

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 recently proposed 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 the PLMs BART and T5 achieve new state-of-the-art results and that task-adaptive pretraining strategies improve their performance even further. In particular, we report new state-of-the-art BLEU scores of 49.72 on LDC2017T10, 59.70 on WebNLG, and 25.66 on AGENDA datasets - a relative improvement of 31.8%, 4.5%, and 42.4%, respectively. In an extensive analysis, we identify possible reasons for the PLMs' success on graph-to-text tasks. We find evidence that their knowledge about true facts helps them perform well even when the input graph representation is reduced to a simple bag of node and edge labels.¹

1 Introduction

Graphs are important data structures as they represent complex relations between a set of objects. For example, syntactic and semantic structures of sentences can be represented using different graph representations (Bastings et al., 2017; Banarescu et al., 2013) and knowledge graphs (KGs) are used to describe factual knowledge in the form of relations between entities (Gardent et al., 2017).

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 numerous applications such as question answering (Duan et al., 2017), dialogue generation (Moon et al., 2019), and summarization (Fan et al., 2019).

Transfer learning has become ubiquitous in NLP and pretrained Transformer-based architectures have considerably outperformed prior state of the art (Devlin et al., 2019; Liu et al., 2020; Radford et al., 2019). Following this trend, recent works (Mager et al., 2020; Harkous et al., 2020) apply transfer learning to data-to-text generation, where a language model is first pretrained on large corpora before being fine-tuned on the target task.

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 are also the first to investigate 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), our approaches based on PLMs consistently outperform these models, even though PLMs – as sequence models – do not exhibit any graph-specific structural bias.² This puts into question the importance of encoding the structure of a graph in the presence of a strong language model. In our analysis we investigate to what extent fine-tuned

¹Our code and pretrained model checkpoints are available at <https://github.com/UKPLab/plms-graph2text>.

²The 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.

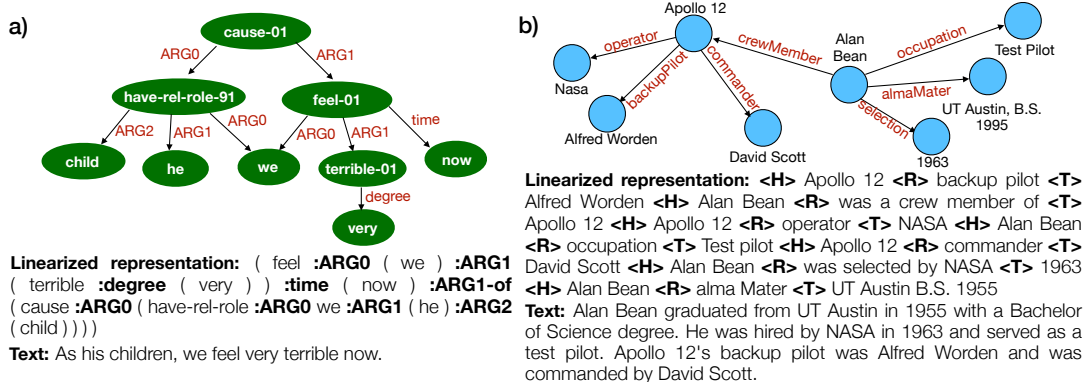


Figure 1: Examples of (a) AMR and (b) WebNLG graphs, the input for the models and the reference texts.

PLMs make use of the graph structure and whether they need it at all. We notably observe that PLMs can 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. We also find evidence that factual knowledge from the pre-training phase poses a strong bias on the texts generated by PLMs – to the extent that even unseen corrupted graph facts lead to correct output texts. In summary, our contributions are the following: (1) We examine and compare two PLMs, BART and T5, for graph-to-text generation, exploring *language model adaptation* (LMA) and *supervised task adaptation* (STA) pretraining strategies, employing additional task-specific data. (2) Our approaches consistently outperform the state of the art by a significant margin on three established graph-to-text benchmarks from different domains. (3) We demonstrate 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. (4) We present evidence that the *knowledge about facts* acquired during pretraining gives PLMs such an advantage on KG-to-text benchmarks that their performance is almost the same with or without access to the graph structure.

2 Related Work

Graph-to-text generation can be divided into two main tasks: generating text (i) from meaning representations (Konstas et al., 2017) and (ii) from KGs (Gardent et al., 2017).

AMR-to-Text Generation. 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. Various neural models have been proposed to generate sentences from AMR graphs. Konstas et al. (2017) propose the first neural approach for AMR-to-text generation that uses a linearized input graph. Recent approaches (Song et al., 2018; Beck et al., 2018; Damonte and Cohen, 2019; Ribeiro et al., 2019; Zhao et al., 2020a) propose architectures based on GNNs to directly encode the AMR graph structure. Other methods (Zhu et al., 2019; Cai and Lam, 2020b; Wang et al., 2020; Song et al., 2020; Yao et al., 2020) employ Transformers to learn node representations injecting the graph structure into the self-attention aggregation.

KG-to-Text Generation. Recent neural approaches for KG-to-text generation linearize the KG triples as input to sequence-to-sequence models (Trisedya et al., 2018; Moryossef et al., 2019; Castro Ferreira et al., 2019). Marcheggiani and Perez Beltrachini (2018) use GNNs to capture node contexts, and demonstrate superior performance compared to LSTMs. Koncel-Kedziorski et al. (2019) propose a Transformer-based approach which directly encodes the input graph structure. Most recent approaches (Ribeiro et al., 2020; Schmitt et al., 2020) propose to encode both global and local node contexts in order to better capture the graph topology.

Pretrained Language Models. Pretrained Transformer-based models, such as BERT (Devlin et al., 2019), XLNet (Yang et al., 2019), 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 on many natural language generation (NLG) tasks. Mager et al. (2020) were the first to employ a pretrained Transformer-based language model – namely GPT-2 – for AMR-to-text generation. Very 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. Different from the above works, we do not only investigate standard fine-tuning approaches but also new task-adaptive pretraining approaches for BART and T5, and we also provide the first analysis aimed at explaining the good performance of PLMs at graph-to-text tasks. Concurrent to our work, Radev et al. (2020) propose DART, a new data-to-text dataset, and apply BART to the WebNLG dataset, augmenting the training data. Our study not only considers more benchmarks and PLMs, but also differs in that it focuses on transfer learning strategies that separate task-adaptive pretraining from fine-tuning on the actual task training data as opposed to the training data augmentation in Radev et al. (2020).

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 consider additional pretraining for NLG with PLMs.

3 PLMs for Graph-to-Text Generation

3.1 Models in this Study

We investigate BART and T5, 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.

BART is pretrained as a text-to-text denoising autoencoder: first, the input text is corrupted with a random noise function; then BART is trained to reconstruct the original text. The training corpus is a combination of books and Wikipedia data. We evaluate BART versions with different capacity: *base* with 140M and *large* with 400M parameters.

T5 generalizes the text-to-text architecture to a variety of NLP tasks. The model is pretrained with randomly corrupted text spans with different mask

ratios and span sizes. The training corpus is C4, a large cleaned corpus of web texts. We experiment with the following variants: *small* with 60M, *base* with 220M, and *large* with 770M parameters.

We fine-tune BART and T5 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-adaptive Pretraining

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 pre-training 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 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. 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 in three graph-to-text benchmarks: AMR17 (LDC2017T10), WebNLG (Gardent et al., 2017), and AGENDA (Koncel-Kedziorski et al., 2019). Table 8 in the Appendix shows statistics for each dataset.

AMR17. An instance in LDC2017T10 consists of a sentence annotated with its corresponding AMR graph. Following Mager et al. (2020), we

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

linearize the AMR graphs using the PENMAN notation (see Figure 1a).

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 AMR17) 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 Gigaword⁴ corpus and use a state-of-the-art AMR parser (Cai and Lam, 2020a) to parse them into AMR graphs.⁵ 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.⁶

⁴<https://catalog.ldc.upenn.edu/LDC2003T05>

⁵We filter out sentences that do not yield well-formed AMR graphs.

⁶Note that Gigaword and AMR17 share similar data sources.

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). Table 9 in the Appendix shows relevant dataset statistics.

5 Experiments

We develop our experiments based on pretrained models released by HuggingFace (Wolf et al., 2019). 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. We add all edge labels seen in the training set to the vocabulary of the models for AMR17. For the KG datasets, we add the $\langle H \rangle$, $\langle R \rangle$, and $\langle T \rangle$ tokens to the models’ vocabulary. Dev BLEU is used for model selection.

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

5.1 Results on AMR-to-Text

Table 2 shows our results for the setting without additional pretraining, with additional self-supervised task-adaptive pretraining solely using the collected

Model	BLEU			METEOR			chrF++		
	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	-	-	-
<i>based on PLMs</i>									
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 _{base}	53.11	62.74	41.53	40.18	44.45	35.36	70.02	76.68	62.76
BART _{large}	54.72	63.45	43.97	42.23	45.49	38.61	72.29	77.57	66.53
T5 _{small}	56.34	65.05	45.37	42.78	45.94	39.29	73.31	78.46	67.69
T5 _{base}	59.17	64.64	52.55	43.19	46.02	41.49	74.82	78.40	70.92
T5 _{large}	59.70	64.71	53.67	44.18	45.85	42.26	75.40	78.29	72.25

Table 1: Results on WebNLG. A, S and U stand for *all*, *seen*, and *unseen* partitions of the test set, respectively.

Model	BLEU	M	BT
Ribeiro et al. (2019)	27.87	33.21	-
Zhu et al. (2019)	31.82	36.38	-
Cai and Lam (2020b)	29.80	35.10	-
Zhao et al. (2020b)	32.46	36.78	-
Wang et al. (2020)	33.90	37.10	-
Yao et al. (2020)	34.10	38.10	-
<i>based on PLMs</i>			
Mager et al. (2020)	33.02	37.68	-
Harkous et al. (2020)	37.70	38.90	-
BART _{base}	36.71	38.64	52.47
BART _{large}	43.47	42.88	60.42
T5 _{small}	38.45	40.86	57.95
T5 _{base}	42.54	42.62	60.59
T5 _{large}	45.80	43.85	61.93
<i>with task-adaptive pretraining</i>			
BART _{large} + LMA	43.94	42.36	58.54
T5 _{large} + LMA	46.06	44.05	62.59
BART _{large} + STA (200K)	44.72	43.65	61.03
BART _{large} + STA (2M)	47.51	44.70	62.27
T5 _{large} + STA (200K)	48.02	44.85	63.86
T5 _{large} + STA (2M)	49.72	45.43	64.24

Table 2: Results on AMR-to-text generation for the AMR17 test set. M and BT stand for METEOR and BLEURT, respectively. **Bold (Italic)** indicates the best score without (with) task-adaptive pretraining.

Gigaword sentences (LMA), and with additional supervised task adaptation (STA), before fine-tuning. We also report several recent results on the AMR17 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. We are the first to employ BART and T5, which have both a Transformer-based encoder and decoder, in AMR-to-text generation.

Only considering approaches without task adaptation, BART_{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_{large} performs best. The other metrics follow similar trends. See Table 10 in the Appendix for evaluation with more automatic metrics.

Task-specific Pretraining. 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 graphs. 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. After task adaptation, T5 achieves 49.72 BLEU points, the new state of the art for AMR-to-text generation. In general, models pretrained on the STA setup converge faster than without task-adaptive pretraining. For example, T5_{large} without additional pretraining converges after 5 epochs of fine-tuning whereas T5_{large} with STA already converges after 2 epochs.

5.2 Results on WebNLG

Table 1 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	-
BART _{base}	22.01	23.54	-13.02
BART _{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
<i>with task-adaptive pretraining</i>			
BART _{large} + LMA	25.30	25.54	-08.79
T5 _{large} + LMA	22.92	24.40	-10.39
BART _{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 pre-training.

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. Zhao et al. (2020a) leverage additional information about the order that the triples are realized and achieve the best performance among approaches that do not employ PLMs. Note that T5 is also used in Kale (2020). A particular difference in our T5 setup is that we add a 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 hypothesize that the performance gap between *seen* and *unseen* sets stems from the advantage obtained by a model seeing examples of relation-text pairs during training. 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: $\langle \text{Abdul Taib Mahmud}, \text{party}, \text{Parti Bumiputera Sarawak} \rangle$. 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 conduct experiments with LMA or STA for WebNLG. However, we explore cross-domain STA in additional experiments, which we discuss in Appendix A.2.

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

Model	AMR17	
	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
BART _{large}	6.70 ^B	5.72 ^{BC}
Reference	5.91 ^A	-
Model	WebNLG	
	F	A
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
BART _{large}	6.53 ^C	6.50 ^B
Reference	5.89 ^B	6.47 ^B

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

fluent text helps when generating paper abstracts, even though they were not pretrained in the scientific domain. BART_{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.

Task-specific Pretraining. 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 AMR17. We hypothesize that this is due to the fact that the graphs do not completely cover the target text, making this dataset more challenging. See Table 11 in the Appendix for more automatic metrics.

5.4 Human Evaluation

To further assess the quality of the generated text, we conduct a human evaluation on AMR17 and WebNLG via crowd sourcing on Amazon Mechanical Turk.⁷ 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 close in meaning is the generated text to the reference sentence?) for AMR17; (ii) *Adequacy* (i.e., does the text clearly express the data?) for WebNLG. We randomly select 100 texts from the

⁷We 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

↓ $T5^{order}$

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

↓ $T5^{shuf}$

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.

Shuffle →

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 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 $T5_{small}$ and a gold reference (bottom). Note that $T5$ can produce a reasonable text even when the input triples are shuffled randomly.

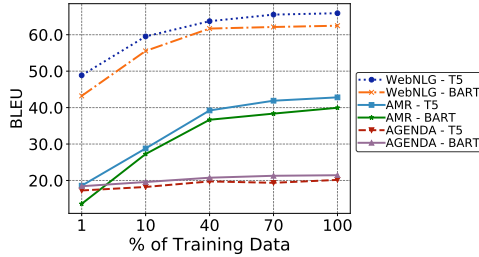


Figure 3: Performance of $BART_{base}$ and $T5_{base}$ in the dev set when experimenting with different amounts of training data.

generations 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.⁸

Table 4 shows the results. There is a similar trend as in the automatic evaluation. Our approaches improve the fluency, meaning similarity, and adequacy on the two 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 AMR17, where our approach also has the largest BLEU improvement (+8.10) over Harkous et al. (2020). Finally, note that both models score higher than the gold sentences in fluency, highlighting their strong language generation abilities.

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 $BART_{base}$ and $T5_{base}$ already achieve similar performance as using the entire training data in all three benchmarks. Also note that in a low-resource scenario $T5_{base}$ considerably

⁸Inter-annotator agreement for the three criteria ranged from 0.40 to 0.79, with an average Krippendorff’s α of 0.56.

Model	AMR17	WebNLG	AGENDA
$T5^{order}$	36.83	63.41	19.86
$T5^{shuf}$	15.56	61.54	19.08

Table 5: Impact (measured with BLEU) of using a bag of entities and relations (*shuf*) as input for $T5_{small}$.

outperforms $BART_{base}$ in both datasets. In particular, with only 1% of training examples, the difference between $T5$ and $BART$ is 4.98 and 5.64 BLEU points for AMR and WebNLG, respectively. This suggests that $T5$ is a good candidate to be employed in low-resource graph-to-text tasks. Interestingly, the amount of training data has very little influence on the models’ performance for AGENDA.

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 $T5_{small}$ ’s performance when its input contains correctly ordered triples ($T5^{order}$) 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. We see that as evidence that the graph structure is similarly well

T/F	Input Fact	T5 ^{order}	T5 ^{shuf}	CGE-LW
(1) F	• Ohio • is Part Of • Cleveland	Ohio is part of Cleveland.	Cleveland is part of Ohio.	ohio is part of cleveland.
(2) F	• United States • is Part Of • Amarillo • Texas	Amarillo, Texas is part of the United States.	Amarillo, Texas is part of the United States.	united states is part of amarillo, texas.
(3) F	• Leinster • is Part Of • Dublin	Leinster is part of Dublin.	Leinster is part of Dublin.	leinster is part of dublin.
(4) T	• Italy • capital • Rome	italy’s capital is rome.	Rome is the capital of Italy.	rome is the capital of italy.
(5) F	• Rome • capital • Italy	Rome’s capital is Italy.	Rome is the capital of Italy.	rome is the capital of rome.
(6) T	• italy • capital • rome	Italy’s capital is rome.	Italy’s capital is rome.	-
(7) F	• rome • capital • italy	The capital of rome is italy.	Italy is the capital of rome.	-

Table 6: Example generations from corrupted (F) and true (T) WebNLG dev set facts by T5_{small} fine-tuned on correctly ordered nodes (*order*) and randomly shuffled nodes (*shuf*) from the WebNLG training set, and CGE-LW.

captured by T5^{order}. Without the graph structure (T5^{shuf}), AMR-to-text performance drops significantly. KG-to-text performance, however, 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. It has recently been argued that large PLMs acquire a certain amount of factual knowledge during pretraining (Petroni et al., 2019). We hypothesize that this knowledge makes it easier to recover KG facts based on a set of entities and relations than to reconstruct a corrupted AMR.

6.2 Qualitative Analysis

To further test our hypothesis that PLMs make use of their knowledge about true facts during KG-to-text generation, we take example facts from the WebNLG dev set, corrupt them, and feed them to both T5^{order} and T5^{shuf}. We also feed those triples to CGE-LW (Ribeiro et al., 2020), a state-of-the-art KG-to-text model trained from scratch, i.e., without any pretraining. Table 6 shows the generated texts.

The model trained on correctly ordered input has learned a bit more to rely on the input graph structure. The false fact in example (1) is reliably transferred to the text by T5^{order} but not by T5^{shuf}, which silently corrects it. But even T5^{order} is not completely free from its factual knowledge bias, as illustrated in example (2) where both models refuse to generate an incorrect fact. This indicates that facts seen during pretraining serve as a strong guide during text generation, even for models that were fine-tuned with a clearly marked graph structure. The fact that CGE-LW, a graph encoder model trained from scratch on the WebNLG training set, has no difficulties in textualizing the false triples (except example 5) further supports this argument.

Interestingly, both T5 models leave the wrong input in (3) uncorrected. The fact that Leinster is a region in Ireland and not, e.g., a neighborhood

of the city Dublin is probably unknown to T5. It seems that T5 falls back to the order of words in the input in such a case. Examples (4)–(7) also illustrate this behavior. While the well-known entities “Rome” and “Italy” produce a similar behavior as “Ohio” and “Cleveland”, i.e., T5^{order} complies with generating a false statement and T5^{shuf} rather follows its factual knowledge, lowercasing the entity names changes that. With the unknown entities “rome” and “italy”, both (case-sensitive) models fall back to the order of the input for their generations.

This experiment is related to testing factuality and trustworthiness of text generation models (Wiseman et al., 2017; Falke et al., 2019). It is important for a generation model to stay true to its input as its practical usefulness can be severely limited otherwise. We are the first to detect this issue with the use of PLMs for data-to-text tasks.

7 Conclusion

We investigated two pretrained language models (PLMs) for graph-to-text generation and show that language model adaptation (LMA) and supervised task adaptation (STA) pretraining strategies are beneficial for this task. Our approaches outperform the state of the art by a substantial margin on three graph-to-text benchmarks. We also examined to what extent the graph structure is taken into account for the text generation process, and we found evidence that factual knowledge is a strong guide for these models. We believe that PLMs will play an important role in future endeavors to solve graph-to-text generation tasks and we expect our work to serve as guidance and as a strong baseline for them. A promising direction for future work is to explore ways of injecting a stronger graph-structural bias into large PLMs to thus possibly leveraging their strong language modeling capabilities and keeping the output faithful to the input graph.

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.
- 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.
- Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima’an. 2017. [Graph convolutional encoders for syntax-aware neural machine translation](#). In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1957–1967, Copenhagen, Denmark. 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.
- 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.
- 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.
- Angela Fan, Claire Gardent, Chloé Braud, and Antoine Bordes. 2019. [Using local knowledge graph construction to scale Seq2Seq models to multi-document inputs](#). 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 4186–4196, Hong Kong, China. 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](#). *arXiv e-prints*.
- 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*.
- 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 pre-training 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 of the 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](#).
- 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, 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*.
- 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, Ante Wang, Jinsong Su, Yue Zhang, Kun Xu, Yubin Ge, and Dong Yu. 2020. [Structural information preserving for graph-to-text generation](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7987–7998, 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.
- 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, Łukasz 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.
- David Wadden, Ulme Wennberg, Yi Luan, and Hananeh 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. 2019. [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.

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.

A 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.1 Input Graph Size

Figure 4 visualizes the $T5_{\text{small}}$ ’s performance with respect to the number of input graph triples in WebNLG dataset. We observe that $T5^{\text{order}}$ and $T5^{\text{shuf}}$ 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.

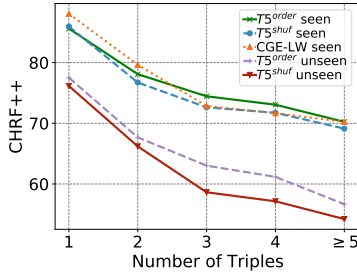


Figure 4: chrF++ scores with respect to the number of triples for WebNLG *seen* and *unseen* test sets.

A.2 Cross-domain Pretraining

For a given task, it is not always possible to collect closely related data – as we saw, e.g., for WebNLG. We therefore investigate how STA can help in a cross-domain setting for different KG-to-text benchmarks. Table 7 shows the results using $BART_{\text{base}}$ and $T5_{\text{base}}$. 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 $BART_{\text{base}}$, STA increases the performance in the cross-domain setting in most of the cases. For $T5_{\text{base}}$, STA in KGAIA improves the performance on WebNLG.

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

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

Table 7: Effect (measured with BLEU score) of cross-domain STA.

	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 #Nodes	15.63	4.0	13.4
Avg #Tokens	16.1	31.5	157.9

Table 8: 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 9: Statistics for the KGAIA dataset.

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

Table 10: Additional results for the AMR17 test set. BS, MS stand for BertScore, MoverScore, respectively. **Bold (Italic)** indicates the best score without (with) task-adaptive pretraining.

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

Table 11: Additional results on AGENDA test set. **Bold (Italic)** indicates best scores without (with) task-adaptive pretraining.