Addressing the Training-Inference Discrepancy in Discrete Diffusion for Text Generation

Masaki Asada[1]
[1]Artificial Intelligence Research Center, National Institute of Advanced Industrial Science and Technology, Japan
masaki.asada@aist.go.jp

Makoto Miwa[1,2]
[2]Toyota Technological Institute, Japan
makoto-miwa@toyota-ti.ac.jp

Abstract

This study addresses the discrepancy between training and inference in discrete diffusion models for text generation. We propose two novel strategies: (1) a training schema that considers two-step diffusion processes, allowing the model to use its own predicted output as input for subsequent steps during training and (2) a scheduling technique that gradually increases the probability of using self-generated text as training progresses. Experiments conducted on four widely used text generation benchmark datasets demonstrate that both proposed strategies improve the performance of discrete diffusion models in text generation[1].

1 Introduction

In recent years, diffusion models have garnered significant attention in the field of text-to-text generative models (Lin et al., 2023; Wu et al., 2023). It is known that by adjusting the diffusion steps, diffusion models can achieve a balance between quality and generation speed, demonstrating higher performance than Non-AutoRegressive (NAR) models, which generate an entire sequence at once, while enabling faster decoding than AutoRegressive (AR) models, which generate sequences token by token (Zhou et al., 2024). Diffusion models for text generation can be broadly categorized into continuous (Lin et al., 2023; Wu et al., 2023) and discrete (He et al., 2023; Zhou et al., 2024) variants. In particular, discrete diffusion models interpret the noise-adding process as replacing tokens with mask tokens. This approach has the advantage of leveraging the performance of pre-trained language models (PLMs) that perform mask infilling as a pre-training task and outperformed existing continuous diffusion models on several datasets (Zhou et al., 2024).

[1]The code is available on https://github.com/aistairc/text-diff-2step-loss

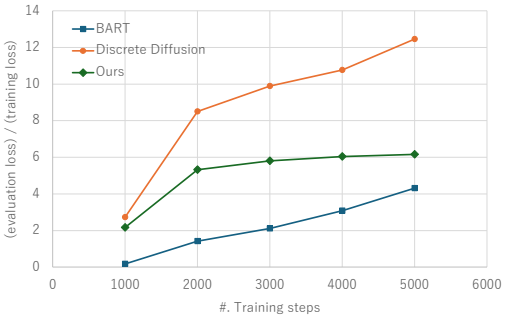

Figure 1: The value of (evaluation loss) / (training loss) for training steps. The traditional discrete diffusion model shows a larger training-inference discrepancy compared to autoregressive model BART, on the other hand, our discrete diffusion model reduces this discrepancy.

Existing discrete diffusion models for text generation have a problem of discrepancy between training and inference. During the training of diffusion models, noise is applied to gold tokens up to a randomly chosen specific time step, and the model learns to predict the correct text from the noise-applied text, minimizing the loss against the gold tokens. On the other hand, during inference, noise is applied to the tokens predicted in the previous step. This can potentially lead to inductive bias, as shown in Figure 1. While there has been research addressing this issue in AR decoding (Zhang et al., 2019), studies on addressing the discrepancy between training and inference in diffusion models are unexplored.

This study proposes two novel approaches to address the discrepancy in dealing with instances between training and inference: a two-step diffusion training schema using the predicted output as subsequent input, and a scheduling technique gradually increasing the probability of using self-generated text as training progresses. Our contributions are summarized as follows:

- We propose a novel discrete diffusion method

Proceedings of the 31st International Conference on Computational Linguistics, pages 7156–7164
January 19–24, 2025.

for text generation, tackling the discrepancy between training and inference.

- Experimental results on four widely used text generation benchmark datasets demonstrate that our proposed approach contributes to improving the performance of text generation on discrete diffusion models.

2 Related Work

In recent years, diffusion models have gained attraction in text generation tasks, demonstrating particular efficacy in text generation (Tang et al., 2023; Li et al., 2022b). Diffusion models (Ho et al., 2020; Dhariwal and Nichol, 2021) are categorized as a class of latent variable models that progressively transform random Gaussian noise into meaningful data samples. These models can be broadly classified into two categories: continuous diffusion models (Ho et al., 2020), which employ the latent space of token embeddings and iteratively refine all target token embeddings through a parameterized denoising process, and discrete diffusion models (Austin et al., 2021; Qian et al., 2024), which apply forward and denoising processes to discrete random variables with vocabulary-sized categories for text data. There have been fewer proposals for discrete diffusion models despite their advantage of being able to leverage PLMs, further research in this area is needed. Therefore, this study chooses the discrete diffusion model as the backbone and proposes an approach to address the discrepancy between training and inference of the model.

3 Discrete Diffusion Models

This section explains the discrete diffusion model for text generation, which serves as the baseline for the proposed method. We followed the base discrete diffusion model employed in Diffusion-NAT (Zhou et al., 2024), without including their proposed self-prompting component.

Text-to-text generation tasks can be formulated as modeling the conditional probability $P(Y|X)$, where $X = \{x_1, x_2, \cdots, x_m\}$ and $Y = \{y_1, y_2 \cdots, y_n\}$ denote the input text and output text respectively, both consisting of a sequence of tokens from a vocabulary $\mathcal{V}$. Discrete diffusion models perform forward and denoising processes using discrete random variables with K categories, where $K = |\mathcal{V}|$ for text data. The forward process of adding noise is defined as:

$$q(Y_t|Y_{t-1}) = v^\top(Y_t)\mathbf{Q}_t v(Y_{t-1}). \quad (1)$$

Here, $v(Y)$ maps each token index to a K-dimensional one-hot vector, $\mathbf{Q}_t$ is the probability transition matrix, and $[\mathbf{Q}_t]_{i,j}$ denotes the probability of a token i being replaced by a token j. Concretely, at the t-step of the forward process, if the i-th token is not the `[MASK]` token, it has the probability of α_t to remain unchanged and γ_t to be replaced by the `[MASK]` token, leaving the probability of $\beta_t = 1 - \alpha_t - \gamma_t$ for transitioning to other tokens in the vocabulary $\mathcal{V}$ as:

$$[\mathbf{Q}_t]_{i,j} = \begin{cases} \alpha_t & \text{if } j = i, \\ \gamma_t & \text{if } j = \texttt{[MASK]}, \\ \beta_t & \text{otherwise}, \end{cases} \quad (2)$$

where α_t and γ_t are determined by the pre-defined noise scheduler.

Discrete diffusion model employs PLMs to recover the masked tokens from the noised target text at each time step, revising the decoding process of PLMs into the NAR manner that can recover all masked tokens simultaneously. Concretely, at the t-step, given the condition text X and the noised target text Y_t containing `[MASK]` tokens, X are fed into the encoder and Y_t are fed into the decoder of PLMs respectively, and simultaneously recover all the `[MASK]` tokens into the target tokens. The employment of PLMs in the denoising process enables leveraging their pre-trained knowledge and generation capacity.

During training, the model predicts all the original tokens $Y_0 = \{y_1^{(0)}, \cdots, y_n^{(0)}\}$ using the architecture of PLM in the NAR manner at each time step as:

$$\text{PLM}(\{y_1^{(t)}, \cdots, \texttt{[M]}\}, X) = \{\hat{y}_1^{(0)}, \cdots, \hat{y}_n^{(0)}\}, \quad (3)$$

where `[M]` means `[MASK]` token. The timestep t is randomly chosen only once per sample. X and intermediate recovered text $Y_t = \{y_1^{(t)}, \cdots, \texttt{[M]}\}$ are passed to the encoder and decoder of the PLM, respectively. The representation embeddings of the input X are passed to the decoder through the cross-attention module. As Y_t usually contains several `[MASK]` tokens, the above process can be regarded as recovering all the masked tokens into the original ones, which follows the pre-training objective of PLMs. Cross-entropy loss is used for this training

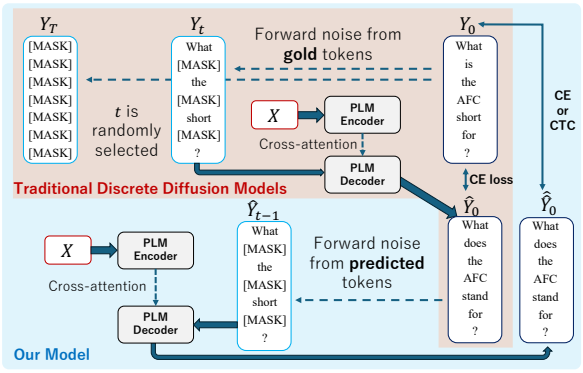

Figure 2: Training workflow of the discrete diffusion model with step-aware loss compared to the existing model

objective, and it is formulated as follows:

$$\mathcal{L}_Y = -\sum_{i=1}^{n} \log p_\theta(y_i^{(0)} | Y_t, X). \tag{4}$$

During inference, given Y_t, the model first estimates $\hat{Y}_0$, and then adds the $(t-1)$-step noise into it for producing Y_{t-1}. The above process will be iterated for multiple steps until the final results of Y_0 are obtained.

4 Methodology

This study aims to address the discrepancy between training and inference in discrete diffusion text generation with two novel approaches: two-step loss and scheduling for adopting gold or predicted sequences. We show the overview of our model in Figure 2.

Two-step loss As mentioned in Section 3, during training, a single timestep t is randomly selected, and Y_t is prepared by adding noise to Y_0 up to step t using the probability transition matrix $\mathbf{Q}_t$, from which $\hat{Y}_0$ is predicted. In other words, during training, Y_t is always derived from the **gold** sequence. On the other hand, during inference, the model starts from Y_T where T is the number of total diffusion steps and predicts $\hat{Y}_0$, and then $\mathbf{Q}_t$ is applied to obtain the Y_t, which means Y_t is always derived from the **predicted** sequence.

To address this discrepancy in training and inference, we propose a discrete diffusion model with a two-step loss. During training, a time step t is randomly selected and the input to the decoder Y_t is prepared by applying the transition matrix $\mathbf{Q}_t$ to Y_0. The model then predicts $\hat{Y}_0$ as follows:

$$\hat{Y}_0 = \text{PLM}(Y_t, X). \tag{5}$$

Subsequently, $\hat{Y}_{t-1}$ is prepared using the $\mathbf{Q}_t$ and the resulting sequence $\hat{Y}_0$. The model then predicts the target sequence at the 0-th step, taking $\hat{Y}_{t-1}$ as input:

$$\hat{\hat{Y}}_0 = \text{PLM}(\hat{Y}_{t-1}, X), \tag{6}$$

where $\hat{\hat{Y}}_0 = \{\hat{\hat{y}}_1^{(0)}, \cdots, \hat{\hat{y}}_n^{(0)}\}$. The cross-entropy loss function is as follows:

$$\mathcal{L}_{\text{CE}} = -\sum_{i}^{n} y_i^{(0)} \log \hat{\hat{y}}_i^{(0)}. \tag{7}$$

Furthermore, we investigate the potential application of Connectionist Temporal Classification (CTC) (Graves et al., 2006), which has been successful in the field of NAR (Asada and Miwa, 2023; Sohrab et al., 2024). Detailed description of CTC is provided in Apeendix A. The loss function for the variant using CTC is expressed as follows:

$$\mathcal{L}_{\text{CTC}} = -\log P_{\text{CTC}}(Y_0 | \hat{\hat{Y}}_0). \tag{8}$$

Scheduling for adoption of gold or predicted sequences The approach of preparing input sequences for the next step from the self-predicted sequences, as shown in Equation 6, becomes too difficult a task for the model in the early stage of training. Therefore, we propose an approach that probabilistically selects tokens from either of the self-predicted sequence and the gold sequence with scheduling the selection probability. For this, using the selection probability p_k, we modify Equation 6 as follows:

$$\hat{\hat{Y}}_0 = \begin{cases} \text{PLM}(\hat{Y}_{t-1}, X) & \text{with } p_k \\ \text{PLM}(Y_t, X) & \text{with } 1 - p_k, \end{cases} \tag{9}$$

where $\hat{Y}_{t-1}$ is the noised sequence from predicted sequence, while Y_t is the noised sequence from gold sequence. The selection probability p_k is determined linearly based on the current training step k and the total number of training steps K.

5 Experiments

5.1 Experimental Settings

Data and task settings We fine-tuned our model on four widely-used text generation benchmark datasets: XSum, MSNews, SQuAD v1.1, and MSQG. The detailed information on datasets and model settings are shown in Appendix B and C.

Baselines We compare our diffusion model with existing popular AR, NAR, Semi-NAR, and diffusion generation models. The details on the setting of baselines is summarized in Appendix D.

	XSum			MSNews		
	R-1	R-2	R-L	R-1	R-2	R-L
AR						
LSTM	-	-	-	30.0	14.6	27.7
Transformer	30.6	10.8	24.4	33.0	15.4	30.0
MASS	39.7	17.2	31.9	-	-	-
ProphetNet	39.8	17.1	32.0	-	-	-
BART	38.7	16.1	30.6	41.8	23.1	38.3
NAR						
NAT	24.0	3.88	20.3	-	-	-
iNAT	24.0	3.99	20.3	-	-	-
CMLM	23.8	3.60	20.1	-	-	-
LevT	24.7	4.18	20.8	-	-	-
BANG	32.5	8.98	27.4	32.7	16.1	30.3
ELMER	38.3	14.1	29.9	35.6	16.1	32.5
BnB	36.1	13.4	30.0	-	-	-
Diffusion						
GENIE	29.3	8.3	21.9	-	-	-
AR-Diff	32.2	10.6	25.2	-	-	-
Diff-NAT	**38.8**	**15.3**	30.8	46.8	31.6	44.2
Ours	38.5	14.8	**30.9**	**50.5**	**35.1**	**48.0**

Table 1: Performance comparison between our model and baselines on text summarization datasets. R-1/2/L mean ROUGE-1/2/L. Baseline scores are collected from Zhou et al. (2024) and Sohrab et al. (2024). The bold texts indicate the highest scores among NAR and Diffusion models.

	SQuAD v1.1			MSQG		
	R-L	B-4	MT	R-L	B-4	MT
AR						
LSTM	-	-	-	25.3	3.5	14.1
Transformer	29.4	4.61	9.86	29.3	5.1	16.6
MASS	49.4	20.1	24.4	-	-	-
ProphetNet	48.0	19.5	23.9	-	-	-
BART	42.5	17.0	23.1	38.1	10.2	22.1
NAR						
NAT	31.5	2.46	8.86	-	-	-
iNAT	32.4	2.33	8.84	-	-	-
CMLM	31.5	2.51	8.85	-	-	-
LevT	31.3	2.27	9.14	-	-	-
BANG	44.0	12.7	18.9	33.1	**11.0**	18.4
ELMER	40.2	13.4	20.0	26.6	5.00	15.7
BnB	41.7	13.8	-	-	-	-
Diffusion						
Diff-NAT	**46.6**	**16.1**	21.9	33.3	6.6	19.3
Ours	43.5	15.4	**23.0**	**39.0**	8.0	**20.5**

Table 2: Performance comparison between our model and baselines on question generation datasets. R-L, B-4, MT mean ROUGE-L, BLEU-4, METOR, respectively. Baseline scores are collected from Zhou et al. (2024) and Sohrab et al. (2024). The bold texts indicate the highest scores among NAR and Diffusion models.

Diffusion settings Our model is initialized with the BART-base checkpoint following Diffusion-NAT (Zhou et al., 2024), comprising 139M parameters, without any additional parameter inclusion. We adopt a linear noise shecduler (Ho et al., 2020) for the diffusion process. During the training phase, we set the diffusion steps to 1,000. For inference, we employ DDIM (Song et al., 2021) to expedite sampling, reducing the diffusion steps to 200.

5.2 Results

Text summarization Table 1 compares the performance of the proposed method and baseline models on text summarization task datasets XSum and MSNews. On the XSum dataset, the proposed method shows comparable performance to Diffusion-NAT, which employed self-prompting that recalculates the encoder output at each step during both training and inference, concatenating the output from the previous step to the context. On the MSNews dataset, the proposed method demonstrates the highest performance, showing even higher performance compared to models with AR decoding.

Question generation Table 2 compares the performance of the proposed method and baseline models on question generation tasks SQuAD and MSQG. For the SQuAD dataset, when compared to Diffusion-NAT, the proposed method scores lower on ROUGE-L and BLEU-4, but it showed a higher score on the METEOR metric. For the MSQG dataset, the proposed method outperforms Diffusion-NAT in all evaluation metrics. ROUGE-L score of our model showed the highest performance among all models, including AR decoding models.

Further analysis Table 3 shows an ablation study on the two components of the proposed method, the two-step loss and its scheduling, for both CTC loss and cross-entropy loss. Both proposed approaches contribute to performance improvement in both losses, demonstrating the effectiveness of the proposed method. Especially when adopting the CTC loss, the two-step loss and scheduling of p_k significantly improved performance. As a result, the model with CTC loss and all proposed components showed the highest performance, except on the BLEU-4 for the MSQG dataset.

Analysis of performance and latency w.r.t. diffusion steps is provided in Appendix E, showing that more diffusion steps consistently enhance performance and that fewer steps achieve comparable performance to the AR model with lower latency. A comparison of latency between our model and BART for sequence length is shown in Appendix F, demonstrating the growing speed advantage of dif-

Models	MSNews			MSQG		
	R-1	R-2	R-L	R-L	B-4	MT
CTC loss	**50.55**	**35.15**	**48.04**	**39.00**	8.09	**20.56**
-w/o two-step loss	48.69	33.71	45.98	36.82	6.62	19.86
-w/o scheduling p_k	40.17	28.14	39.34	26.26	1.58	10.02
Cross-entropy loss	50.14	34.70	47.58	38.96	**8.49**	20.30
-w/o two-step loss	50.12	34.69	47.55	38.82	8.31	20.29
-w/o scheduling p_k	49.35	34.06	47.04	30.16	3.14	13.61

Table 3: Ablation study of the proposed components on cross-entropy loss and CTC loss. "w/o two-step loss" means that the model always receives input sequences that are noised from gold tokens during training. "w/o scheduling p_k" means that while two-step loss is used, p_k is always set to 1

fusion models as target text length increases.

6 Conclusion

In this study, we aim to addressed the discrepancy between training and inference in discrete diffusion models and propose two approaches: two-step loss and scheduling for adoption of gold or predicted tokens. Experimental results show that the proposed model outperforms or is comparable to existing diffusion models, and in some settings, it demonstrates higher performance than AR decoding methods. The ablation study showed that both of the proposed ideas contribute to the performance improvement.

Limitation

This study proposed a new learning method for discrete diffusion models that bridges the gap between training and inference. One important limitation is that the proposed learning method has not been applied to pre-training on unlabeled text, and it would be valuable to explore effective self-supervised learning methods for diffusion models. Additionally, the proposed model uses a smaller PLM, BART-base as its backbone, and it would be worthwhile to investigate whether performance improves when scaling up the model size.

Ethical Considerations

The discrete diffusion model proposed in this study has not undergone pre-training on large-scale unlabeled text datasets. The training data is limited to relatively small benchmark datasets, which were selected solely for the purpose of evaluating the performance of the model architecture. This fine-tuned model is not intended for direct application to other problem settings. Therefore, we believe there is no risk of harmful, hateful, or biased knowledge being disseminated in the real world through this proposed model.

Acknowledgments

This paper is based on results obtained from a project JPNP20006, commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

References

Masaki Asada and Makoto Miwa. 2023. BioNART: A biomedical non-AutoRegressive transformer for natural language generation. In *The 22nd Workshop on Biomedical Natural Language Processing and BioNLP Shared Tasks*, pages 369–376, Toronto, Canada. Association for Computational Linguistics.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. 2021. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, volume 34, pages 17981–17993. Curran Associates, Inc.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. 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 6112–6121, Hong Kong, China. Association for Computational Linguistics.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data

with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor OK Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *International Conference on Learning Representations*.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Zhengfu He, Tianxiang Sun, Qiong Tang, Kuanning Wang, Xuanjing Huang, and Xipeng Qiu. 2023. DiffusionBERT: Improving generative masked language models with diffusion models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4521–4534, Toronto, Canada. Association for Computational Linguistics.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851.

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.

Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2022a. ELMER: A non-autoregressive pre-trained language model for efficient and effective text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1044–1058, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. 2022b. Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems*, 35:4328–4343.

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.

Zhenghao Lin, Yeyun Gong, Yelong Shen, Tong Wu, Zhihao Fan, Chen Lin, Nan Duan, and Weizhu Chen. 2023. Text generation with diffusion language models: A pre-training approach with continuous paragraph denoise. In *International Conference on Machine Learning*, pages 21051–21064. 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, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Weizhen Qi, Yeyun Gong, Jian Jiao, Yu Yan, Weizhu Chen, Dayiheng Liu, Kewen Tang, Houqiang Li, Jiusheng Chen, Ruofei Zhang, Ming Zhou, and Nan Duan. 2021. Bang: Bridging autoregressive and non-autoregressive generation with large scale pretraining. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8630–8639. PMLR.

Weizhen Qi, Yu Yan, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. ProphetNet: Predicting future n-gram for sequence-to-SequencePre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2401–2410, Online. Association for Computational Linguistics.

Lihua Qian, Mingxuan Wang, Yang Liu, and Hao Zhou. 2024. Diffusion glancing transformer for parallel sequence-to-sequence learning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4846–4862, Mexico City, Mexico. Association for Computational Linguistics.

Mohammad Golam Sohrab, Masaki Asada, Matīss Rikters, and Makoto Miwa. 2024. Bert-nar-bert: A non-autoregressive pre-trained sequence-to-sequence model leveraging bert checkpoints. *IEEE Access*, 12:23–33.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021. Denoising diffusion implicit models. In *International Conference on Learning Representations*.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: Masked sequence to sequence pre-training for language generation. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5926–5936. PMLR.

Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. 2019. Insertion transformer: Flexible sequence generation via insertion operations. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5976–5985. PMLR.

Zecheng Tang, Pinzheng Wang, Keyan Zhou, Juntao Li, Ziqiang Cao, and Min Zhang. 2023. Can diffusion model achieve better performance in text generation ? bridging the gap between training and inference ! In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 11359–11386, Toronto, Canada. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Tong Wu, Zhihao Fan, Xiao Liu, Hai-Tao Zheng, Yeyun Gong, Jian Jiao, Juntao Li, Jian Guo, Nan Duan, Weizhu Chen, et al. 2023. Ar-diffusion: Auto-regressive diffusion model for text generation. *Advances in Neural Information Processing Systems*, 36:39957–39974.

Wen Zhang, Yang Feng, Fandong Meng, Di You, and Qun Liu. 2019. Bridging the gap between training and inference for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4334–4343, Florence, Italy. Association for Computational Linguistics.

Kun Zhou, Yifan Li, Xin Zhao, and Ji-Rong Wen. 2024. Diffusion-NAT: Self-prompting discrete diffusion for non-autoregressive text generation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1438–1451, St. Julian's, Malta. Association for Computational Linguistics.

A Connectionist Temporal Classification Loss

CTC was originally proposed for labeling unsegmented sequences. It learns monotonic alignment between acoustic features and transcriptions, which is valid for cross-modal learning like automatic speech recognition (ASR). CTC helps convergence and allows re-scoring decoding through a lightweight output layer, achieving great success in ASR.

An important characteristic of CTC is the addition of a blank label to the output label types and the following reduction process is applied to the model output sequences, where blank label is represented by "-".

- First, consecutive occurrences of the same label in the sequences are condensed into a single occurrence (e.g., `aa-aaa-bb-` $\rightarrow$ `a-a-b-`).

- Next, blank labels are removed (e.g., `a-a-b-` $\rightarrow$ `aab`).

Task	Dataset	#.Train	#.Valid	#.Test
Sum.	XSUM	204,045	11,332	11,334
	MSNews	136,082	7,496	7,562
QG	SQUAD v1.1	75,722	10,570	11,877
	MSQG	198,058	11,008	11,022

Table 4: Statistics of the four datasetes. Sum. and QG stand for text summarization and question generation, respectively.

Given the input sequence X and the corresponding target sequence Y, the CTC loss is defined as:

$$\mathcal{L}_{\text{CTC}} = -\log P_{\text{CTC}}(Y|X) \tag{10}$$

where the probability is calculated by marginalizing over all possible alignments $\Phi(Y)$ between X and Y:

$$P_{\text{CTC}}(Y|X) = \sum_{\pi \in \Phi(y)} P(\pi|X). \tag{11}$$

CTC has the same conditional independence property as NAR generation, where the probability of the path π is the product of the probability $P(\pi_t|x_i)$ at each position i:

$$P(Y|X) \approx \sum_{i=1}^{m} P(\pi_i|x_i), \tag{12}$$

where m is the length of X.

B Data Settings

We describe the data and task settings of the benchmark tasks including text summarization and question generation tasks.

- Text summarization aims to produce a short version of a document while preserving its salient information content. We evaluate our model on the BBC extreme summarization (XSum) dataset and MSNews dataset. The evaluation metric is ROUGE (Lin, 2004), including ROUGE-1, ROUGE-2, and ROUGE-L.

- Question generation aims to generate questions based on given passages and answers. We chose SQUAD v1.1 and MSQG datasets. The evaluation metrics are BLEU-4 (Papineni et al., 2002), ROUGE-L, and METEOR (Banerjee and Lavie, 2005).

The statistics of these datasets are shown in Table 4. We adopted the same data-splitting settings and the

		Ours							BART
#. diffusion steps		2	20	100	200	300	400	1000	-
latency (ms)↓		11.12	24.45	83.61	158.05	258.59	377.92	751.87	220.50
MSQG	R-L↑	33.02	37.54	38.52	39.00	39.30	39.34	39.81	38.1
	B-4↑	4.24	6.96	7.62	8.09	8.38	8.50	8.78	10.2
	MT↑	14.77	19.29	20.13	20.56	20.78	20.89	21.19	22.1

Table 5: The relation between diffusion inference steps and performance, latency, with AR models. Latency refers to the time required to generate one sample.

implementation of those all metrics as the existing studies (Li et al., 2022a; Zhou et al., 2024), and we show the reported scores in (Zhou et al., 2024) for all baselines to make a fair comparison.

C Model Settings

The probabilities for $\mathbf{Q}_t$ that control noise addition are determined by the linear scheduler (Ho et al., 2020). The probability γ_t increases linearly from 0 to 0.005 over T steps, while β_t is set to a constant value of 0.1. The probability p_k for scheduling the multi-step loss is set as follows:

$$p_k = \frac{k}{K}. \tag{13}$$

We employ AdamW as the optimizer with a learning rate of 5e-5. For the SQuAD v1.1 and MSQG datasets, we set the training steps to 10,000, and for XSum and MSNews datasets, we use 80,000 and 20,000 steps, respectively. The total batch size is set to 512 across all datasets. In the comparison of latency, the source text was padded to a length of 512. For our model, we set the target sequence length to 128, meaning that the diffusion process started from step $t = T$ with 128 mask tokens. For a fair comparison, we set the generation maximum sequence length of BART to 128 and the batch size is standardized at 16 for all models. One single NVIDIA V100 GPU is used for the latency calculation.

D Baselines Details

For AR generation, we experiment with a vanilla Transformer model and three PLMs: Transformer (Vaswani et al., 2017), an AR generation model without pre-training. MASS (Song et al., 2019) , BART (Lewis et al., 2020), and ProphetNet (Qi et al., 2020): Three representative PLMs for AR text generation, with pre-training objectives ranging from denoising text to future n-gram prediction.

For NAR and Semi-NAR generation, we compared six models with varying decoding strategies: NAT (Gu et al., 2018): The first proposed NAR text generation model. This model adds a module in the encoder to predict fertilities, serving as a global plan for parallel generation. InsT (Stern et al., 2019): A Semi-NAR text generation model leveraging insertion operations, repeatedly inserting tokens at multiple locations based on the partially inserted sequence. iNAT (Lee et al., 2018), CMLM (Ghazvininejad et al., 2019), LevT (Gu et al., 2019), and BANG (Qi et al., 2021): These four baselines are both NAR and Semi-NAR text generation models. ELMER (Li et al., 2022a) adopts Transformer-based pre-training techniques to further enhance NAR generation performance. BERT-nar-BERT (BnB) (Sohrab et al., 2024) leverages the existing encoder-only model checkpoints for NAR decoding.

For diffusion models, we listed three models: GENIE (Lin et al., 2023) and AR-Diffusion (AR-Diff) (Wu et al., 2023) incorporate pre-training strategies and autoregressive decoding to enhance the generation performance of continuous diffusion models. Diffusion-NAT (Diff-NAT) (Zhou et al., 2024): This is a discrete diffusion model that leverages the capabilities of existing PLMs and proposes a self-prompting approach that recalculates the encoder output at each step during both training and inference, concatenating the output from the previous step to the input of the encoder. This approach improved the quality of generated text at the expense of slower decoding speed.

E Analysis on the Number of Diffusion Steps

Table 5 shows the results of measuring latency and performance on the MSQG dataset when increasing the number of diffusion inference steps. As the number of inference steps increases, we can see that the latency also increases roughly linearly. When compared to BART, a representative model of AR decoding, we found that the latency is approximately equivalent to BART when the number of inference steps is set to 300. Regard-

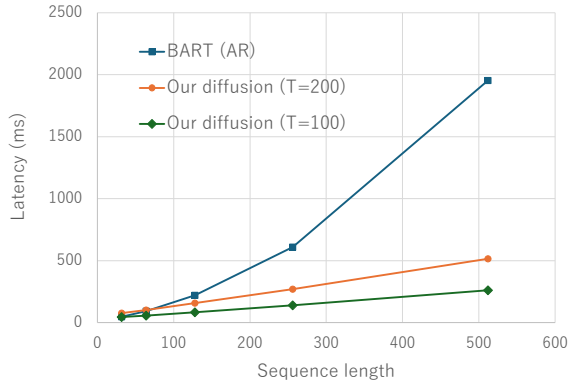

Figure 3: The generation length and latency on BART and our diffusion model. T stands for the number of diffusion steps for inference.

ing performance, we can see that scores improve as the number of inference steps increases, but the improvement becomes more gradual after 100 steps. Considering the balance between performance and speed, we finally adopted 200 inference steps, which has the best performance among the variants capable of faster inference than BART.

F Latency w.r.t Generation Length

Figure 3 shows the latency during inference relative to the maximum length of the target text. This figure shows that while the BART model dramatically slows down as it generates longer sentences, the diffusion model maintains a more consistent speed. This is because even as the maximum sequence length increases, the decoding performed at each diffusion step is done in a NAR manner.