Long Text Generation with Topic-aware Discrete Latent Variable Model

Erguang Yang[1*], **Mingtong Liu**[2], **Deyi Xiong**[3], **Yujie Zhang**[1†], **Jinan Xu**[1], **Yufeng Chen**[1]
[1]School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China
[2]Beijing Lanzhou Technology Co., Ltd., Beijing, China
[3]College of Intelligence and Computing, Tianjin University, Tianjin, China
{egyang,yjzhang,jaxu,chenyf}@bjtu.edu.cn,
liumingtong@langboat.com, dyxiong@tju.edu.cn

Abstract

Generating coherent long texts is an important yet challenging task, particularly for the open-ended generation task. Prior work based on discrete latent codes focuses on the modeling of discourse relation, resulting in discrete codes only learning shallow semantics (Ji and Huang, 2021). A natural text always revolves around several related topics and the transition across them is natural and smooth. In this work, we investigate whether discrete latent codes can learn information of topics. To this end, we build a topic-aware latent code-guided text generation model. To encourage discrete codes to model information about topics, we propose a span-level bag-of-words training objective for the model. Automatic and manual evaluation experiments show that our method can generate more topic-relevant and coherent texts.

1 Introduction

Generating coherent long texts based on a shorter text input is an important yet challenging task. Recent large-scale pre-trained language models (LMs), e.g. GPT-2 (Radford et al., 2019) and BART (Lewis et al., 2020) have shown state-of-the-art performance on various natural language generation (NLG) tasks such as summarization and dialog generation (Radford et al., 2019; Lewis et al., 2020). Although pre-trained language models can generate fluent texts, it is still challenging for them to generate coherent long-form texts.

To address this, various text generation methods have been proposed to improve the coherence of generated texts. The first type of methods attempt to decompose the long text generation procedure into two stages: first producing a skeleton and then generating a text based on the skeleton, where the skeleton can be a set of keywords (Yao et al., 2019; Hua and Wang, 2020; Tan et al., 2021), events (Goldfarb-Tarrant et al., 2020; Li et al., 2022), etc. However, these methods more or less rely on high-precision automatic extraction tools.

The second line of methods introduce latent variables to model high-level structures of texts (Shao et al., 2019; Shen et al., 2019; Ji and Huang, 2021). Shen et al. (2019) develop multi-level VAE models (Bowman et al., 2016) for modeling paragraph-level and sentence-level features. Recently, Ji and Huang (2021) have explored discrete latent variables (Van Den Oord et al., 2017) to model inter-sentence relations and achieved impressive performance. However, through preliminary experiments, we have found that their methods often produced texts unrelated to the input prompts, as shown in Table 5. The possible reason is that latent variables only learn shallow semantic information.

Intuitively, a long text includes a latent topic sequence, and using the topic sequence to guide the generation process can help to generate topic-related and coherent texts. Given the above discussion, we propose a topic-aware latent code-guided text generation model. The main idea is to learn discrete latent codes with topic-aware information. Concretely, as shown in Figure 1, we first abstract a given long text to a discrete code sequence according to the fixed span length, and then it is spliced with the prompt and input into a pre-training language model to reconstruct the given text. To make the discrete latent codes capture information about topics[1], we further propose a span-level bag-of-words prediction auxiliary task to ensure that each code in the discrete sequence can reconstruct the bag-of-words of its corresponding text span. Once discrete latent codes are learned, we use another auto-regressive transformer to fit the prior distribution of them.

[*]This work was done when Erguang Yang was interning at Beijing Lanzhou Technology Co., Ltd., Beijing, China.
[†]Corresponding Author.

[1]The latent topic here is similar to the topic in previous LDA model (Blei et al., 2003) and neural topic model NVDM-GSM (Miao et al., 2017), which defines a topic as a probabilistic distribution over the vocabulary (i.e., a mixture of words).

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 8100–8107
December 7-11, 2022

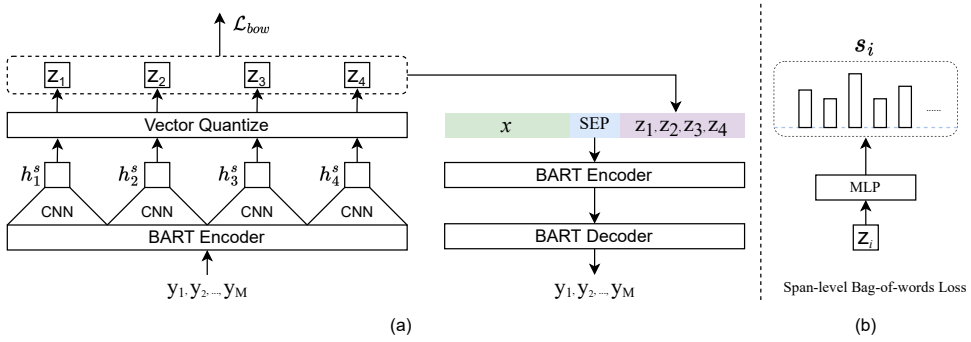


Figure 1: The diagram of our model. (a) Learning a discrete latent code squence $\boldsymbol{z}$ via abstracting and reconstructing the given text $\boldsymbol{y}$. (b) The Span-level Bag-of-words loss enables the latent code z_i to predict the bag-of-words distribution of its corresponding text span $\boldsymbol{s}_i$.

We summarize our contributions as follows: **(1)** We build a topic-aware latent code-guided text generation model, in which we propose a span-level bag-of-words loss to ensure each code can capture the topical information. Additionally, our method can be more easily integrated with the pre-training language models without modifying the structure of them. **(2)** We conduct extensive experiments on two public story generation datasets. Both automatic and manual evaluation results show that our model significantly outperforms strong baselines in generating topic-relevant and coherence texts.

2 Our Approach

We formulate the long text generation task as a conditional generation problem, i.e., generating a long text $\boldsymbol{y} = (y_1, y_2, ..., y_M)$ given an short input prompt $\boldsymbol{x} = (x_1, x_2, ..., x_N)$, where M and N denote the number of tokens.

A natural text can be segmented into successive several text spans, and each span implies a local topic. We suppose the text span length is l,[2] $\boldsymbol{y}$ can be segmented into a span sequence $\boldsymbol{s} = (\boldsymbol{s}_1, ..., \boldsymbol{s}_L)$, where each $\boldsymbol{s}_i$ consists of l tokens, $L = \lceil M/l \rceil$. Our idea is to introduce discrete latent variables $\boldsymbol{z}$ to learn local topics of the text.

Following prior work (Ji and Huang, 2021), we optimize our model by maximizing the following the evidence lower bound (ELBO, Kingma and Welling (2013)):

$$\begin{aligned}\mathcal{L}_{\text{ELBO}} =& \mathbb{E}_{\boldsymbol{z}\sim q_\phi} \log p_\theta(\boldsymbol{y}|\boldsymbol{z}, \boldsymbol{x}) - \\ & D_{\text{KL}}\left(q_\phi(\boldsymbol{z}|\boldsymbol{y}) \| p_\psi(\boldsymbol{z}|\boldsymbol{x})\right)\end{aligned} \tag{1}$$

where the $p_\theta(\boldsymbol{y}|\boldsymbol{z}, \boldsymbol{x})$, $q_\phi(\boldsymbol{z}|\boldsymbol{y})$ and $p_\psi(\boldsymbol{z}|\boldsymbol{x})$ denote the *generator*, *posterior network* and *prior network*, respectively.

[2]The span length is decided by the number of CNN layers.

The training process includes two stages. In the first training stage, we train the *generator* and *posterior network* to learn discrete latent codes of text (§2.1). In the second stage, another auto-regressive model is used as *prior network* to model the prior distribution of discrete codes (§2.2). During text generation, the *prior network* first predicts a discrete code sequence given the input prompt, which is then applied to guide text generation.

2.1 Learning Discrete Latent Codes

We define a latent embedding space $E \in \mathbb{R}^{K\times D}$ where K is the size of the discrete latent space, D is the dimensionality of each latent embedding vector.

To model latent topics of a text, we first use the BART encoder and CNN to obtain text span representations $\boldsymbol{h}^s = [\boldsymbol{h}_1^s, ..., \boldsymbol{h}_L^s]$, and then they are mapped into K-way categorical representations through an MLP layer:

$$\boldsymbol{t}_i = \text{GeLU}\left(\boldsymbol{h}_i^s \boldsymbol{W}_1 + \boldsymbol{b}_1\right)\boldsymbol{W}_2 + \boldsymbol{b}_2 \tag{2}$$

where $\boldsymbol{W}_1, \boldsymbol{b}_1, \boldsymbol{W}_2$ and $\boldsymbol{b}_2$ are trainable parameters. GeLU denotes the gelu activation function.

The Gumbel softmax enables choosing discrete codes in a fully differentiable way (Jang et al., 2016). The probability for choosing the k-th code is

$$p_i^k = \frac{\exp\left(\left(t_i^k + g_i^k\right)/\tau\right)}{\sum_{j=1}^K \exp\left(\left(t_i^j + g_i^j\right)/\tau\right)} \tag{3}$$

where τ is a non-negative temperature. During training, the true gradient of the Gumbel softmax outputs is used. During inference, the discrete code z_i is chosen by:

$$z_i = \underset{k\in K}{\text{argmax}}\, t_i^k \tag{4}$$

Note that the z_i-th embedding in discrete latent space E will be really used.

2.1.1 Training

We use reconstruction loss, diversity loss, span-level bag-of-words loss to train the above model.

Span-Level Bag-of-Words Loss. To ensure each code can reconstruct the bag-of-words of its corresponding text span, we propose a span-level bag-of-words loss. In this way, words with the same features will be merged into the same code during training. Specifically, we take $\boldsymbol{z}_i$ as input and predict the bag-of-words distribution:

$$\boldsymbol{p}_b = \text{softmax}(\boldsymbol{W}_{bow}\boldsymbol{z}_i + \boldsymbol{b}_{bow}) \tag{5}$$

where $\boldsymbol{W}_{bow}$ and $\boldsymbol{b}_{bow}$ are trainable parameters. Then the span-level bag-of-words loss is computed as follows:

$$\mathcal{L}_{bow} = -\sum_{w \in V} \boldsymbol{s}_i(w) \log \boldsymbol{p}_b(w) \tag{6}$$

where V denotes the vocabulary.

Reconstruction Loss. We use the loss to train the model to reconstruct $\boldsymbol{y}$ given $\boldsymbol{x}$ and $\boldsymbol{z}$.

$$\mathcal{L}_{\text{recon}} = -\mathbb{E}_{\boldsymbol{z} \sim q_\phi(\boldsymbol{z}|\boldsymbol{y})} \log p_\theta(\boldsymbol{y}|\boldsymbol{z}, \boldsymbol{x}) \tag{7}$$

Diversity Loss. In our preliminary experiments, we have also found that BART model tends to only utilize one discrete code from the code vocabulary, which harms the expressiveness of the discrete latent space. Following previous works (Baevski et al., 2020; Ji and Huang, 2021), we use a diversity loss to encourage the equal use of the K codes by maximizing the entropy of the averaged softmax distribution $\overline{\boldsymbol{p}} = \frac{1}{L}\sum_{l=i}^{L} \text{softmax}\left(\boldsymbol{t}_i\right)$ across text spans, where $\boldsymbol{t}_i$ is the code logits for the text span $\boldsymbol{s}_i$:

$$\mathcal{L}_d = \frac{1}{K}\sum_{k=1}^{K} \bar{p}^k \log \bar{p}^k \tag{8}$$

The Overall Objective. The final loss function for training is defined as follows:

$$\mathcal{L} = \mathcal{L}_{\text{recon}} + \lambda_1 * \mathcal{L}_d + \lambda_2 * \mathcal{L}_{bow} \tag{9}$$

where λ_* are balancing hyperparameters.

Model	B-1	rB-1	B-1-R	rB-1-R	Distinct-4/5
Dataset: Wikiplots					
Seq2Seq	17.0	21.9	2.7	3.9	47.5 / 63.9
BART	19.8	24.2	4.2	5.6	53.1 / 68.8
DVT	30.2	29.5	6.6	6.6	**89.6 / 97.6**
Ours	31.6	30.4	**7.4**	**7.2**	89.4 / **97.6**
w/o $\mathcal{L}_d$	**32.0**	**31.9**	6.4	6.5	86.2 / 96.7
w/o $\mathcal{L}_{bow}$	30.2	30.2	6.2	6.3	87.9 / 97.3
Dataset: WritingPrompts					
Seq2Seq	20.0	24.0	5.1	7.4	29.8 / 43.6
BART	21.8	25.4	6.1	8.3	40.2 / 54.8
DVT	28.6	26.4	8.2	7.9	84.4 / 95.9
Ours	28.6	26.9	**8.8**	**8.5**	**86.6 / 96.9**
w/o $\mathcal{L}_d$	26.8	24.0	7.8	7.4	85.0 / 96.5
w/o $\mathcal{L}_{bow}$	**29.2**	**27.8**	7.8	7.7	85.9 / 96.7

Table 1: Automatic evaluation results, which are reported as the mean over three runs.

2.2 Prior Modeling

In the second stage, we use another Transformer encoder-decoder model to learn the prior distribution of the discrete latent codes. To train the model, we optimize the following training objective:

$$\mathcal{L}_{prior} = -\sum_{i=1}^{L} \log(z_i|\boldsymbol{z}_{<i}, \boldsymbol{x}) \tag{10}$$

where the ground-truth latent code sequence $\boldsymbol{z}$ is obtained by Eq. 4.

3 Experiments

3.1 Datasets

We evaluated our model on two open story generation datasets, WritingPrompts (Fan et al., 2018) and Wikiplots[3], which are collected from Reddit and Wikipedia, respectively. We only retain the first 25 sentences (split using NLTK[4]) of the texts in both datasets, and filter out short (<50 tokens) texts. The data statistics are shown in Table 4.

3.2 Baselines

We compared our model with the following baselines: (1) **Seq2Seq:** It adopts the same architecture as BART without pretrained parameters. We trained this baseline from a randomly initialized BART on the downstream datasets. (2) **BART:** We fine-tuned the pretrained BART (Lewis et al., 2020) model on the downstream datasets. (3) **DVT:** This method first maps the span-level representations to

[3]The dataset is available at https://github.com/markriedl/WikiPlots

[4]https://www.nltk.org/

Models	Relevance			
	Win	Lose	Tie	κ
Ours vs. BART	19.3	33.3	47.3	0.30
Ours vs. DVT	30.0	15.3	54.7	0.33
Models	**Coherence**			
	Win	Lose	Tie	κ
Ours vs. BART	61.3	11.3	27.3	0.31
Ours vs. DVT	46.0	24.6	29.4	0.32

Table 2: Human evaluation results on Wikiplots dataset. The scores indicate the percentages(%). κ denotes Fleiss' kappa (Fleiss, 1971) to measure the interannotator agreement.

Code ID	Top Code-Related Words
11	**friend**, **daughter**, long, high, **father**, old, **son**, saves, **mother**, **girlfriend**, **man**, **wife**, love, life, best, side, **young**, **lawyer**, **brother**, ...
64	**two**, time, day, **next**, night, **first**, years, film, **one**, story, place, **three**, takes, **end**, begins, town, world, house, **One**, way, game, **last**, ...

Table 3: Code-word distribution.

discrete codes by vector quantizing. Transposed CNNs are adopted to rescale the code embeddings into token-level features. It is then added to decoder's input embedding layer with token embeddings and positional encodings at each decoding position. Additionally, it adopts an auxiliary objective on the latent representations to model the discourse relations (Ji and Huang, 2021). We implemented the model based on the codes provided by the original paper.

3.3 Evaluation Metrics

We adopted the following automatic metrics to evaluate the performance on the test sets. (1) **BLEU-1 (B-1)**: The metric computes n-gram overlap between generated texts and human-written texts (Papineni et al., 2002). (2) **Reverse-BLEU-1 (rB-1)**: The metric measures the recall of generated n-grams (Shi et al., 2018). (3) **BLEU-1-R (B-1-R)** and (4) **Reverse-BLEU-1-R (rB-1-R)**: These two metrics denote that stop words in the generated and referenced text are filtered first, and then the B-1 and rB-1 are computed, respectively. (5) **Distinct-n**: We use $n = 4, 5$ to measure the generation diversity (Li et al., 2016).

3.4 Results

As can be observed in Table 1, compared with B-1 and rB-1, both B-1-R and rB-1-R metrics drop significantly, suggesting that stop words contribute a lot to both B-1 and rB-1. Our model can generate more word overlaps with reference texts as shown by better B-1-R and rB-1-R scores. Besides, in terms of Distinct-4/5 metrics, our model significantly outperforms the Seq2Seq and BART model and is comparative with DVT on the Wikiplots dataset. On the writingprompt dataset, our model surpasses all baselines. These results show that our method can improve the diversity of generation.

Among different variants of our model (ablation study), we see that removing either the diversity loss or the span-level bag-of-words loss has a negative impact on B-1-R, rB-1-R, and diversity metrics. We also note that removing any of the two losses can help B-1 and rB-1 scores. This is because the model tends to use only one discrete code in the whole code vocabulary, as shown in Figure 2, which further leads the code to pay more attention to the high-frequency words, such as stop words.

3.5 Human Evaluation

For human evaluation, we conducted pair-wise comparisons with two strong baseline models (BART and DVT). We randomly sampled 100 texts from the test set of Wikiplots and obtain 300 texts from the three models. We hired three proficient English speakers as human evaluators to give a preference (win, lose, or tie) in terms of relevance and coherence, respectively.[5] *Relevance* measures whether the generation is related to the prompt. *Coherence* measures whether the generation revolves around the same topic and the transition across sentences is natural and smooth.

As shown in Table 2, our model outperforms DVT yet underperforms BART in terms of relevance. For coherence, our model significantly outperforms all baselines, demonstrating the effectiveness of the proposed method. The human evaluation results show *fair* inter-annotator agreement (0.21-0.40).

3.6 Code Study

To verify whether discrete codes can learn topical information about the topic, we show the code-word distribution in Table 3. We can see that different codes correspond to different words, in which code-11 represents "*family*" topic, code-64 represents "*numeral*" topic. These results show that discrete code can automatically merge words of the

[5]We pay humans $18 per hour.

same type during training.

4 Conclusion

In this paper, we have presented a topic-aware code-guided text generation model. The model firstly learns a discrete latent code sequence that abstracts the topic sequence of text, which is then applied to guide the generation of text. To encourage discrete codes to learn topical information, we further propose a span-level bag-of-words loss. Experimental results show that our model can generate more relevant and coherent texts.

Limitations

Here we discuss the limitations of our work. Our method is still a two-stage method which may lead to error propagation. In the future, how to improve the generation quality of discrete codes is worth exploring. Additionally, text coherence is a relatively broad definition including topical relatedness, causal relationship, temporary ordering and discourse structures (Hu et al., 2022). In this work, we only explore the utilization of topical information. It is an important direction to explore other aspects in the future.

Acknowledgements

The present research was supported by the National Nature Science Foundation of China (No. 61876198, 61976015, 61976016). Deyi Xiong was supported by the Natural Science Foundation of Tianjin (Grant No. 19JCZDJC31400). We would like thank the three anonymous reviewers for their constructive suggestions and insightful comments.

Ethics Statement

We recognize that our approach may generate spurious and potentially harmful content due to biases of pre-trained models on the large-scale web corpus. Therefore, we urge users to carefully examine the ethical implications of the output and to apply the system with caution in real-world scenarios.

References

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33:12449–12460.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

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

Seraphina Goldfarb-Tarrant, Tuhin Chakrabarty, Ralph Weischedel, and Nanyun Peng. 2020. Content planning for neural story generation with aristotelian rescoring. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4319–4338, Online. Association for Computational Linguistics.

Zhe Hu, Hou Pong Chan, Jiachen Liu, Xinyan Xiao, Hua Wu, and Lifu Huang. 2022. PLANET: Dynamic content planning in autoregressive transformers for long-form text generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2288–2305, Dublin, Ireland. Association for Computational Linguistics.

Xinyu Hua and Lu Wang. 2020. PAIR: Planning and iterative refinement in pre-trained transformers for long text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 781–793, Online. Association for Computational Linguistics.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Haozhe Ji and Minlie Huang. 2021. DiscoDVT: Generating long text with discourse-aware discrete variational transformer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4208–4224, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

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.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Qintong Li, Piji Li, Wei Bi, Zhaochun Ren, Yuxuan Lai, and Lingpeng Kong. 2022. Event transition planning for open-ended text generation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3412–3426, Dublin, Ireland. Association for Computational Linguistics.

Yishu Miao, Edward Grefenstette, and Phil Blunsom. 2017. Discovering discrete latent topics with neural variational inference. In *International Conference on Machine Learning*, pages 2410–2419. PMLR.

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

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

Zhihong Shao, Minlie Huang, Jiangtao Wen, Wenfei Xu, and Xiaoyan Zhu. 2019. Long and diverse text generation with planning-based hierarchical variational model. 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 3257–3268, Hong Kong, China. Association for Computational Linguistics.

Dinghan Shen, Asli Celikyilmaz, Yizhe Zhang, Liqun Chen, Xin Wang, Jianfeng Gao, and Lawrence Carin. 2019. Towards generating long and coherent text with multi-level latent variable models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2079–2089, Florence, Italy. Association for Computational Linguistics.

Zhan Shi, Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. 2018. Toward diverse text generation with inverse reinforcement learning. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4361–4367. International Joint Conferences on Artificial Intelligence Organization.

Bowen Tan, Zichao Yang, Maruan Al-Shedivat, Eric Xing, and Zhiting Hu. 2021. Progressive generation of long text with pretrained language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4313–4324, Online. Association for Computational Linguistics.

Aaron Van Den Oord, Oriol Vinyals, et al. 2017. Neural discrete representation learning. *Advances in neural information processing systems*, 30.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

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 AAAI Conference on Artificial Intelligence*, volume 33, pages 7378–7385.

A Appendix

A.1 Implementation Setting

We used the pre-trained BART-base[6] model to initialize our model and other baselines due to limited computational resources. Following (Ji and Huang, 2021), we also used 3-layer 1D CNNs with kernel size 2, stride 2, and 0 padding on both sides, which means that every 8 tokens will be mapped to 1 code. The latent code size K was set to 512, D to 768. The Gumbel temperature was annealed from 1.0 to 0.1 after 2.5k steps, and temperature anneal rate was set to $5e-4$. We set training batch size to 4, max length to 512, gradient accumulation step to 8, learning rate to $5e-5$, λ_2 to 0.1. The λ_1 was 10/15 for Wikiplots/writingprompt, respectively.

During inference, we randomly sampled 1,000 prompts from each test set for automatic evaluation. We generated texts using the nucleus sampling with $p = 0.9$ for the trade-off between diversity and fluency. The same inference settings were applied to all baselines for fair comparison.

The prior network also uses BART-base framework, and we only use pre-trained parameters to initialize the encoder. The learning rate is set to $8e-5$. We use the nucleus sampling with $p = 0.9$ to generate discrete code sequences. We implement all models based on the repository of Huggingface's Transformers[7] (Wolf et al., 2020).

A.2 Code Distribution

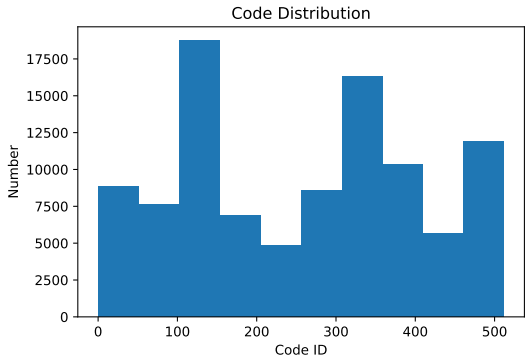
We show code distribution on the test set of the Wikiplots dataset in Figure 2. We can observe that removing span-level bag-of-words loss or diversity loss leads the model to only utilize one code in the whole code vocabulary.

A.3 Case Study

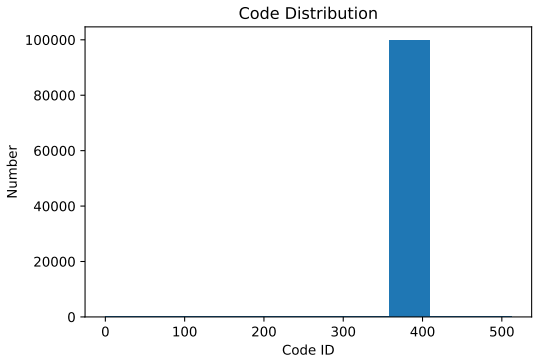
Table 5 shows examples generated by different models on the Wikiplots dataset. We observe that both our model and BART can generate stories related to the prompt and maintain the same topic, in which the story generated by our model contains more plot information and is more attractive. The story generated by DVT has low relevance to the prompt.

[6] https://huggingface.co/facebook/bart-base

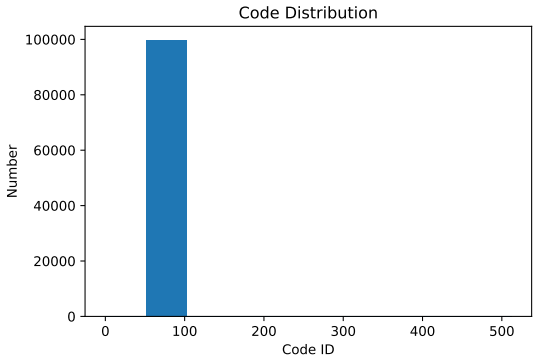
[7] https://huggingface.co/docs/transformers/index



(a) Distribution with the span-level bag-of-words loss and diversity loss



(b) Distribution without the span-level bag-of-words loss



(c) Distribution without the diversity loss

Figure 2: Code distributions on the test set of the Wikiplots dataset.

Dataset	Input	Output	#Sentence	Train	Val	Test
Wikiplots	3.4	288.0	13.9	75k	4k	4k
WritingPrompts	28.4	325.0	21.8	270k	15k	15k

Table 4: The average **input** and **output** lengths (#tokens), the average numbers of **sentences** in the output, and the numbers of stories for the **train**ing/**val**idation/**test** set.

Title	Batman: Year One
BART	The story is set in Gotham City, where Batman and Robin are on their way to the Batcave, where they are ambushed by the Joker and his henchmen. Batman, Robin, and the Joker manage to escape, but the Joker is killed by Batman's henchman. In Gotham, the Joker, who has been tracking them for years, is revealed to be the Joker's son. The Joker is then revealed to have been responsible for the death of his father, who was killed in a car crash. After the Joker kills the Joker in self-defense, he takes the Joker to his hideout, where he tells him that the Joker was responsible for his father's death, and that he wants to take revenge on Batman for killing his father.
DVT	In the aftermath of the death of his wife Anya's parents, the story is told in flashback. Detective Arthur Langton, who once played a police detective of the same name, has become more determined to catch Bruce Wayne, and goes on the run after a shooting at the apartment building. He goes to his old boss Captain Treadstone, a corrupt cop who had been assigned to him. When Alfred stumbles on a murder scene, he has a gun drawn and is knocked out with the help of his partner/defense expert. Meanwhile, Bruce is in danger. While looking through the files of his former partner, Alfred is shot in the chest. It turns out the gun he was found in matched the one on which he had shot and placed the dead man behind bars. He tells Alfred that the gun matches the bullet used by his partner. Later, the detective and Alfred go to the hospital, where Treadstones is to examine the murder weapon used to kill Wayne, as Alfred is there to interview him. Alfred finds the weapon hidden in the murder suspect's home and retrieves it before finding the weapon's remains. Alfred goes to TreadStone's house. He finds the suspect dead with a gunshot wound to the head from a bullet that landed in Alfred's car. The scene then moves to Alfred's house where he discovers a gun barrel on Alfred's head. Alfred tells Alfred he needs to be told the truth in person. Alfred is confronted by an armed man with the gun named "The Lizard", who claims he is Alfred and is looking for the Lizard, the man that Alfred killed in the accident. He and Alfred run out of gasoline, and Alfred is knocked unconscious in the car by his old friend, a mechanic. He realizes that he has been unable to kill him. The Lizard tells Alfred they will be working on a new car for Alfred in the coming days. The Police Commissioner is concerned for his reputation, as the investigation is already ongoing. One of Alfred's friends says it's his birthday, and a second man explains that it was Alfred who killed the man at the beginning. The third man says he's his nephew.
Ours	As the opening scene of the comic shows a group of police (including Batman and Robin) having a gun fight, Robin is shot by a gunman called Captain Aron Dent, and Bruce is knocked unconscious by Aron. Bruce is taken to Arkham and placed in a solitary cell with no chance of escape. He is saved by a man he calls the Joker, who is on trial for the murder of his partner. Batman explains to Bruce that he was once a scientist who killed himself, but in a subsequent struggle, was turned into a man, a member of the criminal underworld. The man then revealed his true identity as a police officer, and was later sentenced to four years in a juvenile detention facility. Joker explains the crimes to the inmates and offers them a deal for their life. The Joker has escaped the institution, and the next morning, his wife, Selina, and five others all have moved to the new apartment. Batman and his team, led by Batman, leave in a train, but Chief Medical Examiner Victoria reveals that her real identity is gone, and Batman deduces that Joker's identity was never discovered. When the rest of the team are questioned by dr John Wayne and dr Arthur Parthiban, the Joker's nemesis, dr Bruce and Bruce's team come to learn that dr Arthur and Bruce are both dead. Batman agrees to take the Joker to the island of Arkham to help Bruce create a cure for the disease of the brain. On board is a ship that has been shot down by a Captain Aronian. Chief Medical Officer dr Aron meets Bruce, Bruce, Batman, and Victoria, who are investigating the deaths of dr Arthur, and a nurse. The nurse with the nurse, Rose, is kidnapped by the Captain and her bodyguard. Once their investigation by Arons arrives, Batman goes to work on the Captain. Rose is the new Captain, with a new woman named Alice, and she joins Batman on the road. Batman soon arrives on Arkham. A prisoner who had teleported by the Doctor's helicopter, finds them and the two escape. Bruce begins to fear the Captain's arrival. Batman rides into the building where his mentor, dr Leo, and dr Arons are holding a patient hostage. Aron follows the prisoner, who reveals that Aron was Aron's first suspect. The Captain's men interrogate him, and also an accomplice named Tony. The Man then kills the prisoner and murders Aron, as he is possessed by a woman, and is about to drown.

Table 5: Examples generated by different models. The generated keywords that match the title are presented in blue. We observe that the story generated by DVT doesn't include the keyword ("Batman") in the title.