Investigating Pretrained Language Models for Graph-to-Text Generation

Leonardo F. R. Ribeiro[†], **Martin Schmitt**[‡], **Hinrich Schütze**[‡] **and Iryna Gurevych**[†]

[†]Research Training Group AIPHES and UKP Lab, Technical University of Darmstadt
[‡]Center for Information and Language Processing (CIS), LMU Munich

www.ukp.tu-darmstadt.de

Abstract

Graph-to-text generation aims to generate fluent texts from graph-based data. In this paper, we investigate two recent pretrained language models (PLMs) and analyze the impact of different task-adaptive pretraining strategies for PLMs in graph-to-text generation. We present a study across three graph domains: meaning representations, Wikipedia knowledge graphs (KGs) and scientific KGs. We show that approaches based on PLMs BART and T5 achieve new state-of-the-art results and that task-adaptive pretraining strategies improve their performance even further. We report new state-of-the-art BLEU scores of 49.72 on AMR-LDC2017T10, 59.70 on WebNLG, and 25.66 on AGENDA datasets - a relative improvement of 31.8%, 4.5%, and 42.4%, respectively, with our models generating significantly more fluent texts than human references. In an extensive analysis, we identify possible reasons for the PLMs' success on graph-to-text tasks. Our findings suggest that the PLMs benefit from similar facts seen during pretraining or fine-tuning, such that they perform well even when the input graph is reduced to a simple bag of node and edge labels.[1]

1 Introduction

Graphs are important data structures in NLP as they represent complex relations within a set of objects. For example, semantic and syntactic structures of sentences can be represented using different graph representations (e.g., AMRs, Banarescu et al., 2013; semantic-role labeling, Surdeanu et al., 2008; syntactic and semantic graphs, Belz et al., 2011) and knowledge graphs (KGs) are used to describe factual knowledge in the form of relations between entities (Gardent et al., 2017; Vougiouklis et al., 2018; Koncel-Kedziorski et al., 2019).

Graph-to-text generation, a subtask of data-to-text generation (Gatt and Krahmer, 2018), aims to create fluent natural language text to describe an input graph (see Figure 1). This task is important for NLP applications such as dialogue generation (Moon et al., 2019) and question answering (Duan et al., 2017). Recently, it has been shown that structured meaning representation, such as AMR or KG, can store the internal state of a dialog system, providing core semantic knowledge (Bonial et al., 2020; Bai et al., 2021) or can be the result of a database query for conversational QA (Yu et al., 2019). Moreover, dialog states can be represented as KGs to encode compositionality and can be shared across different domains, slot types and dialog participators (Cheng et al., 2020).

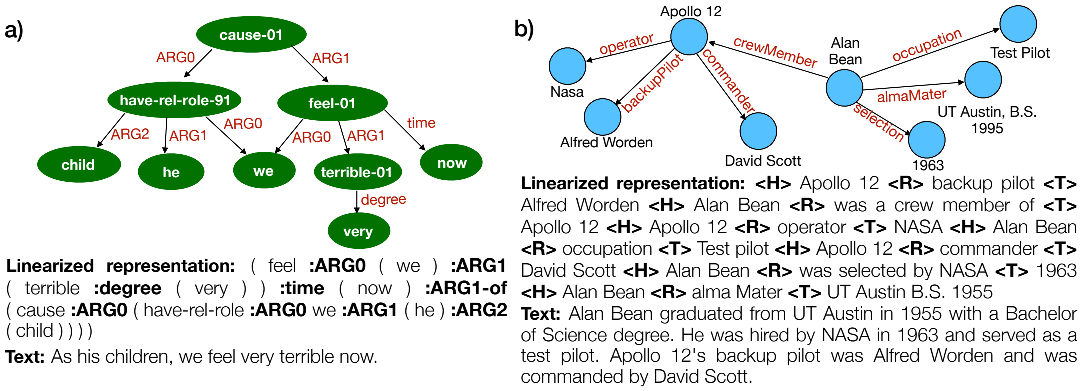
Transfer learning has become ubiquitous in NLP and pretrained Transformer-based architectures (Vaswani et al., 2017) have considerably outperformed prior state of the art in various downstream tasks (Devlin et al., 2019; Yang et al., 2019a; Liu et al., 2020; Radford et al., 2019).

In this paper, we analyze the applicability of two recent text-to-text pretrained language models (PLMs), BART (Lewis et al., 2020) and T5 (Raffel et al., 2019), for graph-to-text generation. We choose these models because of their *encoder-decoder* architecture, which makes them particularly suitable for conditional text generation. Our study comprises three graph domains (meaning representations, Wikipedia KGs, and scientific KGs). We further introduce *task-adaptive* graph-to-text pretraining approaches for PLMs and demonstrate that such strategies improve the state of the art by a substantial margin.

While recent works have shown the benefit of explicitly encoding the graph structure in graph-to-text generation (Song et al., 2018; Ribeiro et al., 2019, 2020; Schmitt et al., 2020; Zhao et al., 2020a, to name a few), our approaches based on PLMs consistently outperform these models, even though PLMs – as sequence models – do not exhibit any

[1] Our code is available at https://github.com/UKPLab/plms-graph2text.

Proceedings of the Third Workshop on Natural Language Processing for Conversational AI, pages 211–227
November 10, 2021. ©2021 Association for Computational Linguistics



Linearized representation: (feel **:ARG0** (we) **:ARG1** (terrible **:degree** (very)) **:time** (now) **:ARG1-of** (cause **:ARG0** (have-rel-role **:ARG0** we **:ARG1** (he) **:ARG2** (child))))

Text: As his children, we feel very terrible now.

Linearized representation: **<H>** Apollo 12 **<R>** backup pilot **<T>** Alfred Worden **<H>** Alan Bean **<R>** was a crew member of **<T>** Apollo 12 **<H>** Apollo 12 **<R>** operator **<T>** NASA **<H>** Alan Bean **<R>** occupation **<T>** Test pilot **<H>** Apollo 12 **<R>** commander **<T>** David Scott **<H>** Alan Bean **<R>** was selected by NASA **<T>** 1963 **<H>** Alan Bean **<R>** alma Mater **<T>** UT Austin B.S. 1955

Text: Alan Bean graduated from UT Austin in 1955 with a Bachelor of Science degree. He was hired by NASA in 1963 and served as a test pilot. Apollo 12's backup pilot was Alfred Worden and was commanded by David Scott.

Figure 1: Examples of (a) AMR and (b) WebNLG graphs, the input for the models and the reference texts.

graph-specific structural bias.[2] Simply representing the graph as a linear traversal (see Figure 1) leads to remarkable generation performance in the presence of a strong language model. In our analysis we investigate to what extent fine-tuned PLMs make use of the graph structure represented in the graph linearization. We notably observe that PLMs achieve high performance on two popular KG-to-text benchmarks even when the KG is reduced to a mere bag of node and edge labels.

Our contributions are the following:

- We investigate and compare two PLMs, BART and T5, for graph-to-text generation, exploring *language model adaptation* (LMA) and *supervised task adaptation* (STA) pretraining, employing additional task-specific data.
- Our approaches consistently outperform the state of the art by significant margins, ranging from 2.6 to 12.0 BLEU points, on three established graph-to-text benchmarks from different domains, exceeding specialized graph architectures (e.g., Graph Neural Networks, GNNs, Kipf and Welling, 2017).
- In a crowdsourcing experiment, we demonstrate that our methods generate texts with significantly better fluency than existing works and the human references.
- We discover that PLMs perform well even when trained on a shuffled linearized graph representation without any information about connectivity (bag of node and edge labels), which is surprising since prior studies showed that explicitly encoding the graph structure improves models trained from scratch (e.g., Zhao et al., 2020a); and investigate the possible reasons for such a good performance.

2 Related Work

Graph-to-text Learning. Various neural models have been proposed to generate sentences from graphs from different domains. Konstas et al. (2017) propose the first neural approach for AMR-to-text generation that uses a linearized input graph. Prior approaches for KG-to-text generation train text-to-text neural models using sequences of KG triples as input (Trisedya et al., 2018; Moryossef et al., 2019; Castro Ferreira et al., 2019; Ribeiro et al., 2021a).

Recent approaches (Marcheggiani and Perez Beltrachini, 2018; Song et al., 2018; Beck et al., 2018; Damonte and Cohen, 2019; Ribeiro et al., 2019; Zhao et al., 2020a; Schmitt et al., 2021; Ribeiro et al., 2021b) propose architectures based on GNNs to directly encode the graph structure, whereas other efforts (Ribeiro et al., 2020; Schmitt et al., 2020; Yao et al., 2020; Wang et al., 2020) inject the graph structure information into Transformer-based architectures. The success of those approaches suggests that imposing a strong relational inductive bias into the graph-to-text model can assist the generation.

Pretrained Language Models. Pretrained Transformer-based models, such as BERT (Devlin et al., 2019), XLNet (Yang et al., 2019b), or RoBERTa (Liu et al., 2020), have established a qualitatively new level of baseline performance for many widely used natural language understanding (NLU) benchmarks. Generative pretrained Transformer-based methods, such as GPT-2 (Radford et al., 2019), BART (Lewis et al., 2020), and T5 (Raffel et al., 2019), are employed in many

[2] The model architecture does not explicitly encode the graph structure, i.e., which entities are connected to each other, but has to retrieve it from a sequence that tries to encode this information.

natural language generation (NLG) tasks.

Mager et al. (2020) were the first to employ GPT-2, a decoder-only PLM, for AMR-to-text generation and use cycle consistency to improve the adequacy. In contrast, we are the first to investigate BART and T5 models, which have both a Transformer-based encoder and decoder, in AMR-to-text generation. Recently, Harkous et al. (2020) and Kale (2020) demonstrate state-of-the-art results in different data-to-text datasets, employing GPT-2 and T5 models respectively. Radev et al. (2020) propose DART, a new data-to-text dataset, and train a BART model gradually augmenting the WebNLG training data with DART data.

Hoyle et al. (2021) explore scaffolding objectives in PLMs and show gains in low-resource graph-to-text settings. Different from the above works, we focus on a general transfer learning strategies for graph-to-text generation, investigating task-adaptive pretraining approaches, employing additional collected task-specific data for different PLMs (BART and T5) and benchmarks. In addition, we provide a detailed analysis aimed at explaining the good performance of PLMs on KG-to-text tasks.

Recently, Gururangan et al. (2020) explored task-adaptive pretraining strategies for text classification. While our LMA (see §3) is related to their DAPT as both use a self-supervised objective on a domain-specific corpus, they notably differ in that DAPT operates on the model input while LMA models the output. We are the first to show the benefits of additional task-specific pretraining in PLMs for graph-to-text tasks.

3 PLMs for Graph-to-Text Generation

3.1 Models in this Study

We investigate BART (Lewis et al., 2020) and T5 (Raffel et al., 2019), two PLMs based on the Transformer *encoder-decoder* architecture (Vaswani et al., 2017), for graph-to-text generation. They mainly differ in how they are pretrained and the input corpora used for pretraining. We experiment with different T5 (*small* - 60M parameters, *base* - 220M, and *large* - 770M) and BART (*base* - 140M and *large* - 400M) capacity models.

We fine-tune both PLMs for a few epochs on the supervised downstream graph-to-text datasets. For T5, in the supervised setup, we add a prefix "translate from Graph to Text:" before the graph input. We add this prefix to imitate the T5 setup, when translating between different languages.

3.2 Task-specific Adaptation

Inspired by previous work (Konstas et al., 2017; Gururangan et al., 2020), we investigate whether leveraging additional task-specific data can improve the PLMs' performance on graph-to-text generation. Task-specific data refers to a pretraining corpus that is more task-relevant and usually smaller than the text corpora used for task-independent pretraining. In order to leverage the task-specific data, we add an intermediate adaptive pretraining step between the original pretraining and fine-tuning phases for graph-to-text generation.

More precisely, we first continue pretraining BART and T5 using language model adaptation (LMA) or supervised task adaptation (STA) training. In the supervised approach, we use pairs of graphs and corresponding texts collected from the same or similar domain as the target task. In the LMA approach, we follow BART and T5 pretraining strategies for language modeling, using the reference texts that describe the graphs. Note that we do not use the graphs in the LMA pretraining, but only the target text of our task-specific data collections. The goal is to adapt the decoder to the domain of the final task (Gururangan et al., 2020). In particular, we randomly mask text spans, replacing 15% of the tokens.[3] Before evaluation, we finally fine-tune the models using the original training set as usual.

4 Datasets

We evaluate the text-to-text PLMs on three graph-to-text benchmarks: AMR (LDC2017T10), WebNLG (Gardent et al., 2017), and AGENDA (Koncel-Kedziorski et al., 2019). We chose those datasets because they comprise different domains and are widely used in prior work. Table 10 in Appendix shows statistics for each dataset.

AMR. Abstract meaning representation (AMR) is a semantic formalism that represents the meaning of a sentence as a rooted directed graph expressing "who is doing what to whom" (Banarescu et al., 2013). In an AMR graph, nodes represent concepts and edges represent semantic relations. An instance in LDC2017T10 consists of a sentence annotated with its corresponding AMR graph. Following Mager et al. (2020), we linearize the AMR graphs

[3] Please, refer to Lewis et al. (2020) and Raffel et al. (2019) for details about the self-supervised pretraining strategies.

using the PENMAN notation (see Figure 1a).[4]

WebNLG. Each instance of WebNLG contains a KG from DBPedia (Auer et al., 2007) and a target text with one or multiple sentences that describe the graph. The test set is divided into two partitions: *seen*, which contains only DBPedia categories present in the training set, and *unseen*, which covers categories never seen during training. Their union is called *all*. Following previous work (Harkous et al., 2020), we prepend $\langle H \rangle$, $\langle R \rangle$, and $\langle T \rangle$ tokens before the head entity, the relation and tail entity of a triple (see Figure 1b).

AGENDA. In this dataset, KGs are paired with scientific abstracts extracted from proceedings of AI conferences. Each sample contains the paper title, a KG, and the corresponding abstract. The KG contains entities corresponding to scientific terms and the edges represent relations between these entities. This dataset has loose alignments between the graph and the corresponding text as the graphs were automatically generated. The input for the models is a text containing the title, a sequence of all KG entities, and the triples. The target text is the paper abstract. We add special tokens into the triples in the same way as for WebNLG.

4.1 Additional Task-specific Data

In order to evaluate the proposed task-adaptive pretraining strategies for graph-to-text generation, we collect task-specific data for two graph domains: meaning representations (like AMR) and scientific data (like AGENDA). We did not attempt collecting additional data like WebNLG because the texts in this benchmark do not stem from a corpus but were specifically written by annotators.

AMR Silver Data. In order to generate additional data for AMR, we sample two sentence collections of size 200K and 2M from the Gigaword[5] corpus and use a state-of-the-art AMR parser (Cai and Lam, 2020a) to parse them into AMR graphs.[6] For supervised pretraining, we condition a model on the AMR silver graphs to generate the corresponding sentences before fine-tuning it on gold AMR graphs. For self-supervised pretraining, we only use the sentences.[7]

Model	BLEU	M	BT
Ribeiro et al. (2019)	27.87	33.21	-
Zhu et al. (2019)	31.82	36.38	-
Zhao et al. (2020b)	32.46	36.78	-
Wang et al. (2020)	33.90	37.10	-
Yao et al. (2020)	34.10	38.10	-
based on PLMs			
Mager et al. (2020)	33.02	37.68	-
Harkous et al. (2020)	37.70	38.90	-
BART$_{\text{base}}$	36.71	38.64	52.47
BART$_{\text{large}}$	43.47	42.88	60.42
T5$_{\text{small}}$	38.45	40.86	57.95
T5$_{\text{base}}$	42.54	42.62	60.59
T5$_{\text{large}}$	**45.80**	**43.85**	**61.93**
with task-adaptive pretraining			
BART$_{\text{large}}$ + LMA	43.94	42.36	58.54
T5$_{\text{large}}$ + LMA	46.06	44.05	62.59
BART$_{\text{large}}$ + STA (200K)	44.72	43.65	61.03
BART$_{\text{large}}$ + STA (2M)	47.51	44.70	62.27
T5$_{\text{large}}$ + STA (200K)	48.02	44.85	63.86
T5$_{\text{large}}$ + STA (2M)	***49.72***	***45.43***	***64.24***

Table 1: Results on AMR-to-text generation for the LDC2017T10 test set. M and BT stand for METEOR and BLEURT, respectively. **Bold** (***Italic***) indicates the best score without (with) task-adaptive pretraining.

Semantic Scholar AI Data. We collect titles and abstracts of around 190K scientific papers from the Semantic Scholar (Ammar et al., 2018) taken from the proceedings of 36 top Computer Science/AI conferences. We construct KGs from the paper abstracts employing DyGIE++ (Wadden et al., 2019), an information extraction system for scientific texts. Note that the AGENDA dataset was constructed using the older SciIE system (Luan et al., 2018), which also extracts KGs from AI scientific papers. A second difference is that in our new dataset, the domain is broader as we collected data from 36 conferences compared to 12 from AGENDA. Furthermore, to prevent data leakage, all AGENDA samples used for performance evaluation are removed from our dataset. We will call the new dataset KGAIA (KGs from AI Abstracts).[8] Table 11 in Appendix shows relevant dataset statistics.

5 Experiments

We modify the BART and T5 implementations released by Hugging Face (Wolf et al., 2019) in order to adapt them to graph-to-text generation. For the KG datasets, we add the $\langle H \rangle$, $\langle R \rangle$, and $\langle T \rangle$ tokens to the models' vocabulary. We add all edge labels seen in the training set to the vocabulary of the

[4] Details of the preprocessing procedure of AMRs are provided in Appendix A.

[5] https://catalog.ldc.upenn.edu/LDC2003T05

[6] We filter out sentences that do not yield well-formed AMR graphs.

[7] Gigaword and AMR datasets share similar data sources.

[8] We will release the collected additional task-specific data.

	BLEU			**METEOR**			**chrF++**		
Model	**A**	**S**	**U**	**A**	**S**	**U**	**A**	**S**	**U**
Castro Ferreira et al. (2019)	51.68	56.35	38.92	32.00	41.00	21.00	-	-	-
Moryossef et al. (2019)	47.24	53.30	34.41	39.00	44.00	37.00	-	-	-
Schmitt et al. (2020)	-	59.39	-	-	42.83	-	-	74.68	-
Ribeiro et al. (2020)	-	63.69	-	-	44.47	-	-	76.66	-
Zhao et al. (2020a)	52.78	64.42	38.23	41.00	46.00	37.00	-	-	-
based on PLMs									
Harkous et al. (2020)	52.90	-	-	42.40	-	-	-	-	-
Kale (2020)	57.10	63.90	52.80	44.00	46.00	41.00	-	-	-
Radev et al. (2020)	45.89	52.86	37.85	40.00	42.00	37.00	-	-	-
BART$_{\text{base}}$	53.11	62.74	41.53	40.18	44.45	35.36	70.02	76.68	62.76
BART$_{\text{large}}$	54.72	63.45	43.97	42.23	45.49	38.61	72.29	77.57	66.53
T5$_{\text{small}}$	56.34	**65.05**	45.37	42.78	45.94	39.29	73.31	**78.46**	67.69
T5$_{\text{base}}$	59.17	64.64	52.55	43.19	**46.02**	41.49	74.82	78.40	70.92
T5$_{\text{large}}$	**59.70**	64.71	**53.67**	**44.18**	45.85	**42.26**	**75.40**	78.29	**72.25**

Table 2: Results on WebNLG. A, S and U stand for *all*, *seen*, and *unseen* partitions of the test set, respectively.

models for AMR. Following Wolf et al. (2019), we use the Adam optimizer (Kingma and Ba, 2015) with an initial learning rate of $3 \cdot 10^{-5}$. We employ a linearly decreasing learning rate schedule without warm-up. The batch and beam search sizes are chosen from {2,4,8} and {1,3,5}, respectively, based on the respective development set. Dev BLEU is used for model selection.

Following previous works, we evaluate the results with BLEU (Papineni et al., 2002), METEOR (Denkowski and Lavie, 2014), and chrF++ (Popović, 2015) metrics. We also use MoverScore (Zhao et al., 2019), BERTScore (Zhang et al., 2020), and BLEURT (Sellam et al., 2020) metrics, as they employ contextual and semantic knowledge and thus depend less on the surface symbols. Additionally, we perform a human evaluation (cf. §5.4) quantifying the fluency, semantic adequacy and meaning similarity of the generated texts.

5.1 Results on AMR-to-Text

Table 1 shows our results for the setting without additional pretraining, with additional self-supervised task-adaptive pretraining solely using the collected Gigaword sentences (LMA), and with additional supervised task adaptation (STA), before fine-tuning. We also report several recent results on the AMR test set. Mager et al. (2020) and Harkous et al. (2020) employ GPT-2 in their approaches. Note that GPT-2 only consists of a Transformer-based decoder.

Only considering approaches without task adaptation, BART$_{\text{large}}$ already achieves a considerable improvement of 5.77 BLEU and 3.98 METEOR scores over the previous state of the art. With a BLEU score of 45.80, T5$_{\text{large}}$ performs best. The other metrics follow similar trends. See Table 13 in Appendix for evaluation with more metrics. The strong performance of both BART and T5 in the AMR dataset suggests that PLMs can infer the AMR structure by a simple linear sequence of the graph, in contrast to GNN-based models that explicitly consider the graph structure using *message-passing* between adjacent nodes (Beck et al., 2018).

Task-specific Adaptation. LMA already brings some gains with T5 benefitting more than BART in most metrics. It still helps less than STA even though we only have automatically generated annotations. This suggests that the performance increases with STA do not only come from additional exposure to task-specific target texts and that the models learn how to handle graphs and the graph-text correspondence even with automatically generated AMRs. After STA, T5 achieves 49.72 BLEU points, the new state of the art for AMR-to-text generation. Interestingly, gains from STA with 2M over 200K are larger in BART than in T5, suggesting that large amounts of silver data may not be required for a good performance with T5.

In general, models pretrained on the STA setup converge faster than without task-specific adaptation. For example, T5$_{\text{large}}$ without additional pretraining converges after 5 epochs of fine-tuning whereas T5$_{\text{large}}$ with STA already converges after 2 epochs.

5.2 Results on WebNLG

Table 2 shows the results for the WebNLG test set. Neural pipeline models (Moryossef et al., 2019; Castro Ferreira et al., 2019) achieve strong performance in the *unseen* dataset. On the other

Model	BLEU	M	BT
Koncel et al. 2019	14.30	18.80	-
An (2019)	15.10	19.50	-
Schmitt et al. (2020)	17.33	21.43	-
Ribeiro et al. (2020)	18.01	22.23	-
$\text{BART}_{\text{base}}$	22.01	23.54	-13.02
$\text{BART}_{\text{large}}$	**23.65**	**25.19**	**-10.93**
T5_{small}	20.22	21.62	-24.10
T5_{base}	20.73	21.88	-21.03
T5_{large}	22.15	23.73	-13.96
with task-adaptive pretraining			
$\text{BART}_{\text{large}}$ + LMA	25.30	25.54	-08.79
T5_{large} + LMA	22.92	24.40	-10.39
$\text{BART}_{\text{large}}$ + STA	***25.66***	***25.74***	***-08.97***
T5_{large} + STA	23.69	24.92	-08.94

Table 3: Results on AGENDA test set. **Bold** (***Italic***) indicates best scores without (with) task-adaptive pretraining.

hand, fully end-to-end models (Ribeiro et al., 2020; Schmitt et al., 2020) have strong performance on the *seen* dataset and usually perform poorly in *unseen* data. Models that *explicitly encode the graph structure* (Ribeiro et al., 2020; Zhao et al., 2020a) achieve the best performance among approaches that do not employ PLMs. Note that T5 is also used in Kale (2020). Differences in our T5 setup include a modified model vocabulary, the use of beam search, the learning rate schedule and the prefix before the input graph. Our T5 approach achieves 59.70, 65.05 and 54.69 BLEU points on *all*, *seen* and *unseen* sets, the new state of the art.

We conjecture that the performance gap between *seen* and *unseen* sets stems from the advantage obtained by a model seeing examples of relation-text pairs during fine-tuning. For example, the relation *party* (political party) was never seen during training and the model is required to generate a text that verbalizes the tuple: ⟨*Abdul Taib Mahmud, party, Parti Bumiputera Sarawak*⟩. Interestingly, BART performs much worse than T5 on this benchmark, especially in the *unseen* partition with 9.7 BLEU points lower compared to T5.

For lack of a suitable data source (cf. §4), we did not explore our LMA or STA approaches for WebNLG. However, we additionally discuss cross-domain STA in Appendix B.

5.3 Results on AGENDA

Table 3 lists the results for the AGENDA test set. The models also show strong performance on this dataset. We believe that their capacity to generate fluent text helps when generating paper abstracts, even though they were not pretrained in the scientific domain. $\text{BART}_{\text{large}}$ shows an impressive performance with a BLEU score of 23.65, which is 5.6 points higher than the previous state of the art.

Model	AMR	
	F	MS
Mager et al. (2020)	5.69^{A}	5.08^{A}
Harkous et al. (2020)	5.78^{A}	5.47^{AB}
T5_{large}	6.55^{B}	6.44^{C}
$\text{BART}_{\text{large}}$	6.70^{B}	5.72^{BC}
Reference	5.91^{A}	-
Model	**WebNLG**	
	F	**SA**
Castro Ferreira et al. (2019)	5.52^{A}	4.77^{A}
Harkous et al. (2020)	5.74^{AB}	6.21^{B}
T5_{large}	6.71^{C}	6.63^{B}
$\text{BART}_{\text{large}}$	6.53^{C}	6.50^{B}
Reference	5.89^{B}	6.47^{B}

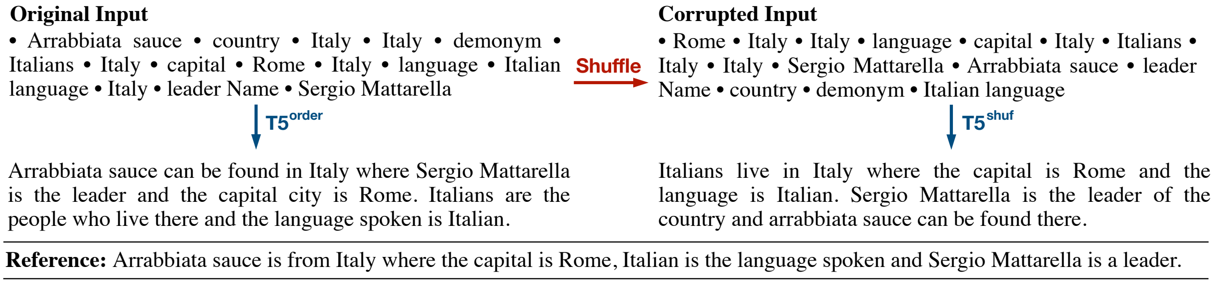
Table 4: Fluency (F), Meaning Similarity (MS) and Semantic Adequacy (SA) obtained in the human evaluation. Differences between models which have a letter in common are not statistically significant and were determined by pairwise Mann-Whitney tests with $p < 0.05$.

Task-specific Adaptation. On AGENDA, BART benefits more from our task-adaptive pretraining, achieving the new state of the art of 25.66 BLEU points, a further gain of 2 BLEU points compared to its performance without task adaptation. The improvements from task-adaptive pretraining are not as large as for AMR. We hypothesize that this is due to the fact that the graphs do not completely cover the target text (Koncel-Kedziorski et al., 2019), making this dataset more challenging. See Table 12 in Appendix for more automatic metrics.

5.4 Human Evaluation

To further assess the quality of the generated text, we conduct a human evaluation on AMR and WebNLG via crowd sourcing on Amazon Mechanical Turk.[9] Following previous works (Gardent et al., 2017; Castro Ferreira et al., 2019), we assess three quality criteria: (i) *Fluency* (i.e., does the text flow in a natural, easy-to-read manner?), for AMR and WebNLG; (ii) *Meaning Similarity* (i.e., how

[9] We exclude AGENDA because its texts are scientific in nature and annotators are not necessarily AI experts.



Original Input
• Arrabbiata sauce • country • Italy • Italy • demonym • Italians • Italy • capital • Rome • Italy • language • Italian language • Italy • leader Name • Sergio Mattarella

Arrabbiata sauce can be found in Italy where Sergio Mattarella is the leader and the capital city is Rome. Italians are the people who live there and the language spoken is Italian.

Corrupted Input
• Rome • Italy • Italy • language • capital • Italy • Italians • Italy • Italy • Sergio Mattarella • Arrabbiata sauce • leader Name • country • demonym • Italian language

Italians live in Italy where the capital is Rome and the language is Italian. Sergio Mattarella is the leader of the country and arrabbiata sauce can be found there.

Reference: Arrabbiata sauce is from Italy where the capital is Rome, Italian is the language spoken and Sergio Mattarella is a leader.

Figure 2: Example graph with 5 triples, from WebNLG dev linearized with the neutral separator tag, denoted •, (top left), its shuffled version (top right), texts generated with two fine-tuned versions of T5$_{\text{small}}$ and a gold reference (bottom). Note that T5 can produce a reasonable text even when the input triples are shuffled randomly.

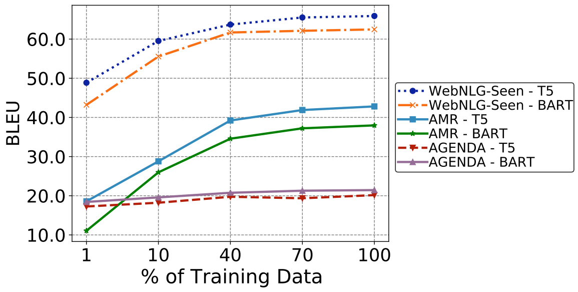


Figure 3: Performance of BART$_{\text{base}}$ and T5$_{\text{base}}$ in the dev set when experimenting with different amounts of training data.

close in meaning is the generated text to the reference sentence?) for AMR; (ii) *Semantic Adequacy* (i.e., does the text clearly express the data?) for WebNLG. We randomly select 100 generated texts of each model, which the annotators then rate on a 1-7 Likert scale. For each text, we collect scores from 3 annotators and average them.[10]

Table 4 shows the results. Our approaches improve the fluency, meaning similarity, and semantic adequacy on both datasets compared to other state-of-the-art approaches with statistically significant margins (p<0.05). Interestingly, the highest fluency improvement (+0.97) is on AMR, where our approach also has the largest BLEU improvement (+8.10) over Harkous et al. (2020). Finally, our models score higher than the references in fluency with statistically significant margins, highlighting their strong language generation abilities.[11]

5.5 Limiting the Training Data

In Figure 3, we investigate the PLMs' performance, measured with BLEU score, while varying (from 1% to 100%) the amount of training data used for fine-tuning. We find that, when fine-tuned with only 40% of the data, both BART and T5 already greatly improve the performance compared to using the entire training data in all three benchmarks. For example, BART fine-tuned on 40% of AMR training data achieves 91% of the BLEU score when fine-tuned on full data.

Model	AMR	WebNLG	AGENDA
T5$^{\text{order}}$	36.83	63.41	19.86
T5$^{\text{shuf}}$	15.56	61.54	19.08

Table 5: Impact (measured with BLEU) of using a bag of entities and relations (*shuf*) as input for T5$_{\text{small}}$.

Note that in a low-resource scenario in AMR and WebNLG, T5 considerably outperforms BART. In particular, with only 1% of training examples, the difference between T5 and BART is 7.51 and 5.64 BLEU points for AMR and WebNLG, respectively. This suggests that T5 is more data efficient when adapting to the new task, likewise our findings in AMR-STA (cf. §5.1).

6 Influence of the Graph Structure

We conduct further experiments to examine how much the PLMs consider the graph structure. To this end, we remove parentheses in AMRs and replace $\langle H\rangle$, $\langle R\rangle$, and $\langle T\rangle$ tokens with neutral separator tokens, denoted •, for KGs, such that the graph structure is only defined by the order of node and edge labels. If we shuffle such a sequence, the graph structure is thus completely obscured and the input effectively becomes a bag of node and edge labels. See Figure 2 for an example of both a correctly ordered and a shuffled triple sequence.

6.1 Quantitative Analysis

Table 5 shows the effect on T5's performance when its input contains correctly ordered triples (T5$^{\text{order}}$)

[10] Inter-annotator agreement for the three criteria ranged from 0.40 to 0.79, with an average Krippendorff's α of 0.56.

[11] Examples of fluent generations can be found in the Tables 15 and 16 in Appendix.

T/F	Input Fact	$T5^{order}$	$T5^{shuf}$
(1) S	• German language • Antwerp • Antwerp • Antwerp International Airport • Belgium • Belgium • Charles Michel • city Served • leader Name • Belgium • language • country	Antwerp International Airport serves the city of Antwerp. German is the language spoken in Belgium where Charles Michel is the leader.	Antwerp International Airport serves the city of Antwerp in Belgium where the German language is spoken and Charles Michel is the leader.
(2) T	• California • is Part Of • US • California • capital • Sacramento	California is part of the United States and its capital is Sacramento.	California is part of the United States and its capital is Sacramento.
(3) F	• US • is Part Of • California • California • capital • Sacramento	California's capital is Sacramento and the United States is part of California.	California is part of the United States and its capital is Sacramento.
(4) T	• Amarillo, Texas • is Part Of • United States	Amarillo, Texas is part of the United States.	Amarillo, Texas is part of the United States.
(5) F	• United States • is Part Of • Amarillo, Texas	Amarillo, Texas is part of the United States.	Amarillo, Texas is part of the United States.

Table 6: Example generations from shuffled (S), true (T), and corrupted (F) triple facts by $T5_{small}$, fine-tuned on correctly ordered triples (*order*) and randomly shuffled input (*shuf*).

vs. shuffled ones ($T5^{shuf}$) for both training and evaluation. We first observe that $T5^{order}$ only has marginally lower performance (around 2-4%) with the neutral separators than with the $\langle H \rangle / \langle R \rangle / \langle T \rangle$ tags or parentheses.[12] We see that as evidence that the graph structure is similarly well captured by $T5^{order}$. Without the graph structure ($T5^{shuf}$), AMR-to-text performance drops significantly. Possible explanations of this drop are: (i) the relative ordering of the AMR graph is known to correlate with the target sentence order (Konstas et al., 2017); (ii) in contrast to WebNLG that contains common knowledge, the AMR dataset contains very specific sentences with higher surprisal;[13] (iii) AMRs are much more complex graph structures than the KGs from WebNLG and AGENDA.[14]

On the other hand, KG-to-text performance is not much lower, indicating that most of the PLMs' success in this task stems from their language modeling rather than their graph encoding capabilities. We hypothesize that a PLM can match the entities in a shuffled input with sentences mentioning these entities from the pretraining or fine-tuning phase. It has recently been argued that large PLMs can recall certain common knowledge facts from pretraining (Petroni et al., 2019; Bosselut et al., 2019).

6.2 Qualitative Analysis

The example in Figure 2 confirms our impression. $T5^{shuf}$ produces a text with the same content as $T5^{order}$ but does not need the correct triple structure to do so. Example (1) in Table 6 shows the output of both models with shuffled input. Interestingly, even $T5^{order}$ produces a reasonable and truthful text. This suggests that previously seen facts serve as a strong guide during text generation, even for models that were fine-tuned with a clearly marked graph structure, suggesting that $T5^{order}$ also relies more on language modeling than the graph structure. It does have more difficulties covering the whole input graph though. The fact that *Antwerp* is located in *Belgium* is missing from its output.

To further test our hypothesis that PLMs make use of previously seen facts during KG-to-text generation, we generate example true facts, corrupt them in a controlled setting, and feed them to both $T5^{order}$ and $T5^{shuf}$ to observe their output (examples (2)–(5) in Table 6). The model trained on correctly ordered input has learned a bit more to rely on the input graph structure. The false fact in example (3) with two triples is reliably transferred to the text by $T5^{order}$ but not by $T5^{shuf}$, which silently corrects it. Also note that, in example (5), both models refuse to generate an incorrect fact. More examples can be found in Table 14 in the Appendix.

Our qualitative analysis illustrates that state-of-the-art PLMs, despite their fluency capacities (cf. §5.4), bear the risk of parroting back training sentences while ignoring the input structure. This issue can limit the practical usage of those models as, in many cases, it is important for a generation model to stay true to its input (Wiseman et al., 2017; Falke et al., 2019).

[12] See a more fine-grained comparison in Appendix C.

[13] Perplexities estimated on the dev sets of AMR and WebNLG datasets, with GPT-2 fine-tuned on the corresponding training set, are 20.9 and 7.8, respectively.

[14] In Appendix D, we present the graph properties of the datasets and discuss the differences.

7 Conclusion

We investigated two pretrained language models (PLMs) for graph-to-text generation and show that the pretraining strategies, language model adaptation (LMA) and supervised task adaptation (STA), can lead to notable improvements. Our approaches outperform the state of the art by a substantial margin on three graph-to-text benchmarks. Moreover, in a human evaluation our generated texts are perceived significantly more fluent than human references. Examining the influence of the graph structure on the text generation process, we find that PLMs may not always follow the graph structure and instead use memorized facts to guide the generation. A promising direction for future work is to explore ways of injecting a stronger graph-structural bias into PLMs, thus possibly leveraging their strong language modeling capabilities and keeping the output faithful to the input graph.

Acknowledgments

We thank our anonymous reviewers for their thoughtful feedback. Leonardo F. R. Ribeiro is supported by the German Research Foundation (DFG) as part of the Research Training Group "Adaptive Preparation of Information form Heterogeneous Sources" (AIPHES, GRK 1994/1) and as part of the DFG funded project UKP-SQuARE with the number GU 798/29-1. Martin Schmitt is supported by the BMBF as part of the project MLWin (01IS18050) and by the German Academic Scholarship Foundation (Studienstiftung des deutschen Volkes).

References

Waleed Ammar, Dirk Groeneveld, Chandra Bhagavatula, Iz Beltagy, Miles Crawford, Doug Downey, Jason Dunkelberger, Ahmed Elgohary, Sergey Feldman, Vu Ha, Rodney Kinney, Sebastian Kohlmeier, Kyle Lo, Tyler Murray, Hsu-Han Ooi, Matthew Peters, Joanna Power, Sam Skjonsberg, Lucy Wang, Chris Wilhelm, Zheng Yuan, Madeleine van Zuylen, and Oren Etzioni. 2018. Construction of the literature graph in semantic scholar. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, pages 84–91, New Orleans - Louisiana. Association for Computational Linguistics.

Bang An. 2019. Repulsive bayesian sampling for diversified attention modeling. In *4th workshop on Bayesian Deep Learning (NeurIPS 2019)*.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *Proceedings of the 6th International The Semantic Web and 2nd Asian Conference on Asian Semantic Web Conference*, ISWC'07/ASWC'07, page 722–735, Berlin, Heidelberg. Springer-Verlag.

Xuefeng Bai, Yulong Chen, Linfeng Song, and Yue Zhang. 2021. Semantic representation for dialogue modeling. 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 4430–4445, Online. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Daniel Beck, Gholamreza Haffari, and Trevor Cohn. 2018. Graph-to-sequence learning using gated graph neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 273–283, Melbourne, Australia. Association for Computational Linguistics.

Anja Belz, Michael White, Dominic Espinosa, Eric Kow, Deirdre Hogan, and Amanda Stent. 2011. The first surface realisation shared task: Overview and evaluation results. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 217–226, Nancy, France. Association for Computational Linguistics.

Claire Bonial, Lucia Donatelli, Mitchell Abrams, Stephanie M. Lukin, Stephen Tratz, Matthew Marge, Ron Artstein, David Traum, and Clare Voss. 2020. Dialogue-AMR: Abstract Meaning Representation for dialogue. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 684–695, Marseille, France. European Language Resources Association.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Deng Cai and Wai Lam. 2020a. AMR parsing via graph-sequence iterative inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1290–1301, Online. Association for Computational Linguistics.

Deng Cai and Wai Lam. 2020b. Graph transformer for graph-to-sequence learning. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7464–7471. AAAI Press.

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.

Jianpeng Cheng, Devang Agrawal, Héctor Martínez Alonso, Shruti Bhargava, Joris Driesen, Federico Flego, Dain Kaplan, Dimitri Kartsaklis, Lin Li, Dhivya Piraviperumal, Jason D. Williams, Hong Yu, Diarmuid Ó Séaghdha, and Anders Johannsen. 2020. Conversational semantic parsing for dialog state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8107–8117, Online. Association for Computational Linguistics.

Marco Damonte and Shay B. Cohen. 2019. Structural neural encoders for AMR-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 3649–3658, Minneapolis, Minnesota. Association for Computational Linguistics.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA. Association for Computational Linguistics.

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

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874, Copenhagen, Denmark. Association for Computational Linguistics.

Tobias Falke, Leonardo F. R. Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. 2019. Ranking generated summaries by correctness: An interesting but challenging application for natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2214–2220, Florence, Italy. Association for Computational Linguistics.

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, Santiago de Compostela, Spain. Association for Computational Linguistics.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61(1):65–170.

Zhijiang Guo, Yan Zhang, Zhiyang Teng, and Wei Lu. 2019. Densely connected graph convolutional networks for graph-to-sequence learning. *Transactions of the Association for Computational Linguistics*, 7:297–312.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for 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.

Alexander Miserlis Hoyle, Ana Marasović, and Noah A. Smith. 2021. Promoting graph awareness in linearized graph-to-text generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 944–956, Online. Association for Computational Linguistics.

Mihir Kale. 2020. Text-to-text pre-training for data-to-text tasks. *arXiv e-prints*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *Proceedings of the 5th International Conference on Learning Representations*, ICLR 2017.

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, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pretraining for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Roberta: A robustly optimized bert pretraining approach. *arXiv e-prints*.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium. Association for Computational Linguistics.

Manuel Mager, Ramón Fernandez Astudillo, Tahira Naseem, Md Arafat Sultan, Young-Suk Lee, Radu Florian, and Salim Roukos. 2020. GPT-too: A language-model-first approach for AMR-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1846–1852, Online. Association for Computational Linguistics.

Diego Marcheggiani and Laura Perez Beltrachini. 2018. Deep graph convolutional encoders for structured data to text generation. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 1–9, Tilburg University, The Netherlands. Association for Computational Linguistics.

Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. OpenDialKG: Explainable conversational reasoning with attention-based walks over knowledge graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 845–854, Florence, Italy. Association for Computational Linguistics.

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.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? 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 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Dragomir Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Nazneen Fatema Rajani, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, Yangxiaokang Liu, Nadia Irwanto, Jessica Pan, Faiaz Rahman, Ahmad Zaidi, Murori Mutuma, Yasin Tarabar, Ankit Gupta, Tao Yu, Yi Chern Tan, Xi Victoria Lin, Caiming Xiong, and Richard Socher. 2020. Dart: Open-domain structured data record to text generation. *arXiv e-prints*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *arXiv e-prints*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*.

Leonardo F. R. Ribeiro, Claire Gardent, and Iryna Gurevych. 2019. Enhancing AMR-to-text generation with dual graph representations. 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 3183–3194, Hong Kong, China. Association for Computational Linguistics.

Leonardo F. R. Ribeiro, Jonas Pfeiffer, Yue Zhang, and Iryna Gurevych. 2021a. Smelting gold and silver for improved multilingual amr-to-text generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Punta Cana, November 7-11, 2021*.

Leonardo F. R. Ribeiro, Yue Zhang, Claire Gardent, and Iryna Gurevych. 2020. Modeling global and local node contexts for text generation from knowledge graphs. *Transactions of the Association for Computational Linguistics*, 8:589–604.

Leonardo F. R. Ribeiro, Yue Zhang, and Iryna Gurevych. 2021b. Structural adapters in pretrained language models for amr-to-text generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Punta Cana, November 7-11, 2021*.

Martin Schmitt, Leonardo F. R. Ribeiro, Philipp Dufter, Iryna Gurevych, and Hinrich Schütze. 2021. Modeling graph structure via relative position for text generation from knowledge graphs. In *Proceedings of the Fifteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-15)*, pages 10–21, Mexico City, Mexico. Association for Computational Linguistics.

Martin Schmitt, Leonardo F. R. Ribeiro, Philipp Dufter, Iryna Gurevych, and Hinrich Schütze. 2020. Modeling graph structure via relative position for better text generation from knowledge graphs. *arXiv e-prints*.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2018. A graph-to-sequence model for AMR-to-text generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1616–1626, Melbourne, Australia. Association for Computational Linguistics.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL 2008 shared task on joint parsing of syntactic and semantic dependencies. In *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 159–177, Manchester, England. Coling 2008 Organizing Committee.

Bayu Distiawan Trisedya, Jianzhong Qi, Rui Zhang, and Wei Wang. 2018. GTR-LSTM: A triple encoder for sentence generation from RDF data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1627–1637, Melbourne, Australia. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Pavlos Vougiouklis, Hady Elsahar, Lucie-Aimée Kaffee, Christophe Gravier, Frédérique Laforest, Jonathon Hare, and Elena Simperl. 2018. Neural wikipedian: Generating textual summaries from knowledge base triples. *Journal of Web Semantics*, 52-53:1 – 15.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. 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 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Tianming Wang, Xiaojun Wan, and Hanqi Jin. 2020. Amr-to-text generation with graph transformer. *Transactions of the Association for Computational Linguistics*, 8:19–33.

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, Copenhagen, Denmark. Association for Computational Linguistics.

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

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019a. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32, pages 5753–5763. Curran Associates, Inc.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019b. Xlnet: Generalized autoregressive pretraining for language understanding. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 5753–5763. Curran Associates, Inc.

Shaowei Yao, Tianming Wang, and Xiaojun Wan. 2020. Heterogeneous graph transformer for graph-to-sequence learning. In *Proceedings of the 58th Annual Meeting of the Association for Computational*

Linguistics, pages 7145–7154, Online. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Heyang Er, Suyi Li, Eric Xue, Bo Pang, Xi Victoria Lin, Yi Chern Tan, Tianze Shi, Zihan Li, Youxuan Jiang, Michihiro Yasunaga, Sungrok Shim, Tao Chen, Alexander Fabbri, Zifan Li, Luyao Chen, Yuwen Zhang, Shreya Dixit, Vincent Zhang, Caiming Xiong, Richard Socher, Walter Lasecki, and Dragomir Radev. 2019. CoSQL: A conversational text-to-SQL challenge towards cross-domain natural language interfaces to databases. 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 1962–1979, Hong Kong, China. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Chao Zhao, Marilyn Walker, and Snigdha Chaturvedi. 2020a. 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, Online. Association for Computational Linguistics.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. 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 563–578, Hong Kong, China. Association for Computational Linguistics.

Yanbin Zhao, Lu Chen, Zhi Chen, Ruisheng Cao, Su Zhu, and Kai Yu. 2020b. Line graph enhanced AMR-to-text generation with mix-order graph attention networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 732–741, Online. Association for Computational Linguistics.

Jie Zhu, Junhui Li, Muhua Zhu, Longhua Qian, Min Zhang, and Guodong Zhou. 2019. Modeling graph structure in transformer for better AMR-to-text generation. 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 5459–5468, Hong Kong, China. Association for Computational Linguistics.

Appendices

In this supplementary material, we provide: (i) additional information about the data used in the experiments, and (ii) results that we could not fit into the main body of the paper.

A AMR Input Representation

We test three variants for the representation of the input AMR graph. Following previous work (Konstas et al., 2017; Mager et al., 2020), we evaluate (i) only node representation, where the edge information is removed from the linearization; (ii) depth-first search (DFS) through the graph and the (iii) PENMAN representation. An example for each representation is illustrated below:

only nodes	`value interrogative commodity`
	`true`

DFS	`value :mode interrogative`
	`:ARG1 commodity :ARG1-of`
	`true`

PENMAN	`( value :mode interrogative`
	`:ARG1 ( commodity ) :ARG1-of`
	`( true ) )`

In this experiment we employ T5_{small}. Table 7 shows the results on the AMR development set. The PENMAN representation leads to best results. Therefore, this representation is used in the rest of the experiments.

Input	BLEU
only nodes	28.22
DFS	34.94
PENMAN	38.27

Table 7: Results on the AMR dev set using T5_{small} for different AMR linearizations.

B Cross-domain Adaptation

For a given task, it is not always possible to collect closely related data – as we saw, e.g., for WebNLG. We therefore report STA in a cross-domain setting for the different KG-to-text benchmarks. Table 8 shows the results using $\text{BART}_{\text{base}}$ and T5_{base}. While the texts in KGAIA and AGENDA share the domain of scientific abstracts, texts in WebNLG are more general. Also note that WebNLG graphs do not share any relations with the other KGs. For $\text{BART}_{\text{base}}$, STA increases the performance in the cross-domain setting in most of the cases. For T5_{base}, STA in KGAIA improves the performance on WebNLG.

In general, we find that exploring additional adaptive pretraining for graph-to-text generation can improve the performance even if the data do not come from the same domain.

STA on	Fine-tuned & Evaluated on	
	WebNLG-*Seen*	AGENDA
	$\text{BART}_{\text{base}}$	
None	58.71	22.01
KGAIA	63.20	23.48
WebNLG	-	21.98
AGENDA	61.25	-
	T5_{base}	
None	62.93	20.73
KGAIA	63.19	22.44
WebNLG	-	20.27
AGENDA	62.75	-

Table 8: Effect (measured with BLEU score) of cross-domain STA.

C Input Graph Size

Figure 4 visualizes T5_{small}'s performance with respect to the number of input graph triples in WebNLG dataset. We observe that T5^{order} and T5^{shuf} perform similarly for inputs with only one triple but that the gap between the models increases with larger graphs. While it is obviously more difficult to reconstruct a larger graph than a smaller one, this also suggests that the graph structure is more taken into account for graphs with more than 2 triples. For the *unseen* setting, the performance gap for these graphs is even larger, suggesting that the PLM can make more use of the graph structure when it has to.

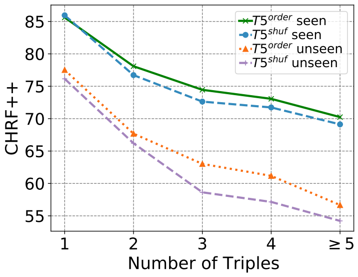


Figure 4: chrF++ scores with respect to the number of triples for WebNLG *seen* and *unseen* test sets.

D Graph Statistics

In Table 9, we present the graph properties of the three datasets. All statistics are calculated using

	AMR			WebNLG			AGENDA		
min, avg and max number of nodes	2	28.6	335	2	6.8	15	2	10.5	80
min, avg and max node degrees	1	2.2	21	1	1.7	7	1	1.67	15
min, avg and max number of edges	1	32.3	554	1	5.9	14	1	8.8	124
min, avg and max graph diameter	1	12.2	40	1	4.1	10	1	3.1	20
min, avg and max shortest path length	0	7.49	40	0	2.4	10	0	2.3	20

Table 9: Graph statistics of AMR, WebNLG and AGENDA datasets. The values are calculated using the training data. Note that AMR graphs contain a more complex structure than WebNLG and AGENDA graphs.

the Levi transformation (Beck et al., 2018) of the undirected version of the graphs, where edges are also considered nodes in the graph. WebNLG and AGENDA datasets contain disconnected graphs, and we use the largest subgraph to calculate the diameter. Note that AMR graphs have a much more complex structure: (i) they have more nodes and edges than WebNLG and AGENDA graphs; (ii) the average graph diameter and the average shortest path between nodes in AMRs are at least three times larger than in WebNLG and AGENDA graphs; (iii) nodes in AMRs have larger degrees than nodes in WebNLG and AGENDA graphs.

	AMR17	WebNLG	AGENDA
#Train	36,521	18,102	38,720
#Dev	1,368	872	1,000
#Test	1,371	1,862	1,000
#Relations	155	373	7
Avg #Tokens	16.1	31.5	157.9

Table 10: Statistics for the graph-to-text benchmarks.

	Title	Abstract	KG
Vocab	48K	173K	113K
Tokens	2.1M	31.7M	9.6M
Entities	-	-	3.7M
Avg Length	11.1	167.1	-
Avg #Nodes	-	-	19.9
Avg #Edges	-	-	9.4

Table 11: Statistics for the KGAIA dataset.

Model	chrF++	BS (F1)	MS
Schmitt et al. (2020)	44.53	-	-
Ribeiro et al. (2020)	46.37	-	-
BART$_{\text{base}}$	48.02	89.36	34.33
BART$_{\text{large}}$	**50.44**	88.74	32.24
T5$_{\text{small}}$	44.91	88.56	30.25
T5$_{\text{base}}$	48.14	88.81	31.33
T5$_{\text{large}}$	48.14	**89.60**	**35.23**
with task-adaptive pretraining			
BART$_{\text{large}}$ + LMA	51.33	89.12	33.42
T5$_{\text{large}}$ + LMA	49.37	89.75	36.13
BART$_{\text{large}}$ + STA	***51.63***	89.27	34.28
T5$_{\text{large}}$ + STA	50.27	***89.93***	***36.86***

Table 12: Results of the chrF++, BertScore (BS) and MoverScore (MS) scores for AGENDA test set. **Bold** (***Italic***) indicates best scores without (with) task-adaptive pretraining.

Model	chrF++	BS (F1)	MS
Guo et al. (2019)	57.30	-	-
Zhu et al. (2019)	64.05	-	-
Cai and Lam (2020b)	59.40	-	-
Wang et al. (2020)	65.80	-	-
Yao et al. (2020)	65.60	-	-
based on PLMs			
Mager et al. (2020)	63.89	-	-
BART$_{\text{base}}$	66.65	95.22	60.78
BART$_{\text{large}}$	71.06	96.08	65.74
T5$_{\text{small}}$	68.78	95.62	63.70
T5$_{\text{base}}$	70.81	95.99	65.63
T5$_{\text{large}}$	**72.57**	**96.27**	**67.37**
with task-adaptive pretraining			
BART$_{\text{large}}$ + LMA	71.14	95.94	64.75
T5$_{\text{large}}$ + LMA	72.83	96.32	67.44
BART$_{\text{large}}$ + STA (200K)	72.26	96.21	66.75
BART$_{\text{large}}$ + STA (2M)	73.58	96.43	68.14
T5$_{\text{large}}$ + STA (200K)	74.09	96.51	68.86
T5$_{\text{large}}$ + STA (2M)	***74.79***	***96.59***	***69.53***

Table 13: Results of the chrF++, BertScore (BS) and MoverScore (MS) scores for the LDC2017T10 test set. **Bold** (***Italic***) indicates the best score without (with) task-adaptive pretraining.

	T/F	Input Facts	T5$^{\text{order}}$	T5$^{\text{shuf}}$
(1)	S	• capital • leader Name • London • Pound sterling • United Kingdom • leader Name • United Kingdom • Elizabeth II • United Kingdom • Boris Johnson • London • currency	The capital city is London, the currency is the Pound sterling and the leader is Elizabeth II. Boris Johnson is also a leader in the UK.	The capital of the United Kingdom is London, the currency is the Pound sterling and the country is lead by Elizabeth II and Boris Johnson.
(2)	T	• Germany • capital • Berlin	Berlin is the capital of Germany.	Berlin is the capital of Germany.
(3)	F	• Berlin • capital • Germany	Berlin's capital is Germany.	Berlin is the capital of Germany.
(4)	F	• Leinster • is Part Of • Dublin	Leinster is part of Dublin.	Leinster is part of Dublin.
(5)	F	• Rome • capital • Italy	Rome's capital is Italy.	Rome is the capital of Italy.
(6)	T	• Italy • capital • Rome	Italy's capital is Rome.	Rome is the capital of Italy.
(7)	T	• Texas • capital • Austin • Andrews County Airport • location • Texas	Austin is the capital of Texas where Andrews County Airport is located.	Austin is the capital of Texas where Andrews County Airport is located.
(8)	F	• Austin • capital • Texas • Andrews County Airport • location • Texas	The capital of Austin is Texas and Andrews County Airport is located in Texas.	Andrews County Airport is located in Texas where Austin is the capital.

Table 14: Example generations from shuffled (S), true (T), and corrupted (F) triple facts by T5$_{\text{small}}$, fine-tuned on correctly ordered triples (*order*) and randomly shuffled input (*shuf*).

D	Model	Examples
AMR	Reference	I had to deal with verbal abuse from my dad for a long 8 years before I came to uni and honestly, the only reason why I'm here is because it was the only way out.
	T5	I had to deal with 8 years of verbal abuse from my dad before coming to university and honestly the only reason I'm here is because it's the only way out.
	BART	I had to deal with my dad's verbal abuse for 8 years long before coming to uni and honestly the only reason I'm here is because it's the only way out.
	Mager et al. (2020)	i've had to deal with verbal abuse from my dad for 8 years (before i came to uni i was honestly the only reason i was here) and it's only because of the way it is.
WebNLG	Reference	Aaron Turner is an electric guitar player who has played with the black metal band Twilight and with Old Man Gloom. Death metal is a musical fusion of black metal.
	T5	Aaron Turner plays the electric guitar and is associated with the band Twilight. He is also a member of the Old Man Gloom band. Black metal and death metal are both genres of music.
	BART	The black metal genre is a variation of the death metal genre. It is also where the band, Twilight, and the alternative rock band, Old Man Gloom, are from. One of the members of the band is Aaron Turner, who plays the electric guitar.
	Harkous et al. (2020)	Aaron Turner, a.k.a. Black Metal, is a member of the Twilight (band) and Old Man Gloom bands. He also plays electric guitar and has a strong connection with the Death Metal genre.

Table 15: Examples of text generated by the different models. D refers to the dataset.

Model	**Examples**
Reference	in this paper, a new array signal processing technique by using particle swarm optimization is proposed to identify multipath channel parameters. the proposed array signal processing technique provides estimates to the channel parameters by finding a global minimum of an optimization problem. since the optimization problem is formulated in the cross-ambiguity function (caf) domain of the transmitted signal and the received array outputs, the proposed array signal processing technique is called as particle swarm optimization. the performance of the particle swarm optimization is compared with the space alternating generalized expectation maximization technique and with another recently proposed pso based technique for various snr values. simulation results indicate the superior performance of the pso based technique over mentioned techniques for all snr values.
T5	this paper, we propose an array signal processing technique based on particle swarm optimization to estimate the multipath channel parameters from the received and the transmitted signal in a cross-ambiguity domain. the proposed array signal processing technique is formulated as an optimization problem and particle swarm optimization is compared with the generalized expectation maximization technique. simulation results show that the proposed particle swarm optimization outperforms the pso based technique for a wide range of snr values. in addition, the proposed particle swarm optimization is computationally more efficient than the gem based technique for small snr values.
BART	in this paper, a new array signal processing technique based on particle swarm optimization is proposed. the proposed array signal processing technique is used to estimate the multipath channel parameters from the transmitted signal. the proposed array signal processing technique is formulated as an optimization problem in the cross-ambiguity domain. the particle swarm optimization is compared with the generalized expectation maximization technique. simulation results show that the proposed particle swarm optimization outperforms the pso based technique for all snr values. furthermore, the proposed particle swarm optimization is able to estimate the channel parameters more accurately than the generalized expectation maximization technique.
Ribeiro et al. (2020)	in this paper, a novel array signal processing technique based on particle swarm optimization is proposed to estimate the multipath channel parameters from the transmitted signal. the proposed array signal processing technique uses particle swarm optimization to estimate the multipath channel parameters. the proposed array signal processing technique is formulated as an optimization problem. simulation results show that the proposed array signal processing technique outperforms the conventional generalized expectation maximization technique and the pso based technique is robust to the snr values.

Table 16: Examples of text generated by the different models trained on the AGENDA dataset.