GPT-3 API.

References

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Jan A Botha, Manaal Faruqui, John Alex, Jason Baldridge, and Dipanjan Das. 2018. Learning to split and rephrase from wikipedia edit history. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 732–737.

Thiago Castro Ferreira, Chris van der Lee, Emiel van Miltenburg, and Emiel Krahmer. 2019. Neural data-to-text generation: A comparison between pipeline and end-to-end architectures. 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 552–562, Hong Kong, China. Association for Computational Linguistics.

Wenhu Chen, Yu Su, Xifeng Yan, and William Yang Wang. 2020a. KGPT: Knowledge-grounded pre-training for data-to-text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8635–8648, Online. Association for Computational Linguistics.

Zhiyu Chen, Harini Eavani, Wenhu Chen, Yinyin Liu, and William Yang Wang. 2020b. 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.

Jordan Clive, Kris Cao, and Marek Rei. 2021. Control prefixes for text generation. *arXiv preprint arXiv:2110.08329*.

Bhuwan Dhingra, Manaal Faruqui, Ankur Parikh, Ming-Wei Chang, Dipanjan Das, and William Cohen. 2019. Handling divergent reference texts when evaluating table-to-text generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4884–4895, Florence, Italy. Association for Computational Linguistics.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. Creating training corpora for NLG micro-planners. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*,

pages 179–188, Vancouver, Canada. Association for Computational Linguistics.

Mor Geva, Eric Malmi, Idan Szpektor, and Jonathan Berant. 2019. DiscoFuse: A large-scale dataset for discourse-based sentence fusion. 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 3443–3455, Minneapolis, Minnesota. Association for Computational Linguistics.

Eli Goldberg, Norbert Driedger, and Richard I Kittredge. 1994. Using natural-language processing to produce weather forecasts. *IEEE Expert*, 9(2):45–53.

Heng Gong, Yawei Sun, Xiaocheng Feng, Bing Qin, Wei Bi, Xiaojiang Liu, and Ting Liu. 2020. TableGPT: Few-shot table-to-text generation with table structure reconstruction and content matching. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1978–1988, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Hamza Harkous, Isabel Groves, and Amir Saffari. 2020. Have your text and use it too! end-to-end neural data-to-text generation with semantic fidelity. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2410–2424, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Peyman Heidari, Arash Einolghozati, Shashank Jain, Soumya Batra, Lee Callender, Ankit Arun, Shawn Mei, Sonal Gupta, Pinar Donmez, Vikas Bhardwaj, et al. 2021. Getting to production with few-shot natural language generation models. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 66–76.

Mihir Kale and Abhinav Rastogi. 2020a. Template guided text generation for task-oriented dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6505–6520.

Mihir Kale and Abhinav Rastogi. 2020b. Text-to-text pre-training for data-to-text tasks. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 97–102, Dublin, Ireland. Association for Computational Linguistics.

Zdeněk Kasner and Ondrej Dusek. 2022. Neural pipeline for zero-shot data-to-text generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3914–3932, Dublin, Ireland. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text generation from knowledge graphs with graph transformers. 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 2284–2293.

Klaus Krippendorff. 2011. Computing krippendorff's alpha-reliability. *Computing*, 1:25–2011.

Karen Kukich. 1983a. Design of a knowledge-based report generator. In *21st Annual Meeting of the Association for Computational Linguistics*, pages 145–150.

Karen Kukich. 1983b. Design of a knowledge-based report generator. In *21st Annual Meeting of the Association for Computational Linguistics*, pages 145–150, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Rémi Lebret, David Grangier, and Michael Auli. 2016a. Neural text generation from structured data with application to the biography domain. *arXiv preprint arXiv:1603.07771*.

Rémi Lebret, David Grangier, and Michael Auli. 2016b. 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, Austin, Texas. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597.

Amit Moryossef, Yoav Goldberg, and Ido Dagan. 2019. Step-by-step: Separating planning from realization in neural data-to-text generation. 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 2267–2277, Minneapolis, Minnesota. Association for Computational Linguistics.

Linyong Nan, Dragomir Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, et al. 2020. Dart: Open-domain structured data record to text generation. *arXiv preprint arXiv:2007.02871*.

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, Saarbrücken, Germany. Association for Computational Linguistics.

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 of the Association for Computational Linguistics, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1470–1480, Beijing, China. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019a. Data-to-text generation with content selection and planning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6908–6915.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019b. Data-to-text generation with entity modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2023–2035, Florence, Italy. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners.

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*, 21:1–67.

Ehud Reiter and Robert Dale. 1997. Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87.

Leonardo F. R. Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2021. Investigating pretrained language models for graph-to-text generation. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 211–227, Online. Association for Computational Linguistics.

Leonardo FR Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2020. Investigating pretrained language models for graph-to-text generation. *arXiv preprint arXiv:2007.08426*.

Somayajulu Sripada, Ehud Reiter, and Ian Davy. 2003. Sumtime-mousam: Configurable marine weather forecast generator. *Expert Update*, 6(3):4–10.

Yixuan Su, Zaiqiao Meng, Simon Baker, and Nigel Collier. 2021. Few-shot table-to-text generation with prototype memory. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 910–917.

Qingyun Wang, Xiaoman Pan, Lifu Huang, Boliang Zhang, Zhiying Jiang, Heng Ji, and Kevin Knight. 2018. Describing a knowledge base. *arXiv preprint arXiv:1809.01797*.

Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. *arXiv preprint arXiv:1508.01745*.

Sam Wiseman, Stuart M Shieber, and Alexander M Rush. 2017. Challenges in data-to-document generation. *arXiv preprint arXiv:1707.08052*.

Chao Zhao, Marilyn Walker, and Snigdha Chaturvedi. 2020. Bridging the structural gap between encoding and decoding for data-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2481–2491.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*.

A GPT-3 Prompt

The prefix in the prompt we use is:

```
Table: Michael | birth Place | USA
Text: Michael was born in the USA.

Table: First Clearing | location | On NYS 52 1 Mi. Youngsville
Text: First Clearing is located at On NYS 52 1 Mi. Youngsville.

Table: Abilene Regional Airport | city Served | Abilene Texas
Text: Abilene Regional Airport serves Abilene Texas.

Table: Alfred Moore Scales | active Years Start Date | 1875-03-04
Text: Alfred Moore Scales started to be active on 1875-03-04.
```

B Experimental Details

We use a batch size of 5 and a beam search size of 5 for zero-shot and few-shot settings. For other settings, we do model selection based on the performance on the validation set, with a batch size chosen from {2, 4, 8} and {1, 3, 5}, respectively. We use sacreBLEU (Post, 2018) for model selection. The URL for the metrics and corpus we use are shown in Table 8 and Table 9, respectively.

Metric	URL
BLEU	https://github.com/moses-smt/mosesdecoder/blob/master/scripts/generic/multi-bleu.perl
METEOR	https://www.cs.cmu.edu/~alavie/METEOR/index.html
PARENT	https://github.com/KaijuML/parent
SacreBLEU	https://github.com/mjpost/sacrebleu

Table 8: The URLs for the metrics we use in the experiments.

Dataset	URL
WebNLG	https://gitlab.com/shimorina/webnlg-dataset/-/tree/master/webnlg_challenge_2017
DART	https://github.com/Yale-LILY/dart
E2E	https://github.com/tuetschek/e2e-dataset
WikiFluent	https://github.com/kasnerz/zeroshot-d2t-pipeline

Table 9: The URLs for the corpus we use in the experiments.

C Zero-/Few-shot Experimental Results

We show the BLEU, METEOR and PARENT-F1 scores for zero-/few-shot experiments on WebNLG and DART in Table 10 and Table 11, respectively.

#Instance	0	10	20	50	100
KGPT	14.19/20.78/20.67	17.50/23.13/ 25.77	18.40/23.44/26.49	21.68/25.30/29.22	24.72/26.71/46.50
T5-large	10.46/25.63/23.67	24.74/32.28/42.48	41.38/36.12/52.77	45.32/39.49/59.39	48.68/39.24/60.66
ASDOT	**43.99/39.32**/58.23	**45.16/38.95**/58.24	**47.46/39.35**/59.85	**49.36/40.08**/61.25	**49.39/40.09**/61.08
- w/o weak-sup	34.47/30.06/51.51	39.38/33.93/56.44	43.67/35.81/57.99	47.56/38.61/60.04	48.60/39.68/60.56
- w/ manual templ.	42.02/38.85/**58.26**	43.37/38.69/**58.80**	46.12/38.88/**60.94**	48.28/39.64/**62.02**	48.32/39.32/**61.92**

Table 10: WebNLG few-shot results. x / y / z denotes the model performance on BLEU / METEOR / PARENT-F1.

#Instance	0	10	20	50	100
KGPT	11.15/19.30/18.92	14.91/19.74/23.76	16.83/21.30/26.67	20.16/23.14/31.13	20.31/23.82/31.35
T5-large	8.43/22.67/23.81	29.97/31.44/46.82	32.96/31.76/47.36	37.08/34.43/54.10	39.92/34.90/55.05
ASDOT	**38.81/36.91/54.10**	**40.50/36.65/56.00**	**41.45/36.45/57.34**	**42.33/36.99/57.63**	**42.87/36.77/58.37**
- w/o weak-sup	31.92/26.15/43.99	38.15/32.11/54.97	37.12/32.80/54.12	40.79/35.70/56.40	41.22/35.15/57.79

Table 11: DART few-shot results. x / y / z denotes the model performance on BLEU / METEOR / PARENT-F1.