GPT-too: A Language-Model-First Approach for AMR-to-Text Generation

Manuel Mager[1*] **Ramón Fernandez Astudillo**[2] **Tahira Naseem**[2] **Md Arafat Sultan**[2]
Young-Suk Lee[2] **Radu Florian**[2] **Salim Roukos**[2]

[1] Institute for Natural Language Processing,
University of Stuttgart, Germany
[2] IBM Research AI, Yorktown Heights, NY 10598, USA
manuel.mager@ims.uni-stuttgart.de
{ramon.astudillo,arafat.sultan}@ibm.com
{tnaseem, ysuklee}@us.ibm.com

Abstract

Abstract Meaning Representations (AMRs) are broad-coverage sentence-level semantic graphs. Existing approaches to generating text from AMR have focused on training sequence-to-sequence or graph-to-sequence models on AMR annotated data only. In this paper, we propose an alternative approach that combines a strong pre-trained language model with cycle consistency-based re-scoring. Despite the simplicity of the approach, our experimental results show these models outperform all previous techniques on the English LDC2017T10 dataset, including the recent use of transformer architectures. In addition to the standard evaluation metrics, we provide human evaluation experiments that further substantiate the strength of our approach.

1 Introduction

Abstract Meaning Representation (AMR) (Banarescu et al., 2013) is a rooted, directed, acyclic graph with labeled edges (relations) and nodes (concepts) expressing "who is doing what to whom". AMR-to-text generates sentences representing the semantics underlying an AMR graph.

Initial works in AMR-to-text used transducers (Flanigan et al., 2016), phrase-based machine translation (Pourdamghani et al., 2016) and neural sequence-to-sequence (seq2seq) models with linearized graphs (Konstas et al., 2017). Cao and Clark (2019) leverage constituency parsing for generation. Beck et al. (2018) improve upon prior RNN graph encoding (Song et al., 2018) with Levi Graph Transformations. Damonte and Cohen (2019) compare multiple representations and find graph encoders to be the best. Guo et al. (2019) use RNN graph encoders with dense graph convolutional encoding. Ribeiro et al. (2019) use RNN encoders with dual graph representations. Transformer-based seq2seq (Vaswani et al., 2017) was first applied to AMR-to-text in (Sinh and Le Minh, 2019). Zhu et al. (2019) greatly improve over the prior state-of-the-art by modifying self-attention to account for AMR graph structure. Using transformers has also been recently explored by Wang et al. (2020) who propose a mutli-head graph attention mechanism.

Pre-trained transformer representations (Radford et al., 2018; Devlin et al., 2019; Radford et al., 2019) use transfer learning to yield powerful language models that considerably outperform the prior art. They have also shown great success when fine-tuned to particular text generation tasks (See et al., 2019; Zhang et al., 2019; Keskar et al., 2019). Given their success, it would be desirable to apply pre-trained transformer models to a graph-to-text task like AMR-to-text, but the need for graph encoding precludes in principle that option. Feeding the network with some sequential representation of the graph, such as a topological sorting, looses some of the graphs representational power. Complex graph annotations, such as AMR, also contain many special symbols and special constructs that departure from natural language and may by not interpretable by a pre-trained language model.

In this paper we explore the possibility of directly fine-tuning a pre-trained transformer language model on a sequential representation of AMR graphs, despite the expected difficulties listed above. For this we re-purpose a GPT-2 language model (Radford et al., 2019) to yield an AMR-to-text system. We show that it is surprisingly easy to fine-tune GPT-2 to learn AMR graph to text mapping that outperforms the previous state-of-the-art on automatic evaluation metrics. Since a single graph AMR, graph corresponds to multiple sentences with the same meaning, we

[*] This research was done during an internship at IBM Research AI.

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 1846–1852
July 5 - 10, 2020. ©2020 Association for Computational Linguistics

also provide human evaluation and semantic similarity metric results (Zhang et al., 2020) which are less dependent on reference text. Human evaluation and semantic similarity results highlight the positive impact of a strong language model strategy. Finally we also introduce a simple re-scoring technique based on cycle-consistency that further improves performance.

2 Fine-tuning GPT-2 for conditional language generation

In order to fine-tune a generative model (GPT-2; Radford et al. (2019)) for conditional text generation, prior works fine-tune the language model to predict target text starting from the additional source text as context. In our experiments, we found it beneficial to fine-tune on the joint distribution of AMR and text instead i.e. also reconstruct the source. Given a tokenized sentence $w_1 \cdots w_N$ and the sequential AMR representation $a_1 \cdots a_M$ we maximized the joint probability

$$p_{\text{GPT-2}}(\mathbf{w}, \mathbf{a}) = \prod_{j=1}^{N} p_{\text{GPT-2}}(w_j \mid w_{1:j-1}, a_{1:M}) \cdot \prod_{i=1}^{M} p_{\text{GPT-2}}(a_i \mid a_{1:i-1})$$

A special separator token is added to mark the end of the sequential AMR representation. Special AMR symbols that should not be interpreted literally are assigned tokens from the GPT-2 unused token list. In addition to this, we also observed that freezing the input embeddings when fine-tuning had positive impact in performance.

At test time, we provide the AMR as context as in conventional conditional text generation:

$$\hat{w}_j = \arg\max_{w_j} \{p_{\text{GPT-2}}(w_j \mid w_{1:j-1}, a_{1:M})\}$$

3 Re-scoring via Cycle Consistency

The general idea of cycle consistency is to assess the quality of a system's output based on how well an external 'reverse' system can reconstruct the input from it. In previous works, cycle-consistency based losses have been used as part of the training objective in machine translation (He et al., 2016) and speech recognition (Hori et al., 2019). It has also been used for filtering synthetic training data for question answering (Alberti et al., 2019). Here we propose the use of a cycle consistency measure to re-score the system outputs.

In particular, we take the top k sentences generated by our system from each gold AMR graph and parse them using an off-the-shelf parser to obtain a second AMR graph. We then re-score each sentence using the standard AMR parsing metric Smatch (Cai and Knight, 2013) by comparing the gold and parsed AMRs.

4 Experimental setup

Following Previous works on AMR-to-text, we Use the standard LDC2017T10 AMR corpus for evaluation of the proposed model. This Corpus contains 36,521 training instances of AMR graphs in PENMAN notation and the corresponding texts. It also includes 1368 and 1371 development and test instances, respectively. We tokenize each input text using The JAMR toolkit (Flanigan et al., 2014). The concatenation of an AMR graph and the corresponding text is split into words, special symbols and sub-word units using the GPT-2 tokenizer. We add all arc labels seen in the training set and the root node :root to the vocabulary of the GPT-2model, but we freeze the embedding layer for training. We use the Hugging Face implementation of (Wolf et al., 2019) for GPT-2 small (GPT-2S), medium (GPT-2M) and large (GPT-2L). Fine-tuning converges after 6 epochs, which takes just a few hours on a V100 GPU[1]. For cycle-consistency re-scoring we use an implementation of Naseem et al. (2019) in PyTorch. For re-scoring experiments, we use a beam size of 15.

AMR input representation. we test three variants of AMR representation. First, a depth-first search (DFS) through the graph following Konstas et al. (2017), where the input sequence is the path followed in the graph. Second, to see if GPT-2 is in fact learning from the graph structure, we remove all the edges from the DFS, keeping only the concept nodes. This has the effect of removing the relation information between concepts, such as subject/object relations. As a third option, we use the PENMAN representation without any modification. The three input representations are illustrated below:

[1] Code for this paper is available at: https://github.com/IBM/GPT-too-AMR2text

Nodes	`recommend advocate-01 it vigorous`
DFS	`recommend :ARG1 advocate-01 :ARG1 it :manner vigorous`
Penman	`(r / recommend-01 :ARG1 (a / advocate-01 :ARG1 (i / it) :manner (v / vigorous)))`

Decoding. For generation, we experiment with greedy decoding, beam search, and nucleus sampling (Holtzman et al., 2019). For beam search, we explore beam sizes of 5, 10 and 15. As the system, in some cases, produces repetitive output at the end of the text, we additionally perform a post-processing step to remove these occurrences.

Metrics. We considered the three automatic evaluation metrics commonly used in previous works. We compute BLEU (Papineni et al., 2002) using SacreBLEU (Ma et al., 2019). We compute chrF++ (Popović, 2017) using both SacreBLEU and the scripts used by authors of the baseline systems. We compute METEOR (Banerjee and Lavie, 2005) with the default values for English of the CMU implementation.[2]

In addition to the standard automatic metrics, we also carry out human evaluation experiments and use the semantic similarity metric BERTScore (Zhang et al., 2020). Both metrics arguably have less dependency on the surface symbols of the reference text used for evaluation. This is particularly relevant for the AMR-to-text task, since one single AMR graph corresponds to multiple sentences with the same semantic meaning. Conventional metrics for AMR-to-text are are strongly influenced by surface symbols and thus do not capture well the ability of the system to produce a diverse sentences with same underlying semantics.

Human evaluations are carried out by three professional annotators on 51 randomly selected sentences from the 1371 test sentences, on a 6 point scale, ranging from 0 to 5.

- 0=Exceptionally poor (No useful information is conveyed at all.)
- 1=Poor (Fundamental errors in grammar and vocabulary make it difficult to understand the meaning.)
- 2=Not good enough (Errors in grammar, vocabulary and style make it difficult to understand the meaning.)
- 3=Good enough (There are errors in the text, but I am reasonably confident that I understand the meaning.)

Model	Input	BLEU	chrF++
GPT-2S Rec.	Only nodes AMR	9.45	41.59
GPT-2S Rec.	Lin. AMR w/o edges.	11.35	43.25
GPT-2S Rec.	Lin. AMR w/edges.	20.14	53.12
GPT-2S Rec.	Penman AMR	22.37	53.92
GPT-2M Rec.	Lin. AMR w/edges.	22.86	55.04
GPT-2M Rec.	Penman AMR	27.99	61.26

Table 1: Results on the LDC2017T10 development set using GPT-2 S(mall) and M(edium) with Rec(onstruction) loss (see §2) for different AMR representations (see §4).

Approach	Decoding	BLEU	chrF++
GPT-2M Conditional	Greedy	25.73	57.2
GPT-2M Rec.	Greedy	30.41	61.36
GPT-2M Rec.	BEAM	31.8	62.56
GPT-2M Rec.	BEAM 10	**32.32**	62.79
GPT-2M Rec.	Sampling	28.75	61.19

Table 2: Results on the LDC2017T10 development set. Rec(onstruction) uses the AMR reconstruction term (see §2) whereas Conditional does not.

- 4=Very good (There may be minor errors in the text, but I am very confident that I understand the meaning.)
- 5=Excellent (The information is presented clearly and with appropriate grammar, vocabulary and style.)

For each system, scores from all annotators are averaged to compute a single score. Inter-annotator agreement was 0.7 when measured by Pearson correlation coefficient.

Our system produces de-tokenized cased output after BPE decoding, whereas previous systems produce traditional tokenized lower-cased output. Therefore, we lowercase and tokenize our system outputs to have fair comparisons with previous systems.

4.1 Results

Regarding the type of AMR representation, as shown in Table 1, using directly the PENMAN notation for AMR representation leads to the best results outperforming DFS. Edge information, indicating relations between concepts, seems also to play a fundamental role since its absence strongly decreases performance in both DFS and PENMAN representations. Penman notation was chosen for the rest of the experiments.

The impact of the use of a reconstruction term explained in §2 is shown in Table 2. The model trained using this additional term achieves 30.41 BLEU and 61.36 chrF++, as opposed to 25.73

[2] https://www.cs.cmu.edu/~alavie/METEOR

System	Performance		
	BLEU	Meteor	chrF++
Beck et al. (2018)	23.30	-	50.40
Damonte and Cohen (2019)	24.54	24.07	-
Guo et al. (2019)	27.60	-	57.30
Cao and Clark (2019)	26.80	-	-
Sinh and Le Minh (2019)	18.36	-	-
Ribeiro et al. (2019)	27.87	33.21	-
Cai and Lam (2020)	29.80	35.10	59.4
Zhu et al. (2019)	31.82	36.38	**64.05**
GPT-2M Rec.	32.10$^{\blacklozenge}$	35.86$^{\diamond}$	61.81$^{\blacklozenge}$
GPT-2L Rec.	32.47$^{\blacklozenge}$	36.80$^{\diamond}$	62.88$^{\blacklozenge}$
GPT-2M Rec. re-scoring	32.98$^{\blacklozenge}$	37.33$^{\diamond}$	63.09$^{\blacklozenge}$
GPT-2L Rec. re-scoring	**33.02**$^{\blacklozenge}$	**37.68**$^{\diamond}$	**63.89**$^{\square}$

Table 3: Results on the LDC2017T10 test set for best performing models compared to other results reported in the literature. $^{\blacklozenge}$ indicates statistical significance at $(P < .01)$, $^{\diamond}$ at $(P < 0.05)$ and $^{\square}$, not significant. All significance tests are with respect to (Zhu et al., 2019).

System	LDC2017T10		
	Human Eval.		SemSim
	Avg.	P45	F1
Guo et al. (2019)	2.48	15.69%	92.68
Ribeiro et al. (2019)	2.42	16.37%	92.63
Zhu et al. (2019)	2.61	20.26%	93.31
GPT-2M Rec.	3.03	37.91%	94.55
GPT-2L Rec.	**3.04**	**41.83%**	**94.63**

Table 4: Human evaluation and semantic similarity (SemSim) results on the LDC2017T10 test set. Human evaluations (Human Eval.) show the average (Avg.) of scores (0 to 5) and the ratio of sentence evaluated between 4 and 5 (P45). All results for human evaluation are on 51 randomly selected sentences and statistically significant at $(P < 0.05)$. SemSim results are significant at $(P < 0.01)$. All significance tests refer to a comparison with (Zhu et al., 2019).

BLEU and 57.2 chrF++ without the term. We therefore use a reconstruction term training in the rest of the experiments.

Beam search improves system performance greatly over the greedy baseline with 1.91 BLEU points (see Table 2). With beam size 10, we obtain 32.32 BLEU and 62.79 chrF++. With nucleus sampling at a cumulative probability mass of 0.9, performance drops to 28.75 BLEU and 61.19 chrF++. Finally, cycle-consistency re-ranking of the beam search outputs improves performance (33.57 BLEU, 64.86 chrF++) over the one best output.

Table 3 compares the best GPT-2M and GPT-2L results, fine-tuned using the reconstruction term and PENMAN notation. For all scores we test statistical significance with a standard two-tailed student t-test. Our model achieves a large improvement of 1.2 BLEU and 1.3 METEOR scores over the previous state-of-the-art model using GPT-2L and re-scoring. For chrF++, we get different scores from SacreBLEU and the scripts provided by the authors of our baseline systems, achieving comparable results with the former (63.89), and improving over the best score with the latter (65.01) $(P < .01)$.

Table 4 shows human Evaluation results and semantic similarity scores of GPT-2L and GPT-2M compared to (Zhu et al., 2019; Ribeiro et al., 2019; Guo et al., 2019). Our approach produces a large number of high-quality sentences with 41.8%, a significant gain over the previous best system (20.26%). Regarding semantic similarity, prior art methods show relatively close scores, a 0.9 points difference, while GPT-2L Rec. improves 1.6 points over the best of these models. It should be noted that differences with (Zhu et al., 2019) for GPT-2L Rec. are statistically significantly with $P < .05$, while differences for GPT-2M Rec are not significant due to the small sample size.

In Table 5 we show three nontrivial examples, where we compare our system outputs with those of previous work. In the first example, the reference sentence contains a grammatical error. Our system not only generates the correct output, but also corrects the error in the reference. The proposed system can generate fluent long sentences as shown in example 2. The third example shows a sentence where all systems including ours fail to generate a correct text.

4.2 Discussion

Due to the large amounts of data they are trained on, pre-trained transformer language models can be expected to generate fluent and diverse text (See et al., 2019). It should however be highlighted that fine-tuned GPT-2 learns to produce not only fluent but also adequate text, despite using a sequential representation of an AMR graph as input. As shown in the experimental setup, encoding of relations plays as well a fundamental role in AMR-to-text performance, indicating that GPT-2 attains a fine-grained understanding of the underlying semantics to reach state of the art performance.

While a sequence of PENMAN notation to-

	System	Generated text
(1)	**REF:**	the doctors gave her medication and it 's made her much better .
	G2S:	the doctor gives her medications and they make her much better .
	Transf:	doctors give her medications and make her much better .
	Our:	the doctor gave her the medication and made her feel much better.
	Our R.:	the doctor gave her the medication and made her " much better " .
(2)	**REF:**	at the state scientific center of applied microbiology there is every kind of deadly bacteria that was studied for use in the secret biological weapons program of the soviet union .
	G2S:	there are every kind of killing <unk> in the state scientific center of applied microbiology to use themselves for soviet union 's secret biological weapons programs .
	Transf:	there is every kind of bacterium , which is studied in using bacterium for the soviet union secret biological weapons program .
	Our:	every kind of bacterium that was studied was found at the state scientific center of applied microbiology and was used in soviet secret weapons programs for biological weapons of biology .
	Our R.:	every kind of bacterium that has been studied and used in soviet secret programs for biological weapons has been in the state scientific center of applied microbiology .
(3)	**REF:**	among the nations that have not signed the treaty only india and israel would qualify for admission to the nsg under the israeli proposal .
	G2S:	only one of the nations who do not sign the treaty are qualified for their proposal to admit the nsg .
	Transf:	india and israel are only qualified for the nations that do not sign the treaty , but they admitted to the nsg .
	Our:	india and israel are the only countries eligible to admit to the nsg by proposing a treaty .
	Our R.:	only india and israel are eligible to admit to the nsg by proposing a treaty .

Table 5: Output examples from four systems of the LDC2017T10 dataset. REF stands for reference, G2S for (Guo et al., 2019) and Transf. for (Zhu et al., 2019). Our is the top beam output for `GPT-2L` and Our R. is with re-scoring.

kens is far from an optimal encoding of a graph, it is noteworthy how far performance-wise current strong language models can go. Furthermore, It is likely that standard metrics (BLEU, Meteor, chrF++) that rely on a reference text do not properly reflect AMR-to-text quality. An AMR graph corresponds to multiple sentences with the same semantics and these measures are likely biased towards the single available reference. In metrics that are less influenced by the reference text such as human evaluation and semantic similarity, the proposed system shows a larger improvement over the previous systems with close to 50% of the generated sentences considered excellent or good.

Finally it is worth considering that leveraging pre-trained transformers greatly expands the vocabulary available on AMR-to-text systems. A single AMR graph can correspond to multiple sentences with markedly different surface realizations, but manual annotation of AMR is a time consuming task. Approaches like the one proposed may be a simple solution for generation of diverse text data for AMR parser training or other applications were diversity play a role.

5 Conclusions

In this work, we present a language model-based approach for the AMR-to-text generation task. We show that a strong pre-trained transformer language model (`GPT-2`) can be fine-tuned to generate text directly from the PENMAN notation of an AMR graph. Comparison with state-of-the-art models in BLUE, chrF++, METEOR as well as SemSim and human evaluation metrics show that while simple, this approach can outperform existing methods including methods training transformers from scratch. We also show that cycle consistency-based re-scoring using a conventional AMR parser and the Smatch metric can notably improve the results. Future work will focus on incorporating better encoding of the AMR graph into the current system and exploring data augmentation techniques leveraging the proposed approach.

Acknowledgments

We thank the reviewers for their valuable suggestions. We would also like to thank Chunchuan Lyu for his valuable feedback and help.

References

Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. 2019. Synthetic QA corpora generation with roundtrip consistency. In *Proceedings of the 57th Annual Meeting of the 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.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

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.

Deng Cai and Wai Lam. 2020. Graph transformer for graph-to-sequence learning. In *34th AAAI conference on artificial intelligence*.

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 748–752.

Kris Cao and Stephen Clark. 2019. Factorising amr generation through syntax. 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 2157–2163.

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.

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.

Jeffrey Flanigan, Chris Dyer, Noah A Smith, and Jaime Carbonell. 2016. Generation from abstract meaning representation using tree transducers. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pages 731–739.

Jeffrey Flanigan, Sam Thomson, Jaime Carbonell, Chris Dyer, and Noah A Smith. 2014. A discriminative graph-based parser for the abstract meaning representation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1426–1436.

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.

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

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Takaaki Hori, Ramon Astudillo, Tomoki Hayashi, Yu Zhang, Shinji Watanabe, and Jonathan Le Roux. 2019. Cycle-consistency training for end-to-end speech recognition. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6271–6275. IEEE.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

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.

Qingsong Ma, Johnny Wei, Ondřej Bojar, and Yvette Graham. 2019. Results of the wmt19 metrics shared task: Segment-level and strong mt systems pose big challenges. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 62–90.

Tahira Naseem, Abhishek Shah, Hui Wan, Radu Florian, Salim Roukos, and Miguel Ballesteros. 2019. Rewarding Smatch: Transition-based AMR parsing with reinforcement learning.

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

Maja Popović. 2017. chrf++: words helping character n-grams. In *Proceedings of the second conference on machine translation*, pages 612–618.

Nima Pourdamghani, Kevin Knight, and Ulf Hermjakob. 2016. Generating english from abstract meaning representations. In *Proceedings of the 9th international natural language generation conference*, pages 21–25.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. 2018. *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/language-unsupervised/language_understanding_paper. pdf*.

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

Leonardo FR Ribeiro, Claire Gardent, and Iryna Gurevych. 2019. Enhancing amr-to-text generation with dual graph representations. *arXiv preprint arXiv:1909.00352*.

Abigail See, Aneesh Pappu, Rohun Saxena, Akhila Yerukola, and Christopher D Manning. 2019. Do massively pretrained language models make better storytellers? *arXiv preprint arXiv:1909.10705*.

Vu Trong Sinh and Nguyen Le Minh. 2019. A study on self-attention mechanism for amr-to-text generation. In *International Conference on Applications of Natural Language to Information Systems*, pages 321–328. Springer.

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.

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 *Advances in neural information processing systems*, pages 5998–6008.

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.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

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*.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2019. Dialogpt: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536*.

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 5462–5471.