FastDiSS: Few-step Match Many-step Diffusion Language Model on Sequence-to-Sequence Generation

Anonymous ACL submission

Abstract

Diffusion Language Models decouple inference cost from sequence length, achieving linear computational complexity compared to the quadratic complexity of autoregressive models, while providing self-correction via iterative decoding. However, few-step sampling in text diffusion suffers from a training–inference mismatch in the self-conditioning mechanism, akin to exposure bias, and from underexposure to high-noise regimes under uniform scheduling. We propose FastDiSS, a unified framework that boosts the sampling efficiency by bridging the self-conditioning training-inference mismatch and improves the training process through an adaptive model-aware noise scaling. Extensive results show that FastDiSS achieves higher BLEU scores on conditional generation benchmarks while achieving a four times speed-up over standard diffusion models. FastDiSS considerably narrows the performance gap between few and many-step diffusion, demonstrating quality and inference speed improvements over current state-of-the-art methods.

1 Introduction

The parallel nature of the diffusion models (Li et al., 2022; Lou et al., 2024) makes them a strong candidate for fast text generation. Coupled with parallel decoding, diffusion language models achieve linear complexity in sequence length, significantly faster than the quadratic complexity of autoregressive models (Radford et al., 2018, 2019; Brown et al., 2020). Specifically, the training and inference processes simultaneously generate every token following the forward and reverse processes. In the forward process, each token in a sample $\boldsymbol{z}_0$ is independently perturbed in T steps with Gaussian noise. After obtaining the series $\boldsymbol{z}_0, ..., \boldsymbol{z}_{t/T}, ..., \boldsymbol{z}_1$, the denoising network D_θ is trained to reverse this process $\hat{\boldsymbol{z}}_{(t-1)/T} = D_\theta(\hat{\boldsymbol{z}}_{t/T}, t)$, starting from $\boldsymbol{z}_1$, following univariate Gaussian distributions (Section 2).

To ensure that the reverse trajectory closely follows the forward process, a sufficiently large decoding step T is required, imposing a computational burden despite parallel decoding. To maintain a small T, several works explicitly utilized the estimation from the previous step to refine the current prediction, known as the self-conditioning mechanism (Chen et al., 2023b; Yuan et al., 2024; Gulrajani and Hashimoto, 2023). Although this method may not produce significant gains in image generation, it is highly effective for text generation, as the natural sparsity of language makes denoising predictions nearly accurate (Gao et al., 2024; Ye et al., 2023).

In this work, we propose Fast Diffusion Sequence to Sequence (FastDiSS), a novel diffusion training method, combining a training-sampling alignment method and a model-aware noise augmentation to enhance sampling efficiency.

First, we discover that the training posterior mismatches of self-conditioning (Section 3) result in the *exposure bias* issues (Schmidt, 2019; Ning et al., 2024), potentially impeding the generation quality. We propose *Leakage-informed Noise Scheduler (SNL)*, a novel scheduler that simulates the self-conditioning behavior during inference to close the training-inference gap. With this modified scheduler, our model achieves competitive performance against DiffuSeq (Gong et al., 2023a), using only two sampling steps. Our method also outperforms DiffuSeq-v2 (Gong et al., 2023b), a strong state-of-the-art baseline for fast text generation.

We then notice that the unbiased training of the diffusion language network slows down convergence and potentially destabilizes it. Due to the significant influence of high noise levels on text quality (Li et al., 2022), we introduce *Model-Aware Noise Scaling (MANS)*, a selective training augmentation strategy that progressively increases the token noise level to accelerate convergence and avoid trivial learning from low-noise inputs.

Experimental results across diverse conditional generation tasks demonstrate that our method consistently narrows the performance gap between few-step and many-step sampling. Moreover, it surpasses existing text diffusion models in generation quality, while preserving high diversity. For example, on the Machine Translation task (Cettolo et al., 2014; Bojar et al., 2014), with only 5 sampling steps, FastDiSS achieves a higher BLEU score (Papineni et al., 2002) than baselines with 20 and 1000 decoding steps, corresponding to a $4\times$ and $200\times$ speed up.

In summary, our contributions are as follows:

- We reveal the limitations of the self-conditioning mechanism and underscore the critical role of high-level noise in enabling robust diffusion model learning, thereby motivating improved training strategies.
- We propose FastDiSS, a novel training method, which consists of (1) a modified noise scheduler (SNL) to effectively mitigate the exposure bias of self-conditioning and (2) a model-aware mechanism (MANS), which gradually increases the token noise level to avoid trivial training.
- Experiments on various benchmarks show that our proposed method significantly boosts the generation quality and drastically speeds up the inference process.

2 Background

Denoising Diffusion Models. We revisit Gaussian diffusion process in its continuous-time formulation (Song et al., 2021; Kingma et al., 2021), which defines a trajectory $\{\boldsymbol{z}_t\}_{t=0}^{1}$ of increasing noise starting from the clean data $\boldsymbol{z}_0 \sim p(\boldsymbol{z}_0)$ and ending with $\boldsymbol{z}_1 \sim \mathcal{N}(0, \mathbf{I})$. For any t, the noise schedule comprises the decay factor α_t and the diffusion rate σ_t, which are strictly positive and monotonic over time.

Given that $q(\boldsymbol{z}_t|\boldsymbol{z}_0)$ satisfies the Markovian property, the forward process is formulated as:

$$q(\boldsymbol{z}_t|\boldsymbol{z}_0) = \mathcal{N}(\boldsymbol{z}_t; \alpha_t \boldsymbol{z}_0, \sigma_t^2 \mathbf{I}), \quad (1)$$

$$q(\boldsymbol{z}_t|\boldsymbol{x}_s) = \mathcal{N}(\boldsymbol{z}_t; (\alpha_t/\alpha_s)\boldsymbol{z}_s, \sigma_{t|s}^2 \mathbf{I}), \quad (2)$$

where $0 \leq s < t \leq 1$ and $\sigma_{t|s}^2 = \sigma_t^2 - (\alpha_t^2/\alpha_s^2)\sigma_s^2$. As $t \to 1$, $\alpha_t \to 0$ and $\sigma_t \to 1$, so the endpoint almost surely follows a Gaussian distribution.

The goal of the diffusion model is to denoise $\boldsymbol{z}_0 \sim p(\boldsymbol{z}_0|\boldsymbol{z}_t)$ through a neural network $D_\theta(\boldsymbol{z}_t) \approx \boldsymbol{z}_0$, which is trained using a mean-squared error loss

$$L_{\text{diffusion}} = \mathbb{E}_{\boldsymbol{z}_0, t\sim\mathcal{U}[0,1]}[||D_\theta(\boldsymbol{z}_t) - \boldsymbol{z}_0||^2]. \quad (3)$$

Reverse sampling $\boldsymbol{z}_s$ from $\boldsymbol{z}_t$ can be viewed as approximating $p(\boldsymbol{z}_s|\boldsymbol{z}_t)$. To do so, we leverage the posterior distribution $q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{z}_0)$ defined by the forward process, which eventually formulates the sampling distribution by

$$p(\boldsymbol{z}_s|\boldsymbol{z}_t) = \mathbb{E}_{p(\boldsymbol{z}_0|\boldsymbol{z}_t)}[q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{z}_0)], \quad (4)$$

with $\boldsymbol{z}_0$ is replaced by the output of the denoising network $D_\theta(\boldsymbol{z}_t)$. Note that the posterior distribution is tractable and can be expressed as

$$q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{z}_0) = \mathcal{N}(\boldsymbol{z}_s; \frac{\alpha_t}{\alpha_s}\frac{\sigma_s^2}{\sigma_t^2}\boldsymbol{z}_t + \alpha_s\frac{\sigma_{t|s}^2}{\sigma_t^2}\boldsymbol{z}_0, \frac{\sigma_s^2}{\sigma_t^2}\sigma_{t|s}^2\mathbf{I}). \quad (5)$$

Full derivation is shown in Appendix A.1. Recursive sampling with this schedule, starting at $\boldsymbol{z}_1 \sim \mathcal{N}(0, \mathbf{I})$, enables generating data from $p(\boldsymbol{z}_0)$.

Conditional Sequence Modeling with Diffusion Models. Diffusion models rely on a continuous space where Gaussian noise can be smoothly added and subtracted. Text, however, is composed of discrete tokens with no inherent notion of "small changes" between them. To address this, DiffusionLM (Li et al., 2022) maps the text sequence $\boldsymbol{x} \in \{0, 1\}^{L\times V}$, where each token is represented as a one-hot vector, into a continuous latent space $\boldsymbol{z}_0 \in \mathbb{R}^{L\times H}$, with sequence length L, hidden dimension H, vocabulary size V. Hence, $\boldsymbol{z}_0 \sim q(\boldsymbol{z}_0|\boldsymbol{x})$ is the embedding codebook of $\boldsymbol{x}$.

The reverse process aims to generate $\boldsymbol{z}_0$, which is then mapped back to $\boldsymbol{x}$. The diffusion model is trained on the latent space with the objective

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{diffusion}} + \mathcal{L}_{\text{rounding}} = \mathbb{E}_{\boldsymbol{z}_0,t}[||D_\theta(\boldsymbol{z}_t) - \boldsymbol{z}_0||^2] + \mathbb{E}_{\boldsymbol{z}_0}\left[-\log p_\theta(\boldsymbol{x}|\boldsymbol{z}_0)\right], \quad (6)$$

where $\mathcal{L}_{\text{rounding}}$ is the reconstruction loss mapping from the latent to the discrete space.

To extend the model for conditional sequence generation, a simple yet effective approach is to incorporate the conditioning sequence $\boldsymbol{c}$ as an additional input to the denoising network $D_\theta(\boldsymbol{z}_t, \boldsymbol{c})$. The target sequence length L can be inferred from the conditioning context via a learned prior $L \sim p(L|\boldsymbol{c})$. Aside from this conditioning, the diffusion process remains unchanged as in Eq. 4.

3 Self-conditioning Diffusion Language Model

Definition. The sampling target is denoted as $\bar{\boldsymbol{z}}_s$ to distinguish it from the training targets $\boldsymbol{z}_s$. Subsequently, they are sampled from $q(\bar{\boldsymbol{z}}_s|\bar{\boldsymbol{z}}_t, \bar{\boldsymbol{z}}_\theta^{tu})$ and $q(\boldsymbol{z}_s|\boldsymbol{z}_t, \bar{\boldsymbol{z}}_\theta^t)$, where $\bar{\boldsymbol{z}}_\theta^{tu} = D_\theta(\bar{\boldsymbol{z}}_t, \bar{\boldsymbol{z}}_\theta^{uk})$ and $\bar{\boldsymbol{z}}_\theta^t = D_\theta(\boldsymbol{z}_t, \hat{\boldsymbol{z}}_\theta^t)$ for any $0 < s < t < u < k \leq 1$. The initial guess $\hat{\boldsymbol{z}}_\theta^t$ can be expressed as $\hat{\boldsymbol{z}}_\theta^t = D_\theta(\boldsymbol{z}_t, 0)$, where 0 implies uncondition. We omit the source condition $\boldsymbol{c}$ since the main focus is on the self-conditioning.

Training procedure. The initial forward prediction $\hat{\boldsymbol{z}}_\theta^t$ acts as an auxiliary condition, which is then fed back into the denoising model. The $\boldsymbol{z}_0$-prediction becomes $\bar{\boldsymbol{z}}_\theta^t = D_\theta(\boldsymbol{z}_t, \text{sg}(\hat{\boldsymbol{z}}_\theta^t))$, where $\text{sg}(\cdot)$ denotes the stop gradient operation, ensuring that gradients are not propagated back through $\hat{\boldsymbol{z}}_\theta^t$. The new training objective becomes:

$$\mathcal{L}_{\text{sc}} = \mathbb{E}_{\boldsymbol{z}_0,t}[||D_\theta(\boldsymbol{z}_t, \text{sg}(\hat{\boldsymbol{z}}_\theta^t)) - \boldsymbol{z}_0||^2]. \quad (7)$$

The training process alternates between optimizing $\mathcal{L}_{\text{diffusion}}$ and $\mathcal{L}_{\text{sc}}$.

Speed up sampling. At each step, estimating $\hat{\boldsymbol{z}}_\theta^t$ followed by $\bar{\boldsymbol{z}}_\theta^t$ double the inference time. Therefore, previous works reused the previous step prediction $\bar{\boldsymbol{z}}_\theta^{uk}$ to avoid additional inference overhead.

3.1 Challenges and Drawbacks

Distribution mismatch. Since $q(\bar{\boldsymbol{z}}_s|\bar{\boldsymbol{z}}_t, \bar{\boldsymbol{z}}_\theta^{tu}) = q(\boldsymbol{z}_s|\boldsymbol{z}_t, \bar{\boldsymbol{z}}_\theta^t)$ holds only if $\bar{\boldsymbol{z}}_\theta^{tu} = \bar{\boldsymbol{z}}_\theta^t$, this requires the network to make no prediction error between $\bar{\boldsymbol{z}}_\theta^{tu}$ and $\bar{\boldsymbol{z}}_\theta^t$. However, $\bar{\boldsymbol{z}}_\theta^{tu} - \bar{\boldsymbol{z}}_\theta^t$ is practically non-zero, so we claim that this prediction error needs to be considered to bridge the training-inference gap.

To measure both *local* (*per-step*) and *global* (*cumulative*) prediction error, we modify the sampling process by substituting the previous estimate $\bar{\boldsymbol{z}}_\theta^{tu}$ with the current-step estimate $\hat{\boldsymbol{z}}_\theta^t$ of $\bar{\boldsymbol{z}}_t$. Local estimation error (vertical direction) is quantified by comparing the training distribution $q(\bar{\boldsymbol{z}}_0)$ with the sampling distribution $q(\boldsymbol{z}_0)$ using BLEU scores at t-step reverse process. Total estimation error (horizontal direction) is assessed across different numbers of denoising steps (NFEs). The results are shown in Table 1.

The table shows that the original self-conditioning leads to the trend: *the smaller the NFEs, the lower the BLEU scores*, which implies greater impact of local approximation error as t decreases. These results reveal a *training and sampling empirical gap*.

Model	Number of denoising steps (NFEs)				
	5	20	50	100	1,000
Original	27.85	29.83	29.97	30.10	30.12
Correct	29.70	30.21	30.34	30.20	30.23
Sup $\mathbb{E}$	0.047	0.008	0.009	0.010	-

Table 1: BLEU scores and average estimation gap ($\forall t$) on the original and corrected self-condition on baselines trained with $T = 2000$, evaluated on IWSLT14 De-En.

3.2 Potential Benefits

Increasing the number of denoising steps decreases prediction error. In contrast to prior studies, which suggested that error could accumulate across steps (Ning et al., 2023), we observed that *the estimation gap decreases as NFEs increase*. This implies that, given a sufficiently large number of denoising steps, the discretization errors between t and u become small enough for the model to estimate the correct self-condition, as shown in the following theorem:

Theorem 1 *Let $t_0, t_1, ..., t_n \in [\epsilon, 1]$ such that $t_0 < t_1 < ... < t_n = 1$; $\Delta t := \max_{i\in[1,n-1]}\{|t_{i+1} - t_i|\}$. Assume D_θ satisfies the Lipschitz condition: there exists $K > 0$ such that for all $t \in [\epsilon, 1]$, $\boldsymbol{x}$, and $\boldsymbol{y}$, we have $||D_\theta(\boldsymbol{z}_t, \boldsymbol{x}) - D_\theta(\boldsymbol{z}_t, \boldsymbol{y})||_2 \leq K||\boldsymbol{x} - \boldsymbol{y}||_2$. Assume further that for all $i \in [0, n-1]$, the denoising estimation at t_{i+1} has local error uniformly bounded by $\mathcal{O}((t_{i+1} - t_i)^{p+1})$ with $p \geq 1$. Then, the supremum of local error expectation:*

$$\sup_{i\sim[0,n-1]} \mathbb{E}\left[\|D_\theta(\boldsymbol{z}_{t_i}, \bar{\boldsymbol{z}}_\theta^{t_{i+1}t_{i+2}}) - D_\theta(\boldsymbol{z}_{t_i}, \hat{\boldsymbol{z}}_\theta^{t_i})\|\right] = \mathcal{O}((\Delta t)^p). \quad (8)$$

Theorem 1 suggests that the self-condition estimation can become arbitrarily accurate, as long as the number of sampling steps T, or NFEs correspondingly, are large enough (we provide the full proof in Appendix A.2). Table 1 (third row) shows that the empirical bound in Eq. 8 is smaller as NFE increases, further confirm NFE $= 20$ is sufficient to close the training-inference gap, while performance gains diminished after NFE > 20.

Few match Many-step sampling. Table 1 (second row) shows that the BLEU score of the corrected self-condition slightly drop when NFE is reduced from 20 to 5, compared to the original version. This observation highlights the importance of a training design that aligns with the inference process, enabling optimal inference efficiency.

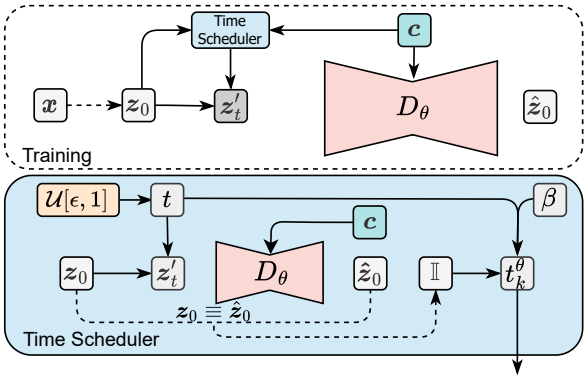

Figure 1: Overall pipeline of FastDiSS, including Leakage-informed Noise Scheduler and Model-Aware Noise Scaling.

Algorithm 1 Training with Leakage-informed Noise Scheduler

Input: Text sequence $\boldsymbol{x}$, denoising network D_θ

while not converged **do**
  $\boldsymbol{z}_0 \sim q_\theta(\boldsymbol{z}_0|\boldsymbol{x})$
  $t \sim \mathcal{U}(\epsilon, 1)$, $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$
  $\boldsymbol{z}'_t = \alpha_t \lambda_t \boldsymbol{z}_0 + \sigma_t \sqrt{1+\gamma_t^2}\boldsymbol{\epsilon}$
  $\hat{\boldsymbol{z}}_\theta^t \leftarrow D_\theta(\boldsymbol{z}'_t, 0)$
  $r \sim \mathcal{U}(0, 1)$
  $\boldsymbol{z}_\theta^t \leftarrow D_\theta(\boldsymbol{z}'_t, \hat{\boldsymbol{z}}_\theta^t)$ if $r < 0.5$ else $\hat{\boldsymbol{z}}_\theta^t$
  $\mathcal{L}_{\text{total}} = ||\boldsymbol{z}_\theta^t - \boldsymbol{z}_0||^2 - \log p_\theta(\boldsymbol{x}|\boldsymbol{z}_0)$
  $\theta \leftarrow \theta - \nabla_\theta \mathcal{L}_{\text{total}}$
end while

4 FastDiSS

In this section, we introduce FastDiSS, a novel modular training framework to improve text diffusion training (Figure 1). In Section 4.1, we introduce *Leakage-informed Noise Scheduler (SNL)* that bridges the training-inference gap. In Section 4.2, we interpret this design via robustness analysis, revealing that the forward perturbation resembles empirical inference behavior. In Section 4.3, we propose *Model-Aware Noise Scaling (MANS)*, which dynamically adjusts token-level noise based on model confidence, effectively tailoring training difficulty to token-level denoising.

4.1 Leakage-informed Noise Scheduler (SNL)

Our objective is to address exposure bias arising from the training-sampling mismatch, thereby improving sampling efficiency. Our approach emphasizes an end-to-end training pipeline, without modification to the sampling stage. Specifically, we modify the forward process with a new leakage schedule, with the implication that it captures the inference estimation $\bar{\boldsymbol{z}}_\theta^{uk}$ $(u > t)$:

$$\boldsymbol{z}'_t = \alpha_t \lambda_t \boldsymbol{z}_0 + \sigma_t \sqrt{1+\gamma_t^2}\boldsymbol{\epsilon}_t. \tag{9}$$

The two terms λ_t and γ_t are linear functions, defined as $\lambda_t = (\lambda_{\min} - \lambda_{\max})t + \lambda_{\max}$ and $\gamma_t = (\gamma_{\min} - \gamma_{\max})t + \gamma_{\max}$, which also decrease/increase monotonically. This setup ensures that the noising process for $\boldsymbol{z}'_t$ is faster than $\boldsymbol{z}_t$, allowing it to approximate the denoising process from $\bar{\boldsymbol{z}}_u$ $(u > t)$. We discuss this in more detail in Section 4.2. The proposed training procedure for SNL is presented in Algorithm 1.

In general, a closer approximation to $\bar{\boldsymbol{z}}_u$ leads to better performance. However, there is no universally optimal scheduling function, as respace sampling (Nichol and Dhariwal, 2021) allows different choices for the number of denoising steps, with each choice requiring a different set of optimal hyperparameters. Essentially, we only align the early phase (large t, see Figure 2), since improving learning at smaller t does not significantly improve performance (Ye et al., 2023; Gao et al., 2024).

4.2 Leakage Noise Robustness

In this section, we show that the modified training process with SNL aligns with the inference process by facilitating the estimation of $\bar{\boldsymbol{z}}_\theta^{uk} \approx D_\theta(\boldsymbol{z}'_t)$. We also verify that the linear design for λ_t and γ_t is reasonable. Akin to Section 3, we trained the standard text diffusion model on the IWSLT14 De-En dataset. At test time, we applied the original 20-step denoising process to the validation set. At each step t, we preserved the previous step's estimation $\bar{\boldsymbol{z}}_\theta^{uk}$ and also computed the corrected self-condition $\hat{\boldsymbol{z}}_\theta^t = D_\theta(\bar{\boldsymbol{z}}_t, 0)$. We empirically found that $\bar{\boldsymbol{z}}_\theta^{uk}$ is normally distributed around $\hat{\boldsymbol{z}}_\theta^t$:

$$\bar{\boldsymbol{z}}_\theta^{uk} \sim \mathcal{N}(\hat{\boldsymbol{\mu}}_{ut}\hat{\boldsymbol{z}}_\theta^t, \hat{\boldsymbol{\sigma}}_{ut}^2 \boldsymbol{I}), \tag{10}$$

with mean vector $\hat{\boldsymbol{\mu}}_{ut}$ and standard deviation vector $\hat{\boldsymbol{\sigma}}_{ut}$ (see Appendix B). From this indication, we derive a side-by-side connection with $\boldsymbol{z}'_t$. Considering at testing, suppose that $\hat{\boldsymbol{z}}_\theta^t$ fully denoise $\bar{\boldsymbol{z}}_t$, we substitute $\hat{\boldsymbol{z}}_\theta^t$ by $\bar{\boldsymbol{z}}_\theta^{uk}$ as:

$$\begin{aligned}
\bar{\boldsymbol{z}}_t &= \alpha_t \hat{\boldsymbol{z}}_\theta^t + \sigma_t \boldsymbol{\epsilon}_t = \alpha_t \frac{\bar{\boldsymbol{z}}_\theta^{uk} - \hat{\boldsymbol{\sigma}}_{ut}\boldsymbol{\epsilon}_{\boldsymbol{u}}}{\hat{\boldsymbol{\mu}}_{ut}} + \sigma_t \boldsymbol{\epsilon}_t \\
&= \alpha_t \frac{1}{\hat{\boldsymbol{\mu}}_{ut}} \bar{\boldsymbol{z}}_\theta^{uk} + \sigma_t \sqrt{1 + \left(\frac{\alpha_t}{\sigma_t}\frac{\hat{\boldsymbol{\sigma}}_{ut}}{\hat{\boldsymbol{\mu}}_{ut}}\right)^2}\boldsymbol{\epsilon} \\
&= \alpha_t \hat{\boldsymbol{\lambda}}_{ut} \bar{\boldsymbol{z}}_\theta^{uk} + \sigma_t \sqrt{1+\hat{\boldsymbol{\gamma}}_{ut}^2}\boldsymbol{\epsilon},
\end{aligned} \tag{11}$$

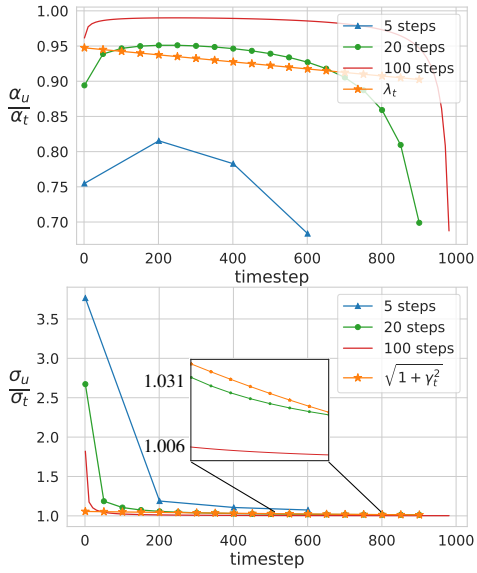

Figure 2: Mean (top) and variance (bottom) ratio differences measured at different denoising steps. We manually select hyperparameters of the scaling terms to fit 20-step sampling.

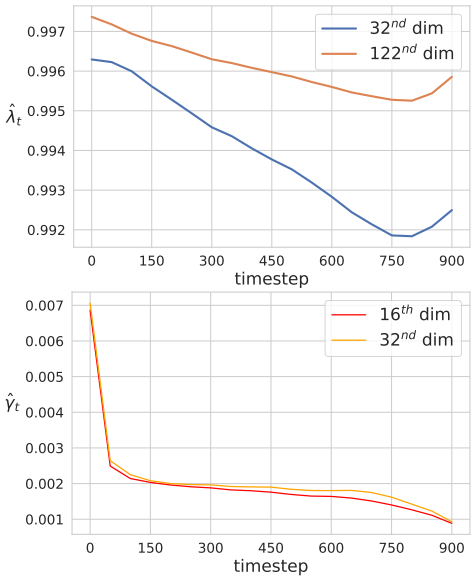

Figure 3: Inference mean $\hat{\lambda}_t$ (top) and variance $\hat{\gamma}_t$ (bottom) scaling factors of prediction mismatch in a pretrained network, plotted across timestep t for randomly selected embedding dimensions.

where $\hat{\boldsymbol{\lambda}}_{ut} = 1/\hat{\boldsymbol{\mu}}_{ut}$, $\hat{\boldsymbol{\gamma}}_t = (\alpha_t/\sigma_t)(\hat{\boldsymbol{\sigma}}_{ut}/\hat{\boldsymbol{\mu}}_{ut})$, and $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$ is obtained using the reparametrization trick on the terms containing $\boldsymbol{\epsilon}_u$ and $\boldsymbol{\epsilon}_t$. Since $\hat{\boldsymbol{\mu}}_{ut}$ and $\hat{\boldsymbol{\sigma}}_{ut}$ vary linearly and independently across dimensions, all operations remain element-wise. Apparently, we observed a similar pattern between Eq. 9 and 11, as well as the empirically estimation between Figure 2 and 3. This indicates that SNL explicitly informs the training process of $\bar{\boldsymbol{z}}_\theta^{uk}$.

In principle, empirical schedules $\{\hat{\boldsymbol{\lambda}}_{ut}, \hat{\boldsymbol{\gamma}}_{ut}\}_{t=1}^T$ could be derived from Eq.11. However, the dependence on discretization steps, along with the linear and dimension-wise variation of $\hat{\boldsymbol{\lambda}}_{ut}$ and $\hat{\boldsymbol{\gamma}}_{ut}$, precludes expression through a single global function. To avoid complicated hyperparameter searches, we instead select anchor points $(\lambda_{\min}, \lambda_{\max})$ and $(\gamma_{\min}, \gamma_{\max})$ *independently of feature space*. This approach enables an end-to-end training procedure that preserves the original dynamics without incurring additional training complexity.

4.3 Model-Aware Noise Scaling (MANS)

Due to the sparsity of the text embedding space, continuous text diffusion models can easily reconstruct clean embeddings from noises without learning the meaning of the whole sequence. This limits the learning capability of continuous text diffusion models, leading to underfitting. Previous research commonly addressed this issue in two ways: *(1) reformulating the forward process (through α_t and σ_t)* (Li et al., 2022) and *(2) modifying the standard uniform t-sampling distribution* (Tang et al., 2023; Ye et al., 2023; Chen et al., 2023a). These scheduling strategies are designed to ensure that training remains nontrivial, i.e., each noise level contributes to generation quality. However, most existing methods assess importance by averaging training losses across the entire batches, accidentally neglecting high-impact tokens. Recent studies have attempted to incorporate token-level awareness through manually designed time sampling schemes (Wu et al., 2023; Yuan et al., 2024).

We hypothesize that during the initial stages of training, the model quickly learns to predict easy tokens (common patterns) (Chen et al., 2023a; Ou and Jian, 2024), while harder tokens (uncommon patterns) remain underfit due to the model's limited capacity to capture them early on. In the later phases, as predictions for easy tokens become trivial, the model tends to overfit on the minority tokens. This is reflected in marginal reductions in loss, while the performance plateaus (Figure 4). We refer to this as the memorization issues of deep networks (Zhang et al., 2016; Bai et al., 2021).

To balance the learning contribution across tokens, we selectively apply higher noise levels to easy tokens. Token importance is quantified based on the model's reconstruction confidence, measured by the likelihood $P[D_\theta(z_t^k) = k]$, where z_t^k

Method	LB	NB	NFE	IWSLT14 DE→EN	IWSLT14 EN→DE	WMT14 DE→EN	WMT14 EN→DE	WMT16 RO→EN	WMT16 EN→RO
Transformer	5		-	33.61	28.30	30.55	26.85	33.08	32.86
CMLM	5		-	29.41	24.34	28.71	23.22	31.13	31.26
DiffusionLM‡	5	10	20	29.11	22.91	19.69	17.41	30.17	29.39
Diffomer‡	7	3	20	28.01	23.31	25.30	23.80	29.37	29.20
DINOISER	5	10	20	31.61	25.70	29.05	24.26	31.22	31.08
DiffuSeq	1	10	2000	29.43	-	22.72	-	-	-
SeqDiffuSeq	1	10	2000	30.45	22.12	23.93	19.76	-	-
AR-Diffusion	1	10	20	31.80	26.01	-	-	-	-
FastDiSS	10	1	5	32.46	25.73	29.54	24.33	31.55	30.88
FastDiSS	10	2	5	32.70	26.02	<u>29.75</u>	**24.69**	31.81	30.90
FastDiSS	10	1	20	<u>32.81</u>	<u>26.29</u>	29.47	24.50	<u>31.89</u>	<u>31.37</u>
FastDiSS	10	2	20	**32.88**	**26.39**	**29.83**	<u>24.57</u>	**31.99**	**31.44**

Table 2: Main results on Machine Translation. The best NAR results are **bold** and the second-best results are <u>underlined</u>. ‡ indicates results reported by Ye et al. (2023). Other results are from their original papers.

represents the noisy embedding of token $k \in [V]$ at time t, and V denotes the vocabulary size. Based on this evaluation metric, we propose a noise scaling schedule that progressively increases the denoising difficulty of $\boldsymbol{z}_t$. Specifically, for each token k that is fully reconstructed to match the ground truth, we scale its noise level by incrementing the corresponding time step t_k according to a model-aware scaling function $\beta(n)$ at iteration n,

$$t_k^\theta = \begin{cases} \beta(n) \cdot t_k & \text{if} \quad D_\theta(z_t^k) = k, \\ t_k & \text{otherwise.} \end{cases} \tag{12}$$

The β-function divides training into several phases, and we apply a fixed scale for each phase. Essentially, MANS leverages the instant feedback at every training iteration to ensure effective training.

5 Experiments

5.1 Settings

Datasets. For Machine Translation, we follow previous works with benchmark datasets IWSLT14 (En-De/De-En) (Cettolo et al., 2014), WMT14 (En-De/De-En), and WMT16 (En-Ro/Ro-En) (Bojar et al., 2014). For Text Summarization, experiments are conducted on Gigaword (Narayan et al., 2018). For Question Paraphrase, experiments are conducted on QQP from the community question answering forum Quora. For Text Simplification, we follow DiffuSeq (Gong et al., 2023a) and conduct experiments on Wiki-Auto.

Evaluation Metrics. We report the SacreBLEU for Machine Translation, following Ye et al. (2023); Gong et al. (2023a). We follow Qi et al. (2020) and report the Rouge-1/2/L for the summarization task. For Question Paraphrase and Text Simplification, we follow DiffuSeq to employ sentence-level BLEU under the tokenizer of `bert-base-uncased`, Rouge-L, and BERTScore (Zhang et al., 2019) for quality assessment, and sentence-level self-BLEU (Zhu et al., 2018) for diversity assessment.

Baselines. We include three groups of baselines: (1) The autoregressive models with Transformer (AR) (Vaswani et al., 2017), GRU with attention, and the fine-tuned GPT2-large; (2) The non-autoregressive model (NAR) with CMLM (Ghazvininejad et al., 2019) and LevT (Gu et al., 2019); (3) Continuous diffusion-based language models (Continuous DLM) include DiffusionLM (Li et al., 2022), Diffomer (Gao et al., 2024), DINOISER (Ye et al., 2023), DiffuSeq (Gong et al., 2023a), SeqDiffuSeq (Yuan et al., 2024), and AR-Diffusion (Wu et al., 2023). For Text Summarization, we also include LSTM (Greff et al., 2016) for AR, NAG-BERT (Su et al., 2021) for NAR. For Question Paraphrase, we include Discrete DLM with RDM (Zheng et al., 2023).

Training and Inference. During training, we adopt `sqrt` noise schedule (Li et al., 2022) with $T = 2000$ diffusion steps. The anchor points $(\lambda_{\min}, \lambda_{\max})$ and $(\gamma_{\min}, \gamma_{\min})$ are $(0.9, 0.95)$ and $(0.15, 0.35)$, respectively. MANS is randomly applied with 50% probability to ensure that the model experiences noise at every level, including low-level noise. The detailed scaling settings are specified in Appendix E.1. Our implementation is based on Diffomer, using `fairseq`, with Transformer and sampling configurations from prior work, with NFE $= 2, 5, 20$. For every task, the vocabulary is constructed based on Byte Pair Encoding (BPE) (Kudo and Richardson, 2018). Further details are

Method	LB	NB	NFE	QQP BLEU↑	QQP Rouge-L↑	QQP BertScore↑	Wiki-Auto BLEU↑	Wiki-Auto Rouge-L↑	Wiki-Auto BertScore↑
Transformer‡	-	-	-	27.22	57.48	83.81	26.93	49.07	73.81
GPT2-large FT‡	-	-	-	20.59	54.15	83.63	26.93	51.11	78.82
CMLM×	-	-	10	21.78	56.12	-	35.26	58.46	81.83
LevT‡	-	-	-	22.68	57.95	83.44	20.52	44.02	72.54
Differmer*	1	10	20	27.95	59.24	82.97	34.78	54.55	78.86
DINOISER†	1	10	20	19.49	53.16	80.36	23.88	48.21	67.87
DiffuSeq	1	10	2000	24.13	58.80	**83.65**	36.22	58.49	81.26
SeqDiffuSeq	1	10	2000	24.34	-	**84.00**	37.12	-	82.14
FastDiSS	10	1	2	27.94	58.47	81.81	40.23	59.10	81.60
FastDiSS	10	1	5	**28.88**	**59.34**	82.58	**40.90**	**59.64**	82.16
FastDiSS	10	1	20	28.32	58.88	82.62	40.81	**59.64**	**82.17**

Table 3: Main results on QQP and Wiki-Auto. The best NAR results are **bold** and the second-best results are underlined. ‡, ×, and † indicate results reported by Gong et al. (2023a), Tang et al. (2023), and Chuang et al. (2024), respectively. ∗ indicates reproduced results. Other results are from their original papers.

Method	LB	NB	NFE	Rouge-1 (↑)	Rouge-2 (↑)	Rouge-L (↑)
LSTM	5		-	34.2	16.0	31.8
CMLM	-	-	-	34.4	15.6	32.2
NAG-BERT	-	-	-	35.1	16.5	**33.3**
Differmer*	5	2	20	34.9	17.0	32.4
DiffuSeq	1	10	2000	31.2	12.2	29.2
SeqDiffuSeq	1	10	2000	31.9	12.4	29.2
FastDiSS	5	2	5	34.9	16.9	32.5
FastDiSS	5	2	20	**35.3**	**17.3**	32.8

Table 4: Main results on Text Summarization. The best NAR results are **bold** and the second-best results are underlined. Baseline results are from Differmer (Gao et al., 2024). ∗ indicates reproduced results.

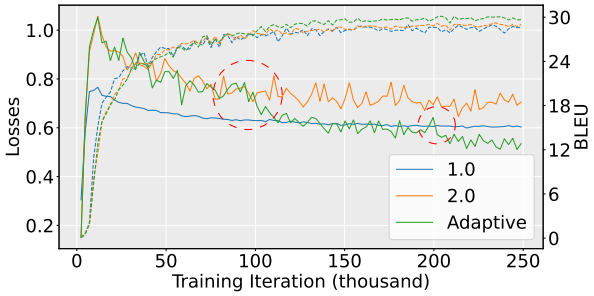

Figure 4: Loss curve and BLEU score follow the training iterations. The dashed line represents the estimated BLEU score, with colors corresponding to the loss curves for each respective setting.

described in Appendix E. We also apply Minimum Bayes Risk (MBR) (Kumar and Byrne, 2004) following previous works (Li et al., 2022; Gong et al., 2023a; Dieleman et al., 2022) with different length beam (**LB**) and noise beam (**NB**).

5.2 Main Results

Overall performance. The results of different datasets are shown in Table 2, 3, and 4. FastDiSS outperforms both DLM and NAR baselines on most datasets with different choices of beam size and NFEs, and even performs comparably with AR. In some cases, such as in WMT14, the 5-step even surpasses 20-step sampling, demonstrating that SNL effectively bridges the gap between few and many-step sampling. With SNL training, the sampling speed is 4× faster than DINOISER and Differmer.

Training Convergence. Figure 4 compares the validation loss during training when not applying scaling (1.0), applying fixed scaling (2.0), and adaptive scaling. Fixed scaling does not improve performance, while adaptive scaling *continues to improve performance and reduce the loss*. The result confirms that gradually increasing the noise scale allows the model to converge better, even with high noise levels in the late training phase.

Accelerated sampling comparison. We compare our method with DiffuSeq-v2 (Gong et al., 2023b) to showcase the setting in a practical application. Figure 6 shows that FastDiSS dominates DiffuSeq-v2 at every NFEs, with optimal performance achieving at low NFE values.

Sampling Diversity. Figure 5 shows the diversity and quality correlations. FastDiSS-2NFE matches the performance of the best baseline, Differmer-20NFE, with only 2 sampling steps. Increasing the number of denoising steps trades the inference speed for quality. FastDiss-5NFE is slower than FastDiss-2NFE but has significantly higher BLEU and about the same Self-BLEU scores.

5.3 Ablation Studies

Effect of different components. As can be seen from Table 5, both MANS and SNL enhance the

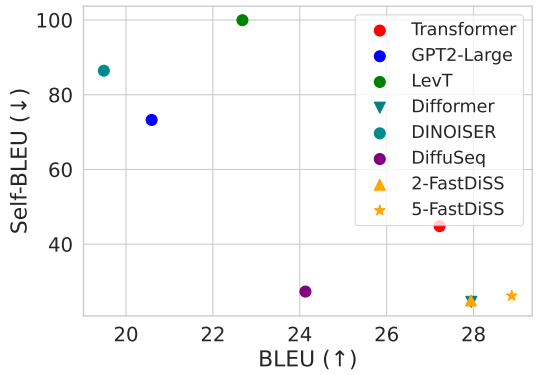

Figure 5: Diversity and quality comparison between baseline and FastDiSS. The results are from DiffuSeq (Gong et al., 2023a).

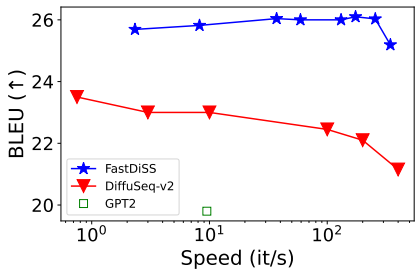

Figure 6: Generation speed and quality with different NFE. The speed is averaged over 3 runs.

SNL	MANS	NFE=5	NFE=20	ΔNFE
		27.98	29.78	1.80
✓		29.64	30.36	0.72
	✓	30.77	31.49	0.72
✓	✓	31.17	31.66	0.49

Table 5: Ablation on each component of the proposed methods. Results are reported on IWSLT14 De-En with LB = 2, NB = 2.

performance. Notably, combining MANS with SNL narrows the performance gap between 5 and 20 steps.

Comparison between different numbers of denoising steps. Figure 7 illustrates the effectiveness of our proposed method with few-step denoising. Our method outperforms the 20-step baseline after only 3 sampling steps, yielding $\sim 7\times$ speedup. Additionally, our sampling method converges after only 7 steps, at a significantly higher BLEU score than the 20-step baseline.

Other noise schedulers. Table 6 demonstrates the performance on popular noise schedules, including `linear` (Ho et al., 2020) and `cosine` (Nichol and Dhariwal, 2021). The `linear` schedule is less sensitive to the NFE value. Our method strongly boosts the score in the few-step settings.

Parameters of the noise scaling terms γ_t and λ_t. In Table 7, we show the performance with different

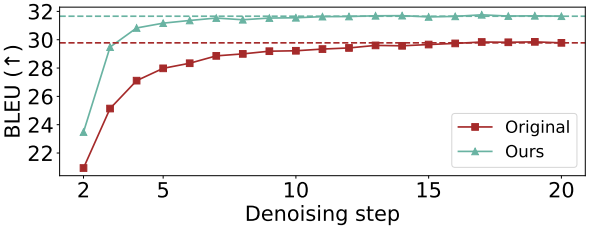

Figure 7: Performance difference between the original and our proposed method. Results are reported on IWSLT14 De-En with the same seed.

Noise Schedule		NFE=2	NFE=5	NFE=20
Linear	Orig.	26.50	27.17	**27.54**
	Ours	**26.84**	**27.52**	**27.54**
Cosine	Orig.	25.43	27.05	27.57
	Ours	**27.32**	**27.77**	**27.94**

Table 6: BLEU Score on QQP with different noise schedules. Results are reported with LB = 2, NB = 1.

Steps	5	20	100	Fixed
λ	(0.60,0.85)	(0.90,0.95)	(0.98,0.99)	0.85
γ	(0.25,0.90)	(0.15,0.35)	(0.05,0.15)	0.50
BLEU	25.48	**26.35**	25.70	25.84
R-L	57.29	**57.47**	56.82	57.18

Table 7: Performance on QQP with different variants of SNL. Results are reported with NFE = 2, LB = 5, NB = 1.

variants of SNL, including both dynamics and fixed schedules, where we set λ_t and γ_t to be constant, instead of a linear-time function. The results indicate that the best performance lies in the leakage-informed level at 20-step denoising. Hence, we use this setting for the remaining benchmarks.

6 Conclusion

In this paper, we propose FastDiSS, an improved training framework for diffusion-based language models that (1) addresses the training–inference mismatch inherent in self-conditioning mechanisms, while (2) accelerates convergence in the later training phases, which are typically slow and difficult to optimize. Experimental results on various benchmarks indicate that the training is more effective and the gap between few and many-step decoding is bridged. While we believe that our work has a substantial impact on the real-world application of diffusion language models, we encourage a better method for aligning self-conditioning to approach the performance of autoregressive models. This study offers a promising perspective on the development of few-step language models.

Limitation

In this study, since we aim to align training to inference, the proposed techniques are constrained by the goodness of the previous self-conditioning prediction. Hence, it would be more promising if we approached the problem from the opposite direction, as refining the prediction to be more accurate would significantly boost the performance. On the other hand, our current noise scaling strategy relies on predefined values at each training phase, which may be suboptimal when earlier scaling stages are insufficiently trained. A more adaptive scaling function could further enhance performance.

References

Marianne Arriola, Aaron Gokaslan, Justin T Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. 2025. Block diffusion: Interpolating between autoregressive and diffusion language models. *arXiv preprint arXiv:2503.09573*.

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.

Yingbin Bai, Erkun Yang, Bo Han, Yanhua Yang, Jiatong Li, Yinian Mao, Gang Niu, and Tongliang Liu. 2021. Understanding and improving early stopping for learning with noisy labels. *Advances in Neural Information Processing Systems*, 34:24392–24403.

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.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

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.

Jiaao Chen, Aston Zhang, Mu Li, Alex Smola, and Diyi Yang. 2023a. A cheaper and better diffusion language model with soft-masked noise. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4765–4775, Singapore. Association for Computational Linguistics.

Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. 2023b. Analog bits: Generating discrete data using diffusion models with self-conditioning. *International Conference on Learning Representations*.

Yun-Yen Chuang, Hung-Min Hsu, Kevin Lin, Chen-Sheng Gu, Ling Zhen Li, Ray-I Chang, and Hung-yi Lee. 2024. Meta-diffub: A contextualized sequence-to-sequence text diffusion model with meta-exploration. *Advances in neural information processing systems*.

Sander Dieleman, Laurent Sartran, Arman Roshannai, Nikolay Savinov, Yaroslav Ganin, Pierre H Richemond, Arnaud Doucet, Robin Strudel, Chris Dyer, Conor Durkan, et al. 2022. Continuous diffusion for categorical data. *arXiv preprint arXiv:2211.15089*.

Zhujin Gao, Junliang Guo, Xu Tan, Yongxin Zhu, Fang Zhang, Jiang Bian, and Linli Xu. 2024. Empowering diffusion models on the embedding space for text generation. 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 4664–4683. Association for Computational Linguistics.

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.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. 2023a. DiffuSeq: Sequence to sequence text generation with diffusion models. In *International Conference on Learning Representations, ICLR*.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. 2023b. Diffuseq-v2: Bridging discrete and continuous text spaces for accelerated seq2seq diffusion models. *arXiv preprint arXiv:2310.05793*.

Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. 2016. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor OK Li, and Richard Socher. 2017. Non-autoregressive neural machine translation. *arXiv preprint arXiv:1711.02281*.

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, Online. Association for Computational Linguistics.

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

Ishaan Gulrajani and Tatsunori B Hashimoto. 2023. Likelihood-based diffusion language models. *Advances in Neural Information Processing Systems*, 36:16693–16715.

Jiaxin Guo, Minghan Wang, Daimeng Wei, Hengchao Shang, Yuxia Wang, Zongyao Li, Zhengzhe Yu, Zhanglin Wu, Yimeng Chen, Chang Su, et al. 2021. Self-distillation mixup training for non-autoregressive neural machine translation. *arXiv preprint arXiv:2112.11640*.

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

Fei Huang, Tianhua Tao, Hao Zhou, Lei Li, and Minlie Huang. 2022. On the learning of non-autoregressive transformers. In *International conference on machine learning*, pages 9356–9376. PMLR.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. 2022. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. 2021. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Shankar Kumar and Bill Byrne. 2004. Minimum bayes-risk decoding for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 169–176.

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.

Min Liu, Yu Bao, Chengqi Zhao, and Shujian Huang. 2023. Selective knowledge distillation for non-autoregressive neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 13246–13254.

Aaron Lou, Chenlin Meng, and Stefano Ermon. 2024. Discrete diffusion language modeling by estimating the ratios of the data distribution. In *International conference on machine learning*.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. 2022a. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. 2022b. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *arXiv preprint arXiv:1808.08745*.

Alexander Quinn Nichol and Prafulla Dhariwal. 2021. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR.

Mang Ning, Mingxiao Li, Jianlin Su, Albert Ali Salah, and Itir Onal Ertugrul. 2024. Elucidating the exposure bias in diffusion models. *International Conference on Learning Representations*.

Mang Ning, Enver Sangineto, Angelo Porrello, Simone Calderara, and Rita Cucchiara. 2023. Input perturbation reduces exposure bias in diffusion models. In *International Conference on Machine Learning*, pages 26245–26265. PMLR.

Yimin Ou and Ping Jian. 2024. Effective integration of text diffusion and pre-trained language models with linguistic easy-first schedule. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 5551–5561.

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.

Weizhen Qi, Yeyun Gong, Yelong Shen, Jian Jiao, Yu Yan, Houqiang Li, Ruofei Zhang, Weizhu Chen, and Nan Duan. 2022. A self-paced mixed distillation method for non-autoregressive generation. *arXiv preprint arXiv:2205.11162*.

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.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

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.

Qiu Ran, Yankai Lin, Peng Li, and Jie Zhou. 2020. Learning to recover from multi-modality errors for non-autoregressive neural machine translation. *arXiv preprint arXiv:2006.05165*.

Subham Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin Chiu, Alexander Rush, and Volodymyr Kuleshov. 2024. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184.

Florian Schmidt. 2019. Generalization in generation: A closer look at exposure bias. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 157–167, Hong Kong. Association for Computational Linguistics.

Chenze Shao, Xuanfu Wu, and Yang Feng. 2022. One reference is not enough: Diverse distillation with reference selection for non-autoregressive translation. *arXiv preprint arXiv:2205.14333*.

Chenze Shao, Jinchao Zhang, Jie Zhou, and Yang Feng. 2023. Rephrasing the reference for non-autoregressive machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 13538–13546.

Samuel Sanford Shapiro and Martin B Wilk. 1965. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3-4):591–611.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*.

Yang Song and Prafulla Dhariwal. 2023. Improved techniques for training consistency models. *arXiv preprint arXiv:2310.14189*.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. 2023. Consistency models.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2021. Score-based generative modeling through stochastic differential equations. *International Conference on Learning Representations*.

Yixuan Su, Deng Cai, Yan Wang, David Vandyke, Simon Baker, Piji Li, and Nigel Collier. 2021. Non-autoregressive text generation with pre-trained language models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 234–243, Online. Association for Computational Linguistics.

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! *arXiv preprint arXiv:2305.04465*.

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.

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.

Jiasheng Ye, Zaixiang Zheng, Yu Bao, Lihua Qian, and Mingxuan Wang. 2023. Dinoiser: Diffused conditional sequence learning by manipulating noises. *arXiv preprint arXiv:2302.10025*.

Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Fei Huang, and Songfang Huang. 2024. Text diffusion model with encoder-decoder transformers for sequence-to-sequence generation. 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 22–39, Mexico City, Mexico. Association for Computational Linguistics.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2016. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*.

Kexun Zhang, Rui Wang, Xu Tan, Junliang Guo, Yi Ren, Tao Qin, and Tie-Yan Liu. 2022. A study of syntactic multi-modality in non-autoregressive machine translation. *arXiv preprint arXiv:2207.04206*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Lin Zheng, Jianbo Yuan, Lei Yu, and Lingpeng Kong. 2023. A reparameterized discrete diffusion model for text generation. *arXiv preprint arXiv:2302.05737*.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 1097–1100.

A Theoretical Details

A.1 Derivation of the posterior distribution $q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{z}_0)$

Since the forward process is a Markov chain, for $t > s$, we have $q(\boldsymbol{z}_s, \boldsymbol{z}_t|\boldsymbol{z}_0) = q(\boldsymbol{z}_s|\boldsymbol{z}_0)q(\boldsymbol{z}_t|\boldsymbol{z}_s)$.

Following the Bayes rule, the posterior equals to $q(\boldsymbol{z}_t|\boldsymbol{z}_s)q(\boldsymbol{z}_s|\boldsymbol{z}_0)/q(\boldsymbol{z}_t|\boldsymbol{z}_0)$. Given that every term is a Gaussian likelihood (Eq. 1 and 2), we plug this into the posterior, which yields:

$$q(\boldsymbol{z}_s|\boldsymbol{z}_t,\boldsymbol{z}_0)=\mathcal{N}(\boldsymbol{z}_s;\tilde{\boldsymbol{\mu}}(\boldsymbol{z}_t,\boldsymbol{z}_0),\tilde{\sigma}_t^2\boldsymbol{I}) \quad (13)$$

$$\text{where}\quad \tilde{\boldsymbol{\mu}}(\boldsymbol{z}_t,\boldsymbol{z}_0)=\frac{\alpha_t}{\alpha_s}\frac{\sigma_s^2}{\sigma_t^2}\boldsymbol{z}_t+\alpha_s\frac{\sigma_{t|s}^2}{\sigma_t^2}\boldsymbol{z}_0 \quad (14)$$

$$\text{and}\qquad \tilde{\sigma}_t=\frac{\sigma_s}{\sigma_t}\sigma_{t|s}. \quad (15)$$

A.2 Proof of Theorem 1

Because $D_\theta(\boldsymbol{z}_{t_i},\cdot)$ is K-Lipschitz, we have

$$\begin{aligned}&\mathop{\mathbb{E}}_{i\sim[0,n-1]}\|D_\theta(\boldsymbol{z}_{t_i},\bar{\boldsymbol{z}}_\theta^{t_{i+1}t_{i+2}})-D_\theta(\boldsymbol{z}_{t_i},\hat{\boldsymbol{z}}_\theta^{t_i})\|\\&\le K\mathop{\mathbb{E}}_{i\sim[0,n-1]}\|\bar{\boldsymbol{z}}_\theta^{t_{i+1}t_{i+2}}-\hat{\boldsymbol{z}}_\theta^{t_i}\|\end{aligned}$$

Furthermore, from our assumption that the local error is bounded by $\mathcal{O}((t_{i+1}-t_i)^{p+1})$, which is partially observed from the empirical estimation in Eq. 10, we have

$$\begin{aligned}&K\mathop{\mathbb{E}}_{i\sim[0,n-1]}\|\bar{\boldsymbol{z}}_\theta^{t_{i+1}t_{i+2}}-\hat{\boldsymbol{z}}_\theta^{t_i}\|\\&\overset{(i)}{\le}\frac{K}{n}\cdot\sum_{i=0}^{n-1}\mathcal{O}((t_{i+1}-t_i)^{p+1})\\&\le\sum_{i=0}^{n-1}\mathcal{O}((t_{i+1}-t_i)^{p+1})\\&=\sum_{i=0}^{n-1}(t_{i+1}-t_i)\mathcal{O}((t_{i+1}-t_i)^{p})\\&\le\mathcal{O}((\Delta t)^p)\sum_{i=0}^{n-1}(t_{i+1}-t_i)\\&=\mathcal{O}((\Delta t)^p)(t_n-t_0)\\&\le\mathcal{O}((\Delta t)^p)(1-\epsilon)\\&\le\mathcal{O}((\Delta t)^p)\end{aligned}$$

where (i) holds due to the uniform sampling distribution of t. Our proof builds on the error bounds for the ODE Solver in Consistency Models (Song et al., 2023), but it is different since we provide the total local error bounds rather than targeting the empirical approximation bounds.

B Estimate the Self-condition Gaussian Distribution Error

In this section, we conduct experiments on IWSLT14 De-En to empirically show that $\bar{\boldsymbol{z}}_\theta^{uk}\sim\mathcal{N}(\hat{\boldsymbol{\mu}}_t\hat{\boldsymbol{z}}_\theta^t,\hat{\boldsymbol{\sigma}}_t^2\mathbf{I})$ (Sec. 4.2), i.e., the prediction error of the previous step is nearly Gaussian distributed around current predictions. To do so, we need to prove that, for each t and each word embedding dimension $i\in\{1,...,H\}$, the dimension-wise error (ϵ_t^i) follows $\epsilon_t^i\sim\mathcal{N}(0,(\hat{\sigma}_t^i)^2)$, where $\boldsymbol{\epsilon}_t=\bar{\boldsymbol{z}}_\theta^{uk}-\hat{\boldsymbol{\mu}}_t\hat{\boldsymbol{z}}_\theta^t$. To test this hypothesis, we uniformly select 20 values of t in $[\epsilon,1]$ using a 0.05 discretization step size, then perform a denoising process through these selected t, then record the value of $\bar{\boldsymbol{z}}_\theta^t$ and $\hat{\boldsymbol{z}}_\theta^t$ in each t. Then, for each t, we use 1,000 sentences $z\in\mathcal{D}$ and flatten every token in each sentence to compute the dimension independent mean $\hat{\mu}_t^i$ and standard deviation $\hat{\sigma}_t^i$ of the error, which we use to standardize the error values for all the dimension ϵ_t^i (i.e., $\bar{\epsilon}_t^i=\frac{\epsilon_t^i-\mu_t}{\hat{\sigma}_t^i}$). Estimating $\hat{\mu}_t^i$ can be considered as solving a *linear regression problem*, using ordinary least squares (OLS): $\hat{\mu}_t^i=\frac{\sum_{z\in\mathcal{D}}\bar{z}_u^i\hat{z}_t^i}{\sum_{z\in\mathcal{D}}(\hat{z}_t^i)^2}$. For $\hat{\sigma}_t^i$, it is simply calculating the standard deviation of the residuals: $\hat{\sigma}_t^i=\sqrt{\frac{1}{|\mathcal{D}|}\sum_{z\in\mathcal{D}}(\bar{z}_u^i-\hat{\mu}_t^i\hat{z}_t^i)^2}$. Then, for each i, we use 50 randomly selected values and the Shapiro–Wilk test (Shapiro and Wilk, 1965) to verify that they follow a standard normal distribution. The confidence level is set at 95% and we reject the null hypothesis if the p-value is less than 0.05. The null hypothesis was rejected only in a small minority of cases, confirming our assumption. Figure 8 shows a few histogram examples for ϵ_t^i computed at different dimensions.

C Related Works

Non-autoregressive Language Generation. Non-autoregressive (NAR) models, introduced by Gu et al. (2017), aim to accelerate text generation compared to autoregressive counterparts. To improve performance, subsequent works adopt iterative refinement strategies (Gu et al., 2019; Ghazvininejad et al., 2019). However, due to the independence assumption between tokens, NAR models often struggle with multi-modality. Efforts to mitigate this include structured prediction techniques (Zhang et al., 2022; Ran et al., 2020; Huang et al., 2022), data selection through reference rephrasing (Shao et al., 2022, 2023), and model distillation to enhance learning (Guo et al., 2021; Qi et al., 2022; Liu et al., 2023).

Diffusion Models for Language Generation. Text diffusion tackle multi-modality in text generation and fall into two main categories: continuous and discrete. In continuous models, tokens

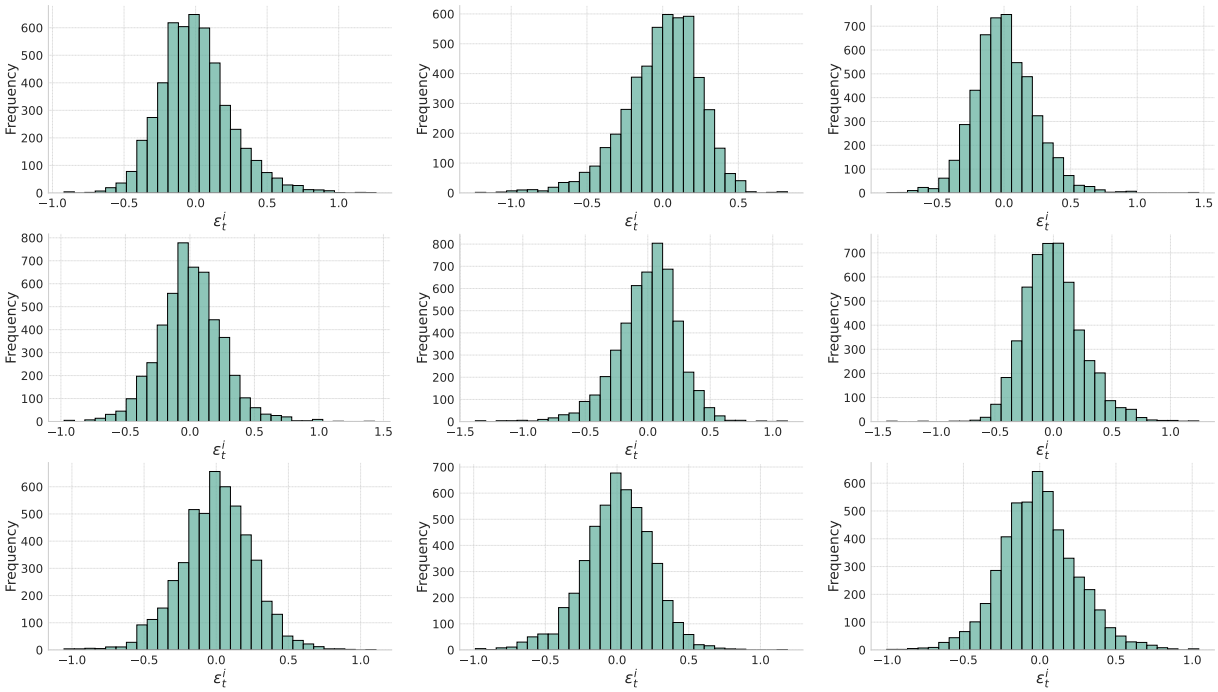

Figure 8: The empirical distribution of ϵ_t^i with different random values of t and i.

are mapped to latent embeddings where diffusion operates. DiffusionLM (Li et al., 2022) pioneered this approach, enabling conditional generation with a Transformer-based encoder-decoder architecture. Subsequent works, Difformer (Gao et al., 2024) and DINOISER (Ye et al., 2023), improve generation quality by addressing representation sparsity, while SeqDiffuSeq (Yuan et al., 2024), AR-Diffusion (Wu et al., 2023), Masked-Diffuse (Chen et al., 2023a)) introduce token-dependent noise schedules. DiffuSeq (Gong et al., 2023a) adopts a decoder-only design for better LLM alignment. In contrast, discrete models define transitions over token space, with early works like D3PM (Austin et al., 2021) and Analog-bits (Chen et al., 2023b) define discrete diffusion via absorbing or uniform transitions, while more recent methods (e.g., MDLM (Sahoo et al., 2024), RDM (Zheng et al., 2023)) adopt masking strategies that have been scaling to LLMs (Arriola et al., 2025).

Accelerating the Diffusion Language Model. While much of the existing literature focuses on improving generation quality and model scalability, fewer studies address the challenge of inference efficiency. In image generation, methods such as DDIM (Song et al., 2020) and Progressive Distillation have demonstrated success in reducing the number of denoising steps. Advances in ODE solvers (Lu et al., 2022a,b) and second-order trajectory approximations like EDM (Karras et al., 2022) have further enhanced the efficiency of the reverse process. More recently, Consistency Models (Song et al., 2023; Song and Dhariwal, 2023) enable high-quality generation with just a single sampling step. In the context of text, DiffuSeq-v2 (Gong et al., 2023b) improves inference by integrating continuous and masked denoising, while Tang et al. (2023) address training–inference discrepancies to enhance generation speed. Our work advances this line by promoting fast, adaptable, and few-step generation, aiming to make diffusion-based language models more viable for real-world applications.

D Comparison with Other Methods

SeqDiffuSeq introduces adaptive token-level noise scaling based on positional importance, assuming that token position, rather than token identity, determines difficulty. In contrast, **Masked-Diffuse** (Chen et al., 2023a) estimates token importance via linguistic rarity. Our approach differs by directly leveraging model confidence to assess token difficulty, with the hypothesis that models perceive language differently from humans.

Tang et al. (2023) also addresses exposure bias in text diffusion. Although their method shares conceptual similarities with ours, it does not incorporate self-conditioning. Moreover, their noise-aware strategy, **Distance Penalty**, applies a fixed penalty

Configurations	WMT14	WMT16	IWSLT14	Gigaword	QQP	Wiki-Auto
Split						
Training	4,500,966	608,319	160,215	3,803,957	144,715	677,751
Validation	3,000	1,999	7,282	189,651	2,048	2,048
Test	3,003	1,999	6,750	1,951	2,500	5,000
Preprocess						
BPE	40,000	30,000	10,000	60,000	15,000	40,000
Vocab	40,624	34,976	15,480	56,392	15,136	45,376
Architecture						
d_{model}	512	512	512	512	768	768
d_{ffn}	2048	2048	1024	2048	3072	3072
Heads	8	8	4	8	12	12
Training						
GPUs	2	2	2	2	2	4
Steps	600K	150K	300K	300K	50K	30K
Tokens/GPU	32K	32K	4K	32K	4K	8K
Phase	[100K,200K,600K]	[50K,100K,150K]	[100K,200K,300K]	[100K,200K,300K]	[10K,20K,30K]	[5K,10K,30K]
Scaling	[2.0,3.0,4.0]	[2.0,3.0,4.0]	[2.0,3.0,4.0]	[2.0,3.0,4.0]	[2.0,3.0,4.0]	[2.0,4.0,8.0]

Table 8: The dataset details, model architectures, and hyperparameters used in our experiments.

across all steps, which, similar to the fixed schedule demonstrated in Table 7, is sub-optimal compared to our adaptive linear scheduling.

E Experimental Settings

For data preprocessing, we follow the instructions in `fairseq` for IWSLT14 and use the preprocessed data by Fully-NAT (Gu and Kong, 2021) for WMT14 and WMT16[1]. For Wiki and QQP, we use preprocessed data provided by DiffuSeq[2]. For Gigaword, we use preprocessed data provided by `huggingface`[3]. These 3 datasets are tokenized with byte-pair encoding (BPE) and vocabularized with `fairseq-preprocess`. The BPEs and vocabulary size are specified in Table 8.

All our implementations are based on `Transformer-base` (Vaswani et al., 2017) for all datasets with 6 Encoder and 6 Decoder layers. The number of attention heads, model hidden dimension size, and other hyperparameters are specified in Table 8. The embedding dimension for the diffusion model is 128. The anchor points $(\lambda_{\min}, \lambda_{\max})$ and $(\gamma_{\min}, \gamma_{\min})$ are designed to fit 20-step sampling, particularly $(0.9, 0.95)$ and $(0.15, 0.35)$, respectively.

For training, all experiments are trained with `fp16`. We used `inverse-sqrt` learning rates with 10,000 warmup steps, $lr_{\max} = 5 \times 10^{-4}$ for every benchmark except Quasar-T with $lr_{\max} = 3 \times 10^{-4}$, norm clipping 1.0, dropout 0.1, and label smoothing 0.1.

All training is conducted on 4 NVIDIA H100 GPUs. It takes approximately 8.5 hours for the WMT and Gigaword datasets, and around 4 hours on average for the other datasets.

During inference, the reverse process remains as in Eq. 4. The self-conditioning techniques reuse previous estimation, similar to previous works. We follow previous work (Li et al., 2022; Gong et al., 2023a; Dieleman et al., 2022) and apply Minimum Bayes-Risk (MBR) decoding (Kumar and Byrne, 2004). We mainly vary the length beam size since it often yields better results.

E.1 MANS Settings

We divide the training process into 3 uniform phases, we gradually increase the scaling value throughout the training process. The MANS phases and corresponding scaling values are specified in Table 8, for example, in WMT14, we apply $\beta(n) = 2.0$ for $n < 100$K, $\beta(n) = 3.0$ for $100\text{K} \leq n < 200$K, and $\beta(n) = 4.0$ otherwise. Empirically, we observe that this modification slightly increases the overall training time by under 5%.

F On the Effectiveness of Minimum Bayesian Risk (MBR) Decoding

MBR decoding is known to enhance the performance of text diffusion models by improving both output quality and diversity. As shown in Table 2, increasing either the length beam or the noise beam leads to higher BLEU scores. In Figure 9, we con-

[1] https://github.com/shawnkx/Fully-NAT
[2] https://github.com/Shark-NLP/DiffuSeq
[3] https://huggingface.co/datasets/Harvard/gigaword

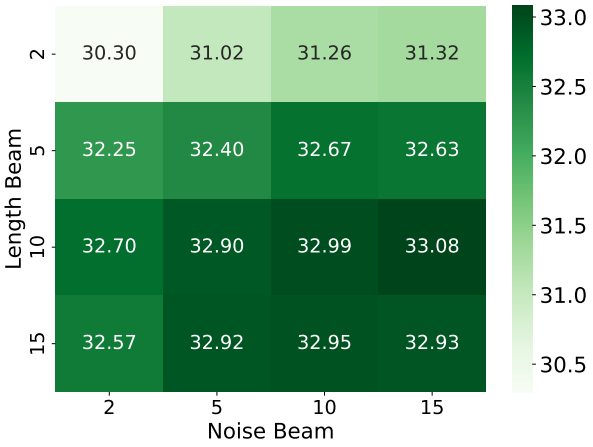

Figure 9: SacreBLEU score on IWSLT14 De-En with various length beams and noise beams.

duct a more thorough evaluation to determine how far BLEU can be improved across different search space configurations. The results indicate that scaling the length beam size (vertical axis) yields faster performance gains than scaling the noise beam size (horizontal axis). This is likely due to our method's ability to reduce faulty predictions during inference. Search with a larger length beam allows the model to better handle errors in length prediction and facilitates exploration of more diverse output sequences. While increasing the total beam size consistently improves performance, the most efficient approach is to first scale the length beam and only minimally increase the noise beam.

Unlike DINOISER, which uses high noise levels solely to leverage source conditioning, potentially limiting diversity, FastDiSS incorporates noise at every step. This allows it to maintain diversity without suffering from the marginal distribution prediction issue. Overall, FastDiSS delivers high-quality and diverse outputs while avoiding significant computational overhead.

G Qualitative Results

To qualitatively examine instance-level generation dynamics, we provide several case studies in Table 9. In the first example, the baseline fails to leverage the previously generated sequence, resulting in grammatical artifacts (e.g., "is" should be "are"). In contrast, our method is aware of the faulty prediction and is able to correct it in the following step, demonstrating the effectiveness of SNL. In the second example, the baseline largely copies the source sequence, leading to lower output diversity. By contrast, the improved learning process enables the model to generate a more fluent and accurate hypothesis that better aligns with the reference, highlighting the strengths of MANS training.

	Wiki-Auto	QQP
	Example 1	
Source	He also twice participated in the Summer Olympics, starting in 1996.	Which is best gadget?
Target	He was also in the 1996 Summer Olympics.	Which is your best gadget?
Step	**Baseline**	
2	He also twice in the Summer Summer Olympics 1996 Summer 1996.	Which is the best gadgets?
1	He also twice in the Summer Summer in the in 1996.	Which is the best gadgets?
Step	**FastDiSS**	
2	He also twice twice in in Summer Summer Olympics, starting in 1996.	What are best best gadgets?
1	He also twice twice in the Summer Olympics, starting in 1996.	What are the best gadgets?
	Example 2	
Source	Whedon served as an executive producer, along with Tim Minear.	Can we create free energy?
Target	Whedon was the executive producer, along with Tim Minear.	How do we make free energy?
Step	**Baseline**	
2	Whedon was an an executive producer with Tim Minear.	Can I create free energy?
1	He was an an executive producer with Tim Minear.	Can I create free energy?
Step	**FastDiSS**	
2	Whedon was an executive producer with Tim Minear.	How can I create a free energy?
1	He was an executive producer with Tim Minear.	How can I make a free energy?

Table 9: Wiki-Auto and QQP Generation with NFE $= 2$, using MBR decoding with LB $= 5$ and NB $= 1$.