Screening through a broad pool: Towards better diversity for lexically constrained text generation

Changsen Yuan [a], Heyan Huang [a*], Yixin Cao [b], Qianwen Cao [c]

a Beijing Institute of Technology, China

b Singapore Management University, Singapore

c China University of Petroleum, China

Published in Information Processing & Management (2024) 61 (2), 103602. DOI: 10.1016/j.ipm.2023.103602

Abstract

Lexically constrained text generation (CTG) is to generate text that contains given constrained keywords. However, the text diversity of existing models is still unsatisfactory. In this paper, we propose a lightweight dynamic refinement strategy that aims at increasing the randomness of inference to improve generation richness and diversity while maintaining a high level of fluidity and integrity. Our basic idea is to enlarge the number and length of candidate sentences in each iteration, and choose the best for subsequent refinement. On the one hand, different from previous works, which carefully insert one token between two words per action, we insert an uncertain number of tokens following a well-designed distribution. To ensure high-quality decoding, the insertion number increases as more words are generated. On the other hand, we randomly mask an increasing number of generated words to force Pre-trained Language Models (PLMs) to examine the whole sentence via reconstruction. We have conducted extensive experiments and designed four dimensions for human evaluation. Compared with important baseline (CBART (He, 2021)), our method improves the 1.3% (B-2), 0.1% (B-4), 0.016 (N-2), 0.016 (N-4), 5.7% (M), 1.9% (SB-4), 0.6% (D-2), 0.5% (D-4) on One-Billion-Word dataset (Chelba et al., 2014) and 1.6% (B-2), 0.1% (B-4), 0.121 (N-2), 0.120 (N-4), 0.0% (M), 6.7% (SB-4), 2.7% (D-2), 3.8% (D-4) on Yelp dataset (Cho et al., 2018). The results demonstrate that our method is more diverse and plausible.

Keywords: Constrained text generation, Pre-trained language models, Randomly insert, Randomly mask, Text diversity

1. Introduction

Lexically constrained text generation (CTG) is the task of generating sentences based on constrained keywords. As shown in Fig. 1, given several keywords faces, allegedly, agents, and questioned, the model prediction is a fluent and plausible sentence He faces federal charges for allegedly telling FBI agents who questioned him, which must include all keywords. Lexically CTG targets controlled text generation and has a wide range of applications, such as story generation (Fan et al., 2018, Fang et al., 2021), advertisements generation (Duan et al., 2021, Hughes et al., 2019), and counterfactual reasoning (Qin, Bosselut, Holtzman, Bhagavatula, Clark, & Choi, 2019).

Compared with other text generation tasks, the outputs of a lexically CTG model are more flexible and diverse — there are various texts including the same set of keywords but with different meanings. Therefore, although one can achieve satisfactory performance by searching in human written corpus (Li, Su, Cai, Wang, & Liu, 2022), generation-based methods (Zhou, Gao, Li, & Shum, 2020) are preferable. The basic idea is to add tokens progressive Iteration between keywords. To make the generated text

Keywords: faces, allegedly, agents, questioned	
Iteration 1: **He** faces **federal** allegedly **telling** agents **who** questioned **him** **Iteration 2:** He faces federal **charges** allegedly telling **FBI** agents who questioned him **.** **Iteration 3:** He faces federal charges **for** allegedly telling FBI agents who questioned him .	
Prediction	**Human Reference**
He **faces** federal charges for **allegedly** telling FBI **agents** who **questioned** him .	Siddiqui **faces** an attempted murder charge for **allegedly** trying to shoot **agents** while she was being **questioned** .

Fig. 1. An example of text generation from the One-Billion-Word (Chelba et al., 2014). **Bold words** are generated words in each iteration.

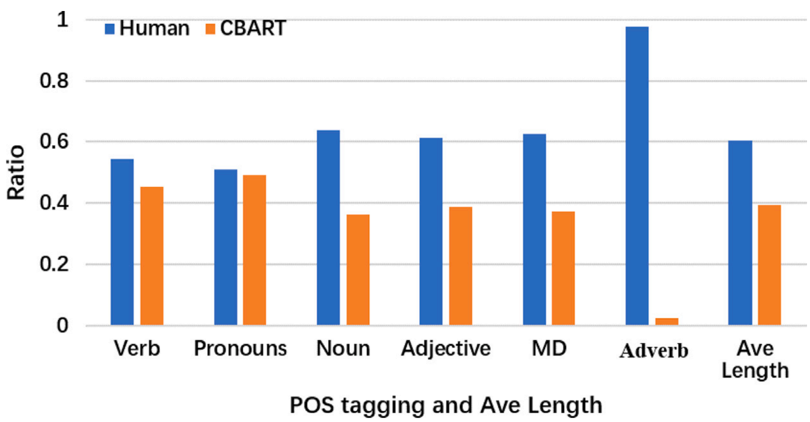


Fig. 2. The ratio of CBART and Human for POS tagging and average length with 4 keywords on One-Billion-Word.

text fluent, additional actions of insertion (e.g., Replace and Keep) are introduced for refinement (Zhang, Wang, Li, Gan, Brockett, & Dolan, 2020). Furthermore, CBART (He, 2021) leverages the pre-trained encoder to provide the decoder with coarse-grained modification for refinement action guidance.

In this paper, we argue that the diversity of existing lexically CTG methods is still unsatisfactory. As shown in Fig. 2, we take CBART and dataset One-Billion-Word (Chelba et al., 2014) as an example and compare with human reference on the token POS tagging results as well as sentence length. We can see that no matter which type of POS tagging, the distinct words used in CBART are much less than that of Human. For example, CTG by the CBART in the test dataset has 2755 verbs and 18 adverbs. However, there are 3316 verbs and 238 adverbs in the Human. The ratios of verbs in CBART and Human are 45.4% and 54.6%. The ratios of adverbs in CBART and Human are 7.0% and 93.0%. Besides, the average sentence length of CBART is much shorter than Human (15.5 vs. 23.6). Short texts limit the variability and flexibility of the text, making them less likely to introduce various words. Of course, the more diverse the generated text, the more difficult it is plausible. That is, it is a great challenge how to improve the generation's richness and diversity while maintaining a high level of fluidity and integrity.

To do so, we propose a lightweight dynamic refinement strategy that takes the best advantages of Pre-trained Language Models (PLMs). Instead of costly pre-training/finetuning, we focus on increasing the randomness of inference. There are two steps. The first step aims at proactively enlarging the pool of decoded sentences. On the one hand, different from one-token-per-insertion in existing methods, we insert an uncertain number of tokens between two keywords at each decoding step, where the number is determined following a well-designed distribution. To ensure high-quality decoding, the probability of inserting more tokens is increasing along with the growing sentence length, because we observe that it is too difficult for PLMs to generate many tokens while knowing only a few. On the other hand, we randomly mask an increasing number of tokens to force PLMs to examine and revise the whole sentence via reconstruction. Although existing works also design supervised/unsupervised substitution actions, the probability of triggering the actions is stationary over time, regardless of a higher demand for more generated content. By repeating the first step, we can obtain a pool of different sentences. In the second step, we screen and select the best sentence from the pool using another PLM, introducing differentiated sentence quality measurements.

In general, our key assumption is that the higher the number of randomly inserted tokens, the higher the uncertainty of the generated sentences, which in turn promotes the increase of sentence diversity. Inserting more tokens at once increases uncertainty.

However, the higher the number of random token insertions, the lower the quality of the generated text. Our aim is to find this balance between improving text quality and diversity.

For evaluation, we have conducted extensive experiments on two publicly available datasets compared with five baseline methods. We also design four dimensions for human evaluation: fluency, complete, informativeness, and correlation (between generated texts). Both results demonstrate that our proposed method can generate more diverse and high-quality sentences. Further ablation and case study provide systematically analysis on diversity.

2. Related work

2.1. Non-autoregressive

Non-autoregressive (NAR) (Gu, Bradbury, Xiong, Li, & Socher, 2018) has an unparalleled advantage that is generating speed for text generation, but the text quality of NAR models has a huge gap with autoregressive models. To alleviate this problem, recent works (Ghazvininejad, Levy, Liu, & Zettlemoyer, 2019; Gu et al., 2018; He, 2021; Lee, Mansimov, & Cho, 2018) find the answer from the autoregressive and NAR models how to trade-off speed and quality, that is, generating the text with multiple iterations. Li and Shi (2021) proposes a grammatical error correction model based on BERT (Devlin, Chang, Lee, & Toutanova, 2019). Liu, Huang, and Mou (2022) proposes a NAR unsupervised summarization, which employs an edit-based search towards a heuristically defined score to generate a summary. Although the performance of the current NAR models gets improved, it is still not comparable with the autoregressive models due to the lack of dependency among target words.

2.2. Lexically CTG

Lexically CTG relies on some keywords to generate the text. Early studies, e.g., B/F-LMs (Liu, Fu, Qu, & Lv, 2019; Mou, Yan, Li, Zhang, & Jin, 2015), use one constrained keyword to generate text. And GBS (Post & Vilar, 2018) costs the quality and diversity to produce the text based on multi-keywords. Recently, Zhang et al. (2020) proposes a POINTER model to insert new tokens between existing keywords in a parallel manner with BERT. But the POINTER model imposes all the generation burden on the decoder, leading the poor text quality. To address this problem, He (2021) proposes a CBART model to convert some generation burden to the encoder, which predicts insertion, replacement, copy actions to guide the modification of decoder. Seo, Jung, Jung, Hwang, Namgoong, and Roh (2023) employs semantic control grammar and re-rank method to obtain the candidate sentences. Yuan, Wang, Yu, and Zhang (2022) proposes a hierarchical template-transformer model to generate sentiment texts with personalized information. Iso (2022) proposes a lexically CTG framework by automatically generating templates given constrained lexicons and replacing placeholders in the templates. In addition, there are some new tasks: Nie, Yang, Chen, Kong, Zhu, and Yang (2022) proposes a novel task that aims at keywords to sentence generation with desired complexity levels for grade reading and language teaching tasks. Recently, lexically constrained text generation with large language models (LLMs) (OpenAI, 2023; Touvron, Lavril, et al., 2023; Touvron, Martin, et al., 2023) represents a dynamic and evolving field, with rapid advancements in both model development and practical applications. And LLM achieves unparalleled results on this task.

However, the above methods only insert one token per iteration, because the performance will degrade if generating many tokens while knowing only a few. This leads to sacrificed textual diversity. Note that LLMs are autoregressive models, and they have massive parameters and training datasets. Our model is a NAR based on PLM. It is unfair to compare our model with LLMs. In this paper, we highlight the advantages of the diversity of generation-based methods and propose to add randomness to inference for a diverse pool of candidate sentences, so that we can select high-quality text while improving the diversity.

This paper is the extension of CBART (He, 2021) . Compared to original work, this paper has several improvements:

Methods: We propose two-step dynamic refinement 4.3 to balance the text of diversity and quality. The first step uses the diffused mask and dynamic refinement to generate a wide range of pool sentences for constrained tokens. And the second step refines the sentences to select the best sentence.

Experiments: (1) We conduct our method on One-Billion-Word and Yelp dataset to demonstrate the effectiveness by automatic metrics (B-2, B-4, and SB-4 et al.); (2) We also design four dimensions (Fluency, Complete, Informativeness, and Correlation) for human evaluation to complement automatic metrics. (3) Compared with CBART (He, 2021), our method achieve great performance on automatic metrics and human evaluation.

Content: Our research focuses on achieving a harmonious balance between text quality and diversity, aiming to enhance both aspects simultaneously. Lexically CTG tasks primarily prioritize the quality of generated texts, often neglecting the diversity of outputs. To foster text diversity, it is essential to generate enough tokens, which prompt for rich variations in the generated text. However, this motivation for diversity may lead to a compromise in text quality. Hence, our core approach centers on keeping the text quality and improving diversity.

3. Research objective

This paper aims to address the following problems of the existing methods for Lexically CTG:

- How to effectively increase the length of generation texts to enhance their diversity?
- When generating long texts means the number of inserted tokens increases, how to ensure the quality of the generated texts?

To address these issues, we propose a lightweight dynamic refinement strategy that takes the best advantages of Pre-trained Language Models (PLMs), which contain two modules. The first module focuses on generating diverse texts by flexible insertion. The core idea is that increase the length of texts by inserting multiple tokens. The second module aims to provide more keywords without relying on extra knowledge to keep the high-quality inserted tokens.

4. Methodology

Our proposed method aims to take the best advantages of PLMs and focuses on improving the inference over diversity. In this section, we first follow CBART (He, 2021) to construct data for training in Sections 4.1 and 4.2. Then, we describe our two-step dynamic refinement strategy in Section 4.3.

4.1. Data preparation

Due to the training datasets (e.g., One-Billion-Word) only containing the text without constrained keywords, we follow He (2021) to design the training data $D = \{(X^e, Y^e, X^d, Y^d)\}$, where X^e and Y^e are inputs and outputs of the encoder. Particularly, at the very beginning, X^e is constrained keywords. X^d and Y^d are the inputs and outputs of the decoder. We use X^e and Y^e to construct X^d — a sequence of text containing `[mask]` and keywords, which will be regarded as modification results to guide the decoder to generate target output sentence Y^d. Next, we will explain the details with examples.

For the label space of Y^e, we define three actions: Keep, Replace, and Insert. **Keep** means that the current token remains original state into the next iteration. **Replace** means that the current token should be replaced by a new token. **Insert** is used to add a new token before the current token.

Example. Give a piece of text T = `First pictures released of the aftermath of a bomb near a police station in Lahore`, we construct the dataset D as follows:

For the encoder, we randomly select some words as keywords from T, e.g., X^e = `of the aftermath of a bomb near a police station in Lahore`. Then, we randomly replace keywords (e.g., replace `aftermath` with `math`) in the X^e by a certain percentage (e.g., 15%). Finally, the input of the encoder is X^e = `of the math of a bomb near a police station in Lahore`, and the encoder label is $Y^e = \{2, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0\}$, where 2 denotes that some tokens should be inserted before `of`, and 1 denotes that the current token should be replaced by another token. 0 denotes that the current token remains unchanged.

For the decoder, we design the input of the decoder based on X^e and Y^e: X^d = `[mask] of the [mask] of a bomb near a police station in Lahore`. The decoder output/label Y^d should be the original text for reconstruction, replacing the `[mask]` with the correct token, e.g., Y^d = `First of the aftermath of a bomb near a police station in Lahore`. Note that if multiple tokens need to be inserted (the first [mask] in X^d), we only pick up one leftmost word as target and ignore the remaining ones.

We can see that it is to insert one token only per iteration, although there are several words to insert. Indeed, we can conduct the generation several times, but this, to some extent, hurts diversity. So, we introduce the inference method with a dynamic number of insertion tokens in Section 4.3 for improvements.

4.2. Training

Encoder. Encoder aims to use the input tokens to predict the action (Keep, Replace, or Insert) for each token. Specially, we use BART (Lewis et al., 2020) with linear transformation to encode constrained keywords and compute actions for keywords. Given an input sequence with n constrained keywords, $\mathbf{X}^e = [\boldsymbol{x}^e_1, \boldsymbol{x}^e_2, \ldots, \boldsymbol{x}^e_n]$, the encoder is expressed as follows:

$$\begin{cases} \mathbf{H}^e = \mathrm{BART}(\mathbf{X}^e) \\ \mathbf{Y}^e = \mathrm{Softmax}(\mathbf{W}^e\mathbf{H}^e + b^e) \end{cases}, \tag{1}$$

where $\mathbf{H}^e = [\boldsymbol{h}^e_1, \boldsymbol{h}^e_2, \ldots, \boldsymbol{h}^e_n]$ is the embedding of tokens, and $\boldsymbol{h}^e_i \in \mathbb{R}^k$. $\mathbf{W}^e \in \mathbb{R}^{k\times 3}$ and b^e are the trained parameters. $\mathbf{Y}^e = [\boldsymbol{y}^e_1, \boldsymbol{y}^e_2, \ldots, \boldsymbol{y}^e_n]$ is the probability of actions, $\boldsymbol{y}^e_i \in \mathbb{R}^3$. And we chose the max probability as the action of the current token. Then, we use the cross-entropy as the loss function:

$$L_{encoder} = -\frac{1}{n}\sum_{t=1}^{n} \log p(\boldsymbol{y}^e_t | \boldsymbol{x}^e_1, \boldsymbol{x}^e_2, \ldots, \boldsymbol{x}^e_n). \tag{2}$$

Decoder. Given the decoder input $\mathbf{X}^d$ and label $\mathbf{Y}^e$, $\mathbf{X}^d = [\boldsymbol{x}^d_1, \boldsymbol{x}^d_2, \ldots, \boldsymbol{x}^d_m]$ contains `[mask]` and all keywords, and $\mathbf{Y}^d = [\boldsymbol{y}^d_1, \boldsymbol{y}^d_2, \ldots, \boldsymbol{y}^d_m]$ is the predicted text. To ensure the keywords appear in the output, the decoder only predicts the `[mask]` in the $\mathbf{X}^d$. Following the Encoder, we also use BART and linear transformation to compute the $\mathbf{Y}^d$ as follows:

$$\begin{cases} \mathbf{H}^{\mathbf{d}} = \mathrm{BART}(\mathbf{X}^d) \\ \mathbf{Y}^d = \mathrm{Softmax}(\mathbf{W}^d\mathbf{H}^{\mathbf{d}} + b^d) \end{cases}, \tag{3}$$

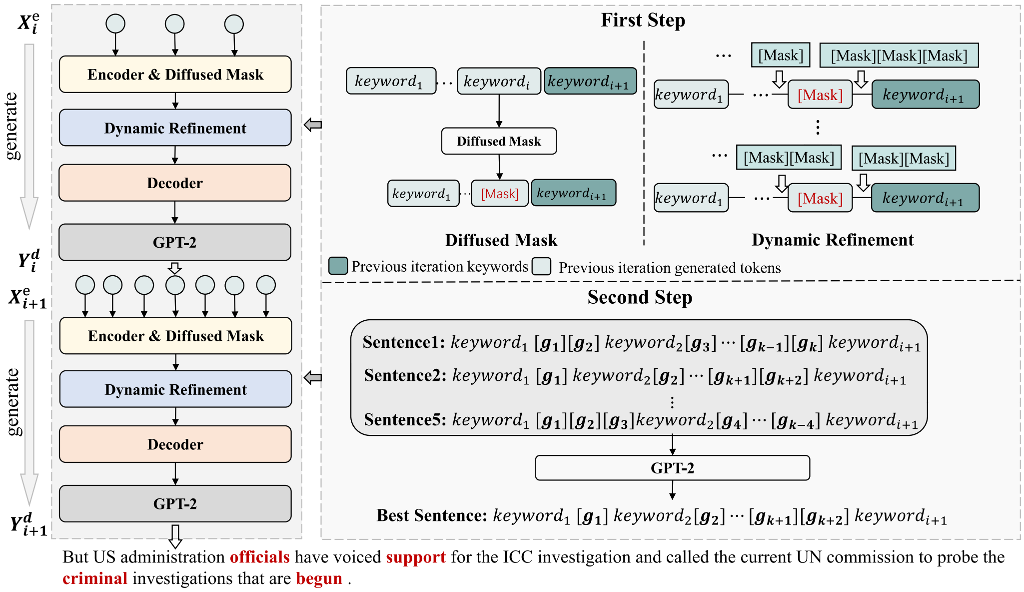


Fig. 3. Architecture of Dynamic Refinement. g_i is the ith inserted token in the sentence.

where $\mathbf{W}^e \in \mathbb{R}^{k\times v}$ and b^d are the trained parameters. v is the vocabulary size. We optimize the decoder by minimizing the reconstruction loss:

$$L_{decoder} = -\frac{1}{m}\sum_{t=1}^{m} \log p(\mathbf{y}_t^d | \mathbf{X}^d, y_{\leq t}^d). \tag{4}$$

Training Loss. In this part, we join the $L_{encoder}$ and $L_{decoder}$ to optimize the total loss:

$$L_{total} = L_{encoder} + L_{decoder}. \tag{5}$$

4.3. Dynamic refinement for inference

In this section, we introduce our two-step dynamic refinement considering both generation diversity and quality. The basic idea is to add more randomness via dynamic insertion and mask to obtain a board pool of candidate sentences (the first step), then select the best one for the next step refinement (the second step).

4.3.1. Overview

Similar to previous work (He, 2021; Zhang et al., 2020), the text is generated through multiple iterations. At each iteration, the output of the decoder would be the input of the encoder in the next. Two example iterations are shown in Fig. 3, each has two steps. Formally, given some constrained keywords X_i^e in the ith iteration:

For the first step, we feed X_i^e into the encoder to obtain the action labels Y_i^e, and in the meantime, we randomly mask the words generated in the previous iteration according to an increasing ratio, namely diffused Mask (Section 4.3.2). Then, we construct several the decoder inputs $\{X_i^d\}$ according to masked X_i^e and action label Y_i^e, such that we can generate various candidate sentences $\{Y_i^d\}$ by feeding them to the pre-trained decoder, namely dynamic insertion (Section 4.3.3).

Finally, at the second step, we utilize GPT-2 (Radford et al., 2019) to select the best candidate Y_i^{d*} from the pool $\{Y_i^d\}$. We choose the negative log-likelihood (NLL) as the measurement. In next iteration, the Y_i^{d*} becomes the input of encoder X_{i+1}^e.

The iteration continues until the model hit a Termination Condition. There are two trigger conditions:(1) All encoder labels are 0, which indicates the tokens remains unchanged. (2) The upper limit of iterations is reached.

4.3.2. Diffused mask

Diffused Mask (DM) aims to ensure the high-quality of generated text along with the increasing content. Currently, one of the major problems for lexically CTG is that we need to generate too many tokens using only a few keywords — it has been demonstrated that using a few words to generate many tokens leads to low-quality predictions, e.g., in large-scale PLMs, only 15% of words are masked. Along with the increasing mask ratio, the pre-training performance presents a downward trend (Devlin et al., 2019). Therefore, we attempt to randomly mask more tokens as more and more tokens are generated, taking advantage of PLMs for fluency.

Note that although there are replace actions in the training phase, they follow a certain percentage in the training data. Thus, no matter how much-generated content we have, the replace action is very limited. And, we cannot directly increase the action ratio in training. Otherwise, it leads to a confused pre-trained encoder/decoder. On the contrary, DM brings more randomness, and under extreme conditions, all generated tokens in the last iteration will be masked. That is, we encourage the model to re-generate them based on all keywords as well as the generated tokens so far.

Specially, we randomly mask the ratio k of generated tokens of the previous iteration. And the ratio of masking is influenced by dynamic refinement (Detailed see 4.3.3); the more words it inserts, the larger the mask ratio is. We thus call it diffused mask. k is defined as follows:

$$k = \begin{cases} T * 10\% & 0 \leq T < \alpha_1 \\ 0 & T \geq \alpha_1 \end{cases}, \tag{6}$$

where T is the number of iteration. α_1 is the hyper-parameter, which is equal to 4.

4.3.3. Dynamic refinement

Dynamic Refinement (DR) explores the effectiveness of inserting multiple `[mask]` between two keywords. We hypothesize that the number of given words affects the number of inserted tokens. Therefore, we have to carefully choose the insertion token number within a controlled range. On the one hand, randomly inserting multiple words between two keywords will force the model to consider longer outputs, increasing the probability of using more diverse words. On the other hand, the model can generate fluent and plausible sentences, only if there are enough contextual words; otherwise, the generation quality will drop dramatically.

In specific, we randomly choose to insert t tokens in the closed interval, $t \in [1, T+1]$, where T is the number of iteration, which can be expressed as follows:

$$t = \begin{cases} \text{Random}([1, T+1]) & 0 \leq T < \alpha_2 \\ 1 & T \geq \alpha_2 \end{cases}, \tag{7}$$

where α_2 is a hyper-parameter, which is equal to 4. Note that when the iteration reaches a certain value (α_2), the number of inserted tokens per iteration between two keywords is reduced to 1. The reason is that after several iteration, the structure of the text has been completed and it is not suitable for too many insertions leading to a decrease in the text quality.

Given the inputs and outputs of the encoder (the refinement actions), we can obtain a list of candidates with varying lengths for decoding by performing DR 5 times. We then leverage the pre-trained decoder with the Top-p strategy (Holtzman, Buys, Du, Forbes, & Choi, 2020) to generate multiple sequences. Finally, we select the best sequence for the next iteration as described in the Section 4.3.1.

5. Experiments

5.1. Datasets and metrics

Datasets. To demonstrate the performance of our model, we evaluate the model on two publicly available datasets One-Billion-Word[1] and Yelp.[2] We largely follow the works (He, 2021; Zhang et al., 2020) for fair comparisons. **One-Billion-Word** is a public dataset from EMNLP2017 WMT News Crawl data, which contains 268,586 sentences. **Yelp** is the Yelp English review dataset from Cho et al. (2018), which contains 160,000 training examples. Following He (2021), we select the sentences with length greater 10 and less than 40 as the Training and Dev dataset. We choose the 1M and 0.1M sentences as the Training dataset and Dev dataset for One-Billion-Word and Yelp. Besides, we construct 6 Test datasets based on 1–6 keywords, and each Test has 1,000 sentences.[3]

Metrics. We evaluate our model from two aspects: the quality and diversity of text. Following the previous work (He, 2021; Zhang et al., 2020), we use BLEU (Papineni, Roukos, Ward, & Zhu, 2002), NIST (Doddington, 2002), and METEOR (Banerjee & Lavie, 2005) as automatic metrics to evaluate the generation quality, which measure the similarity between the generated text and the reference text. For the diversity metrics, we use Self-BLEU score (SB-4) (Zhu et al., 2018), distinct bigrams (D-2) (Li, Galley, Brockett, Gao, & Dolan, 2016), and 4-grams (D-4) (Li et al., 2016) to evaluate the similarity between one generated text with other generated texts. Note that higher scores of BLEU, NIST, or METEOR indicate that the text generation model can generate similar text with reference text and has high-quality content. The lower scores of Self-BLEU or higher distinct n-gram indicate that the model can generate more diverse texts. Results for models are averaged over the six test sets with N = 1 to N = 6. N is the number of constrained keywords.

5.2. Parameter settings

Following CBART (He, 2021), we choose the BART-base (Lewis et al., 2020) as the pre-trained language model. The learning rate is set to $1e^{-5}$, and batch size is 80. We optimize our model with AdamW (Loshchilov & Hutter, 2019). We train our model for 3 epochs, and the number of iteration is 10. α_1 and α_2 are set to 4. In top-p, the p is 0.5.

---

1 http://www.statmt.org/lm-benchmark/
2 https://www.yelp.com/dataset
3 https://github.com/NLPCode/CBART

Table 1
Main results on the One-Billion-Word and Yelp. Results with † are computed based on re-trained models. Other results are reported in He (2021). The best results are in the **bold**, and the second-best results are underlined. "M" refers to METEOR. ↑ denotes that the higher the value, the better. ↓ denotes that the lower the value, the better.

	Model	BLEU ↑		NIST ↑		↑	↓	Distinct ↑	
		B-2	B-4	N-2	N-4	M	SB-4	D-2	D-4
One-Billion-Word	sep-B/F	4.4%	0.7%	0.616	0.618	7.0%	52.1%	46.3%	78.8%
	asyn-B/F	4.3%	0.7%	0.554	0.556	6.8%	50.3%	47.8%	80.9%
	GBS	10.1%	2.8%	**1.487**	**1.497**	13.5%	37.0%	59.3%	87.2%
	CGMH	9.9%	3.5%	1.153	1.165	13.1%	**10.2%**	**78.9%**	**99.3%**
	POINTER$_{BERT-Base}$	2.5%	0.1%	0.961	0.961	10.2%	–	–	–
	Our$_{BERT-Base}$	10.3%	3.7%	0.969	0.969	28.1%	14.3%	69.2%	90.1%
	CBART$_{BART-Base}$†	15.1%	5.8%	0.964	0.965	25.7%	14.8%	69.8%	98.8%
	Our$_{BART-Base}$	**16.4%**	**5.9%**	0.980	0.981	**31.4%**	12.9%	70.4%	**99.3%**
Yelp	sep-B/F	6.9%	2.1%	0.521	0.531	8.7%	67.1%	31.9%	64.6
	asyn-B/F	7.5%	2.3%	0.698	0.711	9.0%	68.0%	31.9%	64.6%
	GBS	13.6%	4.5%	**1.680**	**1.712**	15.3%	59.3%	37.5%	70.2%
	CGMH	12.3%	4.6%	1.413	1.446	14.6%	**23.6%**	**60.7%**	**97.7%**
	POINTER$_{BERT-Base}$	4.0%	0.3%	1.139	1.140	13.0%	–	–	–
	Our$_{BERT-Base}$	18.0%	6.2%	0.935	0.934	25.9%	31.7%	43.2%	90.7%
	CBART$_{BART-Base}$†	18.4%	7.9%	1.102	1.103	28.8%	36.2%	45.8%	91.9%
	Our$_{BART-Base}$	**20.0%**	**8.0%**	1.223	1.223	**28.8%**	29.5%	48.5%	95.7%

5.3. Baseline

We compare our method with several strong baselines for lexically CTG:

- **Sep-B/F** and **Asyn-B/F** (Mou et al., 2015) that provides a novel backward and forward language model to generate previous and subsequent words conditioned.
- **GBS** (Hokamp & Liu, 2017) that proposes Grid Beam Search to allow the inclusion of pre-specified lexical constraints.
- **CGMH** (Miao, Zhou, Mou, Yan, & Li, 2019) that uses Metropolis–Hastings sampling for constrained sentence generation.
- **POINTER** (Zhang et al., 2020) that can use BERT to insert new tokens between existing tokens in a parallel manner.
- **CBART** (He, 2021) that reduces the generation burden from the decoder, improving text quality.

5.4. Overall performance

Table 1 shows the overall performance on One-Billion-Word and Yelp. We can observe that: **(1)** Our method can outperform all baselines in most metrics on different datasets, demonstrating the effectiveness and generalization ability of our model. **(2)** CGMH achieves the best performance on SB-4, D-2, and D-4. Because it comes at the expense of degrading text quality, which is consistent with previous work (He, 2021; He & Li, 2021). **(3)** On the content quality metrics (BLEU, NIST, and M), our proposed method gets improved slightly. Because DM can force to mask some generated tokens, using more keywords to improve the quality of the generated tokens. **(4)** On the diversity metrics (SB-4, D-2, and D-4), our model achieves a great improvement compared with state-of-the-art CBART and POINTER. Because we flexibly insert multiple tokens per action, which brings more different generation distributions over vocabulary, and thus generates longer and more candidate sentences. **(5)** Our approach outperforms previous works (CBART and POINTER) in both the BERT and BART models, showcasing superior performance for our approach and illustrating its generalizability.

5.5. Human evaluation

For complementary to automatic metrics, we conduct a human evaluation. We randomly select 50 sentences and invite three volunteers[4] to compare the generated sentences with CBART and Human Reference. Following previous works (He, 2021; Zhang et al., 2020), we use the **Fluency** and **Complete** to demonstrate the text quality and use the **Informativeness** and **Correlation** to demonstrate the diverse text. For inter-annotator agreement, the values of Cohen's kappa (Fleiss, 1971) are 0.67, 0.79, 0.69, and 0.62 for Fluency, Complete, Informativeness, and Correlation. From the Table 2, we can see that: **(1) Quality.** Compared with Human Reference, the results of our proposed method still have a large gap. But we are preferable to CBART. **(2) Diversity.** Human reference still has an overwhelming advantage, but the performance gap between our proposed method and CBART gets larger, demonstrating the effectiveness of our proposed lightweight refinement strategy.

[4] All volunteers are engaged in NLP research, and they independently annotate the data.

Table 2
Human evaluation on One-Billion-Word.

Fluency: A and B, which is more fluency?				
System A		Neutral	System B	
Our	34.9%	9.9%	**45.2%**	Human
Our	**32.7%**	38.2%	29.1%	CBART
Complete: A and B, which text is more complete?				
System A		Neutral	System B	
Our	3.3%	67.3%	**29.4%**	Human
Our	**15.6%**	72.8.%	12.6%	CBART
Informativeness: A and B, which is more informative?				
System A		Neutral	System B	
Our	18.0%	8.7%	**73.3%**	Human
Our	**62.7%**	15.7%	21.6%	CBART
Correlation: Correlation of generated text and Human?				
System A		Correlation	Non-correlation	
Our		27.9%	**72.1%**	
CBART		36.3%	**63.7%**	

Table 3
An ablation study on One-Billion-Word. The number of constrained keywords is 4.

Model	↑ B-2	↑ N-2	↓ SB-4	↑ D-2
Our	**18.3%**	**1.106**	**11.0%**	72.5%
w/o DR	16.9%	1.104	13.3%	71.0%
w/o DM	18.0%	1.061	11.0%	**74.2%**
w/o DR & DM	16.5%	1.030	13.2%	70.8%

5.6. Ablation study

We conduct an ablation study to illustrate the performance of our main modules and parameters.

Effect of DM and DR. We conduct experiments to analyze the effect of DM and DR, as shown in Table 3. We can see that: **(1)** Without the DM, the performance of B-2 and N-2 is reduced, and the results of SB-4 and D-2 are better than our method. These indicate two facts: First, DR does enhance the diversity of texts, but at the same time, it can cause a decrease in text quality. Second, the quality and diversity of the text are a game, and DM can balance them by providing more keywords, while improving the diversity and quality of the text. **(2)** Compared w/o DM with w/o DR & DM, all results of w/o DM are better than w/o DR & DM, particularly in terms of diversity. This demonstrates that inserting multiple tokens at once makes sentences longer and increases text variety without degrading the text quality. **(3)** Removed DR leads all results worse than our approach, especially in the diversity. The major reason is that DR can flexibly insert multiple tokens between two keywords to improve diversity. **(4)** Compared w/o DR with w/o DR & DM, DM can improve text quality, but it has little impact on diversity.

Effect of Hyper-parameter (α_1 **and** α_1). α_1 and α_2 are thresholds for DM and DR, respectively. We show the evaluation metrics for different α_1 and α_2 in Table 4. From Table 4, we can find that: (1) The experimental results have the same trend for α_1 and α_2. Our performance is the best when $\alpha_1 = 4$ and $\alpha_2 = 4$. Because when DR inserts a large number of words, DM is needed to cooperate with expanding the ratio of the mask to ensure text quality. (2) When α_2 rises, the effectiveness of our method rises significantly at the first few iterations. Because the first few iterations increase the number of insertions, significantly increasing the sentence length.

Effect of the Number of Constraints. As shown in Table 5, as the number of constrained keywords increases, the scores of B-2 and N-2 increase rapidly. These results are consistent with our hypothesis that the model can generate high-quality text only if there is sufficient context information. Therefore, it is reasonable to increase the ratio of masking on generated tokens in the last iteration, because there are more given works for the current iteration. Besides, we also observe a similar trend in terms of diversity. This is because more keywords provide a longer sentence as initialization, and it is relatively easy to generate longer sentences, which brings a higher probability of involving more different words.

5.7. Text analysis

We report the validity of our method in two ways: Sentence Structure and Case Study.

Table 4
The Results with different α_1 and α_2 on One-Billion-Word. The umber of constrained keywords is 4.

Model		↑ B-2	↑ N-2	↓ SB-4	↑ D-2
$\alpha_1 = 1$	$\alpha_2 = 4$	18.1%	1.035	11.7%	72.3%
$\alpha_1 = 2$		**18.4%**	1.082	12.4%	71.5%
$\alpha_1 = 3$		18.2%	1.092	11.4%	72.4%
$\alpha_1 = 4$		18.3%	**1.106**	**11.0%**	72.8%
$\alpha_1 = 5$		18.1%	1.099	11.1%	72.6%
$\alpha_1 = 6$		18.3%	1.104	11.4%	**72.9%**
$\alpha_1 = 4$	$\alpha_2 = 1$	17.2%	1.119	12.2%	72.4%
	$\alpha_2 = 2$	18.3%	**1.149**	11.8%	72.4%
	$\alpha_2 = 3$	18.0%	1.098	12.1%	72.2%
	$\alpha_2 = 4$	**18.3%**	1.106	**11.0%**	72.8%
	$\alpha_2 = 5$	**18.3%**	1.086	11.1%	72.7%
	$\alpha_2 = 6$	18.2%	1.070	11.2%	**73.1%**

Table 5
The number of constrained keywords on One-Billion-Word.

Number Keywords	↑ B-2	↑ N-2	↓ SB-4	↑ D-2
1	4.7%	0.505	14.1%	63.2%
2	8.6%	0.679	12.6%	68.7%
3	13.0%	0.864	11.4%	71.2%
4	18.3%	1.106	**11.0%**	72.7%
5	24.0%	1.365	11.4%	72.9%
6	**29.6%**	**1.665**	11.1%	**73.4%**

Sentence Structure. We can clearly observe that:

(1) In Fig. 4(f), the distribution of CBART concentrates on the right side, human reference is in the middle, while our proposed method is evenly distributed in different lengths. That is, we tend to generate longer sentences than CBART. In Fig. 4 (a∼e), no matter in which position (i.e., between different pairs of keywords), we tend to insert more words than CBART. This also demonstrates that our method does not prefer a specific position, which may reduce the text quality.

(2) From Fig. 5 (Top)[5], we can see that our method has a similar number of Verb, Pronouns, MD, Adjective, and Adverbs with Human. Only the Noun is slightly less than Human. But the number of CBART for POS tagging is far less than that of Human.

Case Study. From Fig. 5 (Bottom), although our method and CBART have good fluency and grammatical rules, etc., generated sentences via our method are longer and contain richer information, such as more Verb (`voiced, called`) and Pronoun (`that`) in first example.

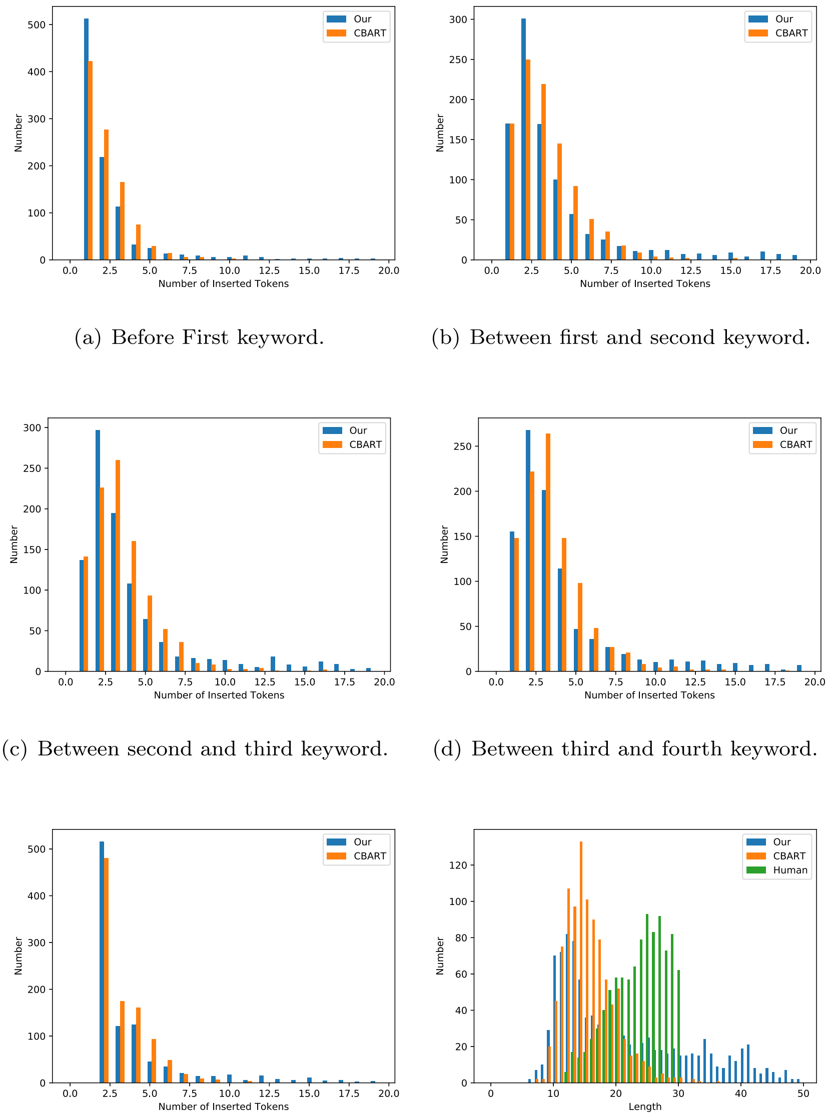
6. Conclusion

In this paper, we propose a lightweight dynamic refinement strategy to improve the diversity of lexically CTG. It can provide a larger number and longer length of candidate texts in each iteration and take the best one from it. And experimental results show that our method is effective and significantly better than competitive baselines. The extensive analysis also provides interesting insights about our method. In the future, we will elicit more information from the language pre-trained model to compensate for the lack of constrained keywords.

7. Limitation

The limitations of our approach include the following two points: (1) Although our approach improves the performance of the model in lexical CTG, there is still a gap compared to Human Reference. (2) Using the prompt-tuning method to elicit more reliable keywords from the pre-trained language model can improve the model results more effectively. (3) Due to data and parameter scale limitations, our method is not compared with ChatGPT.

[5] We use NLTK (https://www.nltk.org/) to analysis texts and obtain the POS tagging.



(a) Before First keyword. (b) Between first and second keyword.

(c) Between second and third keyword. (d) Between third and fourth keyword.

(e) After fourth keywords. (f) Sentence length.

Fig. 4. The number of inserted tokens before keywords and sentence length on One-Billion-Word.

CRediT authorship contribution statement

Changsen Yuan: Conceptualization, Formal analysis, Methodology, Validation, Writing – original draft, Writing – review & editing. **Heyan Huang:** Funding acquisition, Methodology, Writing – original draft, Writing – review & editing. **Yixin Cao:** Formal analysis, Methodology, Writing – review & editing. **Qianwen Cao:** Formal analysis, Methodology.

Data availability

I have shared dataset in the paper.

Acknowledgments

The authors thank all the reviewers for their suggestions and comments. This study is supported by National Natural Science Foundation of China (No. U21B2009).

	Verb	Pronouns	Noun	Adjective	MD	Adverbs
Human	3316	809	6824	1800	239	751
CBART	2755	781	3875	1137	142	18
Our	3513	940	5717	1577	207	731

	Constrained Keywords: officials support criminal begun
CBART	The **officials** said their **support** for the **criminal** investigation had **begun** .
Our	But US administration **officials** have voiced **support** for the ICC investigation and called the current UN commission to probe the **criminal** investigations that are **begun** .
	Constrained Keywords: analysts suggested figures sector
CBART	But **analysts suggested** that the **figures** could dampen demand for private **sector** recovery .
Our	Some **analysts suggested** that the **figures** showed the services index , manufacturing service **sector** and industry **sector** remained weak at 11 .

Fig. 5. At the top, we count the number of POS tagging. At the bottom, we show generated texts by CBART and our method with same keywords extracted from One-Billion-Word test. MD is Modal verb.

References

Banerjee, Satanjeev, & Lavie, Alon (2005). METEOR: an automatic metric for MT evaluation with improved correlation with human judgments. In Jade Goldstein, Alon Lavie, Chin-Yew Lin, & Clare R. Voss (Eds.), *Proceedings of the workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization* (pp. 65–72). Association for Computational Linguistics, URL https://aclanthology.org/W05-0909/.

Chelba, Ciprian, Mikolov, Tomás, Schuster, Mike, Ge, Qi, Brants, Thorsten, Koehn, Phillipp, et al. (2014). One billion word benchmark for measuring progress in statistical language modeling. In Haizhou Li, Helen M. Meng, Bin Ma, Engsiong Chng, Lei Xie (Eds.), *INTERSPEECH 2014, 15th annual conference of the international speech communication association* (pp. 2635–2639). ISCA, URL http://www.isca-speech.org/archive/interspeech_2014/i14_2635.html.

Cho, Woon Sang, Zhang, Pengchuan, Zhang, Yizhe, Li, Xiujun, Galley, Michel, Brockett, Chris, et al. (2018). Towards coherent and cohesive long-form text generation. arXiv preprint arXiv:1811.00511.

Devlin, Jacob, Chang, Ming-Wei, Lee, Kenton, & Toutanova, Kristina (2019). BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, & Thamar Solorio (Eds.), *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, volume 1 (long and short papers)* (pp. 4171–4186). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/n19-1423.

Doddington, George (2002). Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on human language technology research* (pp. 138–145).

Duan, Siyu, Li, Wei, Cai, Jing, He, Yancheng, & Wu, Yunfang (2021). Query-variant advertisement text generation with association knowledge. In Gianluca Demartini, Guido Zuccon, J. Shane Culpepper, Zi Huang, & Hanghang Tong (Eds.), *CIKM '21: The 30th ACM international conference on information and knowledge management* (pp. 412–421). ACM, http://dx.doi.org/10.1145/3459637.3482290.

Fan, Angela, Lewis, Mike, & Dauphin, Yann N. (2018). Hierarchical neural story generation. In Iryna Gurevych, & Yusuke Miyao (Eds.), *Proceedings of the 56th annual meeting of the association for computational linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, volume 1: long papers* (pp. 889–898). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/P18-1082, URL https://aclanthology.org/P18-1082/.

Fang, Le, Zeng, Tao, Liu, Chaochun, Bo, Liefeng, Dong, Wen, & Chen, Changyou (2021). Transformer-based conditional variational autoencoder for controllable story generation. CoRR abs/2101.00828. URL https://arxiv.org/abs/2101.00828.

Fleiss, Joseph L. (1971). Measuring nominal scale agreement among many raters. *Psychological Bulletin*, *76*, 378–382.

Ghazvininejad, Marjan, Levy, Omer, Liu, Yinhan, & Zettlemoyer, Luke (2019). Mask-predict: Parallel decoding of conditional masked language models. In Kentaro Inui, Jing Jiang, Vincent Ng, & Xiaojun Wan (Eds.), *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing* (pp. 6111–6120). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/D19-1633.

Gu, Jiatao, Bradbury, James, Xiong, Caiming, Li, Victor O. K., & Socher, Richard (2018). Non-autoregressive neural machine translation. In *6th international conference on learning representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, conference track proceedings*. OpenReview.net, URL https://openreview.net/forum?id=B1l8BtlCb.

He, Xingwei (2021). Parallel refinements for lexically constrained text generation with BART. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, & Scott Wen-tau Yih (Eds.), *Proceedings of the 2021 conference on empirical methods in natural language processing* (pp. 8653–8666). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2021.emnlp-main.681.

He, Xingwei, & Li, Victor O. K. (2021). Show me how to revise: Improving lexically constrained sentence generation with xlnet. In *Thirty-fifth AAAI conference on artificial intelligence, aaai 2021, thirty-third conference on innovative applications of artificial intelligence, IAAI 2021, the eleventh symposium on educational advances in artificial intelligence* (pp. 12989–12997). AAAI Press, URL https://ojs.aaai.org/index.php/AAAI/article/view/17536.

Hokamp, Chris, & Liu, Qun (2017). Lexically constrained decoding for sequence generation using grid beam search. In Regina Barzilay, & Min-Yen Kan (Eds.), *Proceedings of the 55th annual meeting of the association for computational linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, volume 1: long papers* (pp. 1535–1546). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/P17-1141.

Holtzman, Ari, Buys, Jan, Du, Li, Forbes, Maxwell, & Choi, Yejin (2020). The curious case of neural text degeneration. In *8th international conference on learning representations*. OpenReview.net, URL https://openreview.net/forum?id=rygGQyrFvH.

Hughes, J. Weston, Chang, Keng-hao, & Zhang, Ruofei (2019). Generating better search engine text advertisements with deep reinforcement learning. In Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, & George Karypis (Eds.), *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 2269–2277). ACM, http://dx.doi.org/10.1145/3292500.3330754.

Iso, Hayate (2022). AutoTemplate: A simple recipe for lexically constrained text generation. http://dx.doi.org/10.48550/arXiv.2211.08387, CoRR abs/2211.08387. arXiv:2211.08387.

Lee, Jason, Mansimov, Elman, & Cho, Kyunghyun (2018). Deterministic non-autoregressive neural sequence modeling by iterative refinement. In Ellen Riloff, David Chiang, Julia Hockenmaier, & Jun'ichi Tsujii (Eds.), *Proceedings of the 2018 conference on empirical methods in natural language processing* (pp. 1173–1182). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/d18-1149.

Lewis, Mike, Liu, Yinhan, Goyal, Naman, Ghazvininejad, Marjan, Mohamed, Abdelrahman, Levy, Omer, et al. (2020). BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Dan Jurafsky, Joyce Chai, Natalie Schluter, Joel R. Tetreault (Eds.), *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 7871–7880). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2020.acl-main.703.

Li, Jiwei, Galley, Michel, Brockett, Chris, Gao, Jianfeng, & Dolan, Bill (2016). A diversity-promoting objective function for neural conversation models. In Kevin Knight, Ani Nenkova, & Owen Rambow (Eds.), *NAACL HLT 2016, the 2016 conference of the North American chapter of the association for computational linguistics: Human language technologies* (pp. 110–119). The Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/n16-1014.

Li, Piji, & Shi, Shuming (2021). Tail-to-tail non-autoregressive sequence prediction for Chinese grammatical error correction. In Chengqing Zong, Fei Xia, Wenjie Li, & Roberto Navigli (Eds.), *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing, ACL/IJCNLP 2021, (volume 1: long papers)* (pp. 4973–4984). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2021.acl-long.385.

Li, Huayang, Su, Yixuan, Cai, Deng, Wang, Yan, & Liu, Lemao (2022). A survey on retrieval-augmented text generation. CoRR abs/2202.01110. URL https://arxiv.org/abs/2202.01110.

Liu, Dayiheng, Fu, Jie, Qu, Qian, & Lv, Jiancheng (2019). BFGAN: backward and forward generative adversarial networks for lexically constrained sentence generation. *IEEE ACM Transcations on Audio Speech and Language Processing*, *27*(12), 2350–2361. http://dx.doi.org/10.1109/TASLP.2019.2943018.

Liu, Puyuan, Huang, Chenyang, & Mou, Lili (2022). Learning non-autoregressive models from search for unsupervised sentence summarization. In Smaranda Muresan, Preslav Nakov, & Aline Villavicencio (Eds.), *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers)* (pp. 7916–7929). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2022.acl-long.545.

Loshchilov, Ilya, & Hutter, Frank (2019). Decoupled weight decay regularization. In *7th international conference on learning representations*. OpenReview.net, URL https://openreview.net/forum?id=Bkg6RiCqY7.

Miao, Ning, Zhou, Hao, Mou, Lili, Yan, Rui, & Li, Lei (2019). CGMH: constrained sentence generation by Metropolis-Hastings sampling. In *The thirty-third AAAI conference on artificial intelligence, AAAI 2019, the thirty-first innovative applications of artificial intelligence conference, IAAI 2019, the ninth AAAI symposium on educational advances in artificial intelligence* (pp. 6834–6842). AAAI Press, http://dx.doi.org/10.1609/aaai.v33i01.33016834.

Mou, Lili, Yan, Rui, Li, Ge, Zhang, Lu, & Jin, Zhi (2015). Backward and forward language modeling for constrained sentence generation. *Computation and Language*, arXiv.

Nie, Jinran, Yang, Liner, Chen, Yun, Kong, Cunliang, Zhu, Junhui, & Yang, Erhong (2022). Lexical complexity controlled sentence generation. http://dx.doi.org/10.48550/arXiv.2211.14540, CoRR abs/2211.14540arXiv:2211.14540.

OpenAI (2023). GPT-4 technical report. http://dx.doi.org/10.48550/arXiv.2303.08774, CoRR abs/2303.08774. arXiv:2303.08774.

Papineni, Kishore, Roukos, Salim, Ward, Todd, & Zhu, Wei-Jing (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the association for computational linguistics* (pp. 311–318). ACL, http://dx.doi.org/10.3115/1073083.1073135, URL https://aclanthology.org/P02-1040/.

Post, Matt, & Vilar, David (2018). Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In Marilyn A. Walker, Heng Ji, & Amanda Stent (Eds.), *Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: human language technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, volume 1 (long papers)* (pp. 1314–1324). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/n18-1119.

Qin, Lianhui, Bosselut, Antoine, Holtzman, Ari, Bhagavatula, Chandra, Clark, Elizabeth, & Choi, Yejin (2019). Counterfactual story reasoning and generation. In Kentaro Inui, Jing Jiang, Vincent Ng, & Xiaojun Wan (Eds.), *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing* (pp. 5042–5052). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/D19-1509.

Radford, Alec, Wu, Jeffrey, Child, Rewon, Luan, David, Amodei, Dario, Sutskever, Ilya, et al. (2019). Language models are unsupervised multitask learners. *OpenAI Blog*, *1*(8), 9.

Seo, Hyein, Jung, Sangkeun, Jung, Jeesu, Hwang, Taewook, Namgoong, Hyuk, & Roh, Yoon-Hyung (2023). Controllable text generation using semantic control grammar. *IEEE Access*, *11*, 26329–26343. http://dx.doi.org/10.1109/ACCESS.2023.3252017.

Touvron, Hugo, Lavril, Thibaut, Izacard, Gautier, Martinet, Xavier, Lachaux, Marie-Anne, Lacroix, Timothée, et al. (2023). LLaMA: Open and efficient foundation language models. http://dx.doi.org/10.48550/arXiv.2302.13971, CoRR abs/2302.13971. arXiv:2302.13971.

Touvron, Hugo, Martin, Louis, Stone, Kevin, Albert, Peter, Almahairi, Amjad, Babaei, Yasmine, et al. (2023). Llama 2: Open foundation and fine-tuned chat models. http://dx.doi.org/10.48550/arXiv.2307.09288, CoRR abs/2307.09288. arXiv:2307.09288.

Yuan, Li, Wang, Jin, Yu, Liang-Chih, & Zhang, Xuejie (2022). Hierarchical template transformer for fine-grained sentiment controllable generation. *Information Processing & Management*, *59*(5), Article 103048. http://dx.doi.org/10.1016/j.ipm.2022.103048.

Zhang, Yizhe, Wang, Guoyin, Li, Chunyuan, Gan, Zhe, Brockett, Chris, & Dolan, Bill (2020). POINTER: constrained progressive text generation via insertion-based generative pre-training. In Bonnie Webber, Trevor Cohn, Yulan He, & Yang Liu (Eds.), *Proceedings of the 2020 conference on empirical methods in natural language processing* (pp. 8649–8670). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2020.emnlp-main.698.

Zhou, Li, Gao, Jianfeng, Li, Di, & Shum, Heung-Yeung (2020). The design and implementation of XiaoIce, an empathetic social chatbot. *Computational Linguistics*, *46*(1), 53–93. http://dx.doi.org/10.1162/coli_a_00368.

Zhu, Yaoming, Lu, Sidi, Zheng, Lei, Guo, Jiaxian, Zhang, Weinan, Wang, Jun, et al. (2018). Texygen: A benchmarking platform for text generation models. In Kevyn Collins-Thompson, Qiaozhu Mei, Brian D. Davison, Yiqun Liu, Emine Yilmaz (Eds.), *The 41st international ACM SIGIR conference on research & development in information retrieval* (pp. 1097–1100). ACM, http://dx.doi.org/10.1145/3209978.3210080.