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

Anonymous ACL submission

Abstract

Self-conditioning has been central to the success of continuous diffusion language models, as it allows models to correct previous errors. Yet its ability degrades precisely in the regime where diffusion is most attractive for deployment: few-step sampling for fast inference. In this study, we show that when models only have a few denoising steps, inaccurate self-conditioning induces a substantial approximation gap; this mistake compounds across denoising steps and ultimately dominate the sample quality. To address this, we propose a novel training framework that handles these errors during learning by perturbing the self-conditioning signal to match inference noise, improving robustness to prior estimation errors. In addition, we introduce a token-level noise-awareness mechanism that prevents training from saturation, hence improving optimization. Extensive experiments across conditional generation benchmarks demonstrate that our framework surpasses standard continuous diffusion models while providing up to 400x faster inference speed, and remains competitive against other one-step diffusion frameworks.

1 Introduction

Diffusion models have recently emerged as a compelling alternative to autoregressive text generation, matching and in some settings surpassing autoregressive models in quality and controllability (Nie et al., 2025; Prabhudesai et al., 2025). A key advantage of diffusion models lies in their ability to generate all tokens in parallel, offering a linear-time decoding rather than a quadratic complexity as in autoregressive decoding (Radford et al., 2018, 2019; Brown et al., 2020). This property makes diffusion models attractive for a broad range of natural language processing tasks beyond unconditional generation, such as conditional and structured sequence modeling (Gong et al., 2023a; Ye et al., 2024, 2025).

Despite these advantages, diffusion language models face a central efficiency bottleneck: high-quality generation typically requires a long reverse process with many denoising steps (Gong et al., 2023a). The resulting iterative sampling cost eventually offset the latency gains from parallel token generation. To mitigate this issue, a widely adopted trick is to use self-conditioning (Chen et al., 2023b; Gulrajani and Hashimoto, 2023), which reuses previous predictions as an additional conditioning signal to improve prediction under fewer steps. While self-conditioning indeed strengthens few-step sampling, we find that it introduces underappreciated failure modes that become critical precisely in the fast-inference application.

Mismatching between training-inference self-condition under few-step denoising. Self-conditioning introduces an intrinsic mismatch between training and inference (Schmidt, 2019; Ning et al., 2024; Gao et al., 2024). During training, a model can be conditioned on ground-truth targets, whereas at inference, it must be conditioned on its own imperfect previous predictions. This distribution shift induces error accumulation along the reverse trajectory, leading to sampling drift (Daras et al., 2023) and degraded generation quality. Our analysis shows that the problem is amplified in few-step settings: predictions made at high noise levels differ significantly from those made later at low noise, turning the reused self-conditioning signal into a biased condition. Consequently, self-conditioning can become unstable, and in the worst case, the reused estimates can steer subsequent denoising updates in the wrong direction.

Loss saturation in the late-stage training. Diffusion language models often fit the denoising objective quickly early in training, but subsequently exhibit a pronounced loss plateau. This slow improvement suggests that the sampled noise levels become insufficiently informative: the training sig-

nal is dominated by "easy" cases where tokens are already predicted with high confidence, leading to inefficient learning. Prior works attribute this behavior in part to applying a uniform noise schedule across tokens, which ignores token-wise heterogeneity in denoising difficulty (Yuan et al., 2024). Consistent with this view, our analysis shows that uniform noise sampling is suboptimal. In particular, increasing noise for well-predicted tokens yields a more effective learning signal and improves optimization efficiency, thus achieving lower evaluation loss than models trained with the uniform schedule.

To address these challenges, we propose **Fast Diffusion Sequence-to-Sequence** (FastDiSS), a novel training framework designed to improve the robustness and efficiency of the diffusion model in the few-step setting. Building on self-conditioning, FastDiSS introduces two complementary components that directly target the above failure modes. First, we propose **Self-conditioning Perturbation** (SCP), a simple regularization strategy for self-conditioning. Initially, during training, we obtain this condition by running the denoising network on a more-noised forward process, producing a weaker and noisier estimation. We then train the network conditioned on this corrupted signal, better matching inference-time errors and reducing sampling drift. Second, we introduce **Model-aware Noise Scaling** (MANS), a token-level noise allocation strategy that dynamically adjusts noise based on denoising confidence. MANS applies higher noise to high-confidence tokens, prevents trivial training, and further reduces self-conditioning errors at high noise levels.

We evaluate FastDiSS on six benchmarks covering diverse sequence-to-sequence tasks and few-step generation settings. Across settings, FastDiSS consistently narrows the gap between few-step and many-step sampling, outperforming prior text diffusion baselines in both quality and efficiency. In particular, FastDiSS improves *BLEU* score (Papineni et al., 2002) while achieving substantial speedups ranging from 4× to 400×, and remains competitive with other few-step diffusion approaches.

In summary, our contributions are threefold: (1) we identify and analyze two bottlenecks that limit self-conditioned diffusion language models under few-step samplings, highlighting the roles of discretization-induced mismatch and late-stage training saturation; (2) we introduce FastDiSS, combining SCP to regularize self-conditioning under realistic inference noise and MANS to avoid trivial denoising via confidence-driven token-wise noise allocation; and (3) we demonstrate consistent gains on six benchmarks, showing that FastDiSS improves generation quality while substantially accelerating inference.

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

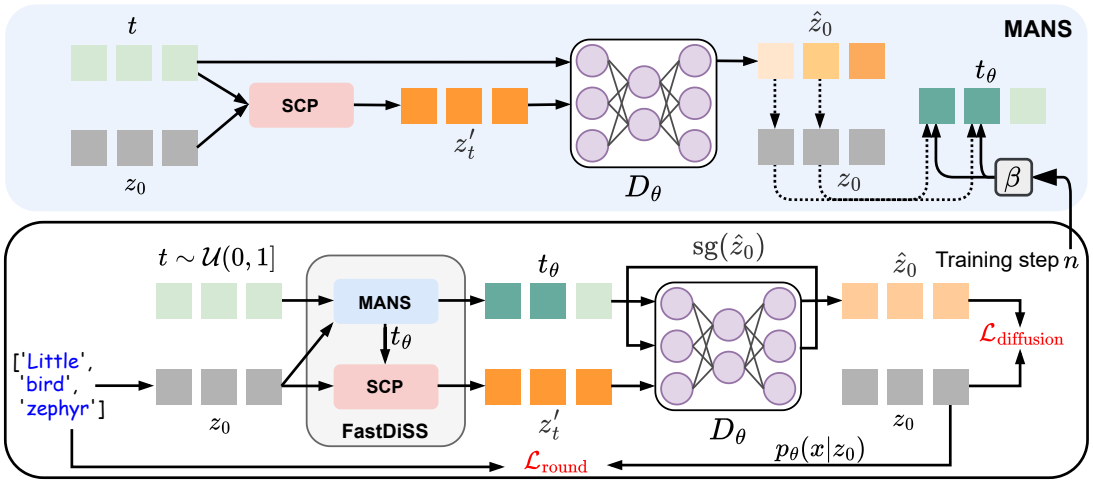

Figure 1: Overview of FastDiSS. The tokenized sequence is first encoded to z_0, while concurrently sampling the initial timestep t. Both z_0 and t are passed into MANS to obtain the new timestep t_θ. Subsequently, noise level at t_θ is added to z_0 using SCP to obtain z'_t. The rest is the same as in the training objective in Eq. 5.

space $z_0 \in \mathbb{R}^{L\times H}$, with sequence length L, hidden dimension H, and vocabulary size V. Hence, $z_0 \sim q(z_0|x)$ is the embedding codebook of x.

The reverse process aims to generate z_0, which is then mapped back to x. The diffusion model is trained on the latent space with the objective

$$\begin{aligned}\mathcal{L}_{\text{total}} &= \mathcal{L}_{\text{diffusion}} + \mathcal{L}_{\text{round}} \\ &= \mathbb{E}_{z_0,t}[||D_\theta(z_t) - z_0||^2] + \mathbb{E}_{z_0}\left[-\log p_\theta(x|z_0)\right], \end{aligned} \tag{5}$$

where $\mathcal{L}_{\text{round}}$ is the $z_0 \to x$ reconstruction loss.

To extend the model for conditional generation, a simple yet effective approach is to incorporate the conditioning sequence c as an additional input to the denoising network, i.e., $D_\theta(z_t, c)$. The target sequence length L can be inferred from the conditioning context via a learned prior $L \sim p(L|c)$. 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 $\hat{z}_\theta^t$ is fed back into the denoising model as an auxiliary condition. The z_0-prediction then becomes $\bar{z}_\theta^t = D_\theta(z_t, \mathrm{sg}(\hat{z}_\theta^t))$, where $\mathrm{sg}(\cdot)$ denotes the stop-gradient operation, preventing gradient propagation through $\hat{z}_\theta^t$. For clarity, we omit the source condition c, as the discussion here focuses solely on the self-conditioning mechanism. The self-conditioning training objective becomes:

$$\mathcal{L}_{\text{sc}} = \mathbb{E}_{z_0,t}[||D_\theta(z_t, \mathrm{sg}(\hat{z}_\theta^t)) - z_0||^2]. \tag{6}$$

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

At each step, estimating $\hat{z}_\theta^t$ followed by $\bar{z}_\theta^t$ 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 $\bar{z}_\theta^{tu}$, for any $0 < s < t < u \leq 1$.

3 Methodology

In this section, we describe the design of FastDiSS for efficient sequence-to-sequence language generation. Fig. 1 provides an overview. Building upon the standard text diffusion architecture, FastDiSS augments training with two tightly coupled components: Self-conditioning Perturbation (SCP) and Model-aware Noise Scaling (MANS). Together, they target the two bottlenecks identified in Sec. 1: (i) mismatch between training-time and inference-time self-conditioning under few-step discretization, and (ii) late-stage training saturation caused by uninformative, token-unaware noise.

3.1 Self-conditioning Limitations

As described in Sec. 2.3, using the reused estimation $\bar{z}_\theta^{tu}$ in place of the step-matched prediction $\bar{z}_\theta^t$ introduces a training-inference mismatch. Intuitively, estimation from a coarser step u carries larger uncertainty than that from step t (Bao et al., 2022; Ning et al., 2024), so $\bar{z}_\theta^{tu}$ is more likely to deviate from the target embedding z_0. To measure the effect of this estimation gap, we report the *BLEU* score on the generation of **IWSLT14** De-En dataset, using the correct $\bar{z}_\theta$ and the original $\bar{z}_\theta^{tu}$ self-condition sampling. In Tab. 1, we conduct experiments 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 Self-conditioning Perturbation

We propose SCP to reduce the training-inference mismatch of self-conditioning. Intuitively, SCP simulates inference self-condition behaviors by regularizing the correct self-condition in training, thereby improving stability and reducing drift without changing the sampling procedure.

The perturbed self-conditioning is obtained from the denoising output of a modified forward process, in which a perturbed version $\boldsymbol{z}'_t$ of $\boldsymbol{z}_t$ is introduced:

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

where $\boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I})$. The factors λ_t and γ_t corrupt the signal term and inflate the effective noise level, yielding a more aggressively noised input and thus a less reliable self-conditioning estimate. We parameterize them as linear schedules, $\lambda_t = (\lambda_{\min} - \lambda_{\max})t + \lambda_{\max}$ and $\gamma_t = (\gamma_{\min} - \gamma_{\max})t + \gamma_{\max}$, with monotonic variation over t. The hyperparameter choice is guided by the forward ratios α_u/α_t and σ_u/σ_t, so that $\boldsymbol{z}'_t$ matches the scale of a later state along the trajectory and approximates the conditioning statistics induced by prior estimation.

Empirically, we observe that the reused estimate behaves as an approximately Gaussian perturbation around the current prediction $\hat{\boldsymbol{z}}_\theta^t$, supporting the view that SCP captures the noise pattern introduced by inference-time self-conditioning. A simplified derivation and detailed distributional analysis are provided in Appx. B.

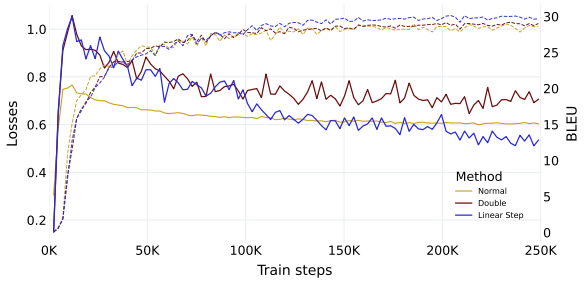

Figure 2: Validation loss and *BLEU* during training under fixed, double, and linear step noise scaling. Dashed lines denote *BLEU* scores, color-matched to the corresponding loss curves.

We summarize the training procedure in Appendix (see Alg. 1). Conventionally, we randomly apply self-conditioning with probability 50%, alternating between conditioned and unconditioned updates to keep the initial prediction meaningful.

3.3 Model-aware Noise Scaling

We introduce MANS to make the diffusion objective token-aware, mitigating late-stage loss saturation. The key idea is to adaptively *increase* noise for tokens the model already denoises confidently, while leaving uncertain tokens unchanged. This strengthens supervision at higher noise levels where few-step discretization is most brittle, and improves self-conditioning even under heavier corruption.

To quantify denoising confidence, we evaluate reconstructed embedding quality $\hat{\boldsymbol{z}}_\theta^t$ by mapping it to the nearest token embeddings $\boldsymbol{e}_m$:

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

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

where i and j denote the ground-truth and reconstructed tokens, respectively. We treat $i = j$ as a high-confidence token, indicating that the denoiser can already recover it reliably at noise level t.

We then define a *noise scaling schedule* that increases denoising difficulty for these high-confidence tokens. Specifically, at training iteration n, we rescale the effective timestep used to corrupt the token by a model-aware factor $\beta(n)$:

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

Intuitively, correctly reconstructed tokens are pushed to a higher-noise regime so the model continues to receive informative gradients, while incorrectly reconstructed tokens remain at the original difficulty to avoid destabilizing learning.

Method	MBR	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‡	50	20	29.11	22.91	19.69	17.41	30.17	29.39
Diffomer‡	20	20	28.01	23.31	25.30	23.80	29.37	29.20
DINOISER	50	20	31.61	25.70	29.05	24.26	31.22	31.08
DiffuSeq	10	2000	29.43	-	22.72	-	-	-
SeqDiffuSeq	10	2000	30.45	22.12	23.93	19.76	-	-
AR-Diffusion	10	20	31.80	26.01	-	-	-	-
FastDiSS	10	5	32.46	25.73	29.54	24.33	31.55	30.88
FastDiSS	20	5	32.70	26.02	<u>29.75</u>	**24.69**	31.81	30.90
FastDiSS	10	20	<u>32.81</u>	<u>26.29</u>	29.47	24.50	<u>31.89</u>	<u>31.37</u>
FastDiSS	20	20	**32.88**	**26.39**	**29.83**	<u>24.57</u>	**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 <u>underlined</u>. ‡ indicates results reported by Ye et al. (2023). Other results are from their original papers.

While $\beta(\cdot)$ can in principle take many forms, our ablations indicate that *progressively increasing* the noise is crucial to overcoming loss saturation. As shown in Fig. 2, a constant doubling of noise ($\beta = 2.0$) yields limited gains and does not consistently break saturation, whereas a Linear schedule continues to improve both validation loss and BLEU even in late training.

Accordingly, we instantiate $\beta(\cdot)$ as a linear stepping schedule, where the scaling factor increases after predefined iteration milestones. This ensures that once the model has sufficiently optimized the current noise regime, it is presented with a systematically harder denoising target, keeping the training signal non-trivial throughout optimization. Qualitative examples illustrating high and low confidence tokens are provided in Appx. G.

4 Experiments

4.1 Experimental Settings

Datasets. Following prior works (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); 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 models: Transformer (Vaswani et al., 2017), GRU with attention, and fine-tuned GPT2-Large; (2) Non-autoregressive model: CMLM (Ghazvininejad et al., 2019) and LevT (Gu et al., 2019); and (3) Diffusion language models: DiffusionLM (Li et al., 2022), Differmer (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 Summarization, we 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 also include few-step generation benchmarks, DiffuSeq-v2 (Gong et al., 2023b), FlowSeq (Hu et al., 2024) and DLM-One (Chen et al., 2025).

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 fasten training. Our implementation is based on Differmer, with the same sampling configurations at NFE $\in \{2, 5, 20\}$. For every task, we construct the vocabulary with Byte Pair Encoding (Kudo and Richardson, 2018). We also apply Minimum Bayes Risk (MBR) decoding (Kumar and Byrne, 2004) following previous works (Li et al., 2022; Gong et al., 2023a). Further details are described in Appx. E and F.

Method	MBR	NFE	QQP			Wiki-Auto		
			BLEU↑	*Rouge-L*↑	*BertScore*↑	*BLEU*↑	*Rouge-L*↑	*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*	10	20	27.95	59.24	82.97	34.78	54.55	78.86
DINOISER†	10	20	19.49	53.16	80.36	23.88	48.21	67.87
DiffuSeq	10	2000	24.13	58.80	83.65	36.22	58.49	81.26
SeqDiffuSeq	10	2000	24.34	-	**84.00**	37.12	-	82.14
DiffuSeq-v2*	10	10	23.07	58.35	82.36	26.60	51.33	77.04
FlowSeq*	10	1	14.30	46.10	66.90	29.02	53.74	72.46
DLM-One	10	1	22.13	57.41	82.97	36.30	58.39	80.84
FastDiSS	10	1	27.16	58.10	81.16	38.20	57.66	80.35
FastDiSS	10	2	27.94	58.47	81.81	40.23	59.10	81.60
FastDiSS	10	5	**28.88**	**59.34**	82.58	**40.90**	**59.64**	82.16
FastDiSS	10	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.

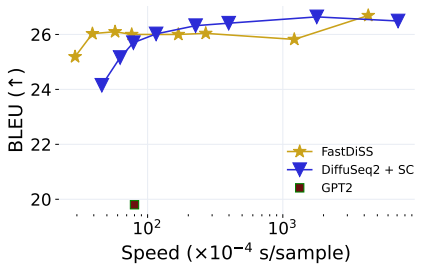

Figure 3: Generation speed and quality with different NFE. The speed is averaged over 3 runs.

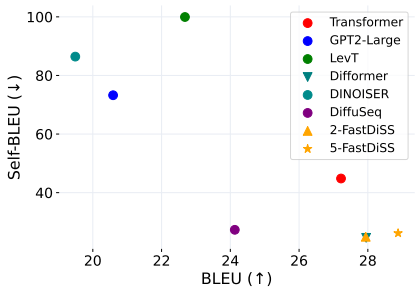

Figure 4: Diversity and quality comparison.

4.2 Main Results

Overall Performance. Tabs. 2 and 3 summarize results across tasks and datasets. Overall, FastDiSS improves few-step diffusion generation, outperforming both Diffusion and Non-autoregressive baselines on most settings (varying MBR and NFEs), while approaching Autoregressive performance. On **WMT14**, the 5-step model even surpasses 20-step sampling, indicating that SCP can effectively close the gap between few and many inference.

In terms of efficiency, FastDiSS is 4× faster than DINOISER, Differmer, and up to 400× faster than long-trajectory methods such as DiffuSeq and SeqDiffuSeq. Compared to one-step baselines, FastDiSS remains competitive without relying on distillation (DLM-One) or flow-matching (FlowSeq) objectives (Tab. 3). Additional results on Text Summarization are reported in Appx. D.

Sampling Speed. We analyze the quality–speed trade-off on **QQP** by comparing FastDiSS with DiffuSeq-v2 using its original self-conditioning mechanism (Gong et al., 2023b). Fig. 3 shows that FastDiSS consistently achieves higher quality at low NFEs, highlighting its advantage in the few-step regime. The margin narrows at medium NFEs, where DiffuSeq-v2 slightly overtakes FastDiSS, suggesting that the benefit of SCP diminishes once discretization error becomes small.

Sampling Diversity. We evaluate diversity on **QQP** using *BLEU* and self-*BLEU*. Fig. 4 shows the trade-off between diversity and quality. Fig. 4 shows that FastDiSS-2NFE matches the strongest baseline, Differmer-20NFE, with 10× smaller bugdet. Increasing NFEs improves quality at the expected cost of latency. Notably, FastDiSS-5NFE achieves substantially higher *BLEU* while maintaining similar self-*BLEU*, indicating that quality gains do not come at the expense of diversity.

SCP	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 4: Ablation study.

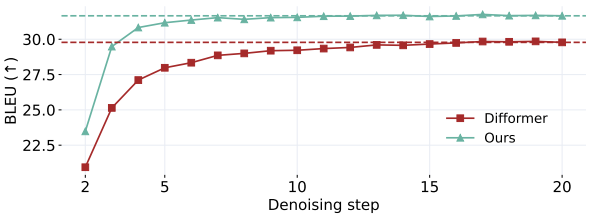

Figure 5: Effect of Number of Denoising Steps.

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 5: Effect of γ_t and λ_t.

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: Effect of Noise Schedulers.

4.3 Ablation Studies

Effect of Each Component. We quantify the contribution of SCP and MANS in Tab. 4. The evaluation is conducted on **IWSLT14** De-En using *BLEU*. The results show that both components provide consistent gains over the base model. MANS is particularly effective at small NFEs, suggesting it improves prediction at large steps, reducing self-conditioning errors. In contrast, SCP mainly improves inference by reducing the training-inference gap, which is most visible in the few-step regime. Combining SCP and MANS yields the strongest performance and significantly narrows the gap between 5-step and 20-step sampling.

Effect of Varying NFEs. We vary NFEs on **IWSLT14** De-En and compare `FastDiSS` against the original codebase, `Differmer`. Fig. 5 shows that `FastDiSS` is markedly more effective under few-step inference: it surpasses the 20-step baseline within 3 sampling steps, corresponding to approximately 7× speedup. Moreover, performance converges after roughly 7 steps, reaching a higher *BLEU* than the 20-step baseline.

Effect of γ_t and λ_t. We study sensitivity to γ_t and λ_t in SCP (Eq. 7) on **QQP**. Tab. 5 compares linear-time schedules against fixed variants where λ_t and γ_t are held constant. The best results are obtained with settings derived from the 20-step ratios, where γ_t ranges from 0.90 to 0.95 and λ_t ranges from 0.15 to 0.35. We use this configuration in the remaining experiments.

Effect of Noise Schedulers. Finally, we compare `FastDiSS` under standard diffusion noise schedules, including `linear` (Ho et al., 2020) and `cosine` (Nichol and Dhariwal, 2021), on **QQP**. Tab. 6 shows that the `linear` schedule is generally less sensitive to the choice of NFE. `FastDiSS` consistently yields improvements in the few-step regime across schedulers, confirming that our gains are not tied to a specific base schedule.

4.4 Extension to Large-scale Benchmark

We evaluate SCP adaptability to reasoning benchmark **GSM8K** (Cobbe et al., 2021) and discrete setting in Tabs. 7 and 8.

Reasoning benchmark. We adopt DoT (Ye et al., 2024), using `Plaid-1B` (Gulrajani and Hashimoto, 2023) as the backbone. We introduce SCP as a lightweight modification following `Plaid` training recipe. Tab. 7 shows that SCP consistently improves accuracy under both standard diffusion inference and Chain-of-Thought (CoT) inference. More importantly, when reducing NFEs, SCP substantially mitigates the drop in accuracy, indicating improved robustness in the low-compute regime. Qualitative examples comparing `Plaid` with and without SCP are provided in Appx. H.

Discrete Diffusion Model Extension. MDLM, the masking diffusion framework behind LLaDA (Nie et al., 2025), operates via iterative masking and unmasking. In this setting, SCP corresponds to increasing the masking rate during training, producing a more corrupted conditioning context that better matches few-step inference. As shown in Tab. 8, SCP improves robustness at small NFEs compared to self-conditioning, and yields consistent gains over the original model as NFEs increase.

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

Table 7: Comparison of *Accuracy* on **GSM8K** benchmark. We experimented with two settings, DoT with Plaid inference and DoT^{mp} with CoT reasoning. The *Accuracy* is averaged over 5 runs.

NFE	Gen Perplexity (↓)		
	None	SC	SCP
2	3178.74	3250.22	**2873.67**
5	1144.42	1433.79	**1215.41**
20	243.81	254.72	**244.40**
50	143.82	133.79	**130.52**
100	113.34	97.88	**95.48**
1000	55.23	45.95	**45.21**
5000	32.12	**27.78**	28.14

Table 8: Main results on MDLM. The best Generation Perplexity results are **bold**.

5 Related Works

Non-autoregressive Language Generation. Non-autoregressive models were early introduced to reduce generation latency by predicting tokens in parallel (Gu et al., 2017). To improve accuracy, later work incorporated iterative refinement and editing-style decoding (Gu et al., 2019; Ghazvininejad et al., 2019). Despite these advances, the conditional independence assumption in many NAR formulations leads to a single mode selection issue. Prior efforts mitigate this issue via structured prediction (Zhang et al., 2022; Ran et al., 2020; Huang et al., 2022), data augmentation and selection (e.g., reference rephrasing) (Shao et al., 2022, 2023), and distillation to transfer knowledge from autoregressive teachers (Guo et al., 2021; Qi et al., 2022; Liu et al., 2023).

Diffusion Models for Language Generation. Text diffusion models can be broadly categorized into *discrete* and *continuous* formulations. Discrete diffusion defines Markov transitions directly over token space. Early approaches such as D3PM (Austin et al., 2021) and Analog-bits (Chen et al., 2023b) rely on absorbing or uniform transitions, while more recent methods adopt masking-style transitions that scale more naturally to language, including MDLM (Sahoo et al., 2024) and RDM (Zheng et al., 2023). These advances have enabled large-scale diffusion LLMs such as LLaDA (Nie et al., 2025), Dream-7B (Ye et al., 2025), and Block Diffusion (Arriola et al., 2025). In contrast, continuous diffusion maps tokens into a continuous embedding space and applies Gaussian processes, enabling denoising in latent space. Representative models include DiffusionLM (Li et al., 2022), Differmer (Gao et al., 2024), DINOISER (Ye et al., 2023), SeqDiffuSeq (Yuan et al., 2024), AR-Diffusion (Wu et al., 2023), and DiffuSeq (Gong et al., 2023a), spanning encoder-decoder and decoder-only designs. Several works further tailor diffusion to language by modifying the noise schedule or corruption process, including Masked-Diffuse (Chen et al., 2023a) and Meta-Diffu$B$ (Chuang et al., 2024). Our method is developed in the continuous latent setting to explore the full potential of continuous modeling.

Accelerating the Diffusion Language Model. Widely adopted methods, such as DDIM (Song et al., 2021a), have demonstrated success in prior text generation works. Advances in ODE solvers (Lu et al., 2022, 2025) have made great success in enhancing the efficiency of the reverse process in DiffuSeq-v2 (Gong et al., 2023b). Tang et al. (2023) also addresses training-inference discrepancies to enhance generation speed. Recently, FlowSeq (Hu et al., 2024) adapts flow matching for sequence modeling, while DLM-One (Chen et al., 2025) distills the teacher model, DiffuSeq, to learn a one-step generator. 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.

6 Conclusion

We introduced FastDiSS, a novel training framework for diffusion language models, combining Self-conditioning Perturbation (SCP) to align self-conditioning under few-step discretization and Model-aware Noise Scaling (MANS) to increase noise on high-confidence tokens for more informative training. 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 models, future work will focus on further refining self-conditioning and closing 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.

Fan Bao, Chongxuan Li, Jiacheng Sun, Jun Zhu, and Bo Zhang. 2022. Estimating the optimal covariance with imperfect mean in diffusion probabilistic models. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1555–1584.

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.

Tianqi Chen, Shujian Zhang, and Mingyuan Zhou. 2025. Dlm-one: Diffusion language models for one-step sequence generation. *CoRR*, abs/2506.00290.

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.

Giannis Daras, Yuval Dagan, Alex Dimakis, and Constantinos Daskalakis. 2023. Consistent diffusion models: Mitigating sampling drift by learning to be consistent. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural*

Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023.

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)*.

Vincent Tao Hu, Di Wu, Yuki Markus Asano, Pascal Mettes, Basura Fernando, Björn Ommer, and Cees Snoek. 2024. Flow matching for conditional text generation in a few sampling steps. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2024 - Volume 2: Short Papers, St. Julian's, Malta, March 17-22, 2024*, pages 380–392. Association for Computational Linguistics.

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.

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.

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, 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.

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.

Algorithm 1 Training with SCP

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

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:

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

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

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

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.

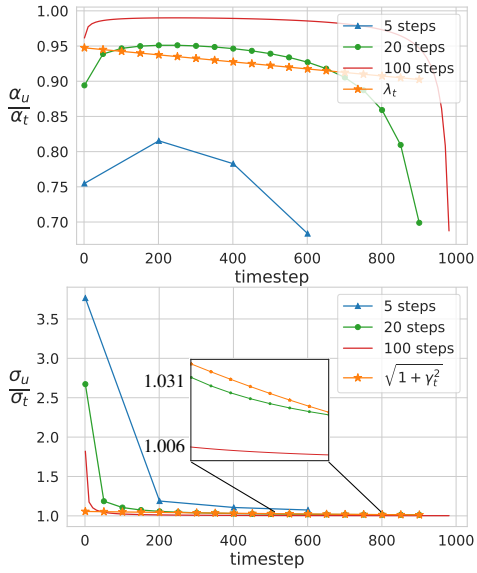

Figure 6: Mean (top) and variance (bottom) ratio differences measured at different denoising steps. We manually select hyperparameters of the scaling terms to fit 20-step sampling.

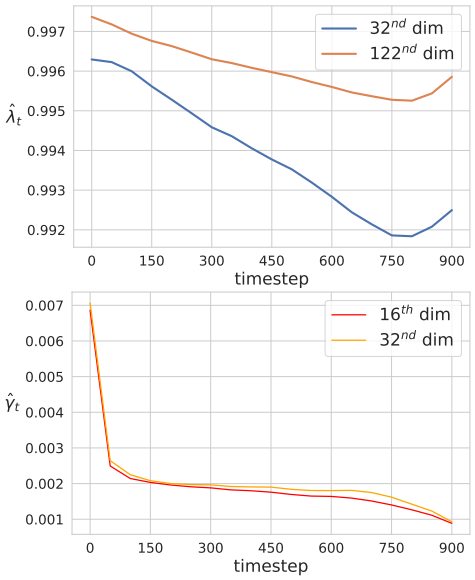

Figure 7: Inference mean $\hat{\lambda}_t$ (top) and variance $\hat{\gamma}_t$ (bottom) scaling factors of prediction mismatch in a pretrained network, plotted across timestep t for randomly selected embedding dimensions.

B Self-conditioning Error Formulation

This section provides empirical and analytical support for SCP. We show that the perturbed forward sample $\boldsymbol{z}_t'$ used in training induces self-conditioning statistics that closely match those observed at inference, and we motivate the linear parameterization of λ_t and γ_t.

B.1 Approximate Gaussian Distribution Between Consecutive Estimates

Following Sec. 3.1, we train a standard continuous diffusion language model on **IWSLT14** De-En and run the original 20-step denoising procedure on the validation set. At each step, we store the previous-step reused estimate $\bar{\boldsymbol{z}}_\theta^{tu}$ and compute the corrected self-conditioning estimate for the current step, denoted $\hat{\boldsymbol{z}}_\theta^s$.

Empirically, we observe that $\bar{\boldsymbol{z}}_\theta^{tu}$ is well-approximated by a Gaussian perturbation around $\hat{\boldsymbol{z}}_\theta^s$:

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

where $\hat{\boldsymbol{\mu}}_{st}$ and $\hat{\boldsymbol{\sigma}}_{st}$ are dimension-wise mean and standard deviation vectors (details in Appx. B.2). This relation makes the inference-time self-conditioning mismatch explicit and allows us to connect it to the perturbed forward construction in Eq. 7.

Assuming that $\hat{\boldsymbol{z}}_\theta^s$ perfectly denoises $\bar{\boldsymbol{z}}_s$ at step s, we start from the standard forward parameterization,

$$\bar{\boldsymbol{z}}_s = \alpha_s\hat{\boldsymbol{z}}_\theta^s + \sigma_s\boldsymbol{\epsilon}_s, \tag{16}$$

and substitute $\hat{\boldsymbol{z}}_\theta^s$ using Eq. 15. Rearranging terms yields

$$\begin{aligned}\bar{\boldsymbol{z}}_s &= \alpha_s\frac{\bar{\boldsymbol{z}}_\theta^{tu} - \hat{\boldsymbol{\sigma}}_{st}\boldsymbol{\epsilon_u}}{\hat{\boldsymbol{\mu}}_{st}} + \sigma_s\boldsymbol{\epsilon}_s \\ &= \alpha_s\frac{1}{\hat{\boldsymbol{\mu}}_{st}}\bar{\boldsymbol{z}}_\theta^{tu} + \sigma_s\sqrt{1+\left(\frac{\alpha_s}{\sigma_s}\frac{\hat{\boldsymbol{\sigma}}_{st}}{\hat{\boldsymbol{\mu}}_{st}}\right)^2}\boldsymbol{\epsilon} \\ &= \alpha_s\hat{\boldsymbol{\lambda}}_{st}\bar{\boldsymbol{z}}_\theta^{tu} + \sigma_s\sqrt{1+\hat{\boldsymbol{\gamma}}_{st}^2}\boldsymbol{\epsilon},\end{aligned} \tag{17}$$

where $\hat{\boldsymbol{\lambda}}_{st} = 1/\hat{\boldsymbol{\mu}}_{st}$, $\hat{\boldsymbol{\gamma}}_{st} = (\alpha_s/\sigma_s)(\hat{\boldsymbol{\sigma}}_{st}/\hat{\boldsymbol{\mu}}_{st})$, and $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$ is obtained via reparameterization of the combined noise terms. Since $\hat{\boldsymbol{\mu}}_{st}$ and $\hat{\boldsymbol{\sigma}}_{st}$ vary independently across dimensions, all operations are element-wise.

Eq. 17 mirrors the structure of our perturbed forward process (Eq. 7): both corrupt the signal term and inflate the effective noise through a multiplicative factor. Consistent with this connection, the empirically observed scaling patterns in Fig. 7 match the behavior induced by SCP in Fig. 6, supporting the view that SCP exposes the model during training to the noise pattern encountered when reusing self-conditioning at inference.

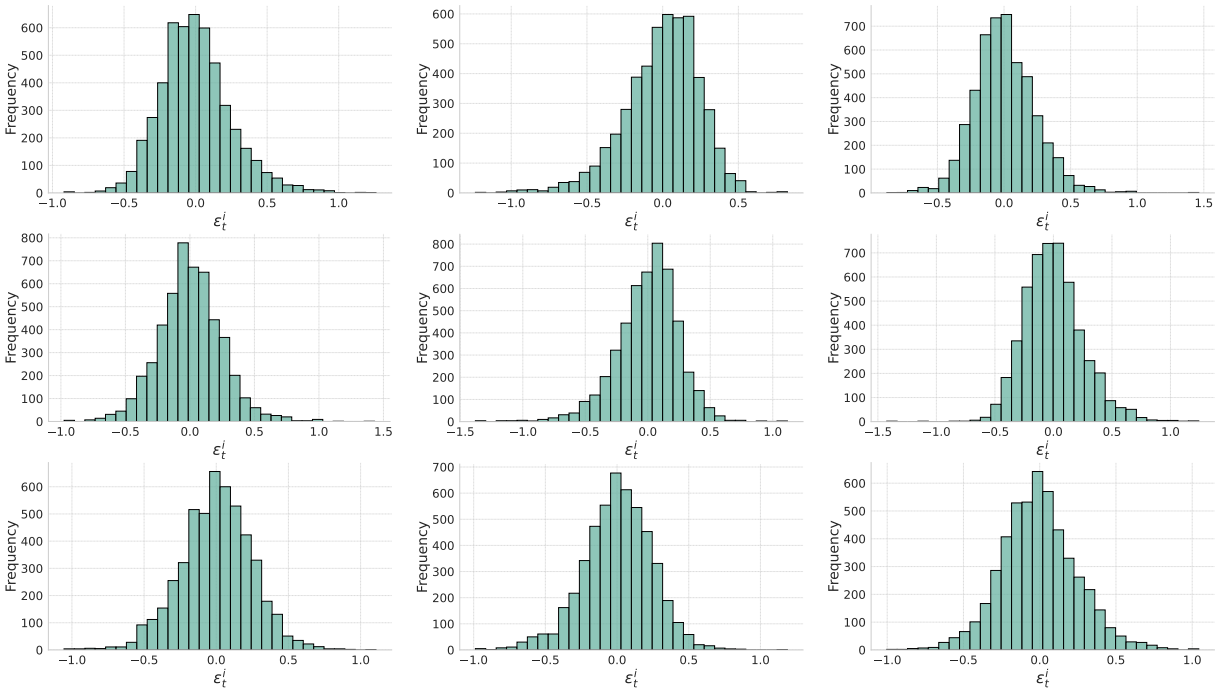

Figure 8: The empirical distribution of ϵ_t^i with different random values of t and i.

In principle, the empirical schedules $\{\hat{\boldsymbol{\lambda}}_{st}, \hat{\boldsymbol{\gamma}}_{st}\}_{t=1}^{T}$ could be estimated directly from Eq. 17. However, these quantities depend on the discretization size and vary across dimensions, making them difficult to express with a single global function. To avoid high-dimensional hyperparameter search, we instead use feature-independent anchor points $(\lambda_{\min}, \lambda_{\max})$ and $(\gamma_{\min}, \gamma_{\max})$ to define monotone linear schedules. This yields a simple end-to-end procedure that preserves the intended dynamics without introducing additional preprocessing overhead.

B.2 Estimate The Self-condition Gaussian Distribution Error

We now describe how we estimate the statistics in Eq. 15 on **IWSLT14** De-En. For each diffusion time t and each embedding dimension $i \in \{1, \ldots, H\}$, we consider the dimension-wise residual

$$\epsilon_t^i = \bar{z}_\theta^{tu,i} - \hat{\mu}_t^i \hat{z}_\theta^{s,i}, \tag{18}$$

and test whether ϵ_t^i is approximately Gaussian with variance $(\hat{\sigma}_t^i)^2$.

We uniformly select 20 timesteps in $[\epsilon, 1]$, run denoising at these timesteps, and record paired values of $\bar{\boldsymbol{z}}_\theta^{tu}$ and $\hat{\boldsymbol{z}}_\theta^{s}$. For each selected t, we sample 1,000 sentences $z \in \mathcal{D}$, flatten all tokens, and estimate $\hat{\mu}_t^i$ and $\hat{\sigma}_t^i$ for each dimension.

Estimating $\hat{\mu}_t^i$ reduces to a one-dimensional linear regression (OLS) per dimension:

$$\hat{\mu}_t^i = \frac{\sum_{z \in \mathcal{D}} \bar{z}_u^i \hat{z}_t^i}{\sum_{z \in \mathcal{D}} (\hat{z}_t^i)^2}, \tag{19}$$

and $\hat{\sigma}_t^i$ is computed as the standard deviation of the residuals:

$$\hat{\sigma}_t^i = \sqrt{\frac{1}{|\mathcal{D}|} \sum_{z \in \mathcal{D}} (\bar{z}_u^i - \hat{\mu}_t^i \hat{z}_t^i)^2}. \tag{20}$$

We then standardize residuals and apply the Shapiro-Wilk test (Shapiro and Wilk, 1965) to 50 randomly selected standardized residuals per dimension. Using a 95% confidence level, we reject normality if $p < 0.05$. The null hypothesis is rejected only in a small minority of cases, supporting our Gaussian approximation. Fig. 8 shows histograms of ϵ_t^i across different dimensions.

C Comparison with Other Methods

C.1 Training And Inference Mismatch

`Distance Penalty` (Tang et al., 2023) is proposed to perturb the forward process for train-test discrepancy reduction. While conceptually related, it does not target the *self-conditioning* mismatch that is central in our analysis. In addition, this strategy applies a fixed penalty across steps, which mirrors the fixed variants in Tab. 5 and is empirically less effective than our time-varying perturbation.

Method	MBR	NFE	*Rouge-1* (↑)	*Rouge-2* (↑)	*Rouge-L* (↑)
LSTM	5	-	34.2	16.0	31.8
CMLM	-	-	34.4	15.6	32.2
NAG-BERT	-	-	<u>35.1</u>	16.5	**33.3**
Differmer*	10	20	34.9	<u>17.0</u>	32.4
DiffuSeq	10	2000	31.2	12.2	29.2
SeqDiffuSeq	0	2000	31.9	12.4	29.2
FastDiSS	10	5	34.9	16.9	32.5
FastDiSS	10	20	**35.3**	**17.3**	<u>32.8</u>

Table 10: Main results on **Gigaword**. The best NAR results are **bold** and the second-best results are <u>underlined</u>. Baseline results are from Differmer (Gao et al., 2024). $*$ indicates reproduced results.

TREC also discusses collapse phenomena under self-conditioning, however their explanation emphasizes shortcut behavior induced by a predicted prior rather than the discretization-amplified self-conditioning mismatch that we studied. Our approach directly regularizes the self-condition so that training better matches inference-time reuse errors under few-step updates.

C.2 Modified Noise Scheduler

SeqDiffuSeq (Yuan et al., 2024) assigns token difficulty by *position*, yielding position-specific training trajectories. However, the far later position schedules become similar (e.g., their Fig. 2), suggesting diminishing improvement as sequence length grows. In contrast, MANS is length-agnostic: it adjusts noise based on model confidence at the token level, and therefore independent of the sequence length.

DINOISER (Ye et al., 2023) emphasizes high-intensity noise to strengthen learning, but high-noise training can introduce large variance and bias learning toward marginal distributions, eventually reducing sample diversity. Our MANS instead increases noise *selectively* for high-confidence tokens while leaving uncertain tokens unchanged, maintaining meaningful supervision.

Meta-Diffu$B$ (Chuang et al., 2024) learns a planning schedule using an auxiliary controller (LSTM) optimized with reinforcement learning. This adds extra modules and increases training complexity, which can be unstable in practice, and its generalization to unseen contexts or long sequences depend largely on the planner choice. In contrast, FastDiSS avoids training an auxiliary planner, and are simple yet effective.

D Additional Results

Tab. 10 shows the experimental results on Text Summarization benchmark. Consistent with previous results in Tabs. 2 and 3, we observe that our model outperforms most of the diffusion and non-autoregressive baselines on all metrics.

E Experimental Settings

Data. For preprocessing, we use `fairseq` library for **IWSLT14**, and use the preprocessed data released by Fully-NAT (Gu and Kong, 2021) for **WMT14** and **WMT16**[1]. For **Wiki** and **QQP**, we use the ones from DiffuSeq[2], and for **Gigaword** we use the HuggingFace version[3]. All datasets are tokenized with byte-pair encoding (BPE) and processed with `fairseq-preprocess`. BPE settings and vocabulary sizes are reported in Tab. 11.

Model. We use a Transformer-base backbone (Vaswani et al., 2017) with 6 encoder and 6 decoder layers for all datasets. The number of attention heads, hidden size, and related hyperparameters are listed in Tab. 11. The diffusion embedding dimension is 128. For SCP, we set anchor points tuned to 20-step sampling, setting $(\gamma_{\min}, \gamma_{\max}) = (0.90, 0.95)$ and $(\lambda_{\min}, \lambda_{\max}) = (0.15, 0.35)$.

Training. All models are trained with `fp16` on 4 NVIDIA H100 GPUs. We use an `inverse-sqrt` learning-rate schedule with 10,000 warmup steps and $lr_{\max} = 5 \times 10^{-4}$ for all benchmarks. We set gradient norm clipping to 1.0, dropout to 0.1, and label smoothing to 0.1. Runtime is approximately 8.5 hours for **WMT** and **Gigaword**, and around 4 hours on average for the other datasets.

Inference. At inference, the reverse process follows Eq. 4. Self-conditioning reuses the previous-step estimate, as in prior work. We apply Minimum Bayes-Risk (MBR) decoding (Kumar and Byrne, 2004) following DiffusionLM and DiffuSeq.

MANS. We implement MANS with three training phases and increase the scaling factor $\beta(n)$ over time. The phase interval and scaling factor are given in Tab. 11. For example, on **WMT14** we use $\beta(n) = 2.0$ for $n < 100$K, $\beta(n) = 3.0$ for $100\text{K} \leq n < 200$K, and $\beta(n) = 4.0$ thereafter. This modification increases total training time by less than 5% in our experiments.

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

F On The Effectiveness Of Minimum Bayesian Risk Decoding

Since FastDiSS builds on Differmer, we evaluate MBR decoding with both length beam search and noise-candidate search to improve output quality while maintaining diversity. We note that this search overhead can often be reduced by alternative stochastic length strategies (Gong et al., 2023a; Wu et al., 2023). Here we focus on characterizing the search-space trade-offs under the standard MBR setup.

As shown in Tab. 2, increasing either the length beam or the noise beam improves *BLEU*. Fig. 9 further determine which axis scales better. The results reveal that scaling the length beam (vertical axis) yields faster gains than scaling the noise beam (horizontal axis). Intuitively, a larger length beam mitigates length prediction errors and expands the set of plausible candidates, which in turn improves the effectiveness of MBR selection. Noise beam, otherwise is not as effective, since our method already ensures convergence for each length.

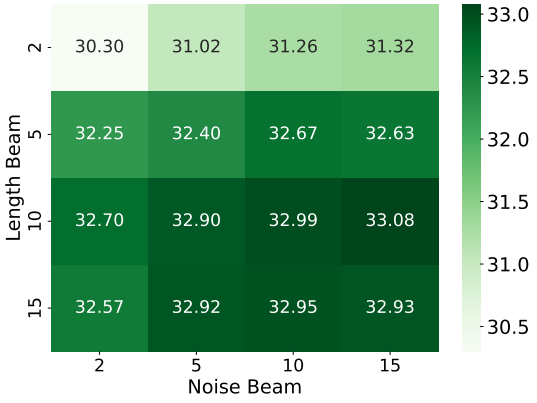

Figure 9: *SacreBLEU* score on **IWSLT14** De-En with various length beams and noise beams.

G High And Low Confidence Tokens

Fig. 10 visualizes tokens identified by MANS as high or low confidence during training. High-confidence tokens are predominantly common words, indicating that the model can already reconstruct them reliably. MANS increases their effective noise level, forcing the model to denoise these "easy" tokens even under high noise, strengthening the self-conditioning signal.

In contrast, low-confidence tokens tend to be rarer or domain-specific words that appear less often in the training data. Their noise level is left unchanged, allowing the model to learn them under a less aggressive corruption regime. This behavior aligns with a coarse-to-fine denoising process: common words are forced to generate faster, while rare words use reliable generated words to refine prediction.

H Qualitative Results

We present qualitative case studies in Tabs. 12 and 13 to illustrate how FastDiSS changes generation dynamics at the instance level. In Tab. 12, the first example shows that the baseline fails to effectively use the reused estimate, leading to persistent artifacts across steps (e.g., the repetition "Summer Summer" is not corrected in the next update). In

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

contrast, FastDiSS repairs the error in subsequent steps, consistent with SCP improving the robustness of the model to erroneous self-conditioning under few-step discretization. The second example highlights that FastDiSS produces a sharper, more accurate reconstruction target, aligning with the role of MANS in strengthening supervision on high-confidence tokens.

Tabs. 14, 15, and 16 further evaluate SCP adaptability to Plaid. Across these settings, SCP yields more coherent and consistent generations, reducing drift and improving paragraph-level continuity under limited denoising steps.

	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 12: Examples of generation throughout the denoising process, with NFE $= 2$.

	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 13: 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 + SCP	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+SCP	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 14: Examples of 32-step long sequence generation on **Prefix Completion** with Plaid-1B. The output results indicate that SCP 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+SCP	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 15: 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+SCP	... 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 16: 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.