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 cost of autoregressive models, while offering self-correction through iterative decoding. However, few-step sampling in text diffusion suffers from a training–inference mismatch in the self-conditioning mechanism—analogous to exposure bias–and from underexposure to high-noise regimes under uniform scheduling. We propose Fast Diffusion Sequence-to-Sequence (FastDiSS), a unified framework that enhances sampling efficiency and stability through two key components: a *Leakage-informed Noise Scheduler* (LINS) that bridges the training–inference gap, and a *Model-aware Adaptive Noise Scaling* (MANS) mechanism that dynamically balances easy and hard token learning. Extensive experiments across conditional text generation benchmarks demonstrate that FastDiSS achieves higher *BLEU* scores while providing up to 400x faster inference than standard diffusion models. By substantially narrowing the quality gap between few- and many-step diffusion, FastDiSS establishes a new direction for efficient and robust diffusion-based text generation.

1 Introduction

Diffusion models have recently emerged as a powerful paradigm for text generation, achieving competitive or even superior performance compared to autoregressive approaches (Nie et al., 2025; Prabhudesai et al., 2025). A key advantage of diffusion models lies in their ability to generate all tokens in parallel. By coupling diffusion processes with parallel decoding, they operate with linear complexity in sequence length, in contrast to the quadratic complexity of autoregressive transformers (Radford et al., 2018, 2019; Brown et al., 2020). During both training and inference, diffusion models simultaneously generate every token by iteratively applying forward and reverse processes. This parallelism not only accelerates decoding but also positions diffusion models as promising candidates for a broad range of natural language processing tasks (Li et al., 2022; Gong et al., 2023a; Ye et al., 2024, 2025).

Despite these advantages, diffusion-based text generation still faces fundamental challenges. A major drawback is the requirement of a large number of reverse steps to produce high-quality outputs (Gong et al., 2023a). The long iterative sampling chain is computationally expensive, often offsetting the efficiency gains from parallel decoding. To mitigate this issue, self-conditioning mechanisms (Chen et al., 2023b; Yuan et al., 2024; Gulrajani and Hashimoto, 2023) have been introduced, where intermediate predictions are fed back into subsequent iterations to refine estimates. Although self-conditioning substantially improves generation quality with fewer steps, it introduces additional problems.

The first problem is the exposure bias (Schmidt, 2019; Ning et al., 2024). Similar to autoregressive models, diffusion models suffer from a mismatch between training and inference: during training, the model is conditioned on ground-truth samples, whereas during inference it must depend on its own previously generated outputs. This discrepancy leads to error accumulation across reverse steps, resulting in sampling drift and degraded generation quality. Our analysis shows that the self-conditioning mechanism further exacerbates this problem. Because self-conditioning relies on only a few denoising steps, the mismatch between training and inference becomes pronounced, as the model cannot gradually adapt to the changes across reverse steps. Consequently, large deviations emerge in the early stages of generation and are quickly propagated and amplified throughout the generating process.

The second problem arises from the uniform

treatment of tokens in existing diffusion models. Current methods typically assume that all tokens contribute equally to the denoising process, irrespective of their frequency. However, we hypothesize that frequent tokens, which encode common linguistic regularities, are mastered early in training, whereas infrequent tokens remain more challenging. As training progresses, the model continues to allocate capacity to these trivial tokens, leading to overfitting and inefficient learning—while harder or infrequent tokens remain underoptimized. This leads to the problem where the denoising capacity is wasted on low-impact tokens, convergence is slowed, and training stability may degrade. As a result, existing diffusion models often operate suboptimally, with both computational efficiency and generative quality.

To address these challenges, we introduce Fast Diffusion Sequence-to-Sequence (`FastDiSS`), a novel training framework that enhances diffusion models with improved sampling efficiency. Building upon the foundation of self-conditioning, our framework integrates two key innovations designed to alleviate exposure bias and improve noise allocation. First, we propose the Leakage-informed Noise Scheduler (`LINS`), a novel noise scheduling strategy that simulates the behavior of self-conditioning during inference. By modifying the forward process to incorporate inference-like estimations, `LINS` narrows the gap between training and inference, thereby mitigating exposure bias. Second, we develop Model-aware Noise Scaling (`MANS`), a selective noise augmentation strategy that dynamically adjusts the noise level of tokens according to their difficulty. Tokens are categorized into easy (i.e., frequent tokens) and hard (i.e., infrequent tokens) classes based on model confidence during reconstruction. `MANS` then applies higher noise to easy tokens, forcing the model to refine its contextual understanding rather than converging prematurely on easy patterns. This selective noise allocation not only accelerates convergence by reallocating capacity toward more challenging or informative tokens but also improves overall training efficiency and enhances generation quality.

We evaluate `FastDiSS` across six benchmark datasets covering diverse sequence-to-sequence tasks, including machine translation, text summarization, question paraphrasing, and text simplification. Experimental results demonstrate that our framework consistently reduces the performance gap between few-step and many-step sampling, surpasses existing text diffusion models in both generation quality and efficiency. More specifically, `FastDiSS` achieves a higher *BLEU* score (Papineni et al., 2002) while achieves substantial efficiency gains with speedups ranging from 4× to 400× compared to competitive baselines. These findings highlight that diffusion models, when equipped with training–sampling alignment and model-aware noise strategies, can serve as both effective and efficient sequence-to-sequence generation.

In summary, we make three main contributions: (1) We analyze the limitations of the self-conditioning mechanism and highlight the critical role of high-level noise in achieving robust and efficient learning in diffusion models, thereby motivating the need for improved training strategies; (2) We introduce `FastDiSS`, a novel training framework that incorporates two components: a `LINS`, which modifies the noise schedule to effectively mitigate the exposure bias of self-conditioning, and a `MANS`, which selectively augments noise by progressively increasing the noise level of easy tokens to prevent trivial training; and (3) We conduct extensive experiments across six benchmarks, demonstrating that our proposed framework substantially improves generation quality while accelerating inference, achieving significant efficiency gains over existing approaches.

2 Background

2.1 Denoising Diffusion Probabilistic Models

We revisit Gaussian diffusion process in its continuous-time formulation (Song et al., 2021b; Chuang et al., 2024), which defines a trajectory $\{\boldsymbol{z}_t\}_{t=0}^{1}, t \in \mathbb{R}$ 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 follows.

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

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

Here, $0 \leq s < t \leq 1$ 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 $\sigma_t \to 1$, the endpoint 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 $D_\theta(\boldsymbol{z}_t)$,

which is trained using a mean-squared error loss:

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

Here, $\mathcal{U}[0,1]$ denotes the continuous uniform distribution. Recursive sampling from the distribution

$$\begin{aligned} p(\boldsymbol{z}_s|\boldsymbol{z}_t) &= \mathbb{E}_{p(\boldsymbol{z}_0|\boldsymbol{z}_t)}[q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{z}_0)] \\ &\approx \mathbb{E}_{p(\hat{\boldsymbol{z}}_\theta^t|\boldsymbol{z}_t)}[q(\boldsymbol{z}_s|\boldsymbol{z}_t, \hat{\boldsymbol{z}}_\theta^t)], \end{aligned} \quad (4)$$

where $\hat{\boldsymbol{z}}_\theta^t = D_\theta(\boldsymbol{z}_t)$, starting at $\boldsymbol{z}_1 \sim \mathcal{N}(0, \mathbf{I})$, enables generating data from $p(\boldsymbol{z}_0)$. Full expression of $q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{z}_0)$ is demonstrated in Appx. A.1.

2.2 Conditional Sequence Modeling with Diffusion Models

Diffusion models rely on a continuous space where Gaussian noise can be smoothly added and subtracted. However, texts are 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, and 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

$$\begin{aligned} \mathcal{L}_{\text{total}} &= \mathcal{L}_{\text{diffusion}} + \mathcal{L}_{\text{round}} \\ &= \mathbb{E}_{\boldsymbol{z}_0,t}[||D_\theta(\boldsymbol{z}_t) - \boldsymbol{z}_0||^2] + \mathbb{E}_{\boldsymbol{z}_0}\left[-\log p_\theta(\boldsymbol{x}|\boldsymbol{z}_0)\right], \end{aligned} \quad (5)$$

where $\mathcal{L}_{\text{round}}$ is the $\boldsymbol{z}_0 \to \boldsymbol{x}$ reconstruction loss.

To extend the model for conditional generation, a simple yet effective approach is to incorporate the conditioning sequence $\boldsymbol{c}$ as an additional input to the denoising network, i.e., $D_\theta(\boldsymbol{z}_t, \boldsymbol{c})$. 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.

2.3 Self-conditioning Diffusion Models

During training, the initial forward prediction $\hat{\boldsymbol{z}}_\theta^t$ is fed back into the denoising model as an auxiliary condition. The $\boldsymbol{z}_0$-prediction then 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, preventing gradient propagation through $\hat{\boldsymbol{z}}_\theta^t$. For clarity, we omit the source condition $\boldsymbol{c}$, as the discussion here focuses solely on the self-conditioning mechanism. The self-conditioning training objective becomes:

$$\mathcal{L}_{\text{sc}} = \mathbb{E}_{\boldsymbol{z}_0,t}[||D_\theta(\boldsymbol{z}_t, \text{sg}(\hat{\boldsymbol{z}}_\theta^t)) - \boldsymbol{z}_0||^2]. \quad (6)$$

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

At each step, estimating $\hat{\boldsymbol{z}}_\theta^t$ followed by $\bar{\boldsymbol{z}}_\theta^t$ double the inference time. Instead, the prediction from the previous step u is reused as the conditioning to avoid additional inference overhead. We denote this estimation as $\bar{\boldsymbol{z}}_\theta^{tu}$, for any $0 < s < t < u \leq 1$.

3 Methodology

3.1 Self-conditioning Limitations

As discussed in Sec. 2.3, $\bar{z}_\theta^{tu} - \bar{z}_\theta^t$ is practically non-zero, which will create estimation error between training and generation. To measure the effect of this estimation gap, we report the *BLEU* score on the generation of **IWSLT14** De-En dataset, using the correct $\bar{\boldsymbol{z}}_\theta$ and the original $\bar{\boldsymbol{z}}_\theta^{tu}$ self-condition sampling. In Tab. 1, we experimented with different numbers of denoising steps (NFEs).

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

Table 1: BLEU scores.

The table shows that the original self-conditioning signifies the approximation error when NFE is small, revealing a *training and inference empirical gap*. In contrast, the performance of the corrected self-condition slightly drops when NFE is reduced from 20 to 5, while it remains consistent for the rest NFEs. This observation highlights the importance of a training design that aligns with the inference process, which not only boosts performance but also enables more efficient inference. We conduct a more thorough theoretical analysis of the estimation gap in Appx. A.2.

3.2 Fast Diffusion Sequence-to-Sequence Overview

Fig. 1 shows Fast Diffusion Sequence-to-Sequence (`FastDiSS`) overview. Building upon the standard text diffusion architecture, `FastDiSS` introduces two complementary components: Leakage-informed Noise Scheduler (`LINS`) and Model-aware Noise Scaling (`MANS`) to jointly improve sampling alignment and training dynamics. `LINS`

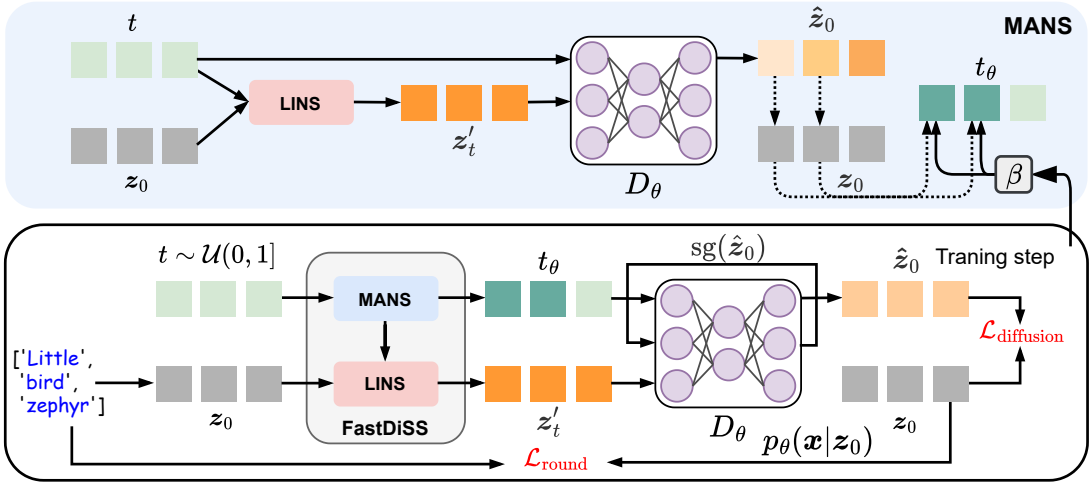

Figure 1: Overview of FastDiSS. The tokenized sequence is first encoded to $\boldsymbol{z}_0$, while concurrently sampling the initial timestep t. Both $\boldsymbol{z}_0$ and t are passed into MANS to obtain the new timestep t_θ. Subsequently, noise level at t_θ is added to $\boldsymbol{z}_0$ using LINS to obtain $\boldsymbol{z}'_t$. The rest is the same as in the training objective in Eq. 5.

Algorithm 1 Training with LINS

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

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

modifies the forward diffusion process to simulate inference-like conditions, thereby reducing exposure bias and improving stability during few-step sampling. In parallel, MANS adaptively adjusts token-level noise based on reconstruction confidence, enabling the model to focus more on informative and difficult tokens while avoiding overfitting to trivial ones.

3.3 Leakage-informed Noise Scheduler

We propose LINS to mimic the effectiveness of the correct self-condition. Intuitively, LINS aims to simulate inference behavior during training by injecting a controlled amount of "leakage" from future-step predictions into the current training step. Given that the training and inference are aligned, we can achieve performance equivalent to a larger computational budget with fewer generation steps. LINS is integrated into the end-to-end training pipeline without requiring any modification to the sampling procedure. During the forward process, we introduce a variable $\boldsymbol{z}'_t$ that captures the leakage effect from later inference steps ($u > t$):

$$\boldsymbol{z}'_t = \alpha_t \lambda_t \boldsymbol{z}_0 + \sigma_t \sqrt{1+\gamma_t^2}\,\boldsymbol{\epsilon}_t, \tag{7}$$

where $\boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I})$. Next, λ_t and γ_t are linear scheduling 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. The hyper-parameter selection strictly follows the leakage ratios α_u/α_t and σ_u/σ_t. Their monotonic variation ensures that $\boldsymbol{z}'_t$ is noised faster than $\boldsymbol{z}_t$, thus approximating the latent representation $\bar{\boldsymbol{z}}_u$ at inference ($u > t$).

Empirically, we observe that the prediction at the previous step u follows a normal distribution around $\hat{\boldsymbol{z}}^t_\theta$, confirming that LINS effectively captures the leakage pattern between consecutive denoising steps. A simplified derivation of this relationship, the close connection with the selection of hyperparameters for λ_t and γ_t, and detailed analysis of the empirical distributions are presented in Appx. B.

The proposed training procedure with LINS is presented in Alg. 1. Specifically, we modify the forward noise schedule with LINS (Eq. 7). Then, we randomly apply self-conditioning with probability 50%. This allows a consistent target for the denoising network.

3.4 Model-aware Noise Scaling

While LINS aligns training with inference at the sequence level, it treats all tokens equally. To further improve the training performance, we introduce

Model-aware Noise Scaling (MANS). Specifically, MANS aims to automatically incorporate token-level awareness into the generation process. We start from the hypothesis that frequent tokens are easier to reconstruct and quickly learned during training, as they represent common linguistic patterns and general knowledge that dominate natural text. Conversely, infrequent tokens are harder to learn and often receive insufficient optimization because the model allocates excessive capacity to already well-learned frequent tokens. To address this imbalance, we categorize tokens into *easy* (frequent) and *hard* (infrequent) groups based on the model's reconstruction confidence, and increase the noise level for easy tokens to encourage the model to focus on more informative and challenging representations. More specifically, we quantify token difficulty using the reconstruction objective $p_\theta(\boldsymbol{x}|\hat{\boldsymbol{z}}_\theta^t)$ instead of $p_\theta(\boldsymbol{x}|\boldsymbol{z}_0)$, which better captures the model's actual denoising capacity. Then, we identify the nearest token embeddings $\boldsymbol{e}_m$ that correspond to the ground-truth and reconstructed tokens:

$$i = \arg\min_{m=1:V} ||\boldsymbol{z}_0 - \boldsymbol{e}_m||_2, \tag{8}$$

$$j = \arg\min_{m=1:V} ||\hat{\boldsymbol{z}}_\theta^t - \boldsymbol{e}_m||_2. \tag{9}$$

Here, i and j represent the ground-truth and reconstructed tokens, respectively.

Inspired by recent studies (Ye et al., 2023) suggesting that higher noise tokens can guide diffusion models to focus more on informative components during denoising, we adopt a complementary strategy: increasing the noise of easy tokens to encourage the model to extract higher-quality and more meaningful representations. We then introduce a *noise scaling schedule* that progressively increases the denoising difficulty for easy tokens. For each token correctly reconstructed ($i = j$), we scale its noise level by adjusting the corresponding time step t according to a model-aware scaling function $\beta(n)$ at iteration n:

$$t_\theta = \begin{cases} \beta(n) \cdot t & \text{if} \quad i = j, \\ t & \text{otherwise.} \end{cases} \tag{10}$$

The β-function divides training into several phases, assigning a fixed scaling factor within each phase. This mechanism ensures that easy tokens receive gradually higher noise, forcing the model to focus on more informative and difficult tokens. In essence, MANS leverages instant feedback at every training iteration to achieve more balanced and efficient learning.

4 Experiments

4.1 Experimental Settings

Datasets. Following existing studies (Gong et al., 2023a), we evaluate Machine Translation on **IWSLT14** (En–De/De–En) (Cettolo et al., 2014), **WMT14** (En-De/De-En), and **WMT16** (En-Ro/Ro-En) (Bojar et al., 2014); Text Summarization on **Gigaword** (Narayan et al., 2018); Question Paraphrase on **QQP**; and Text Simplification on **Wiki-Auto**.

Evaluation Metrics. We report *SacreBLEU* for Machine Translation (Ye et al., 2023; Gong et al., 2023a); *ROUGE-1/2/L* for Summarization (Qi et al., 2020); and for Question Paraphrasing and Text Simplification, we follow the evaluation setup of Gong et al. (2023a), using sentence-level *BLEU*, *ROUGE-L*, and *BERTScore* (Zhang et al., 2020) to assess quality, along with sentence-level self-*BLEU* (Zhu et al., 2018) to measure diversity.

Baselines. We consider three baseline groups: (1) Autoregressive (AR) models: Transformer (Vaswani et al., 2017), GRU with attention, and fine-tuned GPT2-Large; (2) Non-autoregressive (NAR) model: CMLM (Ghazvininejad et al., 2019) and LevT (Gu et al., 2019); and (3) Diffusion-based language (DL) models: DiffusionLM (Li et al., 2022), Difformer (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., 2017) and NAG-BERT (Su et al., 2021). For Question Paraphrase, we include Discrete Diffusion with RDM (Zheng et al., 2023). We refer to these baselines in detail in Appx. C.

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. Our implementation is based on Difformer (Gao et al., 2024), with Transformer and sampling configurations from prior work, with a number of denoising steps—NFE $= \{2, 5, 20\}$. For every task, we construct the vocabulary with Byte Pair Encoding (Kudo and Richardson, 2018). Further details are described in Appx. E. We also apply Minimum Bayes Risk decoding (Kumar and

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	29.75	**24.69**	31.81	30.90
FastDiSS	10	1	20	32.81	26.29	29.47	24.50	31.89	31.37
FastDiSS	10	2	20	**32.88**	**26.39**	**29.83**	24.57	**31.99**	**31.44**

Table 2: Main results on **IWSLT14**, **WMT14**, and **WMT16**. The best NAR results are **bold** and the second-best results are underlined. ‡ indicates results reported by Ye et al. (2023). Other results are from their original papers.

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.

Byrne, 2004) following previous works (Li et al., 2022; Gong et al., 2023a) with different length beam (LB) and noise beam (NB), where a detailed ablation study is described in Appx. F.

4.2 Main Results

Overall Performance. The results on datasets across tasks are shown in Tabs. 2, 3, and 4. FastDiSS outperforms both DL 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 LINS effectively bridges the gap between few and many-step sampling. Importantly, with LINS training, the sampling speed is 4× faster than DINOISER and Differmer and 400x faster than DiffuSeq and SeqDiffuSeq.

Findings on Training. In the early stages of training, we hypothesize that the model rapidly learns to predict easy tokens (Chen et al., 2023a; Ou and Jian, 2024), whereas harder or rarer tokens remain challenging to reconstruct. As training progresses, predictions for easy tokens become trivial, causing the model to overfit minority tokens–a phenomenon reminiscent of memorization effects observed in

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 **Gigaword**. 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 2: Comparison of the validation loss and BLEU score during training when $\beta = 1.0$, 2.0, $\beta(n)$ (Adaptive). The dashed line shows the *BLEU* score, with colors matching the loss curves for each setting.

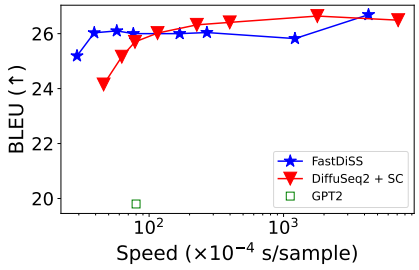

Figure 3: Generation speed and quality with different NFE. The speed is averaged over 3 runs.

prior studies (Zhang et al., 2017; Bai et al., 2021). This is reflected in marginal loss reductions, while performance plateaus (see Fig. 2, β=1.0). A fixed scaling factor ($\beta = 2.0$) also fails to overcome this limitation, whereas the proposed `Adaptive` scaling continues to enhance performance even under high noise levels in later training stages. Examples illustrating easy and hard token distinctions are provided in Appx. G.

Accelerated Sampling Comparison. We compare the trade-off between generation quality and the number of generated tokens between our method and `DiffuSeq-v2` with its original self-conditioning mechanism (Gong et al., 2023b), to demonstrate performance in practical applications. We report results on the **QQP** dataset. As shown in Fig. 3, `FastDiSS` consistently outperforms `DiffuSeq-v2` at low NFEs, demonstrating its superior efficiency in the few-step regime. However, its advantage diminishes at medium NFEs, where performance slightly falls behind `DiffuSeq-v2`. This trend suggests that the benefit of `LINS` decreases as the approximation error margin becomes sufficiently small, revealing a limitation of relying solely on correcting previous predictions once the model approaches convergence.

Sampling Diversity. We study the sampling diversity on **QQP**. More specifically, we measure the performance of baseline w.r.t. both *BLEU* and self-*BLEU*. Fig. 4 shows the trade-off between diversity

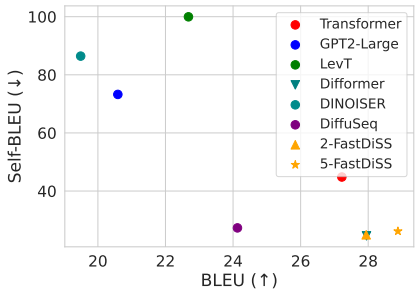

Figure 4: Diversity and quality comparison.

LINS	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 study.

and quality. `FastDiSS`-2NFE achieves comparable performance to the strongest baseline, `Difformer`-20NFE, using only two sampling steps. As the number of denoising steps increases, inference becomes slower, but generation quality improves. Notably, `FastDiSS`-5NFE attains substantially higher *BLEU* scores while maintaining nearly the same self-*BLEU*, indicating that quality gains do not come at the cost of diversity.

4.3 Ablation Studies

Effect of Each Component. We conduct an ablation study to examine the contribution of each component in `FastDiSS`. Specifically, we remove individual components and evaluate the resulting models on the **IWSLT14** De–En dataset using LB = 2 and NB = 2, reporting performance w.r.t. *BLEU* score. Table 5 shows that both `LINS` and `MANS` contribute to performance improvements. Interestingly, `MANS` yields stronger gains at smaller NFEs, suggesting that it accelerates convergence during training by emphasizing informative tokens. In contrast, `LINS` primarily focuses on bridging the gap between training and inference in the self-conditioning mechanism, thus improving the inference performance. Notably, combining `LINS` and `MANS` narrows the performance gap between 5 and 20 steps.

Effect of Number of Denoising Steps. We study the impact of the number of denoising steps by comparing the *BLEU* scores of `FastDiSS` and `Difformer` when varying this number on the **IWSLT14** De–En dataset. Fig. 5 illustrates the effectiveness of our proposed method with few-step denoising. Our method outperforms the 20-step baseline after only 3 sampling steps, yielding $\sim 7\times$

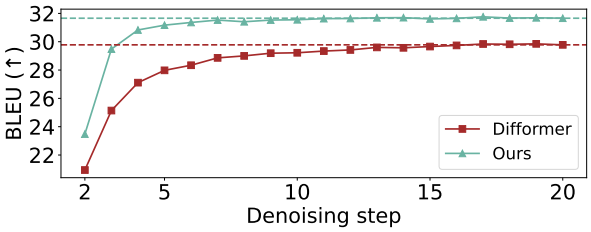

Figure 5: Effect of Number of Denoising Steps.

Steps	5	20	100	Fixed
λ_t	0.60 - 0.85	0.90 - 0.95	0.98 - 0.99	0.85
γ_t	0.25 - 0.90	0.15 - 0.35	0.05 - 0.15	0.50
BLEU	25.48	**26.35**	25.70	25.84
ROUGE-L	57.29	**57.47**	56.82	57.18

Table 6: Effect of γ_t and λ_t.

speedup. Additionally, our sampling method converges after only 7 steps, at a significantly higher *BLEU* score than the 20-step baseline.

Effect of γ_t and λ_t. We study the effect of the hyperparameters γ_t and λ_t in LINS (see Eq. 7). The experiments are conducted with NFE $= 2$, LB $= 5$, and NB $= 1$ on the **QQP** dataset. Tab. 6 shows the performance with different variants of LINS, 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 show that optimal performance is achieved at the leakage-informed level with 20 denoising steps, where γ_t ranges from 0.90 to 0.95 and λ_t ranges from 0.15 to 0.35. Hence, we use this setting for the remaining benchmarks.

Effect of Noise Schedulers. We compare our proposed method with popular noise schedulers, including `linear` (Ho et al., 2020) and `cosine` (Nichol and Dhariwal, 2021) on **QQP**. Tab. 7 shows the results. The `linear` schedule is less sensitive to the NFE value. Our method strongly boosts the score in the few-step settings.

4.4 Experiments on Reasoning Benchmark

To evaluate the adaptability of our approach to LLMs, we applied LINS to DoT (Ye et al., 2024), a diffusion language reasoning model. The base model of DoT is built on Plaid (Gulrajani and Hashimoto, 2023) with 1 billion parameters and trained using the self-conditioning technique. We introduced minor modifications to the training process through Alg. 1, followed by fine-tuning on the downstream task of grade school math problem solving using the **GSM8K** dataset (Cobbe et al., 2021). As shown in Tab. 8, LINS enhances performance in both traditional diffusion inference and CoT reasoning inference, compared to the original method. Moreover, the decline in accuracy is substantially reduced when the number of reasoning steps is decreased, underscoring the potential of this approach in studies that aim to balance computational cost and accuracy. It is worth noting that LINS configurations were pretrained using only two-thirds of the available compute budget. This suggests that the performance of our approach could be further improved with full-scale training, highlighting additional room for optimization. We provide some generated examples from FastDiSS and Plaid in Appx. H.

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 7: *BLEU* Score on **QQP** with different noise schedules. Results are reported with LB $= 2$, NB $= 1$.

T	DoT (Plaid)		DoT^{mp} (CoT)	
	Normal	LINS	Normal	LINS
64	41.3 ± 0.2	$\mathbf{43.7 \pm 0.2}$	36.6 ± 0.6	$\mathbf{40.8 \pm 0.6}$
32	39.9 ± 0.3	$\mathbf{43.0 \pm 1.1}$	37.2 ± 0.7	$\mathbf{40.4 \pm 0.9}$
16	39.0 ± 0.5	$\mathbf{41.1 \pm 0.7}$	36.1 ± 0.7	$\mathbf{39.8 \pm 0.8}$
8	35.4 ± 0.5	$\mathbf{38.2 \pm 0.7}$	30.9 ± 0.6	$\mathbf{35.1 \pm 1.0}$

Table 8: Comparison of *Accuracy* on **GSM8K** benchmark. We experimented with two settings, DoT with Plaid inference and DoT^{mp} with CoT reasoning, each is trained and fine-tuned with and without LINS. The *Accuracy* is averaged over 5 runs.

5 Conclusion

We introduced FastDiSS, a novel training mechanism for diffusion language models that addresses two key challenges. First, the Leakage-informed Noise Scheduler mitigates the training–inference mismatch inherent in self-conditioning. Second, Model-aware Noise Scaling accelerates convergence by selectively applying higher noise to common tokens that contribute more to the generation quality. Experiments on six benchmarks demonstrate that FastDiSS enables more effective training, narrows the gap between few-step and many-step decoding, and achieves significant improvements in both efficiency and generation quality. While our method advances the practicality of fast inference with diffusion-based models, future work is needed to further refine self-conditioning and fully close the gap with autoregressive approaches.

Limitation

Since we aim to align training to inference, the proposed techniques are constrained by the goodness of the previous self-conditioning prediction. 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. Next, 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.

Broader Impact

This work advances diffusion-based language modeling by proposing mechanisms that improve training efficiency and training-inference alignment. By reducing exposure bias and accelerating convergence, our approach contributes to the development of faster, more robust text generation systems. These improvements can benefit a wide range of natural language applications, including machine translation, summarization, and question answering, where efficiency and quality are critical.

Despite these potential benefits, diffusion-based models, like other large generative models, may still propagate social biases or produce harmful or misleading content when trained on unfiltered data. We emphasize the importance of responsible deployment, including careful dataset selection, bias evaluation, and human oversight.

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. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. 2021. Structured denoising diffusion models in discrete state-spaces. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 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. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 24392–24403.

Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Ales Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Workshop on Statistical Machine Translation (WMT@ACL)*, pages 12–58.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th IWSLT evaluation campaign. In *Proceedings of the International Workshop on Spoken Language Translation: Evaluation Campaign (IWSLT)*, 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 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4765–4775.

Ting Chen, Ruixiang Zhang, and Geoffrey E. Hinton. 2023b. Analog bits: Generating discrete data using diffusion models with self-conditioning. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

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. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

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. *CoRR*, abs/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 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, pages 4664–4683.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6111–6120.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. 2023a. DiffuSeq: sequence to sequence text generation with diffusion models. In *Proceedings of the 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. In *Findings of the EMNLP*, pages 9868–9875.

Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R. Steunebrink, and Jürgen Schmidhuber. 2017. LSTM: A search space odyssey. *IEEE Trans. Neural Networks Learn. Syst.*, 28(10):2222–2232.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2017. Non-autoregressive neural machine translation. *CoRR*, abs/1711.02281.

Jiatao Gu and Xiang Kong. 2021. Fully non-autoregressive neural machine translation: Tricks of the trade. In *Findings of ACL*, pages 120–133.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 11179–11189.

Ishaan Gulrajani and Tatsunori B. Hashimoto. 2023. Likelihood-based diffusion language models. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Jiaxin Guo, Minghan Wang, Daimeng Wei, Hengchao Shang, Yuxia Wang, Zongyao Li, Zhengzhe Yu, Zhanglin Wu, Yimeng Chen, Chang Su, Min Zhang, Lizhi Lei, Shimin Tao, and Hao Yang. 2021. Self-distillation mixup training for non-autoregressive neural machine translation. *CoRR*, abs/2112.11640.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Fei Huang, Tianhua Tao, Hao Zhou, Lei Li, and Minlie Huang. 2022. On the learning of non-autoregressive transformers. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 9356–9376.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. 2022. Elucidating the design space of diffusion-based generative models. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 66–71.

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

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B. Hashimoto. 2022. Diffusion-LM improves controllable text generation. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.

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 (AAAI)*, pages 13246–13254.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. 2022. Dpm-solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. 2025. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *Mach. Intell. Res.*, 22(4):730–751.

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. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing EMNLP*, pages 1797–1807.

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

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. 2025. Large language diffusion models. *CoRR*, abs/2502.09992.

Mang Ning, Mingxiao Li, Jianlin Su, Albert Ali Salah, and Itir Önal Ertugrul. 2024. Elucidating the exposure bias in diffusion models. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

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

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 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC/COLING)*, 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 Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318.

Mihir Prabhudesai, Mengning Wu, Amir Zadeh, Katerina Fragkiadaki, and Deepak Pathak. 2025. Diffusion beats autoregressive in data-constrained settings. *CoRR*, abs/2507.15857.

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. *CoRR*, abs/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-sequence pre-training. In *Findings of EMNLP*, pages 2401–2410.

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. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3059–3069.

Subham S. Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu, Alexander Rush, and Volodymyr Kuleshov. 2024. Simple and effective masked diffusion language models. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Florian Schmidt. 2019. Generalization in generation: A closer look at exposure bias. In *Proceedings of the Workshop on Neural Generation and Translation (GNT@EMNLP-IJCNLP)*, pages 157–167.

Chenze Shao, Xuanfu Wu, and Yang Feng. 2022. One reference is not enough: Diverse distillation with reference selection for non-autoregressive translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, pages 3779–3791.

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 (AAAI)*, 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. 2021a. Denoising diffusion implicit models. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Yang Song and Prafulla Dhariwal. 2024. Improved techniques for training consistency models. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. 2023. Consistency models. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 32211–32252.

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

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 Conference of the European Chapter of the Association for Computational Linguistics: Main Volume (EACL)*, pages 234–243.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 5998–6008.

Tong Wu, Zhihao Fan, Xiao Liu, Hai-Tao Zheng, Yeyun Gong, Yelong Shen, Jian Jiao, Juntao Li, Zhongyu Wei, Jian Guo, Nan Duan, and Weizhu Chen. 2023. AR-Diffusion: auto-regressive diffusion model for text generation. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Jiacheng Ye, Shansan Gong, Liheng Chen, Lin Zheng, Jiahui Gao, Han Shi, Chuan Wu, Xin Jiang, Zhenguo Li, Wei Bi, and Lingpeng Kong. 2024. Diffusion of thought: Chain-of-thought reasoning in diffusion language models. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. 2025. Dream 7b: Diffusion large language models. *CoRR*, abs/2508.15487.

Jiasheng Ye, Zaixiang Zheng, Yu Bao, Lihua Qian, and Mingxuan Wang. 2023. DINOISER: diffused conditional sequence learning by manipulating noises. *CoRR*, abs/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 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, pages 22–39.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2017. Understanding deep learning requires rethinking generalization. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

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. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, pages 1747–1757.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Lin Zheng, Jianbo Yuan, Lei Yu, and Lingpeng Kong. 2023. A reparameterized discrete diffusion model for text generation. *CoRR*, abs/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 *Proceedings of the ACM SIGIR International Conference on Research & Development in Information Retrieval (SIGIR)*, pages 1097–1100.

A Theoretical Details

A.1 Derivation of the posterior distribution

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:

$$q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{z}_0) = \mathcal{N}(\boldsymbol{z}_s; \tilde{\boldsymbol{\mu}}(\boldsymbol{z}_t, \boldsymbol{z}_0), \tilde{\sigma}_t^2 \boldsymbol{I}) \quad (11)$$

$$\text{where} \quad \tilde{\boldsymbol{\mu}}(\boldsymbol{z}_t, \boldsymbol{z}_0) = \frac{\alpha_t}{\alpha_s}\frac{\sigma_s^2}{\sigma_t^2}\boldsymbol{z}_t + \alpha_s \frac{\sigma_{t|s}^2}{\sigma_t^2}\boldsymbol{z}_0 \quad (12)$$

$$\text{and} \quad \tilde{\sigma}_t = \frac{\sigma_s}{\sigma_t}\sigma_{t|s}. \quad (13)$$

A.2 Analysis on the Estimation Gap

In contrast to prior studies, which suggested that error could accumulate across steps (Ning et al., 2023), we showed that, given a sufficiently large number of denoising steps, the discretization errors between consecutive steps become small enough for the model to estimate the correct self-condition. Specifically, we first denote t_{i+1} and t_i as two selected consecutive steps used during generation, and $\bar{\boldsymbol{z}}_\theta^{t_i t_{i+1}}$ as the estimation with the condition is the denoising output from the previous step t_{i+1}. Then, we state 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:*

$$\sup \mathop{\mathbb{E}}_{i\sim[0,n-1]} \left[\|D_\theta(\boldsymbol{z}_{t_i}, \bar{\boldsymbol{z}}_\theta^{t_{i+1}t_{i+2}}) - D_\theta(\boldsymbol{z}_{t_i}, \hat{\boldsymbol{z}}_\theta^{t_i})\| \right] = \mathcal{O}((\Delta t)^p). \quad (14)$$

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

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

Furthermore, from our assumption that the local error is bounded by $\mathcal{O}((t_{i+1} - t_i)^{p+1})$, we have the upper bound of the total local error is

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

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.

This theorem suggests that the self-condition estimation can become arbitrarily accurate, as long as the number of sampling steps is large enough. In Tab. 9, we practically demonstrate this estimation error through different numbers of denoising steps.

NFEs	5	20	50	100
Sup $\mathbb{E}$	0.047	0.008	0.009	0.010

Table 9: Estimation Error.

B Leakage Noise Robustness

In this section, we show that the modified training process with LINS 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.

B.1 Normal Distribution between Estimation

Akin to Sec. 3.1, 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 stored the previous step's estimation $\bar{\boldsymbol{z}}_\theta^{tu}$ and also

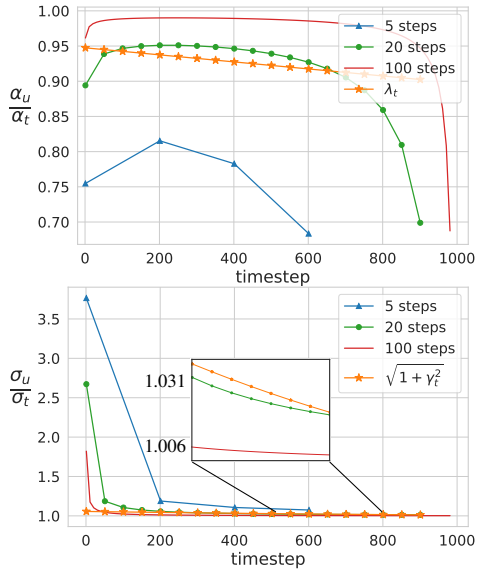

Figure 6: Mean (top) and variance (bottom) ratio differences measured at different denoising steps. We manually select hyperparameters of the scaling terms to fit 20-step sampling.

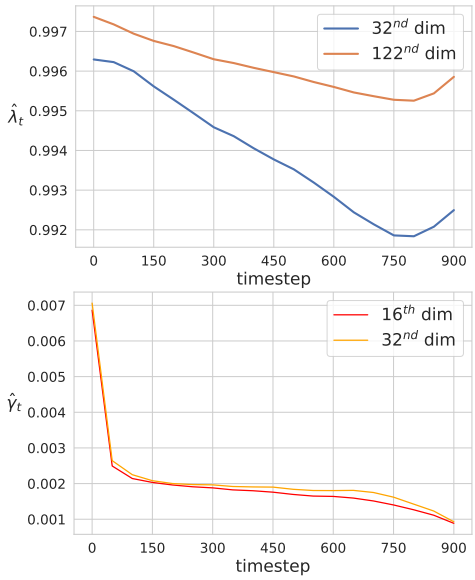

Figure 7: Inference mean $\hat{\lambda}_t$ (top) and variance $\hat{\gamma}_t$ (bottom) scaling factors of prediction mismatch in a pre-trained network, plotted across timestep t for randomly selected embedding dimensions.

computed the corrected self-condition $\hat{\boldsymbol{z}}_\theta^s$ for the current step s.

We empirically found that $\bar{\boldsymbol{z}}_\theta^{tu}$ is normally distributed around $\hat{\boldsymbol{z}}_\theta^s$:

$$\bar{\boldsymbol{z}}_\theta^{tu} \sim \mathcal{N}(\hat{\boldsymbol{\mu}}_{st}\hat{\boldsymbol{z}}_\theta^s, \hat{\boldsymbol{\sigma}}_{st}^2\boldsymbol{I}), \tag{15}$$

with mean vector $\hat{\boldsymbol{\mu}}_{st}$ and standard deviation vector $\hat{\boldsymbol{\sigma}}_{st}$ (see Sec. B.2). From this indication, we derive a side-by-side connection with $\boldsymbol{z}'_s$. Considering at testing, suppose that $\hat{\boldsymbol{z}}_\theta^s$ fully denoise $\bar{\boldsymbol{z}}_s$, we substitute $\hat{\boldsymbol{z}}_\theta^s$ by $\bar{\boldsymbol{z}}_\theta^{tu}$ as:

$$\begin{aligned}\bar{\boldsymbol{z}}_s &= \alpha_s\hat{\boldsymbol{z}}_\theta^s + \sigma_s\boldsymbol{\epsilon}_s = \alpha_s\frac{\bar{\boldsymbol{z}}_\theta^{tu} - \hat{\boldsymbol{\sigma}}_{st}\boldsymbol{\epsilon}_{\boldsymbol{u}}}{\hat{\boldsymbol{\mu}}_{st}} + \sigma_s\boldsymbol{\epsilon}_s \\ &= \alpha_s\frac{1}{\hat{\boldsymbol{\mu}}_{st}}\bar{\boldsymbol{z}}_\theta^{tu} + \sigma_s\sqrt{1+\left(\frac{\alpha_s}{\sigma_s}\frac{\hat{\boldsymbol{\sigma}}_{st}}{\hat{\boldsymbol{\mu}}_{st}}\right)^2}\boldsymbol{\epsilon} \\ &= \alpha_t\hat{\boldsymbol{\lambda}}_{st}\bar{\boldsymbol{z}}_\theta^{tu} + \sigma_s\sqrt{1+\hat{\boldsymbol{\gamma}}_{st}^2}\boldsymbol{\epsilon},\end{aligned} \tag{16}$$

where $\hat{\boldsymbol{\lambda}}_{st} = 1/\hat{\boldsymbol{\mu}}_{st}$, $\hat{\gamma}_{st} = (\alpha_s/\sigma_s)(\hat{\boldsymbol{\sigma}}_{st}/\hat{\boldsymbol{\mu}}_{st})$, and $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$ is obtained using the reparametrization trick on the terms containing $\boldsymbol{\epsilon}_u$ and $\boldsymbol{\epsilon}_s$. Since $\hat{\boldsymbol{\mu}}_{st}$ and $\hat{\boldsymbol{\sigma}}_{st}$ vary linearly and independently across dimensions, all operations remain element-wise. Apparently, we observed a similar pattern between Eq. 7 and 16, as well as the empirically estimation between Figs. 6 and 7. This indicates that LINS explicitly informs the training process of $\bar{\boldsymbol{z}}_\theta^{tu}$.

In principle, empirical schedules $\{\hat{\boldsymbol{\lambda}}_{st}, \hat{\boldsymbol{\gamma}}_{st}\}_{t=1}^T$ could be derived from Eq.16. However, the dependence on discretization steps, along with the linear and dimension-wise variation of $\hat{\boldsymbol{\lambda}}_{st}$ and $\hat{\boldsymbol{\gamma}}_{st}$, 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.

B.2 Estimate the Self-condition Gaussian Distribution Error

In this section, we conduct experiments on **IWSLT14** De-En to empirically show the connection between $\bar{\boldsymbol{z}}_\theta^{tu}$ and $\hat{\boldsymbol{z}}_\theta^s$. 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 $\boldsymbol{\epsilon}_t = \bar{\boldsymbol{z}}_\theta^{tu} - \hat{\boldsymbol{\mu}}_t\hat{\boldsymbol{z}}_\theta^s$. 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., $\bar{\epsilon}_t^i = \frac{\epsilon_t^i - \mu_t}{\hat{\sigma}_t^i}$).

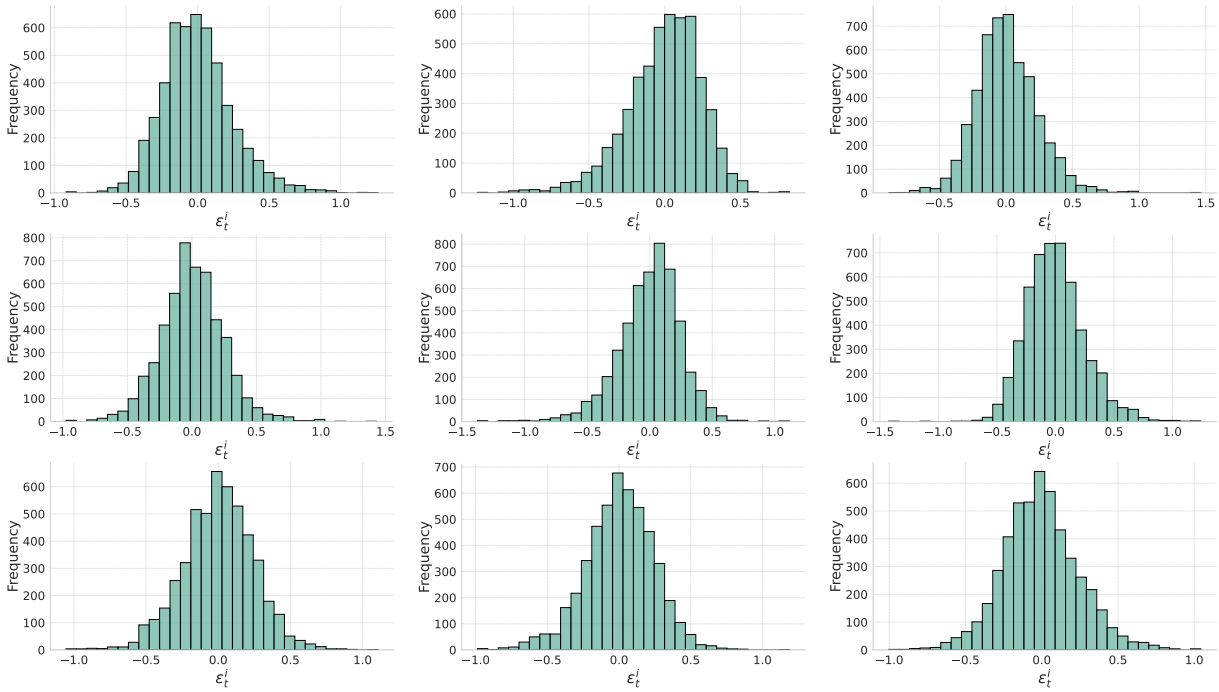

Figure 8: The empirical distribution of ϵ_t^i with different random values of t and i.

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. Fig. 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. Diffusion models for text tackle multi-modality in text generation and fall into two main categories: continuous and discrete. In continuous models, tokens 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. Recently, `Meta-Diffu`B (Chuang et al., 2024) proposes customizing the whole noise schedule plan for each context, making it more flexible and better aligned during inference. `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., 2021a) and Progressive Distillation have demonstrated success in reducing the number of denoising steps. Advances in ODE solvers (Lu et al., 2022, 2025) and second-order trajectory approximations such as 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, 2024) 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) addresses 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.

Several works that coincide with ours are described in more detail in the next section.

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 across all steps, which, similar to the fixed schedule demonstrated in Tab. 6, is sub-optimal compared to our adaptive linear scheduling.

Meta-Diffu$B$ employs a simple architecture (LSTM) for encoding contextual information, which raises concerns about its generalization to unseen contexts—such as in zero-shot downstream tasks or long-sequence processing. Additionally, optimizing the planning schedule using a basic reinforcement learning strategy introduces instability. In contrast, our noise scheduling method avoids the need for optimizing an auxiliary network, effectively using the network's direct feedback to adjust the noise level. Since this approach is used solely for efficient training, the sampling process remains unchanged, thereby avoiding reliance on a planning network, which may be sensitive to out-of-distribution (OOD) scenarios.

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 Tab. 10.

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 Tab. 10. 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.

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

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

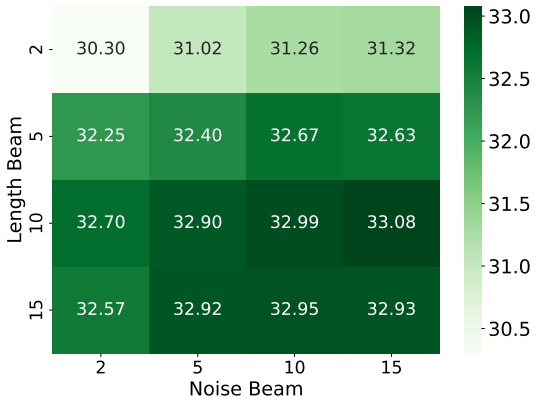

Figure 9: *SacreBLEU* score on **IWSLT14** De-En with various length beams and noise beams.

For MANS, we divide the training process into 3 uniform phases, we gradually increase the scaling value throughout the training process. The phases of MANS and corresponding scaling values are specified in Tab. 10, for example, in **WMT14**, we apply $\beta(n) = 2.0$ for $n < 100$K, $\beta(n) = 3.0$ for $100\text{K} \leq n < 200\text{K}$, 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 Tab. 2, increasing either the length beam or the noise beam leads to higher *BLEU* scores. In Fig. 9, we conduct 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). 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 Visualization of Easy and Hard Tokens

Fig. 10 demonstrates the tokens under the effect of noise scaling with MANS. The "easy" tokens are obviously common words, which the noise scaling forces the model to denoise these tokens even at high-level noise, which allows them to be predicted more precisely and faster during inference. While the "hard" tokens are tokens that appear less in the training set. Its noise level remains, which demonstrates that it might be predicted better at a smaller noise level, corresponding to a coarse-to-fine process.

Figure 10: Word cloud of the easy and hard tokens during training.

H Qualitative Results

To qualitatively examine instance-level generation dynamics, we provide case studies in Tabs. 11, 12, 13, 14, and 15. In Tab. 11, the baseline fails to leverage the previously generated sequence, resulting in grammatical artifacts (e.g., "Summer Summer" is not repaired in the next step). In contrast, `FastDiSS` is aware of the faulty prediction and is able to correct it in the following step, demonstrating the effectiveness of `LINS`. In Tab. 12, the baseline does not preserve the general and fine-grained details, leading to information loss or wrong context understanding. By contrast, our method enables a more fluent and accurate hypothesis that better aligns with the reference.

	Step	Example 1
Source	-	He also twice participated in the Summer Olympics, starting in 1996.
Target	-	He was also in the 1996 Summer Olympics.
Baseline	2	He also twice in the Summer ***Summer*** Olympics 1996 Summer 1996.
	1	He also twice in the Summer ***Summer*** in the in 1996.
FastDiSS	2	He also twice twice in ***in*** Summer ***Summer*** Olympics, starting in 1996.
	1	He also twice twice in the Summer Olympics, starting in 1996.
	Step	**Example 2**
Source	-	Whedon served as an executive producer, along with Tim Minear.
Target	-	Whedon was the executive producer, along with Tim Minear.
Baseline	2	Whedon was ***an*** an executive producer with Tim Minear.
	1	He was ***an*** an executive producer with Tim Minear.
FastDiSS	2	Whedon was an executive producer with Tim Minear.
	1	He was an executive producer with Tim Minear.

Table 11: Examples of generation throughout the denoising process, with NFE $= 2$, using MBR decoding with LB $= 5$ and NB $= 1$.

	Example 1
Source	How do you think Federer, Nadal, Djokovic, Murray, Wawrinka, Delpo & Cilic rank in terms of potential when it comes to greatest spot in the tennis?
Target	How would you rank Federer, Nadal, Djokovic, Murray, Wawrinka, Delpo and Cilic in terms of greatness achieved and potential spot in the history of sport?
Baseline	How do you think about ***the Federer, rank in a good of Mury Wawrinka***, Delgreatest rank and Cilic rank in terms of for tennis?
FastDiSS	What is the review of vic, MurAur, Murray, Dray, Wawrinka, DelPO & Cilic rank in tennis?
	Example 2
Source	How will the ban on 500 and 1000 rupee notes bring out the black money of the big shots who have lots of it in the Swiss bank in a different currency?
Target	How will the decision to illegalize the 500 and 1000 Rs notes help to get rid of black money in the Swiss bank or maybe in other foreign banks and currencies?
Baseline	What will be the effect of banning 500 and 1000 notes on people ***having money*** in Swiss banks?
FastDiSS	How is demonetizing the 500 and 1000 rupees notes affect Indian economy? How it affect the black money in Swiss bank accounts?

Table 12: Examples of generation on **Wiki-Auto** dataset. The resulting outputs demonstrate higher coherence in the long-term context.

	Example 1
Source	This easy chicken curry recipe is made with just a handful of ingredients
Plaid	This easy chicken curry recipe is made with just a handful of ingredients make provide add that helps easy add help give add that helps welcome give help offer help for how you learn give help help help for act give help help help and perform give help help help for for how pay for help help help and even add help help help for give help that add give help up for the add provide help provide for help help help and add for how pay for help give help provide help and add give help help help and add help provide help to create help give help help help for for how doit add this give help help help that add help help give help help for and that give help add help give help give help for the run in the middle of the day add help help provide help help that gives simply give help help give help help provide help help provide help help for pay help help give help help pay give help give help give help help pay help the member on your family make help shop it ...
Plaid + LINS	This easy chicken curry recipe is made with just a handful of ingredients at your disposal including a bow for glowing red ropes for strutter. It alone will be the trick since making anything else would normally be a lengthy method. Once that completes the dungeon is start laid out again towards the end where you learn the way to make naughty but once you find this little hidden dungeon point you will also need the basics of the tools to help you find your way to the recipe for cider seeds.↩↩So many would say these simple caves and dungeons are like paster glue in a Oddband environment. It is not just the recipe that is perhaps also the highlight in that the have side rooms for dungeon for completion in small spells or just locked in a spooky cave somewhere. The lack of questline support will prevent really good and fun faciettes. The formula the ingredients are a lot worth then anything else but for an hour of stats capable of requiring three seemed some how effective when you have the basics ...
	Example 2
Source	Generative models of text are very versatile: they can be used
Plaid	Negative models of text are very versatile: they can be used in the workplace and in the secondary market provide availability provides published versions varies and competitive paid provided available for tender available as the selection of availability choices were very common on a high ... fair to read the initial sales framework very often ... provide selection for the except immediately in commonly based on how pricing control edited ask that everyone creates as a matter of how much published provide is actually apply control on that immediately on offered provide on exchange for them provide paid write line of commissioned provide access made applied to more creative offered works cover different parts of the market agree that paid sharing is much more what use sharing work on the ability to in understanding in support of a contribution of a meaningful piece on say you can provide featured provide paid paid assignment paid submissions management provided applied arrangements with mixed paid paid shared fit easily perform create commissioned work in private ... why is not offered applied availability on the subject or backing and reuse ... to ... more successful provide writing for working provide featured work submission creative exclusively created collection of icons is a symbol medium always learn how to make use of owns offering portfolio offering total free commissioned formats provided published licensing paid
Plaid+LINS	Generative models of text are very versatile: they can be used to extract depth and status information cheaply. Others would argue that static models of text are a powerful aid for explersion; however, it was in the core caricatures that it didn't offer much benefit because in the instance they were not comprehend, the translation to say showed a very high degree of complexity. If you are looking for help overcoming the CPP constraints, you will want to consider means with which the manual models have been effectively replaced. How much structure really can be leveraged here is left as an observation regarding numeric computation and the need to adapt.↩↩Even if the topic had not made its true contribution in the Enchanted Mongoose core system, the switch to primary structures would be significant. When I first saw the stream, I was told that the intent was to take the existing core system and replace it entirely with a trim and circular setup. That was described very well and perhaps would have been an option but as inconsistent with the extra-mainruction requirement as new core support for and directly related to static features. The details the stream provided suggest that they are not based on current text models, which probably means that the team is hoping for a design fix as you no longer get to the point of accurate input even in corramures. Primary layers, on the other hand, still provide a significant degree of input. You will have to navigate over simulated character emulation with replacement and recreate the basic static dynamic of the characters at a much condensed level. However, we are told that it will still be possible to use a simple filter in the secondary system to generate specific surface plans.

Table 13: Examples of 32-step long sequence generation on **Prefix Completion** with Plaid-1B. The output results indicate that LINS helps maintain the coherence and fluency of the paragraph, while ensuring consistency across sentences.

	Example
Source	Lexically Constraints: Donald
Plaid	Sample 1: ... focused in comments on newly uncovered FBI collusion with President **Donald** Trump, Bill Nunes revelation that almost no trump prosecute matters, despite widespread evidence, making him the primary target of these very crimes ...
	Sample 2: ... It also includes some mud slinging with Obama FGA Chairman John Masters attacking **Donald Trump** for loose familiarity with the banking cartel. It's no secret that Trump was beaten to the drapes by President Barack Obama on the actual exam.
	Sample 3: ... March in London Members of Parliament looking at paintings of Prime Minister **Donald** trump during EU festival day Getty Images Brexit protest ...
Plaid+LINS	Sample 1: ... It is also an equal, far more debasive sellout than the academia; some may have reasonably attributed the whole spread to **Donald** Nhutbout not clicking into the comments I have written this time; it seems to be unfortunate because both the coverage and commentary involved that continued focus could just seem to call out bias and denial ...
	Sample 2: ... But at least on campuses, the so called feminist war on radical conservatives may actually be having a negative effect on Loomer attendance. When it comes to that **Donald** Trump presidency, we might actually find men who don't have the viciousness, sorry love, of the feminist left ...
	Sample 3: ... The overproduction chells fields and destroys towns, creates great energy exhaustion and destabilization and allot in the free residents and the general approach to produce any other good or industry fails to expand operations and flows.↩↩**Donald** Stone: Within one of the smaller areas, on the street of **Donald** Stone's home, lies the Exit Man, now the best live erifot and Pisa _example courtesy of writer B. JonKing ...

Table 14: Examples of 32-step long sequence generation on **Lexically Constrained Generation** with Plaid-1B. The Plaid-1B samples lack diversity while ours cover more word combinations.

	Example
Source	Negation: Donald but not Trump
Plaid	... [profile] Frederic Blanchfield Britain's earliest basalt monument in Little Choyside, Chepole, Dorset. Source: MORI.mauipomwork-8x17-1.jpg... [profile] Marco Bori Special Projects Marco Bastoni Participants in discussion with Russian crionic residents in Venice. Artist/21-ponoco-8x17-1.jpg... [profile] Marco Baba.alexnet-8x17-1.jpg... [profile] Giananna Boone Interior Photography Histature from Rome Client – Social Architecture – Vanguard.photo/46-photograph-164x72-r.jpg... [profile] **Donald** Buckingham Industrial Art Izomage ProKnakon Call to exercise Tradarpones in Real Materials-stone-piping-8x17-608px.jpg... [slide] artmagazine-20x26x24.jpg... [slide] ...
Plaid+LINS	... It was dress for contempt. The dishonesty of this whole development was clear, an astonishing and obvious, necessary to humanity, but seen in the remark Il Quartilver made which showed an almost impervolent fit of irony, necessary to discern truth from record. ' Our exile and our exile have been secured. The ocean has risen. Hell, **Donald** haath been a fool, before he came to save me, Taracharvi, and finally ... not only in these troubled times it was taking a giant it-own for their charisma, but sol gallur phir bariet die Pope **Donald**. If they couldn't say 'wrong', where did they get it? The slanders at Timon knew to just make radios, they could have solved ...

Table 15: Examples of 32-step long sequence generation on **Negation** with Plaid-1B. The results demonstrate that the Plaid-1B sample is likely memorized from the dataset when against an unusual pattern, while our method produces a normal paragraph that satisfies the constraint.