Controlling keywords and their positions in text generation

Yuichi Sasazawa, Terufumi Morishita, Hiroaki Ozaki,
Osamu Imaichi, Yasuhiro Sogawa
Hitachi, Ltd. Research and Development Group
{yuichi.sasazawa.bj, terufumi.morishita.wp, hiroaki.ozaki.yu,
osamu.imaichi.xc, yasuhiro.sogawa.tp}@hitachi.com

Abstract

One of the challenges in text generation is to control text generation as intended by the user. Previous studies proposed specifying the keywords that should be included in the generated text. However, this approach is insufficient to generate text that reflect the user's intent. For example, placing an important keyword at the beginning of the text would help attract the reader's attention; however, existing methods do not enable such flexible control. In this paper, we tackle a novel task of controlling not only keywords but also the position of each keyword in the text generation. To this end, we propose a task-independent method that uses special tokens to control the relative position of keywords. Experimental results on summarization and story generation tasks show that the proposed method can control keywords and their positions. The experimental results also demonstrate that controlling the keyword positions can generate summary texts that are closer to the user's intent than baseline.

1 Introduction

One of the challenges in text generation is to generate text that is consistent with the user's intent. Many methods for specifying the keywords that should be included in the generated text to reflect the user's intent have been proposed. As for summarization, by providing the model with keywords that should be included in the summary, it is possible to generate summaries that focus on specific parts of the document (Fan et al., 2018; He et al., 2022; Dou et al., 2021). As for story generation, keywords are used to control the narrative storyline (Jain et al., 2017; Fan et al., 2019; Yao et al., 2019). As for other tasks, such as e-commerce generation, review generation, and question generation, keywords are also used to control text generation (Chan et al., 2019; Shao et al., 2021; Ni and McAuley, 2018; Chan et al., 2021; Zhang and Zhu, 2021). In addition, more-advanced methods that specify the order of keywords to be included in the generated text to control the rough storyline have been proposed (Su et al., 2021; Shao et al., 2021).

The above-described methods, however, cannot generate texts that reflect more fine-grained intentions. Specifically, the user may want to reflect the intended importance of each keyword in the generated text. An effective way to reflect the intended importance of each keyword is to adjust the position of keywords within the text. For example, important keywords such as topic words and eye-catching words can be placed at the beginning of the text to attract the reader's attention, while the keywords for supplementary information can be placed in the middle or later in the text. By controlling the specific position of each keyword according to its importance, it is possible to generate appropriate text for each situation. That is, controlling the specific position of keywords in the generated text is a challenge in terms of reflecting more-specific user intentions and generating texts that attract readers. However, as far as we know, no previous work has tackled this challenge.

In this paper, we tackle a novel task of controlling keywords and the position of each keyword in text generation. Inspired by previous work that controlled text attributes by using special tokens (Iwama and Kano, 2019; Lakew et al., 2019; Martin et al., 2020), we propose a task-independent method that uses special tokens to control text generation. Specifically, the position of the keyword is specified by providing the model with a special token that represents the target relative position of the keyword (0-10%, 10-20%, etc.) and length of target text (20-24 words, 25-29 words, etc.). We use relative positions (rather than absolute positions) because it is more practical to specify relative positions such as "at the beginning," "in the middle," or "at the end" of the target text. Moreover, length of

Proceedings of the 16th International Natural Language Generation Conference, pages 407–413
September 11–15, 2023. ©2023 Association for Computational Linguistics

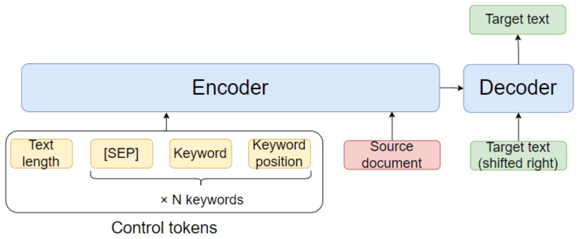

Figure 1: Overview of proposed method. The model is provided with control tokens: keywords in the target text, positions of each keyword, and target-text length to control text generation.

the target text is controlled because text length is considered to be one of the important factors that users want to control when considering where to place keywords. During training of the model, the model is provided with control tokens, including keywords randomly extracted from the target text, the positions of each keyword, and the length of the target text. The model is trained with cross-entropy loss in the same manner as conventional text generation; as a result, the model can learn the correspondence between the input control tokens and the target text.

The proposed "task-independent text-generation-control method" ("proposed method" hereafter) was comprehensively evaluated by applying it to summarization and story-generation tasks. The results of the evaluation show that the proposed method can control keywords and their positions in both tasks (Section 3.2). They also show that the proposed method can generate summary texts that are more similar to the gold summary than the baseline, indicating that text closer to the user's intent can be generated (Section 3.3). Case studies show that a model specifying keyword position can reflect the user's fine-grained intention (Section 3.4).

2 Method

2.1 Models

A BART model (Lewis et al., 2020) is used for the summarization task, and a GPT model (Radford et al., 2018) is used for the story-generation task. When the BART model is used, the source document is combined with the control tokens (i.e., keywords in the text to be generated, positions of each keyword, and length of the text to be generated) and given to the encoder as shown in Figure 1. When the GPT model is used, the control tokens are given to the decoder. As with regular text generation using BART and GPT models, the model is trained to maximize the conditional probabilities $p(y_i|y_{<i}, x)$ by using cross-entropy loss, where y denotes the target text and x denotes the input to the model, including the control tokens and the source document used in summarization task.

2.2 Control tokens

Inspired by existing methods that control text attributes by special tokens (Iwama and Kano, 2019; Lakew et al., 2019; Martin et al., 2020), the model is provided with the position of each keyword and text length as special tokens. For example, if the keyword phrase "two dogs" is located in the first 20-30% of the text and text length is in the range of 50-54 words, "`[LENGTH50]` `[SEP]` two dogs `[POSITION20]`" will be given to the model as the control token. Here, `[LENGTH50]` and `[POSITION20]` are new tokens added to the vocabulary, and the corresponding word embedding is initialized randomly.

Note that control tokens that represent the oracle information of the target text are given to the model during both training and inference. This setting is appropriate because we aim to generate the intended text by providing additional information to the model. It is also possible that the model automatically determines keywords and their positions (i.e., control tokens are not given to the model), but that approach is left for future work.

Control tokens are extracted from the target text as follows. More details are given in Appendix A.3.

Keywords Keywords in this paper are not limited to important words in the target text; they can also be any phrase consisting of one to three consecutive words in the target text. For example, from the target text "Marcia was looking forward to trying hang gliding.", the phrases "Marsha", "was", "looking forward", "to trying", and "trying hang gliding" are first extracted as keyword candidates. However, frequent words with little meaning such as "was" and "to trying" are excluded from the keyword candidates, because they are considered unlikely to be given as keywords by the user. During training, a random number of phrases from the keyword candidates are given to the model as keywords. During inference, the user has the flexibility to give arbitrary keywords to the model. However, for the experiments conducted in this paper, we follow the same approach as during training: the keywords are randomly selected from the keyword

Control	CNN/DM		XSum		ROCStories	
	Include	Pos	Include	Pos	Include	Pos
One keyword						
w/o Control	27.5	8.3	23.4	9.4	0.5	0.1
Keyword	71.3	18.7	86.4	28.7	53.0	14.3
+Len	72.7	20.4	85.8	30.8	50.9	13.5
+Pos	80.8	47.0	**92.1**	63.0	57.2	27.4
+Pos+Len	**85.8**	**48.8**	91.8	**64.1**	**58.8**	**29.1**
Two keywords						
Keyword	52.4	5.1	74.1	14.1	22.9	1.6
+Pos+Len	**75.9**	**28.6**	**85.9**	**46.4**	**31.1**	**7.9**
Three keywords						
Keyword	39.1	2.0	62.5	9.8	9.2	0.3
+Pos+Len	**70.6**	**21.8**	**80.5**	**37.3**	**15.5**	**2.2**

Table 1: Evaluation of the control of keywords and their positions in terms of (i) accuracy of generating text **Includ**ing all of the target keywords and (ii) accuracy of generating text in which all of the target keywords are placed in each target **Pos**ition.

candidates and given to the model.

Keyword Position The position of each keyword is expressed as a relative position. Specifically, the absolute position of the target keyword when counted from the beginning of the text is divided by the number of words in the text, and the quantized position in units of 10% are given to the model.

Text Length Number of words in the target text (quantized in 5 word units) is given to the model.

3 Experiment

3.1 Experiment setting

The proposed method was comprehensively evaluated by applying it to well-established summarization and story-generation tasks. These two tasks have different characteristics. As for summarization, the model extracts information from a source document and compresses it into a short text by using the given control tokens. As for story generation, the model generates text solely on the basis of the given control tokens. For summarization, we used the CNN/DailyMail (Hermann et al., 2015) and the XSum (Narayan et al., 2018) dataset and the $\text{BART}_{\text{LARGE}}$ model (400M parameters) (Lewis et al., 2020). For story generation, we used the ROCStories (Mostafazadeh et al., 2016) dataset and the GPT2 model (120M parameters) (Radford et al., 2018).

We extract candidate keywords from a target text by using the method described in Section 2.2. During training, no more than three keywords were randomly selected from the keyword candidates for each epoch and given to the model. During inference, one to three keywords randomly selected were given to the model in the experiment of Table 1, and one keyword randomly selected was given to the model in the experiment of Table 2 and Table 3.

In all experiments, training and inference were performed three times, and the mean score was reported. See Appendix A for more details on the experimental setup.

3.2 Evaluation of keyword-position control

Whether the given keywords are placed at given positions was evaluated first in terms of (i) the accuracy of generating text including all target keywords and (ii) the accuracy of generating text in which all target keywords are placed in each target position. As shown in Table 1, the proposed method using special tokens (+Pos and +Pos+Len) can generate text that includes the target keyword at the target position. Providing text-length information along with position information (+Pos+Len) improves the accuracy of keyword-position control, particularly in datasets with long text lengths (CNN/DM and ROCStories). In other words, combining relative position and length information enables the model to place the keywords in appropriate positions. The accuracy of the keyword inclusion is also improved when the keyword position is given. We suspect that the model was informed in advance of where the keywords should be placed; as a result, preventing the model from forgetting to place keywords in the text. It is clear that control accuracy is much lower in the case of story generation compared to summarization. This finding may be because the model is not given the source document and generates text from condition tokens only, so the model is more likely to generate the inappropriate context for keyword inclusion.

A more detailed evaluation is given in Table 2. For each target relative position of the keyword, the keyword position in the text was classified as (i) located in the target position, (ii) located at a positional deviation within 10%, (iii) located at a positional deviation greater than 10%, or (iv) not included in the text. It is clear from the results in the table that at all target positions, the accuracy of the keyword-position control is improved compared with that achieved using keyword-only

Keyword position in the generated summary	Target-keyword position (relative position)									
	0-10%	10-20%	20-30%	30-40%	40-50%	50-60%	60-70%	70-80%	80-90%	90-100%
Keyword only Control										
Correct position	52.6	23.8	14.5	9.5	9.5	9.1	8.7	11.8	12.7	15.6
Keyword + Position + Length Control										
Correct position	**84.0**	**57.9**	**49.1**	**41.4**	**36.0**	**36.2**	**33.7**	**36.0**	**46.2**	**47.9**
Within 10% diff	8.1	27.5	31.9	34.4	36.1	34.1	35.5	34.3	23.3	8.9
Over 10% diff	3.2	5.3	8.3	12.8	15.1	15.5	15.1	11.4	6.7	10.9
Not included	4.7	9.4	10.7	11.4	12.8	14.1	15.7	18.4	23.7	32.4

Table 2: Detailed evaluation of the control of the keyword and its position in the CNN/DM dataset. For each target relative position of the keyword, the keyword position in the text was classified as (i) located in the target position (**Correct position**), (ii) located at a positional deviation within 10% (**Within 10% diff**), (iii) located at a positional deviation greater than 10% (**Over 10% diff**), or (iv) not included in the text (**Not included**).

Control	CNN/DM			XSum		
	R1	R2	RL	R1	R2	RL
w/o Control	43.6	20.6	40.5	44.3	21.1	36.5
Keyword	44.4	21.4	41.3	45.9	22.7	38.4
+Len	45.7	22.1	42.5	47.0	23.5	39.3
+Pos	44.9	21.9	41.8	46.7	23.6	40.2
+Pos+Len	**46.4**	**22.8**	**43.2**	**47.8**	**24.5**	**41.2**

Table 3: Summarization evaluation by ROUGE score. To reduce the effect on the ROUGE score due to giving target keywords, target keywords are excluded from both the target and generated summaries.

control, and that finding suggests the effectiveness of the proposed method. The results also show a high success rate of keyword inclusion and positional control near the beginning of the text, and a low success rate in the middle and at the end of the text. This may be because the closer to the end of the text, the more difficult it becomes for the model to generate text that contains the specified keywords while maintaining consistency with the context provided by the preceding words.

3.3 Evaluation of summary-content control

We show that controlling the text makes it easier for the user to generate the intended text in summarization. The results of the evaluation of summary-content control in summarization by ROUGE score (Lin, 2004) are shown in Table 3. Note that to reduce the effect on the ROUGE score due to giving target keywords, target keywords are excluded from both the target and generated summaries. It is clear from the results in the table that the score is improved by controlling keyword positions and text length, and that finding indicates that such control makes it easier to generate text that is close to the user's intended content.

3.4 Case study

To better understand how the proposed model behaved, representative examples of generated texts are shown in Table 4 and Table 5 [1]. In these examples, the keywords and their positions were controlled, although in some examples, the position of the keyword deviates slightly from the target position. It is clear from the table that by assigning different positions for the keywords, it was possible to generate several valid texts with different characteristics. For example, in the example in Table 4, placing the keyword "true miracle dog" at the 0% position generates a text that draws the reader's attention with an eye-catching keyword at the beginning of the text. In contrast, placing that keyword at the 90% position generates a narrative-style text that describes events in chronological order. It is also clear that even when multiple keywords are given, the order of the keywords can be adjusted by controlling the position of each keyword.

We also show some cases in which the proposed model produced errors. When a keyword position near the end of the text is specified, the instruction is often ignored, and the keyword is placed in a completely different position or not included in the text. As can be seen from the results in Table 2, the model tends to be poor at placing keywords at the back of the text.

When comparing the generated text of the summarization task with that of the story generation task, we observed that each of the specified keywords is usually used only once in the generated text of the summarization task, while each of the specified keywords is sometimes used multiple times in the generated text of the story generation

[1] A source document of summarization, gold texts, and additional examples of generated texts are given in Appendix D.

Keyword & Position	Generated text
true miracle dog (0%)	"She's a **true miracle dog** and she deserves a good life," foster mother says. Theia was apparently hit by a car and buried in a field. Four days later, she was found emaciated and dirt-covered by a farm worker. A fundraising page has raised more than $10,000.
true miracle dog (40%)	A stray dog apparently hit by a car and buried in a field survives. The dog, named Theia, is "a **true miracle dog**," her foster owner says. She suffered a dislocated jaw, leg injuries and a caved-in sinus cavity. A fundraising page has raised more than $10,000.
true miracle dog (90%)	Stray dog apparently hit by a car and buried in a field survives. Four days later, she is found emaciated and dirt-covered on a farm. She has a dislocated jaw, leg injuries and a caved-in sinus cavity. A good Samaritan has set up a fundraising page for "the **true miracle dog**"
hit by (40%) bully breed mix (60%)	A stray dog in Washington State apparently died after being **hit by** a car. Four days later, she was found emaciated and dirt-covered in a field. Theia, a **bully breed mix**, has a dislocated jaw, leg injuries and a caved-in sinus cavity.
hit by (10%) bully breed mix (0%)	Theia, a **bully breed mix**, was apparently **hit by** a car and buried in a field. Four days later, she was found emaciated and dirt-covered by a farm worker. She suffered a dislocated jaw, leg injuries and a caved-in sinus cavity. A fundraising page has raised more than $10,000.
Keyword & Position	Generated text (Error cases)
buried in (80%)	Stray dog apparently hit by a car, apparently killed and **buried in** a field. Four days later, the dog manages to stagger to a nearby farm. She was found by a farm worker, who took her to a vet for help. Theia is now receiving treatment.
whacked (70%)	A stray dog in Washington State apparently died after being hit by a car. Four days later, she was found emaciated and dirt-covered by a farm worker. She suffered a dislocated jaw, leg injuries and a caved-in sinus cavity. A fundraising page has raised more than $10,000.

Table 4: Examples of generated texts from the CNN/DM dataset

Keyword & Position	Generated text
dealership (0%)	I went to the **dealership** to look at a car. I was very nervous about buying a car. I asked the salesman if I could try out a certain car. The salesman told me that I could try out a blue car. I drove the blue car home and loved it so much, I bought it.
dealership (30%)	I went to the **dealership** to buy a new car. I was very nervous about the car and the price. I went to the **dealership** and looked at the price. I decided to buy the car and I was very happy with it. I am glad I went to the **dealership** because it was a great deal.
dealership (80%)	I went to the **dealership** to look at a car. I was very nervous because I didn't know what I wanted. I asked the salesman if I could pick out a car. The salesman told me that I could get a new car if I paid $40,000. I drove to the **dealership** and bought a new car.
drive to work (50%) saved enough (70%)	I was driving to work one day when I saw a car in the road. I pulled over and asked if I could **drive to work**. The driver told me that he had **saved enough** money to buy a new car. I drove to work and paid him back. I drove to work and paid him back and he was very happy.
drive to work (20%) saved enough (0%)	I **saved enough** money to buy a new car. I went to the car dealership to test drive my new car. I drove the car for a few hours before I left. When I got home, I realized I had forgotten my wallet. I had to **drive to work** to get my wallet back, but I was happy.
Keyword & Position	Generated text (Error cases)
began saving money (90%)	I was in a hurry to get to work. I had to hurry because I didn't have my car keys. I looked everywhere for my keys, but couldn't find them. I finally found them under my coat, and I was relieved. I was able to grab my keys and walk to work without losing my car keys.
local (0%) Bob (40%) enough (60%)	**Bob** was a **local** handyman. He was hired to fix up a leaky roof on his home. **Bob** was very handy and did a good job at it. Unfortunately, the roof was too deep and the water would not come out. **Bob** had to call a **local** handyman to fix the leaky roof.

Table 5: Examples of generated texts from the ROCStories dataset

task. This may be because the story generation task requires the model to generate text content conditionally only on the specified keywords, causing the model to become overly dependent on them.

4 Conclusion

A method for controlling keywords and the position of each keyword in generated text is proposed and evaluated experimentally by applying it to two tasks: summarization and story generation. The results of the evaluation show that the proposed method, which uses special tokens, can control the keyword positions in both tasks. They also show that the method can generate summary texts that are more similar to the gold summary than the baseline, and that finding indicates that text closer to the user's intent can be generated.

Supplementary Materials Availability Statement

Source code

- The source code is available at Github[2].

Dataset

- The CNN/DM dataset is available at Github[3].
- The XSum dataset is available at HuggingFace[4].
- The ROCStories dataset is available at here[5].

Acknowledgments

We thank Atsuki Yamaguchi, Gaku Morio, and Yuta Koreeda for their support.

[2] https://github.com/ckdjrkffz/controlling_keyword_position
[3] https://github.com/icml-2020-nlp/semsim/tree/master/datasets
[4] https://huggingface.co/datasets/xsum
[5] https://cs.rochester.edu/nlp/rocstories/

References

Ying-Hong Chan, Ho-Lam Chung, and Yao-Chung Fan. 2021. Improving controllability of educational question generation by keyword provision. *CoRR*, abs/2112.01012.

Zhangming Chan, Xiuying Chen, Yongliang Wang, Juntao Li, Zhiqiang Zhang, Kun Gai, Dongyan Zhao, and Rui Yan. 2019. Stick to the facts: Learning towards a fidelity-oriented E-commerce product description 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 4959–4968, Hong Kong, China. Association for Computational Linguistics.

Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, and Graham Neubig. 2021. GSum: A general framework for guided neural abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4830–4842, Online. Association for Computational Linguistics.

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 45–54, Melbourne, Australia. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2019. Strategies for structuring story generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2650–2660, Florence, Italy. Association for Computational Linguistics.

Junxian He, Wojciech Kryscinski, Bryan McCann, Nazneen Rajani, and Caiming Xiong. 2022. CTRLsum: Towards generic controllable text summarization. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5879–5915, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Kango Iwama and Yoshinobu Kano. 2019. Multiple news headlines generation using page metadata. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 101–105, Tokyo, Japan. Association for Computational Linguistics.

Parag Jain, Priyanka Agrawal, Abhijit Mishra, Mohak Sukhwani, Anirban Laha, and Karthik Sankaranarayanan. 2017. Story generation from sequence of independent short descriptions. *CoRR*, abs/1707.05501.

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*.

Surafel Melaku Lakew, Mattia Di Gangi, and Marcello Federico. 2019. Controlling the output length of neural machine translation. In *Proceedings of the 16th International Conference on Spoken Language Translation*, Hong Kong. Association for Computational Linguistics.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1173–1182, Brussels, Belgium. 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.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Louis Martin, Éric de la Clergerie, Benoît Sagot, and Antoine Bordes. 2020. Controllable sentence simplification. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4689–4698, Marseille, France. European Language Resources Association.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Jianmo Ni and Julian McAuley. 2018. Personalized review generation by expanding phrases and attending on aspect-aware representations. In *Proceedings* *of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 706–711, Melbourne, Australia. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Huajie Shao, Jun Wang, Haohong Lin, Xuezhou Zhang, Aston Zhang, Heng Ji, and Tarek Abdelzaher. 2021. Controllable and diverse text generation in e-commerce. In *Proceedings of the Web Conference 2021*, WWW '21, page 2392–2401, New York, NY, USA. Association for Computing Machinery.

Yixuan Su, David Vandyke, Sihui Wang, Yimai Fang, and Nigel Collier. 2021. Plan-then-generate: Controlled data-to-text generation via planning. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 895–909, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826.

Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19. AAAI Press.

Wonjin Yoon, Yoon Sun Yeo, Minbyul Jeong, Bong-Jun Yi, and Jaewoo Kang. 2020. Learning by semantic similarity makes abstractive summarization better. *CoRR*, abs/2002.07767.

Zhiling Zhang and Kenny Zhu. 2021. Diverse and specific clarification question generation with keywords. In *Proceedings of the Web Conference 2021*, WWW '21, page 3501–3511, New York, NY, USA. Association for Computing Machinery.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, page 1097–1100, New York, NY, USA. Association for Computing Machinery.