Unsupervised Syntactically Controlled Paraphrase Generation with Abstract Meaning Representations

Kuan-Hao Huang[*†] **Varun Iyer**[*◇] **Anoop Kumar**[‡]
Sriram Venkatapathy[‡] **Kai-Wei Chang**[†‡] **Aram Galstyan**[‡]
[†]University of California, Los Angeles
[◇]Johns Hopkins University, [‡]Amazon Alexa AI
{khhuang, kwchang}@cs.ucla.edu, viyer3@jhu.edu
{anooamzn, vesriram, kaiwec, argalsty}@amazon.com

Abstract

Syntactically controlled paraphrase generation has become an emerging research direction in recent years. Most existing approaches require annotated paraphrase pairs for training and are thus costly to extend to new domains. Unsupervised approaches, on the other hand, do not need paraphrase pairs but suffer from relatively poor performance in terms of syntactic control and quality of generated paraphrases. In this paper, we demonstrate that leveraging Abstract Meaning Representations (AMR) can greatly improve the performance of unsupervised syntactically controlled paraphrase generation. Our proposed model, **AMR**-enhanced **P**araphrase **G**enerator (AMRPG), separately encodes the AMR graph and the constituency parse of the input sentence into two disentangled semantic and syntactic embeddings. A decoder is then learned to reconstruct the input sentence from the semantic and syntactic embeddings. Our experiments show that AMRPG generates more accurate syntactically controlled paraphrases, both quantitatively and qualitatively, compared to the existing unsupervised approaches. We also demonstrate that the paraphrases generated by AMRPG can be used for data augmentation to improve the robustness of NLP models.

1 Introduction

Syntactically controlled paraphrase generation approaches aim to control the format of generated paraphrases by taking into account additional parse specifications as the inputs, as illustrated by Figure 1. It has attracted increasing attention in recent years since it can diversify the generated paraphrases and benefit a wide range of NLP applications (Iyyer et al., 2018; Huang and Chang, 2021; Sun et al., 2021), including task-oriented dialog generation (Gao et al., 2020), creative generation (Tian et al., 2021), and model robustness (Huang and Chang, 2021).

[*]The authors contribute equally.

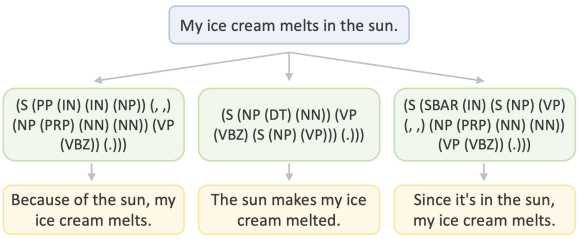

Figure 1: An illustration of syntactically controlled paraphrase generation. Given a source sentence and different parse specifications, the model generates different paraphrases following the parse specifications.

Recent works have shown success in training syntactically controlled paraphrase generators (Iyyer et al., 2018; Chen et al., 2019; Kumar et al., 2020; Sun et al., 2021). Although their models can generate high-quality paraphrases and achieve good syntactic control ability, the training process needs a large amount of supervised data, e.g., parallel paraphrase pairs. Annotating paraphrase pairs is usually expensive because it requires intensive domain knowledge and high-level semantic understanding. Due to the difficulty in collecting parallel data, the ability of supervised approaches are limited, especially when adapting to new domains.

To reduce the annotation demand, unsupervised approaches can train syntactically controlled paraphrase generators without the need for parallel pairs (Zhang et al., 2019; Bao et al., 2019; Huang and Chang, 2021). Most of them achieve syntactic control by learning disentangled embeddings for semantics and syntax separately (Bao et al., 2019; Huang and Chang, 2021). However, without parallel data, it is challenging to learn a good disentanglement and capture semantics well. As we will show later (Section 4.1), unsupervised approaches can generate bad paraphrases by mistakenly swapping object and subject of a sentence.

In this work, we propose to use *Abstract Meaning Representations (AMR) (Banarescu et al., 2013) to learn better disentangled semantic embeddings*

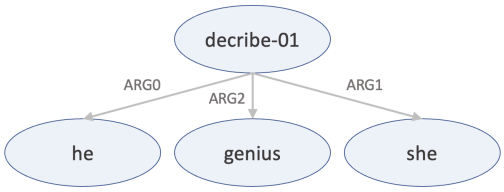

Figure 2: The same AMR graph for a pair of paraphrased sentences "He described her as a genius." and "She was a genius, according to his description."

for unsupervised syntactically controlled paraphrase generation. AMR is a semantic graph structure that covers the abstract meaning of a sentence. As shown in Figure 2, two sentences would have the same (or similar) AMR graph as long as they carry the same abstract meaning, even they are expressed with different syntactic structures. This property makes AMRs a good resource to capture sentence semantics.

Based on this, we design an **AMR**-enhanced **P**araphrase **G**enerator (AMRPG), which separately learns (1) *semantic embeddings* with the AMR garphs extracted from the input sentence and (2) *syntactic embeddings* from the constituency parse of the input sentence. Then, AMRPG trains a decoder to reconstruct the input sentence from the semantic and syntactic embeddings. The reconstruction objective and the design of the disentanglement of semantics and the syntax makes AMRPG learn to generate syntactically controlled paraphrases without using parallel pairs. Our experiments show that AMRPG performs better syntactic control than existing unsupervised approaches. Additionally, we demonstrate that the generated paraphrases of AMRPG can be used for data augmentation to improve the robustness of NLP models.

2 Related Work

Paraphrase generation. Traditional paraphrase generators are usually based on hand-crafted rules (Barzilay and Lee, 2003) or seq2seq models (Cao et al., 2017; Gupta et al., 2018; Fu et al., 2019). To generate diverse paraphrases, different techniques are proposed, including random pattern embeddings (Kumar et al., 2019), latent space perturbation (Roy and Grangier, 2019; Zhang et al., 2019; Cao and Wan, 2020), multi-round generation (Lin and Wan, 2021), reinforcement learning (Liu et al., 2020), prompt-tuning (Chowdhury et al., 2022), order control (Goyal and Durrett, 2020), and syntactic control (Iyyer et al., 2018; Kumar et al., 2020; Huang and Chang, 2021; Sun et al., 2021).

Abstract meaning representation (AMR). Since AMR (Banarescu et al., 2013) captures high-level semantics, it has been applied for various NLP tasks, including summarization (Sachan and Xing, 2016), dialogue modeling (Bai et al., 2021), information extraction (Zhang et al., 2021). Some works also focus on training high-quality AMR parsers with graph encoders (Cai and Lam, 2020), seq2seq models (Konstas et al., 2017; Zhou et al., 2020), and decoder-only models (Bevilacqua et al., 2021).

3 Unsupervised Syntactically Controlled Paraphrase Generation

3.1 Problem Formulation

We follow previous works (Iyyer et al., 2018; Huang and Chang, 2021) and consider constituency parses (without terminals) as the control signals. Given a source sentence s and a target parse p, the goal of the syntactically controlled paraphrase generator is to generate a target sentence t which has similar semantics to the source sentence s and has syntax following the parse p. In the unsupervised setting, the paraphrase generator cannot access any target sentences and target parses but only the source sentences and source parses during training.

3.2 Proposed Method: AMRPG

Motivated by previous approaches (Bao et al., 2019; Huang and Chang, 2021), we design AMRPG to learn separate embeddings for semantics and syntax, as illustrated by Figure 3. Then, AMRPG learns a decoder with the objective to reconstruct the source sentence. The challenge here is how to learn embeddings such that the semantic embedding contains only semantic information while the syntactic embedding contains only syntactic information. We introduce the details as follows.

Semantic embedding. Given a source sentence, we first use a pre-trained AMR parser[1] to get its AMR graph. Next, we use a semantic encoder to encode the AMR graph into the semantic embedding e_{sem}. Specifically, the semantic encoder consists of two parts: a fixed pre-trained AMR encoder (Ribeiro et al., 2021) followed by a learnable Transformer encoder. We additionally perform *node masking* when training the semantic encoder. Specifically, every node in the AMR graph has a

[1]https://github.com/bjascob/amrlib-models

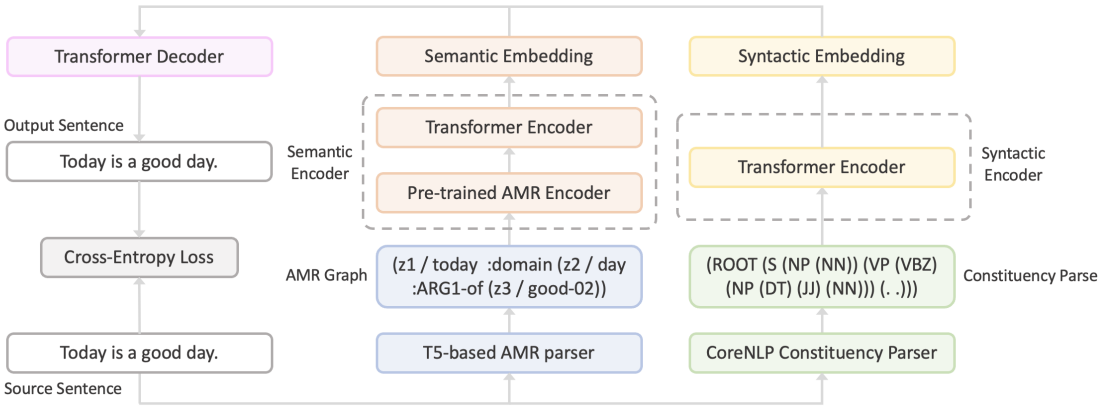

Figure 3: AMRPG's framwork. It separately encodes the AMR graph and the constituency parse of the input sentence into two disentangled semantic and syntactic embeddings. A decoder is then learned to reconstruct the input sentence from the semantic and syntactic embeddings.

probability to be masked out during training. This can improve the robustness of AMRPG.

As mentioned above, two semantically similar sentences would have similar AMR graphs regardless of their syntax. This property encourages AMRPG to capture only semantic information in semantic embeddings. Compared with previous work (Huang and Chang, 2021), which uses bag-of-words to learn the semantic embeddings, using AMR can capture semantics better and lead to better performance, as shown in Section 4.

Syntactic embedding. Given a source sentence, we use the Stanford CoreNLP toolkit (Manning et al., 2014) to get its constituency parse. Then, we remove all the terminals in the parse and learns a Transformer encoder to encode the parse into the syntactic embedding e_{syn}. Since we remove the terminals, the syntactic embedding contains only the syntactic information of the source sentence.

Decoder. We train a Transformer decoder that takes the semantic embedding e_{sem} and the syntactic embedding e_{syn} as the input, and reconstructs the source sentence with a cross-entropy loss. The reconstruction objective makes AMRPG not require parallel paraphrase pairs for training.

Inference. Given a source sentence s and a target parse p, we use the semantic encoder to encode the AMR graph of s into the semantic embedding, use the syntactic encoder to encode p into the syntactic embedding, and use the decoder to generate the target sentence t.

4 Experiments

4.1 Syntactically Controlled Paraphrase Generation

Datasets. We consider ParaNMT (Wieting and Gimpel, 2018) for training and testing. We use *only the source sentences* in ParaNMT to train AMRPG and other unsupervised baselines, and use both the source sentences and target sentences to train supervised baselines. To further test the model's ability to generalize to new domains, we directly use the models trained with ParaNMT to test on Quora (Iyer et al., 2017), MRPC (Dolan et al., 2004), and PAN (Madnani et al., 2012)

Evaluation metrics. Following the previous work (Huang and Chang, 2021), we consider the BLEU score to measure the similarity between the gold target sentences and the predicted target sentences, and consider the *template matching accuracy*[2] (TMA) to evaluate the goodness of syntactic control. More details about the evaluation can be found in Appendix B.2.

Baselines. We consider the following unsupervised models: SIVAE (Zhang et al., 2019), SynPG (Huang and Chang, 2021), AMRPG, and T5-Baseline, which replaces the AMR encoder with a T5-encoder. We also consider SCPN (Iyyer et al., 2018) as the supervised baseline.

Results. Table 1 shows the results of syntactically controlled paraphrase generation. AMRPG performs the best among the unsupervised approaches. Specifically, AMRPG outperforms SynPG, the

[2]Template matching accuracy is defined as the exact matching accuracy of top-2 levels of parse trees.

Model	ParaNMT		Quora		PAN		MRPC	
	TMA	BLEU	TMA	BLEU	TMA	BLEU	TMA	BLEU
Unsupervised Approaches (without using parallel pairs)								
SIVAE (Zhang et al., 2019)	30.0	12.8	48.3	13.1	26.6	11.8	21.5	5.1
SynPG (Huang and Chang, 2021)	71.0	32.2	82.6	33.2	**66.3**	26.4	**74.0**	26.2
T5-Baseline	57.1	22.8	66.1	22.2	55.3	21.0	66.2	18.8
AMRPG	**74.3**	**39.1**	**84.8**	**33.9**	65.6	**31.0**	71.9	**34.8**
Unsupervised Approaches (using target domain source sentences)								
SynPG (Huang and Chang, 2021)	-	-	86.3	44.4	66.4	34.2	**80.7**	44.6
AMRPG	-	-	**86.5**	**45.4**	**67.5**	**37.6**	76.8	**45.9**
Supervised Approaches (using additional parallel pairs in ParaNMT; not compariable to ours)								
SCPN (Iyyer et al., 2018)	83.9	58.3	87.1	41.0	72.3	37.6	80.1	41.8

Table 1: Results of syntactically controlled paraphrase generation. AMRPG performs the best among all unsupervised approaches and can outperform supervised approaches when considering the target domain source sentences.

Input	The dog chased the cat on the street.
Parse template	`(S(NP(DT)(NN))(VP(VBN)(PP))(.))`
Target	The cat was chased by the dog on the street.
SynPG	The dog was chased by the cat on the street.
AMRPG	The cat was chased by a dog in the street.
Input	John will send a gift to Tom when Christmas comes.
Parse template	`(S(SBAR (WHADVP)(S))(,)(NP(NNP))(VP(MD)(VP))(.))`
Target	When Christmas comes, John will send a gift to Tom.
SynPG	When Tom comes, John will send a gift to Christmas.
AMRPG	When Christmas comes, John will send a gift to Tom.

Table 2: Paraphrase examples generated by SynPG and AMRPG. AMRPG captures semantics better and generates higher quality of paraphrases than SynPG.

state-of-the-art unsupervised model, with a large gap in terms of BLEU score. This justifies that using AMR can learn better disentangled embeddings and capture semantics better.

We observe that there is indeed a performance gap between AMRPG and SCPN (supervised baseline). However, since AMRPG is an unsupervised model, it is possible to use the source sentences from the target domains to further fine-tune AMRPG without additional annotation cost. As shown in the table, AMRPG with further fine-tuning can achieve even better performance than SCPN when considering domain adaptation (Quora, MRPC, and PAN). This demonstrates the flexibility and the potential of unsupervised paraphrase models.

Qualitative examples. Table 2 lists some paraphrases generated by SynPG and AMRPG. As we mentioned in Section 3, SynPG uses bag-of-words to learn semantic embeddings and therefore SynPG is easy to get confused about the relations between entities or mistake the subject for the object. In contrast, AMRPG can preserve more semantics.

4.2 Improving Robustness of NLP Models

We demonstrate that the paraphrases generated by AMRPG can improve the robustness of NLP models by data augmentation. Following the setting of previous work (Huang and Chang, 2021), we consider three classification tasks in GLUE (Wang et al., 2019): MRPC, RTE, and SST-2. We compare three baselines: (1) the classifier trained with original training data, (2) the classifier trained with original training data and augmented data generated by SynPG, and (3) the classifier trained with original training data and augmented data generated by AMRPG. Specifically, for every instance in the original training data, we generate four paraphrases as the augmented examples by considering four common syntactic templates. More details can be found in Appendix C.1.

Table 3 shows the clean accuracy and the broken rate (the percentage of examples being attacked) after attacked by the syntactically adversarial examples[3] generated with SCPN (Iyyer et al., 2018). Although the classifiers trained with data augmen-

[3]Appendix C.2 has more details of the adversarial attack.

Model	MRPC		RTE		SST-2	
	Acc.	Brok.	Acc.	Brok.	Acc.	Brok.
Base	**83.3**	52.9	**62.1**	58.1	**92.2**	38.8
+ SynPG	80.6	42.2	61.7	40.3	91.5	38.5
+ AMRPG	80.6	**38.3**	58.8	**39.3**	91.6	**36.7**

Table 3: Augmenting paraphrases generated by AMRPG improves the robustness of NLP models. Acc denotes the clean accuracy (the higher is the better). Brok denotes the percentage of examples being successfully attacked (the lower is the better).

tation have slightly worse clean accuracy, they have significantly lower broken rates, which implies that data augmentation improves the model robustness. Also, data augmentation with AMRPG performs better than data augmentation with SynPG in terms of the broken rate. We attribute this to the better quality of paraphrase generation of AMRPG.

5 Conclusion

We propose AMRPG that utilizes AMR to learn a better disentanglement of semantics and syntax without using any parallel data. This enables AMRPG to captures semantics better and generate more accurate syntactically controlled paraphrases than existing unsupervised approaches. We also demonstrate that how to apply AMRPG to improve the robustness of NLP models.

Limitations

Our goal is to demonstrate the potential of AMR for syntactically controlled paraphrase generation. The current experimental setting follows previous works (Iyyer et al., 2018; Huang and Chang, 2021), which considers the *full* constituency parses as the control signals. In real applications, getting full constituency parses before the paraphrase generation process might take additional efforts. One potential solution is to consider relatively noisy or simplified parse specifications (Sun et al., 2021). In addition, some parse specifications can be inappropriate for certain source sentences (e.g., the source sentence is long but the target parse is short). How to score and reject some of the given parse specifications is still an open research question. Finally, although training AMRPG does not require any parallel paraphrase pairs, it does require a pretrained AMR parser, which can be a potential cost for training AMRPG.

Broader Impacts

Our proposed method focuses on improving syntactically controlled paraphrase generation. It is intended to be used to improve the robustness of models and facilitate language generation for applications with positive social impacts. All the experiments are conducted on open benchmark datasets. However, it is known that the models trained with a large text corpus could capture the bias reflecting the training data. It is possible for our model to potentially generate offensive or biased content learned from the data. We suggest to carefully examining the potential bias before deploying models in any real-world applications.

References

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 (ACL/IJCNLP)*.

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 (LAW-ID@ACL)*.

Yu Bao, Hao Zhou, Shujian Huang, Lei Li, Lili Mou, Olga Vechtomova, Xin-Yu Dai, and Jiajun Chen. 2019. Generating sentences from disentangled syntactic and semantic spaces. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*.

Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021. One SPRING to rule them both: Symmetric AMR semantic parsing and generation without a complex pipeline. In *Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*.

Deng Cai and Wai Lam. 2020. AMR parsing via graph-sequence iterative inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Yue Cao and Xiaojun Wan. 2020. Divgan: Towards diverse paraphrase generation via diversified generative adversarial network. In *Findings of the Association for Computational Linguistics: (EMNLP-Findings)*.

Ziqiang Cao, Chuwei Luo, Wenjie Li, and Sujian Li. 2017. Joint copying and restricted generation for paraphrase. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI)*.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. 2019. Controllable paraphrase generation with a syntactic exemplar. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*.

Jishnu Ray Chowdhury, Yong Zhuang, and Shuyi Wang. 2022. Novelty controlled paraphrase generation with retrieval augmented conditional prompt tuning. In *Proceedings of the Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI)*.

Bill Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *20th International Conference on Computational Linguistics (COLING)*.

Yao Fu, Yansong Feng, and John P. Cunningham. 2019. Paraphrase generation with latent bag of words. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019 (NeurIPS)*.

Silin Gao, Yichi Zhang, Zhijian Ou, and Zhou Yu. 2020. Paraphrase augmented task-oriented dialog generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Tanya Goyal and Greg Durrett. 2020. Neural syntactic preordering for controlled paraphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A deep generative framework for paraphrase generation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*.

Kuan-Hao Huang and Kai-Wei Chang. 2021. Generating syntactically controlled paraphrases without using annotated parallel pairs. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Shankar Iyer, Nikhil Dandekar, and Korn el Csernai. 2017. First quora dataset release: Question pairs. *data.quora.com*.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

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 (ACL)*.

Ashutosh Kumar, Kabir Ahuja, Raghuram Vadapalli, and Partha P. Talukdar. 2020. Syntax-guided controlled generation of paraphrases. *Transactions of the Association for Computational Linguistics*, 8:330–345.

Ashutosh Kumar, Satwik Bhattamishra, Manik Bhandari, and Partha P. Talukdar. 2019. Submodular optimization-based diverse paraphrasing and its effectiveness in data augmentation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Zhe Lin and Xiaojun Wan. 2021. Pushing paraphrase away from original sentence: A multi-round paraphrase generation approach. In *Findings of the Association for Computational Linguistics (ACL/IJCNLP-Findings)*.

Mingtong Liu, Erguang Yang, Deyi Xiong, Yujie Zhang, Yao Meng, Changjian Hu, Jinan Xu, and Yufeng Chen. 2020. A learning-exploring method to generate diverse paraphrases with multi-objective deep reinforcement learning. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING)*.

Nitin Madnani, Joel R. Tetreault, and Martin Chodorow. 2012. Re-examining machine translation metrics for paraphrase identification. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, System Demonstrations*.

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

Aurko Roy and David Grangier. 2019. Unsupervised paraphrasing without translation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*.

Mrinmaya Sachan and Eric P. Xing. 2016. Machine comprehension using rich semantic representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Jiao Sun, Xuezhe Ma, and Nanyun Peng. 2021. AESOP: paraphrase generation with adaptive syntactic control. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yufei Tian, Arvind Krishna Sridhar, and Nanyun Peng. 2021. Hypogen: Hyperbole generation with commonsense and counterfactual knowledge. In *Findings of the Association for Computational Linguistics: (EMNLP-Findings)*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations (ICLR)*.

John Wieting and Kevin Gimpel. 2018. Paranmt-50m: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Xinyuan Zhang, Yi Yang, Siyang Yuan, Dinghan Shen, and Lawrence Carin. 2019. Syntax-infused variational autoencoder for text generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*.

Zixuan Zhang, Nikolaus Nova Parulian, Heng Ji, Ahmed Elsayed, Skatje Myers, and Martha Palmer. 2021. Fine-grained information extraction from biomedical literature based on knowledge-enriched abstract meaning representation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL/IJCNLP)*.

Qiji Zhou, Yue Zhang, Donghong Ji, and Hao Tang. 2020. AMR parsing with latent structural information. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

A Implementation Details

We use around 20 millions of examples[4] in ParaNMT (Wieting and Gimpel, 2018) to train AMRPG and all baselines. The semantic encoder and the syntactic decoder are trained from scratch, with the default architecture and the default parameters of `torch.nn.Transformer`. The max length for input sentences, the linearized constituency parses, and the linearized AMR graph are set to 40, 160, and 250, respectively. The word dropout rate is 0.4 while the node masking rate is 0.6. We consider Adam optimizer with the learning rate being 10^{-4} and the weight decay being 10^{-5}. The total number of epochs is set to 10. When generating the outputs, we use random sampling with temperature being 0.5. The model is trained with 4 NVIDIA V100 GPUs with 16 GB memory each. It takes around 7 days to finish the training process.

B Experimental Settings of Syntactically Controlled Paraphrase Generation

B.1 Datasets

Following previous work (Huang and Chang, 2021), our test data is: (1) 6,400 examples of ParaNMT (Wieting and Gimpel, 2018), (2) 6,400 examples of Quora (Iyer et al., 2017), (3) 2,048 examples of PAN (Madnani et al., 2012), and (4) 1,920 examples of MRPC (Dolan et al., 2004).

B.2 Evaluation

Following previous work (Huang and Chang, 2021), we consider paraphrase pairs to evaluate the performance. Given a paraphrase pairs (s_1, s_2), we use the Standford CoreNLP constituency parser (Manning et al., 2014) to get their parses (p_1, p_2). The input of all baselines would be (s_1, p_2) and the ground truth would be s_2.

Assuming the generated paraphrase is g, We use BLEU score to measure the similarity between the generated paraphrase g and the ground truth s_2. We also calculate the template matching accuracy (TMA) by computing the exact matching accuracy of the top-2 levels of p_g and p_2 (p_g is the constituency parse of g).

[4] https://github.com/uclanlp/synpg

C Experimental Settings of Model Robustness

C.1 Training Details

We use the pre-trained SynPG parse generator to generate the full parse for each instance with the following parse templates: "`(S(NP)(VP)(.))`", "`(S(VP)(.))`", "`(NP(NP)(.))`", and "`(FRAG(SBAR)(.))`". Then, we use the generated full parses as the parse specifications to generate paraphrases for data augmentation. When training classifiers with data augmentation, the original instances have four times of weights as the augmented instances when computing the loss. We use the scripts from Huggingface[5] with default values to train the classifiers.

C.2 Generating Adversarial Examples

We use the official script[6] of SCPN (Iyyer et al., 2018) to generate syntactically adversarial examples. Specifically, we consider the first five parse templates for RTE and SST-2 and first three parse templates for MRPC to generate the adversarial examples. As long as one of the adversarial examples makes the classifier change the prediction, we count it as a successful attack on this instance.

[5] https://github.com/huggingface/transformers/blob/main/examples/pytorch/text-classification/run_glue.py

[6] https://github.com/miyyer/scpn