Empowering Diffusion Models on the Embedding Space for Text Generation

Zhujin Gao[1,2*], **Junliang Guo**[3*], **Xu Tan**[3], **Yongxin Zhu**[1,2], **Fang Zhang**[1,2], **Jiang Bian**[3], **Linli Xu**[1,2†]

[1]School of Computer Science and Technology, University of Science and Technology of China
[2]State Key Laboratory of Cognitive Intelligence
[3]Microsoft Research Asia

gaozhujin@mail.ustc.edu.cn, {junliangguo, xuta}@microsoft.com
{zyx2016, fangzhang}@mail.ustc.edu.cn, jiabia@microsoft.com
linlixu@ustc.edu.cn

Abstract

Diffusion models have achieved state-of-the-art synthesis quality on both visual and audio tasks, and recent works further adapt them to textual data by diffusing on the embedding space. In this paper, we conduct systematic studies of the optimization challenges encountered with both the embedding space and the denoising model, which have not been carefully explored. Firstly, the data distribution is learnable for embeddings, which may lead to the collapse of the embedding space and unstable training. To alleviate this problem, we propose a new objective called the anchor loss which is more efficient than previous methods. Secondly, we find the noise levels of conventional schedules are insufficient for training a desirable denoising model while introducing varying degrees of degeneration in consequence. To address this challenge, we propose a novel framework called noise rescaling. Based on the above analysis, we propose Differmer, an embedding diffusion model based on Transformer. Experiments on varieties of seminal text generation tasks show the effectiveness of the proposed methods and the superiority of Differmer over previous state-of-the-art embedding diffusion baselines.[1]

1 Introduction

A wave of diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2020b) is sweeping the generation tasks (*e.g.*, image and audio synthesis) recently, showing their great capacity for high-quality data generation. Diffusion models are a family of iterative generative models, which are trained to recover corrupted data and then generate data by gradually refining samples from the pure noise. This procedure enables the model to make subtle refinements of output samples in a multi-step denoising process, and thus generate high-fidelity and diverse samples (Dhariwal and Nichol, 2021; Nichol and Dhariwal, 2021; Ho and Salimans, 2021; Rombach et al., 2022; Chen et al., 2020; Kong et al., 2020).

The booming achievements in vision and audio domains inspire researchers to delve into the realm of text generation. Diffusion models introduce a novel noising paradigm and a training objective other than token prediction, establishing an alternative form of language models, which exhibits the potential to foster an enhanced comprehension of language modeling. From a higher perspective, this investigation generalizes the diffusion model across modalities, and further contributes to a unified multimodal framework (Bao et al., 2023; Tang et al., 2024). Nonetheless, the exploration is still at an initial stage. Recent works (Li et al., 2022; Gong et al., 2022; Strudel et al., 2022) basically convert the discrete tokens to embeddings and then utilize continuous diffusion models to generate them, which can be termed embedding diffusion models. These preliminary attempts follow the original model to deal with the embeddings, with little consideration of the unique properties and the optimization challenges of the embedding space and the denoising model.

In this paper, we explore the embedding diffusion model from two perspectives separately, *i.e.*, the embedding space and the denoising model, based on which we conduct a thorough study respectively. Firstly, for diffusion models on image and audio generation, the ground truth data is stationary during training. In contrast, it is learnable for the textual data (*i.e.*, embeddings), which may cause the collapse of the embedding space and introduce instability to the training of the model. To avoid the collapse caused by dynamically shifting embedding parameters, we propose an anchor loss function to attain well-distributed embeddings and stabilize the training process. The detailed analysis

[*]Equal contribution.
[†]Corresponding author.
[1]Code is available at https://github.com/zhjgao/differmer

Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 4664–4683

is presented in Section 3.1.

Secondly, in Section 3.2, we find that in the high dimensional embedding space, the insufficient noise results in a simple denoising task, which causes the degeneration of the model. To tackle this challenge, we propose a novel framework named noise rescaling, which is orthogonal to the choice of the noise schedule and applicable to any existing schedules. Specifically, we define an index termed degeneration score as a measurement of the degree of degeneration. Guided by the degeneration score, we can apply a noise rescaling procedure to prevent the model from degenerating.

Based on the above discussion, we propose an integrated framework of Difformer, a denoising diffusion Transformer model. We conduct experiments on a variety of important text generation tasks including machine translation, text summarization, paraphrasing, text simplification, and question generation. On these benchmark datasets, Difformer outperforms diffusion-based and iteration-based non-autoregressive baselines and achieves state-of-the-art performance among embedding diffusion models. Further experiments demonstrate the superiority of Difformer over baselines including LLMs in quality, diversity, and efficiency, emphasizing the potential of diffusion models for text generation in the era of LLMs.

2 Background

Diffusion Models Denoising diffusion probabilistic models (Sohl-Dickstein et al., 2015; Ho et al., 2020) utilize a forward process to perturb the data with Gaussian noise, and a reverse process to restore the data symmetrically. Ho et al. (2020) develop the approach by specific parameterizations, achieving comparable sample quality with state-of-the-art generative models such as GANs (Goodfellow Ian et al., 2014). After that, great improvements have been made by many following works (Song et al., 2020a; Dhariwal and Nichol, 2021; Nichol and Dhariwal, 2021; Rombach et al., 2022) both in quality and efficiency. Given a data sample $\mathbf{z}_0 \in \mathbb{R}^d$, the denoising diffusion probabilistic model gradually perturbs it into a pure Gaussian noise $\mathbf{z}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ through a series of latent variables $\mathbf{z}_1, \cdots, \mathbf{z}_T$ in the forward process:

$$q(\mathbf{z}_t|\mathbf{z}_0) = \mathcal{N}\left(\mathbf{z}_t; \sqrt{\bar{\alpha}_t}\mathbf{z}_0, \bar{\beta}_t\mathbf{I}\right),$$

where $\bar{\alpha}_t, \bar{\beta}_t$ are hyper-parameters controlling the noise level added at timestep t, which form the noise schedule. Usually, these hyper-parameters are set to satisfy $\bar{\alpha}_t := \prod_{i=0}^{t} \alpha_i$, $\alpha_t + \beta_t = 1$, and $\bar{\alpha}_t + \bar{\beta}_t = 1$. The reverse process is parameterized as:

$$p_\theta(\mathbf{z}_{t-1}|\mathbf{z}_t) = \mathcal{N}\left(\mathbf{z}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{z}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{z}_t, t)\right),$$

where $\boldsymbol{\mu}_\theta(\cdot)$ and $\boldsymbol{\Sigma}_\theta(\cdot)$ are the predicted mean and covariance of $q(\mathbf{z}_{t-1}|\mathbf{z}_t)$, and θ denotes the model parameters. After parameterization, we utilize a simplified variational lower-bound as the objective function

$$\mathcal{L}_{\text{vlb}} = \mathbb{E}_{\mathbf{z}_0, \mathbf{z}_t, t}\left[\|\hat{\mathbf{z}}_0(\mathbf{z}_t, t) - \mathbf{z}_0\|^2\right], \quad (1)$$

where $\hat{\mathbf{z}}_0(\mathbf{z}_t, t)$ is the model prediction of the original data $\mathbf{z}_0$ given $\mathbf{z}_t$. The detailed derivation can be found in Appendix B.

Diffusion Models for Text Generation The breakthrough of diffusion models on continuous data encourages people to explore their potential on discrete textual data. The definition of forward and reverse processes is the key question for diffusion models. Recent works mainly follow two directions.

Firstly, discrete diffusion models on categorical distributions are proposed (Hoogeboom et al., 2021; Austin et al., 2021; Savinov et al., 2021; Reid et al., 2022), by which sentences are corrupted and refined at the token level. However, these kinds of corruption are coarse-grained. Attempts have been made to explore modeling on surrogate representations of discrete data such as analog bits (Chen et al., 2022) and simplex (Han et al., 2023). Nevertheless, these representations carry little semantic information about tokens, which implies that the distances in this space can not accurately reflect semantic correlations between tokens.

In contrast, embedding diffusion models (Li et al., 2022; Strudel et al., 2022; Gong et al., 2022; Ye et al., 2023) introduce an additional embedding step and rounding step in the forward and reverse processes respectively. The embedding step converts tokens into learnable or pre-trained embeddings, which carry semantic information, and then a continuous diffusion process is able to add Gaussian noise to these embeddings, achieving a fine-grained noising procedure. Mathematically, given a sequence of tokens $\boldsymbol{y} = [y_1, y_2, \cdots, y_n]$, the embedding step can be denoted as $\mathbf{z}_0 \sim \mathcal{N}(\boldsymbol{e}_\phi(\boldsymbol{y}), \beta_0\mathbf{I})$ where $\boldsymbol{e}_\phi(\cdot)$ denotes the embedding lookup function. The rounding step turns predicted embeddings back to discrete tokens, which

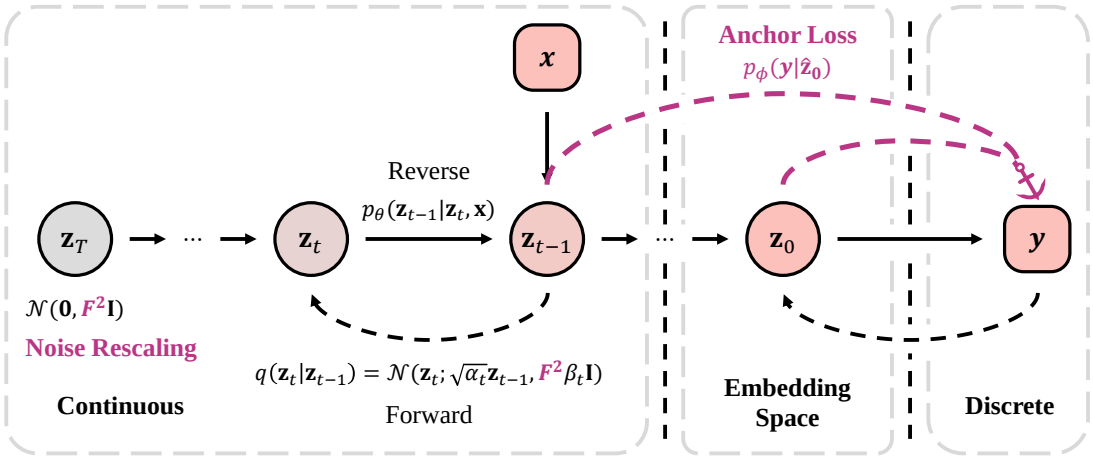


Figure 1: An overview of the Differmer, including the proposed techniques, *i.e.*, the anchor loss, and the noise rescaling.

can be expressed as a softmax distribution over the vocabulary $p_\phi(\boldsymbol{y}|\mathbf{z}_0)$, and is trained by an extra loss function $\mathcal{L}_{\text{round}} = \mathbb{E}_{\boldsymbol{y},\mathbf{z}_0}[-\log p_\phi(\boldsymbol{y}|\mathbf{z}_0)]$. The parameters of this step and the embedding step are tied. The final loss function is written as:

$$\mathcal{L}_{\text{text}} = \mathcal{L}_{\text{vlb}} + \mathcal{L}_{\text{round}}.$$

Nevertheless, these works directly adapt continuous diffusion models to embeddings, without considering the gap between the learnable embedding space and the stationary image or audio data, as well as the distinctive requirements of the denoising model established on the embedding space.

3 Methodology

This section elucidates the challenges inherent in optimizing embedding diffusion models and presents our corresponding solutions. We start with an introduction to the model architecture. The model architecture is based on Transformer (Vaswani et al., 2017), which consists of an encoder and a decoder. The decoder, as the main stem component, is considered as two separate parts in this paper, namely the embeddings $\boldsymbol{e}_\phi = [\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_V] \in \mathbb{R}^{d\times V}, \boldsymbol{e}_\phi(\boldsymbol{y}) = [\boldsymbol{e}_{y_1}, \boldsymbol{e}_{y_2}, \cdots, \boldsymbol{e}_{y_n}]$ and the denoising model $f_\theta(\cdot)$, which denotes the stacked decoder layers. Notably, this paper define $\mathbf{z}_0 = \boldsymbol{e}_\phi(\boldsymbol{y})$. The encoder provides the representation $\mathbf{x} = \text{Encoder}(\boldsymbol{x})$ of the condition sentence $\boldsymbol{x} = [x_1, x_2, \cdots, x_m]$.

3.1 Collapse of the Embedding Space

Analysis of the Collapse Problem The data space is usually fixed for continuous data (*e.g.*, image and audio), while it is learned from scratch for discrete textual data (*i.e.*, embeddings), which therefore shifts dynamically during training. Original diffusion models rely on the loss function Eq. (1) to learn to estimate the clean data sample $\mathbf{z}_0$. Nevertheless, when directly adapting this objective to the embedding diffusion model, the embedding space will collapse. As a result, the embeddings of different tokens will be less distinguishable and non-uniformly distributed in the space, which considerably limits the representation capacity and quality of the embeddings. On the contrary, the model could achieve better performance with more isotropic embeddings (Gao et al., 2018; Li et al., 2020).

Recent works of diffusion on embeddings (Li et al., 2022; Gong et al., 2022) introduce the rounding loss $\mathcal{L}_{\text{round}}$ from the derivation of the variational lower bound, which discriminates the correct embeddings from others given their noised counterparts, therefore enforces the embeddings are distinguishable and informative, alleviating the collapse objectively. We could regard this additional loss function as a regularization term for the embeddings. Nonetheless, only a minor level of perturbation is involved from $\boldsymbol{y}$ to $\mathbf{z}_0$, thereby the rounding loss is only able to apply a relatively weak constraint on the embeddings.

Our empirical evidence also corroborates the limitation of the rounding loss. We observe that the rounding loss undergoes a steep descent and falls to near zero in the initial stages of training, which implies the rounding loss can be effortlessly addressed and fails to conduct strong enough regularization to the embeddings. Therefore, the embedding space is undesirable and eventually leads to unsatisfac-

tory performance. Concurrently, the instability in training also emerges as a problem during training. Even if careful tuning of the hyper-parameters is performed to relieve anisotropy, the performance is still inferior.

Anchor Loss To emphasize the effect of the regularization term, we propose a training objective named the anchor loss

$$\mathcal{L}_{\text{anchor}} = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y}),\mathbf{z}_t,t}[-\log p_\phi(\boldsymbol{y}|\hat{\mathbf{z}}_0(\mathbf{z}_t,\mathbf{x},t))].$$

Compared with $\mathcal{L}_{\text{round}}$, $\mathcal{L}_{\text{anchor}}$ utilizes the model prediction of $\mathbf{z}_0$ as the input, which involves a large discrepancy with $\mathbf{z}_0$ due to the prediction error of the denoising model. Consequently, to ensure these highly noisy representations are identified as the correct tokens, the anchor loss employs a stronger regularization to the embeddings to prevent collapse. Additionally, besides $\mathcal{L}_{\text{vlb}}$, the anchor loss creates another pathway between the denoising model and the target sentences, through which the model could receive feedback from the ground truth, maintaining the training stability. Finally, our training objective is written as

$$\mathcal{L} = \mathcal{L}_{\text{vlb}} + \mathcal{L}_{\text{anchor}}. \tag{2}$$

Empirically, we use self similarity (Ethayarajh, 2019) as the *anisotropy score* to measure the severity of collapse:

$$\text{ANI} = \frac{1}{V(V-1)} \sum_{i=1}^{V} \sum_{j=1,j\neq i}^{V} \cos(\boldsymbol{e}_i, \boldsymbol{e}_j).$$

Essentially, the higher the anisotropy score is, the more severe the collapse is. The anisotropy score as well as the performance obtained by each loss function can be found in Table 1. With only $\mathcal{L}_{\text{vlb}}$ or $\mathcal{L}_{\text{text}}$, the anisotropy score demonstrates that the embeddings are non-uniformly distributed, resulting in unsatisfactory results. On the contrary, the embeddings are well-distributed across the entire space with the anchor loss, and thus the model reaches competitive performance (in BLEU (Papineni et al., 2002)). Alternatively, utilizing pretrained embeddings and freezing them during training could also avoid collapse. As shown in the experimental results, the frozen embeddings alleviate the collapse remarkably, however, they are suboptimal for the problem. Detailed discussion can be found in Appendix C.3.

Loss	ANI	BLEU
$\mathcal{L}_{\text{vlb}}$	0.99	0.07
$\mathcal{L}_{\text{text}}$	0.32	27.89
$\mathcal{L}$	0.03	34.48

Table 1: The anisotropy score and performance of each loss function on the IWSLT14 De-En dataset with *linear* schedule.

3.2 Degeneration of the Denoising Model

Analysis of the Degeneration Problem The design of the noise schedule, which determines the amount of noise added to the data at each step, has significant influences on both forward and reverse processes. Intuitively, denoising is a more challenging task for the model with higher levels of noise, and becomes easier when insufficient corruption is applied, where the model can generate the correct embeddings without depending on the condition and context. As a consequence, the model tends to degenerate to a trivial solution. Here, we provide in-depth analyses of this problem. We start by defining the degenerated model, which discards the conditioning information and generates each embedding by choosing the nearest ones independently:

Definition 1. *For a noised input* $\mathbf{z}_t = [\boldsymbol{z}_{t,1}, \boldsymbol{z}_{t,2}, \cdots, \boldsymbol{z}_{t,n}]$*, the* ***Degenerated Model*** *is defined as*

$$f_{\text{dg}}(\mathbf{z}_t;\mathbf{x}) = \left[\arg\min_{\boldsymbol{e}_y \in \boldsymbol{e}_\phi} \mathcal{L}(\boldsymbol{z}_{t,i}, \boldsymbol{e}_y)\right]_i^n,$$

where

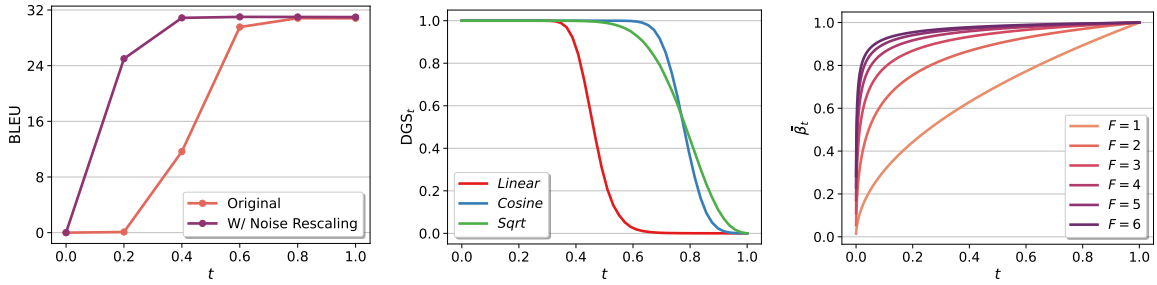
$$\mathcal{L}(\boldsymbol{z}_{t,i}, \boldsymbol{e}_y) = \|\boldsymbol{z}_{t,i} - \boldsymbol{e}_y\|^2 - \log p_\phi(y|\boldsymbol{z}_{t,i}).$$

It can be proved that when insufficient noise is introduced during training, the denoising model tends to fall as the degenerated model defined above.

Theorem 1. *Given embeddings* $\boldsymbol{e}_\phi \sim \mathcal{N}_{d\times V}(\mathbf{0}, \sigma_e\mathbf{I})$*, the probability of the degenerated model being a global minimum of the objective function* $\mathcal{L}$ *for* θ *converges to* 1 *as* $\bar{\beta} \to 0$ *and* $d \to \infty$*.*

We leave the proof and illustrations of this theorem in Appendix A.

This phenomenon could be verified quantitatively. To analyze the capacity of the denoising model at each noise level, we evaluate the BLEU score of $\hat{\mathbf{z}}_0$ generated by the model at different timesteps. To eliminate the impact of the noise



(a) (b) (c)

Figure 2: (a) BLEU score of models fed with pure Gaussian noise $\mathbf{z}_T$ on the IWSLT14 De-En dataset. The value of t is normalized to $[0, 1]$. (b) DGS_t with different widely used schedules. (c) The *sqrt* schedule rescaled with different values of the rescaling factor.

schedule, we feed the model with z_T, *i.e.*, the pure noise, rather than z_t. As illustrated in Fig. 2a, the BLEU score drops dramatically at ts with low noise levels, and the range with low scores occupies nearly half of the axis. In other words, the model degenerates significantly and extensively, implying the noise levels brought by the schedule are far from being sufficient.

Noise Rescaling In most text generation tasks, a degenerated model is undesirable, as it fails to maintain contextual coherence and condition consistency, both of which are crucial for the task. We propose a universal noise rescaling framework to alleviate the degeneration. To achieve this, we start by defining the degree of degeneration to guide the rescaling. Intuitively, as supported by the proof of Theorem 1, the tendency toward degeneration is highly related to the accuracy of the degenerated model. Thus we define a mathematical representation of the overall degree of degeneration:

Definition 2. *The **DeGeneration Score (DGS)** of a specific noise schedule is*

$$\mathrm{DGS}(\bar{\alpha}, \bar{\beta}) = \frac{1}{T} \int_0^T \mathrm{DGS}_t(\bar{\alpha}_t, \bar{\beta}_t)\, \mathrm{d}t,$$

where DGS_t *is the classification accuracy of the degenerated model at* t

$$\mathrm{DGS}_t = P(f_{\mathrm{dg}}(\boldsymbol{z}_t) = \boldsymbol{z}_0),$$

$\bar{\alpha}$ and $\bar{\beta}$ are functions or discrete series representing the noise schedule, and $\boldsymbol{z}_t \sim \mathcal{N}\left(\sqrt{\bar{\alpha}_t}\boldsymbol{z}_0, \bar{\beta}_t\mathbf{I}\right)$. Note that we do not set the constraint $\bar{\alpha}_t + \bar{\beta}_t = 1$ here.

Fig. 2b illustrates DGS_t of several widely utilized noise schedules, including *linear* (Ho et al., 2020), *cosine* (Nichol and Dhariwal, 2021) and *sqrt* (Li et al., 2022), with DGS listed in Table 2. From the table, we can notice that the schedule with a lower degeneration score yields a better BLEU score, which reflects the relationship between the degeneration score and the overall performance.

Schedules	DGS	BLEU
Linear	0.47	32.21
Cosine	0.77	26.61
Sqrt	0.77	22.70

Table 2: The degeneration score of several noise schedules proposed by prior works, and their performance on the IWSLT14 De-En dataset.

Based on the degeneration score, we first specify a threshold $\mathrm{DGS}_{\mathrm{MAX}}$ to the degeneration score, to impose a restriction on the noise added to the embeddings, under which we expect the model will not degenerate. Then we introduce a factor to the noise schedule named the rescaling factor, amplifying the noise added to the embeddings to ensure that the noise schedule satisfies the restriction imposed by $\mathrm{DGS}_{\mathrm{MAX}}$, which can be written as

$$\begin{cases} \bar{\alpha}'_t = \bar{\alpha}_t \\ \bar{\beta}'_t = F^2 \bar{\beta}_t \end{cases}, \tag{3}$$

where $\bar{\alpha}_t$ and $\bar{\beta}_t$ denote the original schedules, $\bar{\alpha}'_t$ and $\bar{\beta}'_t$ denote the schedule coefficients after rescaling, and F is the rescaling factor. Through experiments in Section 4.3, we find this simple but effective adjustment brings significant improvement. In Appendix D, we present an approach for searching F given a specific $\mathrm{DGS}_{\mathrm{MAX}}$, accompanied by a pre-computed function table to facilitate future research. Alternatively, we can also derive a variance-preserving (Song et al., 2020b) variant of the rescaling factor, which satisfies the constraint $\bar{\alpha}'_t + \bar{\beta}'_t = 1$ (details in Appendix C.8). In Fig. 2c

we show the shape of the *sqrt* schedule rescaled by different values of F. As demonstrated in Fig. 2a, with our rescaling technique, the degeneration of the denoising model experiences a substantial alleviation in both degree and occupancy.

3.3 Differmer

Based on the analysis of the challenges encountered with embedding diffusion models, we introduce Differmer, a denoising diffusion model with Transformer, with the proposed techniques including the anchor loss and the noise rescaling technique. An overview of the model is demonstrated in Fig. 1.

Length Prediction and 2D Parallel Decoding Unlike traditional autoregressive models where the sequence length is implicitly decided by the EOS token, diffusion models generate all tokens in a non-autoregressive manner, where the length should be modeled explicitly. Previous works (Li et al., 2022; Gong et al., 2022; Strudel et al., 2022) usually generate a sequence with the maximum length and cut off the content after the EOS token. In this paper, we utilize a more efficient way by explicitly predicting the target length with the encoder output (Lee et al., 2018), *i.e.*, $p_\theta(n|\mathbf{x})$, and a negative log-likelihood loss function is added to Eq. (2) while training.

A unique benefit of this approach is that we can conduct 2D parallel decoding in inference. Firstly, we can consider top-b_1 lengths from the length predictor to generate candidates with different lengths. Secondly, for each length, we can also generate b_2 candidates by sampling different initial noises from the prior. The final prediction is selected from the total $b = b_1 \times b_2$ candidates that minimize the expected risk (Kumar and Byrne, 2004) *w.r.t.* a metric such as BLEU or PPL. We term two kinds of beams as length beam and noise beam respectively. We conduct a study on the impact of b_1 and b_2 in Appendix C.6. In summary, both beams introduce improvements, and b_1 influences more.

Acceleration in Inference Diffusion models are trained with thousands of forward steps, but it would be extremely time-consuming to iterate all steps in inference. For Differmer, we pick a subset $\{\tau_1, \tau_2, \cdots, \tau_K\}$ of size K from the full diffusion trajectory $\{1, 2, \cdots, T\}$ for generation (Song et al., 2020a; Nichol and Dhariwal, 2021). Correspondingly, the generated sample should be drawn from $q(\mathbf{z}_{\tau_{i-1}}|\mathbf{z}_{\tau_i}, \hat{\mathbf{z}}_0(\mathbf{z}_{\tau_i}, \mathbf{x}, \tau_i))$. In consequence, the time complexity of generation is reduced from $\mathcal{O}(T)$ to $\mathcal{O}(K)$.

4 Experiments

4.1 Experimental Setup

To evaluate the proposed Differmer model, we conduct experiments on five conditional text generation tasks including machine translation, text summarization, paraphrasing, text simplification, and question generation.

Datasets For machine translation, mainly following previous works (Gu et al., 2018; Guo et al., 2019; Ghazvininejad et al., 2019), three benchmark datasets WMT14 En-De (Bojar et al., 2014), WMT16 En-Ro (Bojar et al., 2016) and IWSLT14 De-En (Cettolo et al., 2014) are inclued. For text summarization, experiments are conducted on Gigaword (Graff et al., 2003; Rush et al., 2015). In addition, following previous non-autoregressive text generation works (Gu et al., 2018, 2019; Ghazvininejad et al., 2019), for machine translation and text summarization tasks, we adopt sequence-level knowledge distillation (Kim and Rush, 2016) on the original training set to alleviate the multi-modality problem. For paraphrasing, text simplification, and question generation tasks, we mainly follow (Gong et al., 2022) to conduct experiments on Quora Question Pairs (QQP)[2], Wiki-Auto (Jiang et al., 2020) and Quasar-T (QT) (Dhingra et al., 2017) respectively. The data split of the above datasets can be found in Appendix E.

Metrics We report the tokenized BLEU and the SacreBLEU (Post, 2018) for machine translation tasks, and the ROUGE (Lin, 2004) for summarization. As for paraphrasing, text simplification, and question generation tasks, tokenized BLEU, ROUGE-L, and BERTScore (Zhang et al., 2019) are utilized.

Baselines We mainly compare our method with recent embedding diffusion models, including DiffuSeq (Gong et al., 2022), SeqDiffuSeq (Yuan et al., 2023), and DiNoiSer (Ye et al., 2023), which extend Diffusion-LM (Li et al., 2022) to the sequence-to-sequence scenario. We further compare to a recent score-based model CDCD (Dieleman et al., 2022). CMLM (Ghazvininejad et al., 2019) is also included, a non-autoregressive model with iterative

[2] https://www.kaggle.com/c/quora-question-pairs

Models	b	WMT14 En-De BLEU	WMT16 En-Ro BLEU	IWSLT14 De-En BLEU	Gigaword ROUGE-1/2/L
Transformer	1	26.37	32.76	32.62	36.78/17.79/34.10
Transformer	5	27.37	33.59	33.91	37.54/18.80/34.93
CMLM	1	26.56*	32.75*	26.41	34.41/15.61/32.17
CMLM	5	27.03*	33.08*	31.76	36.33/17.82/33.83
DiffuSeq	1	13.73	23.37	27.03	28.50/10.10/26.00
DiffuSeq	10	15.37	25.45	28.78	31.17/12.23/29.24
SeqDiffuSeq	1	23.63†	23.98	28.65	30.28/11.72/28.40
SeqDiffuSeq	10	24.24†	26.17	30.03	31.90/12.36/29.22
DiNoiSer	5	26.08‡	32.57‡	32.23‡	-
DiNoiSer	50	26.29‡	32.59‡	32.48‡	-
Differmer	1	26.74⇑	32.52⇑	32.91⇑	35.45/16.46/32.87⇑
Differmer	10	27.70⇑	33.18⇑	**34.48**⇑	37.12/18.25/34.60⇑
Differmer	20	**27.74**	**33.36**	**34.48**	**37.64/18.75/35.01**

Table 3: The performance of the proposed Differmer and the baseline methods. *, † and ‡ indicate results reported by Ghazvininejad et al. (2019), Yuan et al. (2023) and Ye et al. (2023) respectively. Other results are from our implementation. ⇑ indicates that Differmer outperforms all diffusion-based baselines with the same beam size b.

Models	b	QQP			Wiki-Auto			QT		
		B	R-L	BS	B	R-L	BS	B	R-L	BS
Transformer	1	29.65	59.88	84.28	41.68	58.15	81.40	16.83	35.87	63.97
Transformer	5	30.83	61.20	85.29	43.86	58.48	81.71	16.45	35.59	63.91
DiffuSeq	10	24.13	58.80	83.65	36.22	58.49	81.26	17.31	**36.65**	61.23
SeqDiffuSeq	1	23.28	-	82.91	37.09	-	82.11	**17.20**	-	61.35
SeqDiffuSeq	10	24.34	-	84.00	37.12	-	82.14	17.46	-	61.74
DiNoiSer	10	26.07	-	-	35.36	-	-	-	-	-
DiNoiSer	20	25.42	-	-	36.94	-	-	-	-	-
Differmer	1	28.52⇑	60.15	83.80⇑	40.37⇑	59.56	81.96	16.03	35.06	61.05
Differmer	10	30.43⇑	**61.25**⇑	**85.02**⇑	40.77⇑	59.86⇑	82.21⇑	16.66	36.15	63.29⇑
Differmer	20	**30.52**⇑	61.08	**85.02**	**40.84**⇑	**59.88**	**82.29**	16.88	36.28	**63.32**

Table 4: The performance of the proposed Differmer on the QQP, Wiki-Auto, and QT datasets. B, R-L, and BS stand for the BLEU, ROUGE-L, and BERTScore respectively. The results of DiffuSeq, SeqDiffuSeq, and DiNoiSe are from their paper. ⇑ indicates that Differmer outperforms all diffusion-based baselines with the same beam size b.

decoding, which can be considered as a discrete diffusion model (Austin et al., 2021). In addition, we report the performance of Transformer as the autoregressive baseline.

Implementation Details We set diffusion step $T = 2000$, embedding dimension $d = 128$, the threshold of the degeneration score $\mathrm{DGS}_{\mathrm{MAX}} = 0.15$, and use the *sqrt* noise schedule. Following previous works (Li et al., 2022; Strudel et al., 2022), we also utilize the self-conditioning (Chen et al., 2022) which is shown effective in improving the final performance. More details of experiment settings can be found in Appendix E.

4.2 Results

The main results are listed in Tables 3 and 4. With a little abuse of notation, we use b to represent the size of beam search for the Transformer baseline, as well as the size of parallel decoding (*i.e.*, $b = b_1 \times b_2$). As can be observed from experimental results, the proposed Differmer outperforms both the diffusion-based and iteration-based non-autoregressive baselines on most of the datasets with different choices of b, and even performs comparably with the autoregressive Transformer model. Specifically, the significant improvements of Differmer over the diffusion baselines confirm the challenges that occur to embedding diffusion models for text generation tasks and the effectiveness of the proposed solutions. Compared with CMLM, an iteration-based non-autoregressive baseline, Differmer outperforms on various datasets consistently. Moreover, benefiting from the stochastic nature of diffusion models, Differmer is able to conduct 2D parallel decoding over the length and noise beam at the same time, increasing its flexibil-

Models	b	SacreBLEU
CDCD	1	19.30
CDCD	10	19.70
SeqDiffuSeq	1	19.16
SeqDiffuSeq	10	19.76
DiNoiSer	5	24.25
DiNoiSer	50	24.62
Differmer	1	22.80
Differmer	10	24.10
Differmer	50	**24.90**

Table 5: SacreBLEU scores on the raw WMT14 En-De dataset. The results of the baselines are as reported in their paper.

Models	b	SacreBLEU
DiNoiSer	5	25.70
DiNoiSer	50	25.90
Differmer	10	**26.20**

Table 6: SacreBLEU scores on the distilled WMT14 En-De dataset. The results of DiNoiSer are as reported in their paper.

ity and potential to obtain better results. We further compare with baselines on the raw and distilled WMT14 En-De training set with SacreBLEU[3] as the metric. As shown in Table 5 and Table 6, the proposed Differmer achieves better results with the same number of b. Due to the page limit, we leave results of more metrics and baselines in Appendices C.1 and C.2, which also validate the superiority of Differmer.

4.3 Analyses

Ablation Study We study the effects of the proposed components, which are listed in Table 7. Firstly, while previous embedding diffusion works usually utilize the rounding loss function, we find it does not provide satisfactory results, which echoes our findings in Section 3.1. By replacing $\mathcal{L}_{\text{round}}$ with $\mathcal{L}_{\text{anchor}}$, the problem is largely alleviated with significant performance improvements. The enhancement from noise rescaling also reinforces our findings in Section 3.2 that the model suffers from a degeneration problem. Besides, the integration of the anchor loss and noise rescaling yields the best performance.

Inference Speed Continuous diffusion models usually rely on hundreds or thousands of reverse steps in inference to guarantee the quality of the generated samples (Li et al., 2022; Gong et al., 2022; Strudel et al., 2022). In contrast, we find that Differmer is able to achieve considerably good performance with much fewer reverse steps. In Table 8, we evaluate the BLEU score and inference speed by varying the number of reverse steps. The conclusions are two-fold. Firstly, Differmer performs robustly *w.r.t.* K, especially compared with diffusion-based baselines. We attribute this advantage to the anchor loss, as it facilitates learning a well-distributed embedding space, and connects $\hat{\mathbf{z}}_0$ with solid ground truth labels, which reduce the obscurity of predictions. Correspondingly, the inference speed of Differmer outperforms the autoregressive model Transformer by 6 times and the iterative non-autoregressive model CMLM by 3 times when K is small, showing the potential of deploying Differmer to online systems.

Diversity To conduct a comprehensive evaluation of Differmer, we incorporate 4-gram diversity (div-4) (Deshpande et al., 2019) as the diversity metric on the QQP dataset, and expand our comparison with LLMs, including GPT-2 (Radford et al., 2019) and GPVAE-T5 (Du et al., 2022). According to Fig. 3, we observe that the adjustment

Anchor Loss	Noise Rescaling	BLEU
		16.96
✓		22.70
	✓	27.89
✓	✓	**34.48**

Table 7: The ablation study on the proposed components. Results are conducted on the IWSLT14 De-En dataset with $b = 10$.

Models	K	Speed	BLEU
Transformer	n	6.05	33.91
CMLM	10	11.80	31.76
DiffuSeq	2000	0.06	28.78
DiffuSeq	1000	0.12	23.91
DiffuSeq	500	0.23	0.96
SeqDiffuSeq	2000	0.05	30.03
Differmer	2000	0.03	34.09
Differmer	20	6.30	34.19
Differmer	10	11.40	34.13
Differmer	1	39.51	30.14

Table 8: The inference speed and corresponding performance of the proposed Differmer and the baselines. The speed is represented as sentences per second. Results are conducted on the IWSLT14 De-En with $b = 10$ and batch size $= 1$, without early stopping (see Section 4.3).

[3]The signature is `nrefs:1|case:mixed|eff:no|tok:13a|smooth:exp|version:2.2.0`

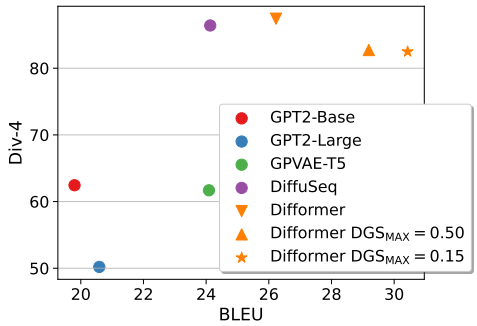


Figure 3: Comparison of generation quality and diversity of baselines and Differmer on the QQP dataset. Results of baselines are from (Gong et al., 2022).

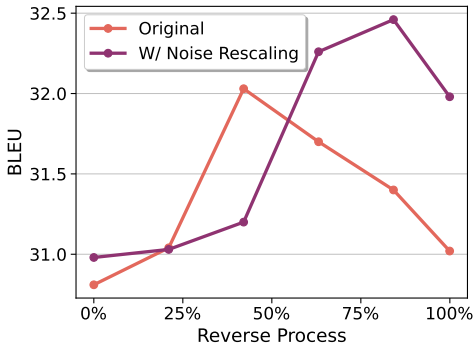


Figure 4: The intermediate BLEU score of $\hat{\mathbf{z}}_0$ within a decoding process.

DGS$_{MAX}$	**Rescaling Factor**	**BLEU**
0.05	7.0	33.88
0.15	4.0	34.48
0.50	2.0	32.54
0.77	1.0	22.70

Table 9: The performance with varying DGS$_{MAX}$. The last row represents the model without noise rescaling. Results are obtained on the IWSLT14 De-En dataset with $b = 10$.

of DGS$_{MAX}$ introduces a trade-off between generation quality and diversity. Specifically, the model trained with a smaller DGS$_{MAX}$ manifests reduced diversity but improved quality. We assume that this is attributed to the stronger restrictions imposed by this DGS$_{MAX}$ leading to a higher determinacy. Notably, under the condition DGS$_{MAX}$ $= 0.15$ and without noising rescaling, Differmer demonstrates superior performance over the baselines including LLMs in terms of quality and diversity respectively.

Generation Quality Dynamics To investigate the dynamics of generation quality during the reverse process, we extract $\hat{\mathbf{z}}_0$ at intermediate reverse steps and evaluate their BLEU scores. The results are illustrated in Fig. 4. For the original noise, in the first third of the reverse process, the generation quality continuously increases as expected. However, there is a noticeable performance decline in the latter part, where the model degenerates gradually. When noise rescaling is applied, a notable improvement can be observed, manifesting that the degeneration problem is alleviated as discussed in Section 3.2. This observation also motivates us to propose an early stopping technique, terminating the decoding process at a proper intermediate step to retain a high-quality output.

Threshold of the Degeneration Score We further analyze the influence of different values of DGS$_{MAX}$ in Table 9. From the table, we can notice that, as DGS$_{MAX}$ decreases, the BLEU score increases at first, and declines afterward. The former is reasonable since a small DGS$_{MAX}$ mitigates the degeneration problem. However, if DGS$_{MAX}$ is too small, a large rescaling factor causes the noise schedule to become a constant, as illustrated in Fig. 2c. Therefore, the multistep denoising process degenerates into a one-step process. In conclusion, a moderate DGS$_{MAX}$ is sufficient to maintain a balance between the degeneration of the model and the denoising process. More results with schedules other than *sqrt* can be found in Appendix C.4.

5 Conclusion

In this paper, we conduct a thorough study of the challenges when optimizing an embedding diffusion model on discrete textual data, and propose the corresponding solutions. Firstly, to tackle the challenge of embedding collapse and instability in training caused by the dynamic nature of embeddings, we introduce an anchor loss to regularize the embeddings and stabilize the training simultaneously. Secondly, we derive a novel noise rescaling framework based on theoretical analysis, which notably alleviates the degeneration of the denoising model resulting from inadequate noise. Finally, integrated with the aforementioned techniques, we present Differmer, a denoising diffusion model based on Transformer. Differmer demonstrates superior performance on various benchmark text generation tasks, outperforming prior diffusion-based models as well as iterative non-autoregressive models.

Limitations

The improvement brought by the proposed techniques is promising. However, embedding diffu-

sion models converge relatively slowly in training. For instance, compared with CMLM, Differmer requires around double the training time to reach convergence, although it is more efficient than prior diffusion models. Moreover, due to the cost of the search for the rescaling factor, it is performed offline and the factor is static during training.

Acknowledgements

This research was supported by the National Natural Science Foundation of China (Grant No. 62276245).

References

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

Fan Bao, Shen Nie, Kaiwen Xue, Chongxuan Li, Shi Pu, Yaole Wang, Gang Yue, Yue Cao, Hang Su, and Jun Zhu. 2023. One transformer fits all distributions in multi-modal diffusion at scale. In *International Conference on Machine Learning*, pages 1692–1717. PMLR.

Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, et al. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the ninth workshop on statistical machine translation*, pages 12–58.

Ondrej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, et al. 2016. Findings of the 2016 conference on machine translation (wmt16). In *First conference on machine translation*, pages 131–198. Association for Computational Linguistics.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th iwslt evaluation campaign. In *Proceedings of the 11th International Workshop on Spoken Language Translation: Evaluation Campaign*, pages 2–17.

Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. 2020. Wavegrad: Estimating gradients for waveform generation. In *International Conference on Learning Representations*.

Ting Chen, Ruixiang ZHANG, and Geoffrey Hinton. 2022. Analog bits: Generating discrete data using diffusion models with self-conditioning. In *The Eleventh International Conference on Learning Representations*.

Aditya Deshpande, Jyoti Aneja, Liwei Wang, Alexander G Schwing, and David Forsyth. 2019. Fast, diverse and accurate image captioning guided by part-of-speech. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10695–10704.

Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794.

Bhuwan Dhingra, Kathryn Mazaitis, and William W. Cohen. 2017. Quasar: Datasets for question answering by search and reading.

Sander Dieleman, Laurent Sartran, Arman Roshannai, Nikolay Savinov, Yaroslav Ganin, Pierre H. Richemond, Arnaud Doucet, Robin Strudel, Chris Dyer, Conor Durkan, Curtis Hawthorne, Rémi Leblond, Will Grathwohl, and Jonas Adler. 2022. Continuous diffusion for categorical data.

Wanyu Du, Jianqiao Zhao, Liwei Wang, and Yangfeng Ji. 2022. Diverse text generation via variational encoder-decoder models with gaussian process priors. *Proceedings of 6th Workshop on Structured Prediction for NLP of the Association for Computational Linguistics*.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. 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 55–65.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tieyan Liu. 2018. Representation degeneration problem in training natural language generation models. In *International Conference on Learning Representations*.

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.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. 2022. Diffuseq: Sequence to sequence text generation with diffusion models. In *The Eleventh International Conference on Learning Representations*.

J Goodfellow Ian, Pouget-Abadie Jean, Mirza Mehdi, Xu Bing, Warde-Farley David, Ozair Sherjil, and C Courville Aaron. 2014. Generative adversarial nets. In *Proceedings of the 27th international conference on neural information processing systems*, volume 2, pages 2672–2680.

David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2003. English gigaword. *Linguistic Data Consortium, Philadelphia*, 4(1):34.

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 and Xiang Kong. 2021. Fully non-autoregressive neural machine translation: Tricks of the trade. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 120–133.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. *Advances in neural information processing systems*, 32.

Junliang Guo, Xu Tan, Di He, Tao Qin, Linli Xu, and Tie-Yan Liu. 2019. Non-autoregressive neural machine translation with enhanced decoder input. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3723–3730.

Xiaochuang Han, Sachin Kumar, and Yulia Tsvetkov. 2023. Ssd-lm: Semi-autoregressive simplex-based diffusion language model for text generation and modular control. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11575–11596.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851.

Jonathan Ho and Tim Salimans. 2021. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. 2021. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in Neural Information Processing Systems*, 34:12454–12465.

Fei Huang, Hao Zhou, Yang Liu, Hang Li, and Minlie Huang. 2022. Directed acyclic transformer for non-autoregressive machine translation. In *International Conference on Machine Learning*, pages 9410–9428. PMLR.

Chao Jiang, Mounica Maddela, Wuwei Lan, Yang Zhong, and Wei Xu. 2020. Neural crf model for sentence alignment in text simplification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7943–7960.

Jungo Kasai, James Cross, Marjan Ghazvininejad, and Jiatao Gu. 2020. Non-autoregressive machine translation with disentangled context transformer. In *International conference on machine learning*, pages 5144–5155. PMLR.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327.

Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. 2020. Diffwave: A versatile diffusion model for audio synthesis. In *International Conference on Learning Representations*.

Shankar Kumar and William Byrne. 2004. Minimum bayes-risk decoding for statistical machine translation. Technical report, JOHNS HOPKINS UNIV BALTIMORE MD CENTER FOR LANGUAGE AND SPEECH PROCESSING (CLSP).

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.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. 2022. 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.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Alexander Quinn Nichol and Prafulla Dhariwal. 2021. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. 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.

Matt Post. 2018. A call for clarity in reporting bleu scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191.

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.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. Comet: A neural framework for mt evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702.

Machel Reid, Vincent J. Hellendoorn, and Graham Neubig. 2022. Diffuser: Discrete diffusion via edit-based reconstruction.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389.

Nikolay Savinov, Junyoung Chung, Mikolaj Binkowski, Erich Elsen, and Aaron van den Oord. 2021. Step-unrolled denoising autoencoders for text generation. In *International Conference on Learning Representations*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020a. Denoising diffusion implicit models. In *International Conference on Learning Representations*.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2020b. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*.

Robin Strudel, Corentin Tallec, Florent Altché, Yilun Du, Yaroslav Ganin, Arthur Mensch, Will Grathwohl, Nikolay Savinov, Sander Dieleman, Laurent Sifre, and Rémi Leblond. 2022. Self-conditioned embedding diffusion for text generation.

Zineng Tang, Ziyi Yang, Chenguang Zhu, Michael Zeng, and Mohit Bansal. 2024. Any-to-any generation via composable diffusion. *Advances in Neural Information Processing Systems*, 36.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Jiasheng Ye, Zaixiang Zheng, Yu Bao, Lihua Qian, and Mingxuan Wang. 2023. Dinoiser: Diffused conditional sequence learning by manipulating noises.

Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Fei Huang, and Songfang Huang. 2023. Seqdiffuseq: Text diffusion with encoder-decoder transformers.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

A Proof of Theorem 1

Following we give the proof of Theorem 1. We start by proving two lemmas. The first one gives the limit of the loss objective if provided insufficient small noise in a high-dimensional space. And the second one proves that the accuracy of the degenerated model converges to 1 on the same condition. Finally, Theorem 1 can be conducted.

Lemma 1. *Assume the embeddings* $\boldsymbol{e}_\phi \sim \mathcal{N}_{d\times V}(\mathbf{0}, \sigma_e \mathbf{I})$*, then*

$$\forall \varepsilon > 0, \lim_{\substack{\bar{\beta}\to 0 \\ d\to\infty}} P(\mathcal{L}(\boldsymbol{z}_t, \boldsymbol{z}_0) < \varepsilon) = 1,$$

which can be rewritten as

$$\mathcal{L}(\boldsymbol{z}_t, \boldsymbol{z}_0) \xrightarrow[\substack{\bar{\beta}\to 0 \\ d\to\infty}]{P} 0.$$

Also, $\forall \boldsymbol{e}_i \in \boldsymbol{e}_\phi$ *and* $\boldsymbol{e}_i \neq \boldsymbol{z}_0$*,*

$$\mathcal{L}(\boldsymbol{z}_t, \boldsymbol{e}_i) \xrightarrow[\substack{\bar{\beta}\to 0 \\ d\to\infty}]{P} 2\sigma_e^2.$$

Proof.

$$\begin{aligned}
\mathcal{L}(\boldsymbol{z}_t, \boldsymbol{z}_0) &= \|\boldsymbol{z}_t - \boldsymbol{z}_0\|^2 - \log p_\phi(y_{\boldsymbol{z}_0}|\boldsymbol{z}_t) \\
&= \|\boldsymbol{z}_t\|^2 + \|\boldsymbol{z}_0\|^2 - \frac{2}{d}\boldsymbol{z}_t \cdot \boldsymbol{z}_0 \\
&\quad - \log \frac{\exp(\boldsymbol{z}_t \cdot \boldsymbol{z}_0)}{\sum_{i=1}^V \exp(\boldsymbol{z}_t \cdot \boldsymbol{e}_i)} \\
&= \|\boldsymbol{z}_t\|^2 + \|\boldsymbol{z}_0\|^2 - \frac{2}{d}\boldsymbol{z}_t \cdot \boldsymbol{z}_0 \\
&\quad + \log \Bigg(1 \\
&\quad + \sum_{i\neq y_{\boldsymbol{z}_0}}^{V} \exp(\boldsymbol{z}_t \cdot \boldsymbol{e}_i - \boldsymbol{z}_t \cdot \boldsymbol{z}_0) \Bigg),
\end{aligned}$$

where $\|\cdot\|^2$ represents mean square error (MSE), and $y_{\boldsymbol{z}_0}$ is the token index of $\boldsymbol{z}_0$. Since $\boldsymbol{z}_t \sim$

$\mathcal{N}\left(\sqrt{1-\bar{\beta}_t}\boldsymbol{z}_0, \bar{\beta}_t\mathbf{I}\right)$, which can be written as $\boldsymbol{z}_t = \sqrt{1-\bar{\beta}_t}\boldsymbol{z}_0 + \sqrt{\bar{\beta}_t}\boldsymbol{\varepsilon}$, where $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, we get

$$\begin{aligned}\mathcal{L}(\boldsymbol{z}_t, \boldsymbol{z}_0) = &\left(2 - \bar{\beta} - 2\sqrt{1-\bar{\beta}}\right)\|\boldsymbol{z}_0\|^2 + \bar{\beta}\|\boldsymbol{\varepsilon}\|^2 \\ &+ 2\left(\sqrt{\bar{\beta}-\bar{\beta}^2} - \sqrt{\bar{\beta}}\right)\frac{\boldsymbol{z}_0 \cdot \boldsymbol{\varepsilon}}{d} \\ &+ \log\Bigg(1 \qquad (4)\\ &+ \sum_{i \neq y_{\boldsymbol{z}_0}}^{V} \exp(g(\boldsymbol{z}_0, \boldsymbol{\varepsilon}, \boldsymbol{e}_i, \boldsymbol{z}_0) \cdot d)\Bigg), \qquad (5)\end{aligned}$$

where

$$\begin{aligned}g(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}) = &\sqrt{1-\bar{\beta}}\frac{\boldsymbol{a}\cdot\boldsymbol{c}}{d} + \sqrt{\bar{\beta}}\frac{\boldsymbol{b}\cdot\boldsymbol{c}}{d} \\ &- \sqrt{1-\bar{\beta}}\frac{\boldsymbol{a}\cdot\boldsymbol{d}}{d} - \sqrt{\bar{\beta}}\frac{\boldsymbol{b}\cdot\boldsymbol{d}}{d}. \qquad (6)\end{aligned}$$

According to Khinchin's law,

$$\|\boldsymbol{z}_0\|^2 = \frac{1}{d}\sum_{i=1}^{d} z_{0,i}^2 \xrightarrow[\substack{\bar{\beta}\to 0\\ d\to\infty}]{P} \mathbb{E}(z_{0,i}^2) = \sigma_e^2.$$

Similarly, the rest terms can be calculated. Based on the continuous mapping theorem, we can substitute these values into Eqs. (4) and (6) as

$$\begin{aligned}g(\boldsymbol{z}_0, \boldsymbol{\varepsilon}, \boldsymbol{e}_i, \boldsymbol{z}_0) &\xrightarrow[\substack{\bar{\beta}\to 0\\ d\to\infty}]{P} 1\cdot 0 + 0\cdot 0 - 1\cdot\sigma_e^2 - 0\cdot 0 \\ &= -\sigma_e^2, \\ \mathcal{L}(\boldsymbol{z}_t, \boldsymbol{z}_0) &\xrightarrow[\substack{\bar{\beta}\to 0\\ d\to\infty}]{P} 0\cdot\sigma_e^2 + 0\cdot 1 + 0\cdot 0 \\ &\quad + \log(1 \\ &\quad + (V-1)\exp(-\sigma_e^2\cdot\infty)) \\ &= 0.\end{aligned}$$

Similarly,

$$\begin{aligned}\mathcal{L}(\boldsymbol{z}_t, \boldsymbol{e}_i) = &\left(1-\bar{\beta}\right)\|\boldsymbol{z}_0\|^2 + \bar{\beta}\|\boldsymbol{\varepsilon}\|^2 + \|\boldsymbol{e}_i\|^2 \\ &+ 2\sqrt{\bar{\beta}-\bar{\beta}^2}\frac{\boldsymbol{z}_0\cdot\boldsymbol{\varepsilon}}{d} \\ &- 2\sqrt{1-\bar{\beta}}\frac{\boldsymbol{z}_0\cdot\boldsymbol{e}_i}{d} \\ &- 2\sqrt{\bar{\beta}}\frac{\boldsymbol{\varepsilon}\cdot\boldsymbol{e}_i}{d} \\ &+ \log\Bigg(1 \\ &+ \sum_{j\neq i}^{V}\exp(g(\boldsymbol{z}_0, \boldsymbol{\varepsilon}, \boldsymbol{e}_j, \boldsymbol{e}_i)\cdot d)\Bigg) \\ &\xrightarrow[\substack{\bar{\beta}\to 0\\ d\to\infty}]{P} 2\sigma_e^2.\end{aligned}$$

□

Lemma 2. *Assume the embeddings $\boldsymbol{e}_\phi \sim \mathcal{N}_{d\times V}(\mathbf{0}, \sigma_e\mathbf{I})$, then for a single noised embedding $\boldsymbol{z}_t$, the output of the degenerated model*

$$f_{\mathrm{dg}}(\boldsymbol{z}_t) \xrightarrow[\substack{\bar{\beta}\to 0\\ d\to\infty}]{P} \boldsymbol{z}_0.$$

Proof. According to Lemma 1, $\forall \boldsymbol{e}_i \in \boldsymbol{e}_\phi$ and $\boldsymbol{e}_i \neq \boldsymbol{z}_0$, we know

$$\mathcal{L}(\boldsymbol{z}_t, \boldsymbol{z}_0) \xrightarrow[\substack{\bar{\beta}\to 0\\ d\to\infty}]{P} 0,$$

and,

$$\mathcal{L}(\boldsymbol{z}_t, \boldsymbol{e}_i) \xrightarrow[\substack{\bar{\beta}\to 0\\ d\to\infty}]{P} 2\sigma_e^2.$$

Since $0 < 2\sigma_e^2$, according to the limiting inequality,

$$\lim_{\substack{\bar{\beta}\to 0\\ d\to\infty}} P(\mathcal{L}(\boldsymbol{z}_t, \boldsymbol{z}_0) < \mathcal{L}(\boldsymbol{z}_t, \boldsymbol{e}_i)) = 1,$$

which means

$$\lim_{\substack{\bar{\beta}\to 0\\ d\to\infty}} P\left(\boldsymbol{z}_0 = \arg\min_{\boldsymbol{e}\in\boldsymbol{e}_\phi}\mathcal{L}(\boldsymbol{z}_t, \boldsymbol{e}_i)\right) = 1.$$

Hence,

$$f_{\mathrm{dg}}(\boldsymbol{z}_t) = \arg\min_{\boldsymbol{e}\in\boldsymbol{e}_\phi}\mathcal{L}(\boldsymbol{z}_t, \boldsymbol{e}_i) \xrightarrow[\substack{\bar{\beta}\to 0\\ d\to\infty}]{P} \boldsymbol{z}_0.$$

□

Proof of Theorem 1. From Lemma 2,

$$f_{\mathrm{dg}}(\boldsymbol{z}_t) \xrightarrow[\substack{\bar{\beta}\to 0\\ d\to\infty}]{P} \boldsymbol{z}_0.$$

Following the proof of Lemma 1, it is obvious that

$$\lim_{\boldsymbol{z}\to\boldsymbol{z}_0} \mathcal{L}(\boldsymbol{z},\boldsymbol{z}_0) = \mathcal{L}(\boldsymbol{z}_0,\boldsymbol{z}_0) \xrightarrow[\substack{\bar{\beta}\to 0\\ d\to\infty}]{P} 0.$$

According to the law of the limit of compositions,

$$\mathcal{L}(f_{\mathrm{dg}}(\boldsymbol{z}_t),\boldsymbol{z}_0) \xrightarrow[\substack{\bar{\beta}\to 0\\ d\to\infty}]{P} 0. \tag{7}$$

$\forall\theta'$, if

$$\mathcal{L}(f_{\theta'}(\boldsymbol{z}_t,t),\boldsymbol{z}_0) \xrightarrow[\substack{\bar{\beta}\to 0\\ d\to\infty}]{P} \mathcal{L}' > 0,$$

from limiting inequality,

$$\lim_{\substack{\bar{\beta}\to 0\\ d\to\infty}} P(\mathcal{L}(f_{\mathrm{dg}}(\boldsymbol{z}_t),\boldsymbol{z}_0) < \mathcal{L}(f_{\theta'}(\boldsymbol{z}_t,t),\boldsymbol{z}_0)) = 1,$$

which means that the probability of f_{dg} being a global minimum of $\mathcal{L}$ for θ converges to 1 as $\bar{\beta}\to 0$ and $d\to\infty$.

Otherwise, if

$$\mathcal{L}(f_{\theta'}(\boldsymbol{z}_t,t),\boldsymbol{z}_0) \xrightarrow[\substack{\bar{\beta}\to 0\\ d\to\infty}]{P} 0,$$

$f_{\theta'}$ satisfies the above conclusion that θ' is also a global minimum.

Finally, if $\mathcal{L}(f_{\theta'}(\boldsymbol{z}_t,t),\boldsymbol{z}_0)$ diverges, then $f_{\theta'}(\boldsymbol{z}_t,t)$ diverges, indicating $f_{\theta'}$ is an unstable model, or converges to infinity. □

Notably, though the condition $\bar{\beta}\to 0$ and $d\to\infty$ seem too strong, our empirical results exhibit a fast coverage speed of Eq. (7) in practice. Fig. 5 showcases the loss of the degenerated model, *i.e.*, $\mathcal{L}(f_{\mathrm{dg}}(\boldsymbol{z}_t),\boldsymbol{z}_0)$ with varying levels of noise and embedding dimensions. According to the figure, even under a dimension of 64, a $\bar{\beta}$ of 0.15 is sufficiently small to ensure the loss converges to 0. Under higher dimensions, like 128, we can observe that approximately 50% of $\bar{\beta}$ obtains zero loss, which significantly emphasizes the tendency of degeneration. A similar tendency is also revealed by the model capacity decreasing in Fig. 2a.

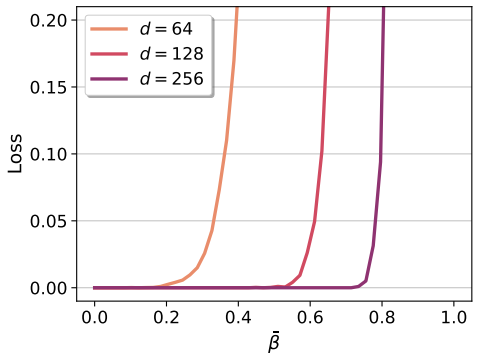


Figure 5: The loss of the degenerated model f_{dg} with varying levels of noise and embedding dimensions.

B Detailed Derivation of the Objective

For a data sample $\mathbf{z}_0$, given a series of latent variables $\mathbf{z}_1,\cdots,\mathbf{z}_T$ which form a Markov chain, we start with the definition of the forward process:

$$q(\mathbf{z}_t|\mathbf{z}_{t-1}) = \mathcal{N}\left(\mathbf{z}_t;\sqrt{\alpha_t}\mathbf{z}_{t-1},\beta_t\mathbf{I}\right),$$

where α_t and β_t represent the noise schedule, and $\alpha_t+\beta_t=1$, $\mathbf{z}_T\sim\mathcal{N}(\mathbf{0},\mathbf{I})$. Then the distribution of the latent variable at any timestep can be determined by

$$q(\mathbf{z}_t|\mathbf{z}_0) = \mathcal{N}\left(\mathbf{z}_t;\sqrt{\bar{\alpha}_t}\mathbf{z}_0,\bar{\beta}_t\mathbf{I}\right),$$

where $\bar{\alpha}_t := \prod_{i=0}^{t}\alpha_i$ and $\bar{\beta}_t := 1-\bar{\alpha}_t$.

The objective of diffusion models is to generate a denoising series to fit the reversion of the forward process, called the reverse process. The reverse process also forms a Markov chain, and is fixed to the learned Gaussian transitions

$$p_\theta(\mathbf{z}_{t-1}|\mathbf{z}_t) = \mathcal{N}\left(\mathbf{z}_{t-1};\boldsymbol{\mu}_\theta(\mathbf{z}_t,t),\boldsymbol{\Sigma}_\theta(\mathbf{z}_t,t)\right),$$

where $\boldsymbol{\mu}_\theta(\cdot)$ and $\boldsymbol{\Sigma}_\theta(\cdot)$ are learnable variables. The covariance is set to be a constant as $\boldsymbol{\Sigma}_\theta(\mathbf{z}_t,t) = \sigma_t^2\mathbf{I}$ following Ho et al. (2020).

The variational lower-bound can be derived from the negative log-likelihood as:

$$\mathbb{E}[-\log p_\theta(\mathbf{z}_0)] \le \mathbb{E}_q\left[-\log\frac{p_\theta(\mathbf{z}_{0:T})}{q(\mathbf{z}_{1:T}|\mathbf{z}_0)}\right] = \mathcal{L}_{\mathrm{vlb}}.$$

Then according to the Markov property,

$$\begin{aligned}\mathcal{L}_{\text{vlb}} &= \mathbb{E}_q\left[-\log\frac{p(\mathbf{z}_T)\prod_{t=1}^{T}p_\theta(\mathbf{z}_{t-1}|\mathbf{z}_t)}{\prod_{t=1}^{T}q(\mathbf{z}_t|\mathbf{z}_{t-1})}\right]\\ &= \mathbb{E}_q\left[-\sum_{t\geq 1}\log\frac{p_\theta(\mathbf{z}_{t-1}|\mathbf{z}_t)}{q(\mathbf{z}_t|\mathbf{z}_{t-1})}\right]+C_1\\ &= \mathbb{E}_q\Bigg[-\sum_{t>1}\log\frac{p_\theta(\mathbf{z}_{t-1}|\mathbf{z}_t)}{q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{z}_0)}\\ &\quad -\log p_\theta(\mathbf{z}_0|\mathbf{z}_1)\Bigg]+C_2\\ &= \mathbb{E}_q\Bigg[\sum_{t>1}\mathbb{KL}[q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{z}_0)\|p_\theta(\mathbf{z}_{t-1}|\mathbf{z}_t)]\\ &\quad -\log p_\theta(\mathbf{z}_0|\mathbf{z}_1)\Bigg]+C_2,\end{aligned}$$

where $\mathbb{KL}[\cdot\|\cdot]$ denotes the KL divergence. Here, $q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{z}_0)$ has the closed form

$$q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{z}_0)=\mathcal{N}\left(\mathbf{z}_{t-1};\tilde{\boldsymbol{\mu}}_t(\mathbf{z}_t,\mathbf{z}_0),\tilde{\beta}_t\mathbf{I}\right),$$

and

$$\begin{cases}\tilde{\boldsymbol{\mu}}_t(\mathbf{z}_t,\mathbf{z}_0)=\xi_t\mathbf{z}_0+\lambda_t\mathbf{z}_t\\ \xi_t=\dfrac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\\ \lambda_t=\dfrac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\end{cases}.$$

Let

$$\mathcal{L}_{t-1}=\mathbb{KL}[q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{z}_0)\|p_\theta(\mathbf{z}_{t-1}|\mathbf{z}_t)],$$

using the formula for the KL divergence of Gaussian distributions, we can derive

$$\mathcal{L}_{t-1}=\frac{1}{2\sigma_t^2}\|\tilde{\boldsymbol{\mu}}_t(\mathbf{z}_t,\mathbf{z}_0)-\boldsymbol{\mu}_\theta(\mathbf{z}_t,t)\|^2.$$

We further parameterize $\boldsymbol{\mu}_\theta(\mathbf{z}_t,t):=\tilde{\boldsymbol{\mu}}_t(\mathbf{z}_t,\hat{\mathbf{z}}_0(\mathbf{z}_t,t))$, where $\hat{\mathbf{z}}_0(\mathbf{z}_t,t)$ is the model prediction of the original data $\mathbf{z}_0$. Consequently,

$$\begin{aligned}\mathcal{L}_{t-1} &= \frac{1}{2\sigma_t^2}\|\xi_t\mathbf{z}_0+\lambda_t\mathbf{z}_t-(\xi_t\hat{\mathbf{z}}_0(\mathbf{z}_t,t)+\lambda_t\mathbf{z}_t)\|^2\\ &\quad +C_3\\ &= \frac{\lambda_t^2}{2\sigma_t^2}\|\mathbf{z}_0-\hat{\mathbf{z}}_0(\mathbf{z}_t,t))\|^2+C_3.\end{aligned}$$

For the last term,

$$-\log p_\theta(\mathbf{z}_0|\mathbf{z}_1)=\frac{1}{2\sigma_1^2}\|\mathbf{z}_0-\tilde{\boldsymbol{\mu}}_1(\mathbf{z}_1,\hat{\mathbf{z}}_0(\mathbf{z}_1,1))\|^2+C_4.$$

Models	WMT14 En-De COMET	IWSLT14 De-En COMET
Transformer	0.8286	0.7894
CMLM	0.8226	0.7736
Differmer	0.8257	0.7875

Table 10: The COMET scores of Differmer and baselines. All results are reported by our implementation.

Noting that

$$\begin{cases}\xi_1=\dfrac{\sqrt{1}(1-\alpha_1)}{1-\alpha_1}=1\\ \lambda_1=\dfrac{\sqrt{\alpha_1}(1-1)}{1-\alpha_1}=0\end{cases},$$

the term can be converted as

$$\begin{aligned}-\log p_\theta(\mathbf{z}_0|\mathbf{z}_1) &= \frac{\lambda_1}{2\sigma_1^2}\|\mathbf{z}_0-\hat{\mathbf{z}}_0(\mathbf{z}_1,1)\|^2+C_4\\ &= \mathcal{L}_{t-1}|_{t=1}+C_4',\end{aligned}$$

which can be combined with the sum of the KL terms.

Finally, ignoring constant terms and weight factors, the training objective becomes

$$\mathcal{L}_{\text{vlb}}=\mathbb{E}_{\mathbf{z}_0,\mathbf{z}_t,t}\left[\|\hat{\mathbf{z}}_0(\mathbf{z}_t,t)-\mathbf{z}_0\|^2\right].$$

C Additional Experimental Results

C.1 More Metrics in Translation

We evaluate the COMET (Rei et al., 2020) score of Differmer as well as baselines on the translation task in Table 10. Combined with BLEU, these results also confirm the performance of Differmer, and its comparability with autoregressive models.

C.2 Comparison with More Baselines

As our research in this work focuses on optimization challenges of embedding diffusion models, we mainly compare Differmer with existing diffusion-based models. To evaluate the promising performance of Differmer, we extend our comparisons to include results from established traditional models, such as mBART (Liu et al., 2020), Levenshtein Transformer (Gu et al., 2019), DisCo (Kasai et al., 2020), Fully NAT (Gu and Kong, 2021), and DA-Transformer (Huang et al., 2022). The results are presented in Table 11. It is worth noting that mBART is a pre-trained language model, which employs significantly larger datasets in training, and it is more than 10 times larger in terms of parameter number. Through the results, the performance of Differmer is still competitive even against more recent and stronger baselines.

Models	b	BLEU
mBART	5	30.50
Levenshtein Transformer	5	27.27
DisCo	5	27.34
Fully NAT	1	27.49
DA-Transformer	200	27.78
Differmer	5	27.61

Table 11: BLEU scores on the WMT14 En-De dataset. All results are as reported in their paper.

Setting	ANI	BLEU
$\mathcal{L}_{\text{vlb}}$	0.99	0.07
$\mathcal{L}_{\text{text}}$	0.32	27.89
$\mathcal{L}$	0.03	**34.48**
$\mathcal{L}$ w/ fixed CMLM embeddings	0.02	30.14

Table 12: The anisotropy score and performance of different settings on the IWSLT14 De-En dataset with the *linear* schedule.

C.3 Study of Frozen Embeddings

Section 3.1 discusses the collapse problem arising from learnable embeddings and introduces the anchor loss to prevent collapse and ensure stability of training. Alternatively, replacing the embedding space with a pre-trained one is also a solution to address this problem, and our preliminary experiments explore this possibility. The corresponding experimental results can be found in Table 12. From the table, the embeddings from pre-trained CMLM alleviate the collapse problem notably, and the performance obtained by each setting is highly related to ANI, which echos our findings in Section 3.1. Moreover, our proposed method exhibits a substantial improvement compared with the model with pre-trained embeddings. We attribute this to the possibility that the embeddings from traditional models are suboptimal for diffusion models.

C.4 Study of DGS_{MAX} with Different Schedules

We further provide more results with different values of DGS_{MAX} and schedules in Table 13. From the table, we can notice that the degeneration problem exists widely in noise schedules, and the proposed noise rescaling framework enhances the performance of schedules consistently. Furthermore, the rescaling factor of *sqrt* is more sensitive to DGS_{MAX}.

C.5 Study of the Dimension of the Embeddings

Prior works mainly use an embedding dimension of 128 (Li et al., 2022; Gong et al., 2022; Yuan et al., 2023), and we also find the model is quite hard to work with a higher embedding dimension like 256 or 512. Intuitively, higher dimensional embedding space is sparser, and embeddings require more noise to diffuse from their nearest neighbor region. Therefore, the model encounters more severe degeneration in this space if provided with insufficient noise. From Table 14, a larger rescaling factor is demanded to reach the same DGS_{MAX} in a higher dimensional embedding space. On the other hand, a low-dimensional embedding space may result in a limited capacity for representation. With noise rescaling, models with different embedding dimensions work successfully and achieve similar results, showing the robustness of the proposed noise rescaling framework. Additionally, the noise rescaling framework augments the scalability of embedding diffusion models and actualizes the potential of application on large-scale datasets and tasks.

C.6 Study of Beam Size

We study the influence of the 2D parallel decoding hyper-parameters, *i.e.*, the length beam size b_1 and noise beam size b_2 described in Section 3.3. As shown in Table 15, we find that length and noise beams both boost the generation quality and are complementary to each other. The length beam brings more significant improvements, while a sufficiently large $b = b_1 \times b_2$ leads to the saturation of the BLEU score.

C.7 Noise Rescaling in Sampling

Since noise rescaling is a technique to alleviate a training problem, it is not applied in sampling. We study the sampling quality when applying the noise rescaling in decoding steps. As illustrated in Table 16, the noise rescaling in sampling is harmful to sampling quality due to no requirement of large noise in the reverse process.

C.8 Variance-Preserving Rescaling Factor

The variance-preserving variant of the rescaling factor can be defined as

$$\begin{cases} \bar{\alpha}'_t = \dfrac{\bar{\alpha}_t}{\bar{\alpha}_t + F^2\bar{\beta}_t} \\ \bar{\beta}'_t = \dfrac{F^2\bar{\beta}_t}{\bar{\alpha}_t + F^2\bar{\beta}_t} \end{cases}, \tag{8}$$

$\mathrm{DGS}_{\mathrm{MAX}}$	*Linear*		*Cosine*		*Sqrt*	
	RF	**BLEU**	**RF**	**BLEU**	**RF**	**BLEU**
0.05	21.0	31.99	41.0	31.64	7.0	33.88
0.15	6.0	33.09	12.5	33.36	4.0	34.48
0.50	-	-	3.0	33.01	2.0	32.54
-	1.0	32.21	1.0	26.61	1.0	22.70

Table 13: The performance with varying $\mathrm{DGS}_{\mathrm{MAX}}$ and noise schedule. Where RF stands for the rescaling factor. Refer to Table 2, DGS of *linear* is less than 0.50, thus the result of *linear* with $\mathrm{DGS}_{\mathrm{MAX}} = 0.50$ is empty. The last row represents the model without noise rescaling. Results are obtained on the IWSLT14 De-En dataset with $b = 10$.

d	**Rescaling Factor**	**BLEU**
64	2.5	32.11
128	4.0	34.48
256	6.0	34.41
512	8.5	34.33

Table 14: The performance with varying dimensions of the embedding space. Results are obtained on the IWSLT14 De-En dataset with $b = 10$ and $\mathrm{DGS}_{\mathrm{MAX}} = 0.15$.

b_1	b_2	**BLEU**
1	1	32.91
1	9	33.53
3	3	34.33
5	2	34.43
9	1	34.48
5	4	34.52
7	3	34.48
9	2	34.44
10	5	34.52

Table 15: The performance with varying b_1 and b_2. Results are obtained on the IWSLT14 De-En dataset.

where F is the rescaling factor. We can derive that $\bar{\alpha}'_t + \bar{\beta}'_t = 1$ and the signal-to-noise ratio of the rescaled schedule satisfies $\mathrm{SNR}'_t = \mathrm{SNR}_t / F^2$, where $\mathrm{SNR}_t = \bar{\alpha}_t / \bar{\beta}_t$. The VP rescaling factor performs similarly to the original version, which is listed in the Table 17.

C.9 Controllable Generation

Due to the distinctive iterative generation process, one of the advantages of diffusion models is that they support flexible and fine-grained control over the outputs, such as syntax tree, length, prefix, and suffix (Li et al., 2022). We validate the length conditioning ability for Differmer in Table 18. Through the cases, the length of generations is well-matched with the condition, highlighting the controllability of Differmer.

NR in Sampling	**BLEU**
	34.48
✓	33.51

Table 16: The BLEU score with noise rescaling ($\mathrm{DGS}_{\mathrm{MAX}} = 0.15$) in sampling. Results are obtained on the IWSLT14 De-En dataset with $b = 10$.

C.10 Case Study

To qualitatively analyze the dynamics of the generation quality at the instance level, we select several representative cases in Table 22. The generation results in Table 22 illustrate the characteristics of Differmer. Specifically, after the first few steps, the model is able to transform random words into noised but human-readable sentences, and with the reverse process progressing, these sentences are refined gradually. However, as the process enters the final steps, the model at these timesteps suffers from the degeneration problem and loses the ability to improve the outputs, even corrupts previously correct words. This dynamics is also reflected in Fig. 4, and emphasizes the necessity of noise rescaling and early stopping techniques. On the other hand, Differmer achieves comparable quality with the autoregressive baseline, and generates more coherent and consistent sentences compared with traditional non-autoregressive models.

D Search of Rescaling Factor

To decide F, we can perform either brute-force or binary search. Without loss of generality, the approach of brute-force search is presented in Algorithm 1. For the computation of DGS, we firstly regard f_{dg} as an equivalent nearest neighbor classifier, then utilize a Monte Carlo method to estimate DGS_t at some of the timesteps, and finally compute the average of all values of DGS_t. Algorithm 2 illustrates this process. Since our theorem is based on the assumption that $\boldsymbol{e}_\phi \sim \mathcal{N}_{d\times V}(\mathbf{0}, \sigma_e \mathbf{I})$,

DGS_{MAX}	Variance Preserving	Rescaling Factor	BLEU
0.05		7.0	33.88
	✓	9.5	33.80
0.15		4.0	34.48
	✓	4.5	33.99
0.50		2.0	32.54
	✓	2.0	31.77
0.77	-	1.0	22.70

Table 17: The performance with varying DGS_{MAX} and variance-preserving rescaling factor. Results are obtained on the IWSLT14 De-En dataset with $b = 10$.

Length	Generations
5	`our imagination is even reality.`
10	`our imagination is a force that can even create reality.`
15	`our imagination, in fact, is a force that can even create a reality.`
Target	`imagination is a force that can actually manifest a reality.`

Table 18: Cases when applying length control.

Algorithm 1 Search F

Input: Noise schedule $\bar{\alpha}, \bar{\beta}$, threshold of degeneration score DGS_{MAX}, search interval ΔF
Output: F

$F \leftarrow 1$
while true do
  Rescale $\bar{\alpha}'$ and $\bar{\beta}'$ from $\bar{\alpha}$ and $\bar{\beta}$ using F according to Eq. (3) or Eq. (8)
  $\text{DGS} \leftarrow \text{DGS}\left(\bar{\alpha}', \bar{\beta}'\right)$
  if $\text{DGS} \leq \text{DGS}_{\text{MAX}}$ **then**
    return F
  end if
  $F \leftarrow F + \Delta F$
end while

we initialize it with a normal distribution, and the computation of DGS can be independent of real embeddings and approximated before training. For the convenience of future research, we provide a pre-computed function table of DGS and the corresponding rescaling factors in Table 19.

E Experimental Settings

We build our model based on Transformer (Vaswani et al., 2017) and use `transformer-iwslt-de-en` config for the IWSLT dataset, `transformer-base` config for WMT and summarization datasets. For other

Algorithm 2 Compute DGS

Input: Noise schedule $\bar{\alpha}, \bar{\beta}$, embeddings $\boldsymbol{e}_\phi$, timestep set $\mathcal{T}$, repeat times N
Output: DGS

$\text{DGS} \leftarrow 0$
for all $t \in \mathcal{T}$ **do**
  $\text{DGS}_t \leftarrow 0$
  for all $\boldsymbol{z}_0 \in \boldsymbol{e}_\phi$ **do**
    for $i \leftarrow 1$ **to** N **do**
      Sample $\boldsymbol{z}_t \sim \mathcal{N}\left(\sqrt{\bar{\alpha}_t}\boldsymbol{z}_0, \bar{\beta}_t\mathbf{I}\right)$
      if $f_{\text{dg}}(\boldsymbol{z}_t) = \boldsymbol{z}_0$ **then**
        $\text{DGS}_t \leftarrow \text{DGS}_t + 1$
      end if
    end for
  end for
  $\text{DGS}_t \leftarrow \text{DGS}_t/(N \times V)$
  $\text{DGS} \leftarrow \text{DGS} + \text{DGS}_t$
end for
$\text{DGS} \leftarrow \text{DGS}/|\mathcal{T}|$
return DGS

datasets, to conduct fair comparisons with DiffuSeq, we set the model dimension to 768 and the feed-forward intermediate dimension to 3072. We tokenize sentences and segment each token into subwords by Byte-Pair Encoding (Sennrich et al., 2016). The training process takes nearly one day on 8 NVIDIA V100 GPUs for the WMT datasets and the Gigaword dataset, while nearly 12 hours on one NVIDIA V100 GPU for the other datasets. In inference, we downsample the diffusion step to 20, *i.e.*, $K = 20$, and stop the decoding process 5 steps earlier, which is much faster than previous works (Li et al., 2022; Gong et al., 2022) while maintaining the performance. Each reported result is the average of 3 runs. The dataset splits we used are listed in Table 20. All datasets can be used for research purposes. The detailed hyper-parameters are listed in Table 21.

$\mathrm{DGS}_{\mathrm{MAX}}$	*Linear*			*Cosine*			*Sqrt*		
	10K	20K	40K	10K	20K	40K	10K	20K	40K
0.05	21.0	20.0	19.0	41.0	38.5	37	7.0	6.5	6.0
0.10	9.5	9.0	9.0	19.0	18.5	18.0	5.0	4.5	4.5
0.15	6.0	5.5	5.5	12.5	12.0	11.5	4.0	4.0	3.5
0.20	4.0	4.0	4.0	9.0	9.0	8.5	3.5	3.5	3.0
0.30	2.5	2.5	2.5	6.0	5.5	5.5	2.5	2.5	2.5
0.50	-	-	-	3.0	2.5	2.5	2.0	2.0	1.5

Table 19: The $\mathrm{DGS}_{\mathrm{MAX}}$ and corresponding rescaling factors with different vocabulary sizes and noise schedules. We perform the search in the 128-dimensional embedding space at the interval of 0.5, and $\sigma_e = 1$.

Splits	**WMT14 En-De**	**WMT16 En-Ro**	**IWSLT14 De-En**	**Gigaword**	**QQP**	**Wiki-Auto**	**QT**
Training	4,500,966	608,319	160,215	3,803,957	144,715	677,751	116,953
Validation	3,000	1,999	7,282	189,651	2,048	2,048	2,048
Test	3,003	1,999	6,750	1,951	2,500	5,000	10,000

Table 20: The dataset splits used in our experiments.

Hyper-parameters	**WMT14 En-De**	**WMT16 En-Ro**	**IWSLT14 De-En**	**Gigaword**	**QQP**	**Wiki-Auto**	**QT**
Architecture							
d_{model}	512	512	512	512	768	768	768
d_{emb}	128	128	128	128	128	128	128
d_{ffn}	2048	2048	1024	2048	3072	3072	3072
Heads	8	8	4	8	12	12	12
Encoder Layers	6	6	6	6	6	6	6
Decoder Layers	6	6	6	6	6	6	6
Activation	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU
Diffusion							
Steps	2000	2000	2000	2000	2000	2000	2000
Schedule	*sqrt*	*sqrt*	*sqrt*	*sqrt*	*sqrt*	*sqrt*	*sqrt*
$\mathrm{DGS}_{\mathrm{MAX}}$	0.15	0.15	0.15	0.15	0.15	0.15	0.20
Self-Conditioning	✓	✓	✓	✓	✓	✓	✓
Training							
Steps	300K	300K	300K	300K	50K	30K	100K
Batch Size (Tokens)	64K	64K	8K	64K	8K	64K	64K
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW	AdamW	AdamW
Adam β	$(0.9, 0.98)$	$(0.9, 0.98)$	$(0.9, 0.98)$	$(0.9, 0.98)$	$(0.9, 0.98)$	$(0.9, 0.98)$	$(0.9, 0.98)$
Weight Decay	0.01	0.01	0.01	0.01	0.01	0.01	0.01
Learning Rate	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}	3×10^{-4}
Warmup	10K	10K	10K	10K	10K	10K	5K
Clip Gradient	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Dropout	0.1	0.1	0.3	0.1	0.1	0.1	0.1
Length Predict Factor	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Label Smoothing	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Inference							
Steps	20	20	20	20	20	20	20
Early Stopping	5	5	5	5	5	5	5

Table 21: The model architectures and hyper-parameters used in our experiments.

	Case 1
Source	ich denke , dass es schwer wird, sie zu treffen, aber ich denke, es ist es auf jeden fall wert, einige wirklich bekannte marken anzusprechen.
Target	i think that you're going to have a hard time meeting with them, but i think it's certainly worth pursuing a couple big, really obvious brands.
Progress	**Generation**
0%	i think it's going to meet you, but i i it it definitely worth talking sing some really known brands.
25%	i think it's hard to meet you, but i think it's definitely worth mating some really known brands.
50%	i think it's hard to meet you, but i think it's definitely worth approaching some really known brands.
75%	i think it's hard to meet you, but i think it's definitely worth approaching some really known brands.
100%	i think it's hard to meet you, but i think it's definitely worth approaching some really known brks.
Transformer	i think it's hard to meet you, but i think it's worth addressing some really familiar brands, i think.
CMLM	i think it's hard to meet you, but i think it's worth saying some really known brands.
	Case 2
Source	und wenn wir mit einem körper konfrontiert sind, der für uns tatsächlich etwas sehr anderes darstellt, verwirrt uns das in hinblick auf diese kategorisierungen.
Target	and when we're faced with a body that actually presents us something quite different, it startles us in terms of those categorizations.
Progress	**Generation**
0%	and when we're confronted with a body which is actually something very different for us, we're confused in in terms of these categorization.
25%	and when we're confronted with a body that is actually something very different for us, we're confused it in terms of categcategorization.
50%	and when we're confronted with a body that's actually something very different for us, we're confusing it in terms of these categorization.
75%	and when we're confronted with a body that's actually something very different for us, we're confusing it in terms of these categorization.
100%	and when we're confronted with a body that's actually something very different for us, we're confusing it in terms of these categorizes.
Transformer	and when we're faced with a body, which actually represents something very different for us, it confuses us in terms of these categorization.
CMLM	and when we're faced with a body that actually represents something very different for us, it confuses us in terms of these categorization.
	Case 3
Source	um also das blinken zu beschleunigen oder zu verlangsamen, drehen sie einfach an diesem knopf und er macht den impuls schneller oder langsamer.
Target	so to make this blink faster or slower, you would just turn this knob and basically make it pulse faster or slower.
Progress	**Generation**
0%	so to accelerate or flck slow slow down, , just turn on buttbuttand it makes pulse faster or slow wer.
25%	so to accelerate the ck or slow it down, just turn on this button and makes the impulse faster or slow down.
50%	so to accelerate the blind or slow it down, just turn on this button and makes the impulse faster or slow down.
75%	so to accelerate the blind or slow it down, just turn on this button and makes the impulse faster or slower.
100%	so to accelerate the blind or slow it down, just turn on this button and makes the impulse faster or slower.
Transformer	so to slow the blind up or slow the blind down, you just turn that button, and it makes the pulse faster or slower.
CMLM	so to speed your blind up or slow down, just turn on that button and it makes the pulse faster or slow.

Table 22: Cases of intermediate generation results during the whole generation process, compared with baselines.