DOS: Dependency-Oriented Sampler for Masked Diffusion Language Models

Xueyu Zhou[1]*, **Yangrong Hu**[1]*, **Jian Huang**[1,2]†
[1]Department of Data Science and Artificial Intelligence
The Hong Kong Polytechnic University, Hong Kong SAR, China
[2]Department of Applied Mathematics
The Hong Kong Polytechnic University, Hong Kong SAR, China
xueyu.zhou@connect.polyu.hk, *yangrong.hu@connect.polyu.hk*, *j.huang@polyu.edu.hk*

Abstract

Masked diffusion language models (MDLMs) have recently emerged as a new paradigm in language modeling, offering flexible generation dynamics and enabling efficient parallel decoding. However, existing decoding strategies for pre-trained MDLMs predominantly rely on token-level uncertainty criteria, while largely overlooking sequence-level information and inter-token dependencies. To address this limitation, we propose **D**ependency-**O**riented **S**ampler (DOS), a training-free decoding strategy that leverages inter-token dependencies to inform token updates during generation. Specifically, DOS exploits attention matrices from transformer blocks to approximate inter-token dependencies, emphasizing information from unmasked tokens when updating masked positions. Empirical results demonstrate that DOS consistently achieves superior performance on both code generation and mathematical reasoning tasks. Moreover, DOS can be seamlessly integrated with existing parallel sampling methods, leading to improved generation efficiency without sacrificing generation quality.

1 Introduction

Large language models (LLMs) have achieved remarkable progress in recent years, demonstrating strong performance in tasks such as code generation and mathematical reasoning (Achiam et al., 2023; Grattafiori et al., 2024; Guo et al., 2025; Yang et al., 2025a). Built upon transformer architectures (Vaswani et al., 2017), most existing LLMs are trained via next-token prediction and generate text in an autoregressive (AR) manner (Radford et al., 2018, 2019; Brown et al., 2020). Despite its strong effectiveness, the AR paradigm inherently enforces a strict left-to-right generation order, which limits parallel decoding and constrains the flexibility of generation dynamics (Xia et al., 2024; Qin et al., 2025; Li et al., 2026). These limitations have motivated the exploration of alternative generation paradigms beyond AR modeling.

Inspired by the success of diffusion models in continuous domains (Ho et al., 2020; Nichol and Dhariwal, 2021; Jing et al., 2022; Esser et al., 2024), masked diffusion language models (MDLMs) have recently emerged as a new paradigm for text generation (Austin et al., 2021a; Shi et al., 2024; Sahoo et al., 2024; Nie et al., 2025; Ye et al., 2025). MDLMs define a forward noising process that progressively masks tokens in a text sequence, and learn a reverse denoising process to reconstruct the original sequence by predicting its masked tokens. Compared to AR models, MDLMs enable flexible generation orders and allow tokens to be predicted in parallel at each denoising step. Some recent studies (Arriola et al., 2025; Yang et al., 2025c; Wang et al., 2026) further extend MDLMs to the block-wise diffusion model, where the sequence is partitioned into multiple blocks, striking a balance between auto-regressive models and diffusion language models.

To further strengthen generation in pre-trained MDLMs, various decoding strategies have been proposed to restructure the decoding order of masked tokens (Chang et al., 2022; Koh et al., 2024; Nie et al., 2025; Kim et al., 2025a) and improve the efficiency of parallel decoding (Wu et al., 2026; Kim et al., 2025a; Ben-Hamu et al., 2025). For example, Kim et al. (2025a) analyzes the influence of generation order in MDLMs and proposes a new inference strategy, leading to significant improvements in generation quality. Although these methods have achieved remarkable results, several aspects of the decoding process remain underexplored.

A key limitation is that existing decoding strategies for MDLMs are primarily based on the output logits of MDLMs and leverage uncertainty-based criteria to guide token updates (Chang et al., 2022;

*Equal contribution.
†Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 17404–17419

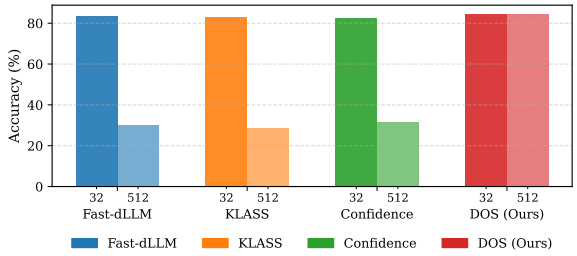

Figure 1: Accuracy on GSM8K using LLaDA-Instruct-8B (Nie et al., 2025) with a fixed generation length of 512 tokens. Block size 32 corresponds to block-wise decoding, while 512 represents the single-block setting. Existing methods (Fast-dLLM (Wu et al., 2026), KLASS(Kim et al., 2025b), Confidence(Chang et al., 2022)) degrade under large block sizes, whereas DOS (ours) remains consistent and robust across both settings.

Koh et al., 2024; Kim et al., 2025a; Nie et al., 2025). These criteria mainly focus on token-level uncertainty, while lacking an explicit mechanism to capture sequence-level information, particularly inter-token dependencies. As a result, they may amplify the discrepancy between marginal token distributions and the true joint distribution over sequences.

In addition, diffusion language models tend to overestimate the confidence of the [EOS] token in long sequences, which can lead to premature or repeated generation of [EOS] tokens and consequently degraded generation quality (Yang et al., 2025b; Nie et al., 2025). As shown in Figure 1, while recent approaches achieve strong performance under block-based decoding (Wu et al., 2026; Kim et al., 2025b), they do not fully address the issue of erroneous [EOS] generation, resulting in a significant performance drop when extended to long-sequence or single-block settings. We attribute these phenomena to a more fundamental limitation of logit-based decoding strategies. Specifically, since such methods rely on token-level confidence signals, they lack an explicit mechanism to capture global sequence-level information and inter-token dependencies, which can lead to inconsistent and suboptimal generations. These observations highlight the need for decoding strategies that are independent of block structures and can explicitly model inter-token dependencies during generation.

From this perspective, we revisit the generation process of diffusion language models from a distributional perspective and analyze the role of decoding order in recovering the target joint distribution. Based on this analysis, we propose **D**ependency-**O**riented **S**ampler (DOS), a training-free decoding strategy that leverages scores derived from the attention matrix to guide token updates. Specifically, we exploit the scaled dot-product attention weights, i.e., $softmax(QK^\top/\sqrt{d})$, as a proxy for inter-token dependency (Vaswani et al., 2017) , and use them to induce a dependency-aware decoding order. This attention-based scoring mechanism enables us to recover dependencies between unmasked tokens and masked tokens, facilitating a dependency-aligned decoding order with respect to the token generation process. Since the attention matrix can be directly obtained from the transformer block in the forward pass, our proposed method does not require any additional training or computational cost. Empirically, DOS achieves high-quality generation in a single-block setting and still outperforms existing decoding strategies, even when those methods employ multiple decoding blocks. Moreover, as a general criterion for decoding order, DOS can easily integrate other accelerated sampling methods, improving sampling efficiency while preserving generation quality.

2 Related Works

Discrete Diffusion Language Models Early research into discrete diffusion generally falls into two paradigms: transition-based frameworks and score-based methods. D3PM (Austin et al., 2021a) pioneered the adaptation of continuous diffusion concepts (Ho et al., 2020) to discrete state spaces via corruption processes defined by transition matrices. Extending this to continuous time, Campbell et al. (2022) utilized continuous-time Markov chain (CTMC) theory to formulate the forward-backward dynamics, deriving a negative ELBO objective in the continuous limit. Alternatively, inspired by the success of denoising score matching (Song et al., 2021), several works have proposed discrete counterparts to the Stein score for modeling discrete data distributions (Meng et al., 2022; Lou et al., 2024). More recently, the focus has shifted towards simplified MDLMs. Studies by Ou et al. (2025); Sahoo et al. (2024); Shi et al. (2024) demonstrate that simplified masking mechanisms can significantly enhance performance, effectively bridging the gap between diffusion-based and AR models. Transitioning from small-scale experiments to LLMs, recent works have begun to investigate the scaling laws of discrete diffusion. LLaDA (Nie et al.,

2025) scales the architecture up to 8 billion parameters, showcasing impressive reasoning capabilities that were previously unseen in smaller discrete diffusion models like GPT-2 (Radford et al., 2019). Furthermore, Dream (Ye et al., 2025) introduces a novel training paradigm by initializing diffusion models with pre-trained AR weights, thereby combining the strengths of both generative approaches.

Decoding Strategies The decoding strategy that determines the order and pace of token generation is pivotal for the efficiency and quality of diffusion language models. While standard approaches like MDLM (Sahoo et al., 2024) typically employ a stochastic unmasking schedule where masked tokens are updated randomly, this naive strategy is often suboptimal for complex reasoning tasks in large-scale models. To address this, recent research focuses on uncertainty-aware decoding. LLaDA (Nie et al., 2025) introduces a confidence-based criterion, prioritizing the unmasking of tokens with higher prediction confidence. Similarly, other works utilize entropy (Koh et al., 2024) or margin confidence (Kim et al., 2025a) as metrics to determine the decoding order. Beyond token-level ordering, LLaDA also explores a semi-autoregressive block-wise strategy, where sequences are generated in parallel blocks; this method has proven particularly effective for structured tasks like coding, significantly outperforming random ordering. Another critical line of research targets acceleration and dynamic scheduling. Instead of using a fixed top-k selection, Fast-dLLM (Wu et al., 2026) employs a flexible confidence threshold to adaptively select multiple tokens per step. To ensure generation stability, KLASS (Kim et al., 2025b) introduces a metric to measure the stabilization between consecutive predictions. Furthermore, the EB-sampler (Ben-Hamu et al., 2025) theoretically derives an upper bound for multi-token prediction errors, utilizing this bound to dynamically optimize the number of unmasked tokens at each step, thereby achieving a balance between speed and accuracy.

3 Preliminaries

In this section, we review the background and training objective of masked diffusion language models, and summarize their forward noising and reverse-time generation processes.

Let $\mathcal{V} = \{1, \dots, V\}$ denote a discrete vocabulary of size V, and let $\boldsymbol{x}_0 \in \mathcal{V}^L$ denote a sequence of length L, where $\boldsymbol{x}_0^i$ is the i-th token. To model the masking process, the state space is extended by introducing a dedicated mask symbol $\texttt{[M]} = V+1$, resulting in an augmented vocabulary of size $V+1$. Let $\delta_a \in \mathbb{R}^{V+1}$ denote the one-hot vector whose a-th coordinate equals 1.

Masked diffusion language models can be formulated as a continuous-time masking (noising) process with factorized transitions $q(\boldsymbol{x}_t|\boldsymbol{x}_s) = \prod_{i=1}^{L} q(\boldsymbol{x}_t^i|\boldsymbol{x}_s^i)$ for $0 \le s < t \le 1$, where $q(\boldsymbol{x}_t^i|\boldsymbol{x}_s^i) = \mathrm{Cat}\Big(\boldsymbol{x}_t^i;\ \frac{\alpha_t}{\alpha_s}\delta_{\boldsymbol{x}_s^i} + \frac{\alpha_s-\alpha_t}{\alpha_s}\delta_{\texttt{[M]}}\Big)$. For example, LLaDA (Nie et al., 2025) uses the linear schedule $\alpha_t = 1 - t$, so $q(\boldsymbol{x}_t^i|\boldsymbol{x}_0^i) = \mathrm{Cat}\Big(\boldsymbol{x}_t^i;\ (1-t)\delta_{\boldsymbol{x}_0^i} + t\delta_{\texttt{[M]}}\Big)$ and the process reaches the fully masked state $\texttt{[M]}^L$ at $t = 1$

Conditioned on $\boldsymbol{x}_0$, the posterior $q(\boldsymbol{x}_s \mid \boldsymbol{x}_t, \boldsymbol{x}_0)$ factorizes as $\prod_{i=1}^{L} q(\boldsymbol{x}_s^i \mid \boldsymbol{x}_t^i, \boldsymbol{x}_0^i)$, with

$$q(\boldsymbol{x}_s^i \mid \boldsymbol{x}_t^i, \boldsymbol{x}_0^i) = \begin{cases} \mathrm{Cat}(\boldsymbol{x}_s^i; \delta_{\boldsymbol{x}_t^i}), \\ \mathrm{Cat}\Big(\boldsymbol{x}_s^i;\ \frac{1-\alpha_s}{1-\alpha_t}\delta_{[\mathrm{M}]} + \frac{\alpha_s-\alpha_t}{1-\alpha_t}\delta_{\boldsymbol{x}_0^i}\Big). \end{cases} \tag{1}$$

where the first case applies when $\boldsymbol{x}_t^i \neq [\mathrm{M}]$, and the second case applies when $\boldsymbol{x}_t^i = [\mathrm{M}]$. Equation (1) (Shi et al., 2024; Sahoo et al., 2024) suggests parameterizing the reverse-time transitions by substituting a learned predicted distribution of $\boldsymbol{x}_0$ into the analytic posterior:

$$\begin{aligned} p_{\boldsymbol{\theta}}(\boldsymbol{x}_s|\boldsymbol{x}_t) &= q(\boldsymbol{x}_s|\boldsymbol{x}_t, \boldsymbol{x}_0 = \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_t)) \\ &= \prod_{i=1}^{L} q(\boldsymbol{x}_s^i|\boldsymbol{x}_t^i, \boldsymbol{x}_0^i = \boldsymbol{f}_{\boldsymbol{\theta}}^i(\boldsymbol{x}_t)), \end{aligned} \tag{2}$$

where $\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_t) \in \mathbb{R}^{L\times(V+1)}$ and $\boldsymbol{f}_{\boldsymbol{\theta}}^i(\boldsymbol{x}_t) \in \mathbb{R}^{V+1}$ is the model's output for the i-th token. $\boldsymbol{f}_{\boldsymbol{\theta}}^i(\boldsymbol{x}_t)$ satisfies $\sum_{j=1}^{V} \boldsymbol{f}_{\boldsymbol{\theta}}^{i,j}(\boldsymbol{x}_t) = 1$ and $\boldsymbol{f}_{\boldsymbol{\theta}}^{i,\texttt{[M]}}(\boldsymbol{x}_t) = 0$ for all i. The training objective is to predict the masked states in each step with the $\boldsymbol{x}_0$, which can be simplified as :

$$\mathcal{L}(\boldsymbol{\theta}) = -\mathbb{E}_{t,\boldsymbol{x}_0,\boldsymbol{x}_t}\Big[\tfrac{1}{t}\textstyle\sum_{i=1}^{L} \mathbf{1}[\boldsymbol{x}_t^i = \texttt{[M]}] \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0^i \mid \boldsymbol{x}_t)\Big]. \tag{3}$$

By minimizing the training objective, f_θ can approximate the conditional distribution of masked tokens given x_t. During inference, $\boldsymbol{x}_0^i$ is first predicted from $\boldsymbol{x}_t$ using $\boldsymbol{f}_{\boldsymbol{\theta}}^i(\boldsymbol{x}_t)$. For masked states $\texttt{[M]}$, the predicted $\boldsymbol{x}_0^i$ is decoded with probability $\frac{\alpha_s-\alpha_t}{1-\alpha_t}$, while the token remains masked with probability $\frac{1-\alpha_s}{1-\alpha_t}$. Unmasked states are kept unchanged.

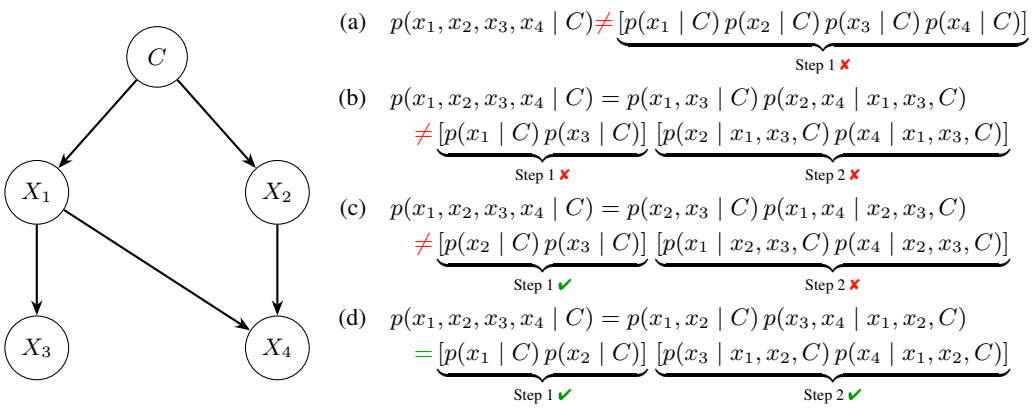

Figure 2: A toy example for parallel decoding in MDLMs, where C denotes the prompt or unmasked tokens and $\{X_i\}_{i=1}^4$ are masked tokens. Different factorizations of $p(X_1, X_2, X_3, X_4|C)$ correspond to different parallel decoding orders. Only factorizations that respect the underlying dependency structure are able to recover the true joint distribution, while improper independence assumptions lead to distribution mismatch.

4 Distributional Analysis of Parallel Decoding

4.1 Distributional Mismatch in Parallel Decoding

As discussed above, masked diffusion models are trained to match marginal distributions by conditioning on some unmasked tokens. However, the ultimate goal of the generation is to reconstruct the target joint distribution.

As illustrated by the toy example in Figure 2, different parallel decoding strategies correspond to different factorizations of the joint distribution $p(X_1, X_2, X_3, X_4 \mid C)$. Formulation (a) predicts all tokens simultaneously and formulation (b) performs parallel decoding under an incorrect block-wise factorization, which induces an invalid hierarchical dependency structure. Simply imposing a block-wise partition does not resolve this issue, as the block structure itself may still be misaligned with the true dependency structure.

Under entropy-constrained decoding (Ben-Hamu et al., 2025), masked tokens can be selected at each step by explicitly exploring conditional independence among masked positions. In formulation (c), this strategy successfully identifies a subset of masked tokens, such as X_2 and X_3, that are conditionally independent given C. This enables an exact recovery of their joint distribution $p(X_2, X_3 \mid C)$ in the first step. However, such conditionally independent subsets are not unique, and more importantly, the independence constraint is imposed only among masked tokens, while the dependency structure between the unmasked context C and masked tokens is not explicitly considered. As a result, although the joint distribution of the selected subset can be correctly recovered locally, the overall factorization remains inconsistent with the true dependency structure, ultimately preventing the recovery of the target joint distribution.

In contrast, formulation (d) induces a factorization that is consistent with the underlying structure by explicitly accounting for the dependencies between the unmasked context and masked tokens. In the first step, masked tokens X_1 and X_2, which are most strongly dependent on the available context C, are prioritized for parallel decoding. Subsequently, the remaining tokens are decoded in parallel via conditioning on the newly unmasked ones. Under such a dependency-consistent factorization, the target joint distribution can be correctly recovered through parallel decoding.

4.2 Uncertainty-Based Decoding Methods

In this subsection, we review and formalize a class of decoding strategies that rely on token-level uncertainty statistics computed from the model's output distributions. Denote the discrete distribution induced by the MDLM f_θ at time t with input x_t for position i as $p_t^i = f_\theta^i(x_t)$.

Definition 1 (Confidence)
(Chang et al., 2022; Nie et al., 2025) For MDLM f_θ at time t and position i, ***Confidence*** *selects tokens based on the maximum value of the discrete*

distribution p_t^i over the vocabulary $\mathcal{V}$, defined as

$$conf_t^i = \max_{v \in \mathcal{V}} p_t^i(v). \tag{4}$$

Definition 2 (Entropy) *(Koh et al., 2024) For MDLM f_θ at time t and position i, **Entropy** measures the uncertainty of the discrete distribution p_t^i over the vocabulary $\mathcal{V}$, defined as*

$$ent_t^i = -\sum_{v \in \mathcal{V}} p_t^i(v) \log p_t^i(v). \tag{5}$$

Definition 3 (Margin confidence) *(Kim et al., 2025a) For MDLM f_θ at time t and position i, **Margin confidence** considers the difference between the highest and the second-highest probabilities in the discrete distribution p_t^i, defined as*

$$margin_t^i = p_t^i(v^*) - \max_{v \in \mathcal{V} \setminus \{v^*\}} p_t^i(v), \tag{6}$$

where $v^* = \arg\max_{v \in \mathcal{V}} p_t^i(v)$.

Despite their differences, these decoding strategies all rely on token-level statistics extracted from the output distribution at individual positions. Therefore, they can effectively identify confident or uncertain tokens locally and can recover marginal distributions for each masked token. However, since methods do not explicitly account for dependencies across tokens, decoding strategies built upon them may fail to reconstruct the target joint distribution over the entire sequence. This observation suggests that effective parallel decoding requires a principled strategy that goes beyond token-level uncertainty and selects tokens based on inter-token dependencies.

5 Methodology

In this section, we propose **D**ependency **O**riented **S**ampler (DOS) for masked diffusion language models, which explicitly accounts for inter-token dependencies during generation. DOS focuses on the problem of which masked tokens should be decoded at each denoising step. Rather than determining the decoding order solely based on token-level uncertainty, our approach selects and updates masked tokens according to their dependency on the currently observed context. In this work, we leverage the scaled dot-product attention weights, which explicitly model how information is aggregated across tokens (Vaswani et al., 2017; Clark et al., 2019; Voita et al., 2019; Raganato and Tiedemann, 2018), as an operational proxy for inter-token dependency.

Algorithm 1 Attention-Based Dependency Scoring

Input: Current sequence state $X_{t+1} \in \mathbb{R}^{L \times V}$, MDLM f_θ, transformer block index i, masked position set $\mathcal{M}$
Output: Dependency score ***dep*** $\in \mathbb{R}^L$

```
outputs ← f_θ(X_{t+1})
Attn^(i) ← outputs.attentions[i]  ▷ Multi-head attention weights from the i-th transformer block, Attn^(i) ∈ R^{H×L×L}
Attn ← (1/H) Σ_{h=1}^{H} Attn_h^(i)  ▷ Head-averaged attention matrix Attn ∈ R^{L×L}
dep ← 0 ∈ R^L
U ← [L] \ M
for each m ∈ M do
    dep(m) ← Σ_{u∈U} Attn(m, u)  ▷ Dependency of masked position m on the unmasked context
end for
return Dependency score dep
```

Concretely, given the attention matrix extracted from a transformer block, each row is interpreted as describing how a query token integrates information from other tokens in the sequence. For example, the entry at row i and column j of the attention matrix reflects how strongly the i-th token depends on information provided by the j-th token during the forward pass. Since we focus on the information aggregated from unmasked tokens when predicting masked tokens, the dependency score for a masked token is defined as the total attention mass assigned to all unmasked tokens. Intuitively, a larger score indicates that the prediction at this masked position relies more heavily on the observed context, and should therefore be prioritized during decoding.

Formally, let $\mathcal{M}$ denote the set of masked positions and $\mathcal{U} = [L] \setminus \mathcal{M}$ the set of unmasked positions. Given the multi-head attention weights $\mathrm{softmax}(QK^\top/\sqrt{d})$ extracted from a transformer block, we average across heads to obtain a token-to-token attention matrix $\mathrm{Attn} \in \mathbb{R}^{L \times L}$. For each masked position $m \in \mathcal{M}$, the dependency score is defined as

$$dep(m) = \sum_{u \in \mathcal{U}} \mathrm{Attn}(m, u), \tag{7}$$

which measures how strongly the prediction at position m attends to the currently unmasked con-

Models	Methods	HumanEval	MBPP	GSM8K	MATH500
LLaDA-Instruct-8B	Confidence	27.44	22.80	31.69	21.40
	Entropy	27.44	19.40	33.13	18.60
	Margin	26.83	26.40	33.76	22.20
	DOS	**42.68**	**38.40**	**84.31**	**41.60**
Dream-v0-Instruct-7B	Confidence	28.66	39.60	31.08	13.60
	Entropy	25.61	33.20	30.55	10.20
	Margin	29.27	42.80	30.78	15.20
	DOS	**59.15**	**50.60**	**80.21**	**45.00**

Table 1: Performance of different decoding strategies under top-1 sampling, in which all methods generate the entire sequence within a single block. For the code benchmarks (HumanEval and MBPP), the generation length is fixed to 256 tokens, while for the math benchmarks (GSM8K and MATH500), the generation length is 512 tokens.

text. In contrast to uncertainty-based criteria that are computed independently for each position, the proposed score explicitly accounts for inter-token dependencies encoded in the attention structure. The complete attention-based dependency scoring procedure is summarized in Algorithm 1.

Based on these scores, DOS selects a subset of masked tokens with the highest scores to decode at each step. The decoding order is explicitly dependency-aware and better reflects the dependency structure encoded by the model. Since DOS provides a dependency score that ranks masked tokens, it can be directly combined with existing parallel decoding methods, such as top-K selection or entropy-constrained decoding (e.g., EB-Sampler).

6 Experiments

In this section, we present a comprehensive empirical evaluation of our proposed DOS across a diverse set of benchmarks and compare it against representative baseline decoding strategies.

6.1 Datasets

We evaluate DOS on a range of benchmarks covering different forms of structured generation, including code generation and mathematical reasoning:

- **Code generation**: **HumanEval** (Chen, 2021) and **MBPP** (Austin et al., 2021b), which consist of Python programming tasks with function-level specifications.
- **Mathematical reasoning**: **GSM8K** (Cobbe et al., 2021), which contains arithmetic word problems of grade school that require multi-step numerical reasoning and **MATH500** (Lightman et al., 2023), which is a subset of the MATH (Hendrycks et al., 2021) benchmark and features more challenging problems.

6.2 Baselines

We compare DOS against the following baselines:

- **Uncertainty-based methods**: selecting tokens at each step based on uncertainty criteria, including confidence (Chang et al., 2022; Nie et al., 2025), entropy (Koh et al., 2024), and margin confidence (Kim et al., 2025a).
- **Entropy-Based (EB) Sampler** (Ben-Hamu et al., 2025): selecting multiple tokens at each step under an entropy-constrained sampling scheme. EB-sampler can be easily integrated with other methods, such as confidence, entropy, margin confidence and our proposed method, by ranking tokens according to these criteria.
- **Fast-dLLM** (Wu et al., 2026): selecting multiple tokens with confidence greater than a threshold at each step.
- **KLASS** (Kim et al., 2025b): selecting multiple tokens with thresholds on both confidence and token-level KL divergence, where the KL divergence measures the difference between the previous distribution and the current distribution of the given token.

We apply these methods to two open source MDLMs, including LLaDA-Instruct-8B (Nie et al., 2025) and Dream-v0-Instruct-7B (Ye et al., 2025). Both models are trained with the masked diffusion objective, without utilizing block-wise masking

Models	Methods	HumanEval		MBPP		GSM8K		MATH500	
		Acc ↑	NFE ↓	Acc ↑	NFE ↓	Acc ↑	NFE ↓	Acc ↑	NFE ↓
LLaDA-Instruct-8B	Fast-dLLM	28.66	83.26	23.00	78.93	33.06	195.46	20.80	219.54
	KLASS	29.88	107.99	24.00	97.09	28.43	265.52	19.60	300.48
	Confidence+EB	28.66	160.10	22.80	167.43	32.15	299.59	21.00	327.07
	Entropy+EB	29.88	175.71	18.80	183.72	25.09	309.12	19.00	334.24
	Margin+EB	26.22	159.58	26.60	166.00	38.51	293.45	21.40	324.10
	DOS+EB	**45.12**	115.84	**38.80**	76.098	**84.61**	163.48	**41.00**	244.24
Dream-v0-Instruct-7B	Fast-dLLM	28.66	141.62	39.60	127.69	31.08	205.52	13.60	273.58
	KLASS	27.44	150.46	38.00	145.09	31.84	239.92	14.20	302.69
	Confidence+EB	28.66	199.66	39.60	191.74	31.01	302.89	13.60	384.77
	Entropy+EB	25.61	222.15	32.60	214.86	30.63	356.59	10.00	408.54
	Margin+EB	29.27	183.25	41.20	177.73	30.86	297.96	15.40	377.15
	DOS+EB	**58.54**	72.47	**51.00**	68.33	**79.98**	182.43	**45.40**	229.36

Table 2: Performance of different decoding strategies within a single block. **EB** stands for EB-sampler (Ben-Hamu et al., 2025), a parallel decoding method that can be combined with different scores. **Acc** denotes task accuracy, and **NFE** denotes the number of model forward evaluations, reflecting decoding efficiency. For HumanEval and MBPP, the generation length is fixed to 256 tokens, while for GSM8K and MATH500, the generation length is 512 tokens.

structures. Further implementation details are provided in Appendix A.2 and hyperparameters are provided in Appendix A.3.

6.3 Main Result

DOS significantly improves performance under single-block decoding. We first evaluate DOS under a single-block decoding setting, where the entire sequence is generated within one diffusion block. As shown in Table 1, DOS consistently outperforms uncertainty-based decoding strategies across all benchmarks and both MDLMs. In contrast, confidence, entropy, and margin struggle to generate long sequences within a single block and exhibit degraded performance, particularly on mathematical reasoning tasks that require strict logical progression across extended generations. By explicitly leveraging inter-token dependencies, DOS can capture long-range dependencies that govern reasoning structure and achieve substantial performance gains, improving accuracy by large margins on GSM8K and MATH500. These results demonstrate that dependency-aware token selection is crucial for generating sequences consistent with the target joint distribution under single-block decoding.

DOS integrates seamlessly with parallel decoding and improves efficiency without sacrificing accuracy. We next evaluate whether DOS can be combined with parallel decoding strategies to improve efficiency. Specifically, we integrate DOS with the EB sampler, which coordinates parallel updates among masked tokens under entropy constraints. As shown in Table 2, DOS+EB consistently improves both accuracy and decoding efficiency for LLaDA and Dream models. Compared with the results of EB sampler integrated with uncertainty-based methods, DOS not only improves the accuracy across tasks but also achieves at least a 1.3× speedup on the LLaDA model. Similar trends are observed for the Dream model. In addition, compared to alternative efficient decoding baselines such as Fast-dLLM and KLASS, DOS+EB achieves substantially higher accuracy while maintaining competitive decoding efficiency. These results indicate that DOS provides a complementary, dependency-aware signal that enhances existing parallel decoding methods.

Method	HumanEval	MBPP	GSM8K	MATH500
Confidence	40.85	38.20	82.56	38.80
Entropy	41.46	37.80	82.79	40.00
Margin	40.85	37.40	82.64	38.80
Fast-dLLM	41.46	38.80	83.40	39.60
KLASS	42.07	38.60	83.02	40.20
DOS (w/o block)	42.68 (+0.61)	38.40 (-0.40)	84.31 (+0.91)	41.60 (+1.40)
DOS (w/ block)	**44.51** (+2.44)	**38.80** (+0.00)	**84.61** (+1.21)	**42.40** (+2.20)

Table 3: Performance comparison on HumanEval, MBPP, GSM8K, and MATH500 using LLaDA-Instruct-8B. All baseline methods (Confidence, Entropy, Margin, Fast-dLLM, and KLASS) adopt the block diffusion with a fixed block size of 32. DOS (ours) is evaluated both without block partitioning (single-block) and with block partitioning (block size = 32).

DOS consistently outperforms block-based decoding strategies with and without block partitioning. We further evaluate DOS against decoding strategies with block structures on the LLaDA model. All baseline methods employ block diffu-

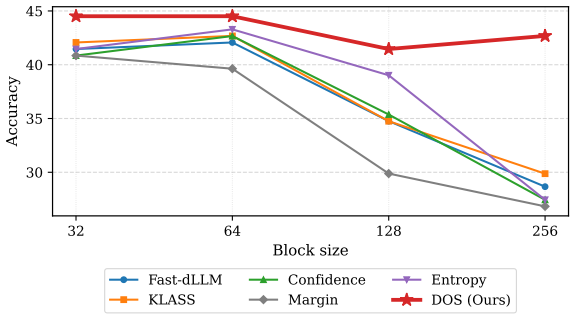

Figure 3: Accuracy on HumanEval using LLaDA-Instruct-8B with a fixed generation length of 256 under varying block sizes. Existing decoding strategies are sensitive to block size and degrade as the block size increases, whereas DOS (ours) demonstrates strong robustness to block size variation and maintains superior consistency across all settings.

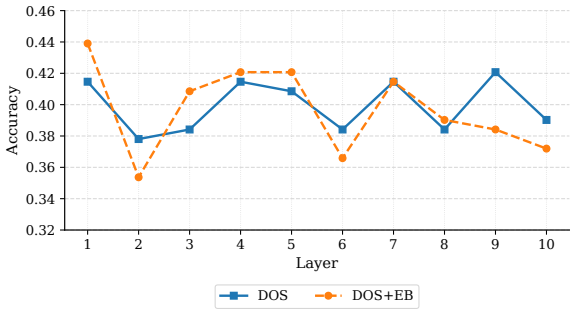

Figure 4: Accuracy of DOS and DOS+EB on the HumanEval benchmark using the LLaDA-Instruct-8B model, where the x-axis indicates the transformer layer from which the attention matrix is extracted.

sion with a fixed block size of 32, whereas DOS is evaluated both with and without block partitioning. As shown in Table 3, DOS consistently achieves superior performance across all benchmarks, even when block partitioning is removed. Although block partitioning improves the performance of existing uncertainty-based decoding strategies, DOS consistently outperforms these methods under both block-based and single-block settings. Moreover, DOS maintains strong performance without relying on block structures and further benefits when block diffusion is introduced. These results demonstrate that explicitly modeling inter-token dependencies is more critical than the choice of block structure for achieving high-quality parallel decoding. We further conduct additional comparisons under block-based decoding with different configurations and details are presented in Appendix B.

DOS is robust to block size and consistently outperforms uncertainty-based decoding. We further investigate the sensitivity of different decoding strategies to the choice of block size and the accuracy of various methods under different block sizes is reported in Figure 3. Existing methods are sensitive to the block configuration and exhibit noticeable performance fluctuations as the block size increases. In contrast, DOS maintains consistently strong performance across a wide range of block sizes and consistently outperforms competing methods. These results indicate that DOS provides a more stable and robust decoding signal by explicitly modeling inter-token dependencies, making it less sensitive to the block configuration. Additional results on the Dream model are provided in Appendix B.2

Effect of Transformer Layer Choice. We further explore the role of layer depth in attention-based dependency scoring within the single-block setting. As illustrated in Figure 4, the performance of DOS and DOS+EB on HumanEval remains robust across the first 10 layers. This phenomenon suggests that the shallow layers can capture the syntactic structures required for effective masked-token selection, which is consistent with prior studies (Tenney et al., 2019; Clark et al., 2019; Jawahar et al., 2019). Additional ablation studies on attention layer selection and head-wise dependency analysis are provided in Appendix B.3 and Appendix B.4.

7 Conclusion

In this work, we study the decoding process of masked diffusion language models from a dependency-aware perspective and show that existing decoding strategies are limited by their reliance on token-level uncertainty and their sensitivity to heuristic block structures, both of which stem from a lack of explicit modeling of inter-token dependencies. To address these limitations, we propose the Dependency-Oriented Sampler (DOS), a training-free decoding strategy that leverages attention-derived dependency signals to guide the decoding order of masked tokens.

By using the attention matrix as a proxy for inter-token dependencies, DOS updates tokens that are better supported by the current unmasked context, leading to a decoding process that is more aligned with the underlying joint distribution. Empirical results on code generation and mathematical reasoning benchmarks demonstrate that DOS consistently

outperforms existing uncertainty-based decoding strategies. Notably, DOS can be seamlessly integrated with parallel sampling methods, improving generation efficiency without sacrificing quality.

Our analysis highlights the importance of respecting inter-token dependencies during sampling and suggests that the choice of decoding order plays a critical role in recovering the target joint distribution in masked diffusion language models. We hope this work encourages further investigation into dependency-aware decoding strategies and contributes to a deeper understanding of parallel generation dynamics in diffusion language models.

Limitations

This work has several limitations that suggest directions for future research.

First, DOS relies on attention weights extracted from transformer blocks as a proxy for inter-token dependencies. While attention provides an effective and readily available signal for guiding decoding order, it does not necessarily correspond to explicit causal or structural dependencies among tokens. Some studies have explored incorporating explicit dependency structures, such as directed acyclic graphs (DAGs), into language model training to better capture compositional and dependency-aware generation dynamics (Huang et al., 2022, 2023). Integrating such structured dependency representations into the decoding process, or combining them with attention-derived signals, may provide a more principled alternative and is an interesting direction for future work.

Second, our current implementation extracts dependency scores from a single transformer layer with head-averaged attention. Although we observe strong performance across different settings, this design does not fully exploit the hierarchical nature of representations in deep transformers. Extending DOS to integrate information across multiple layers, or to selectively aggregate signals from different attention heads, could enable richer dependency modeling and further improve decoding performance.

Finally, our evaluation focuses on structured generation tasks, including code generation and mathematical reasoning, where long-range dependencies play a critical role. The effectiveness of DOS in other generation settings, such as open-ended dialogue or multilingual generation, remains to be systematically investigated. Exploring how dependency-oriented decoding interacts with different task characteristics is an important direction for future work.

Ethics Statement

This work focuses on inference-time decoding strategies for pre-trained masked diffusion language models. It does not involve human subjects, data collection, or additional model training. All experiments are conducted on publicly available benchmarks. The proposed method does not introduce new ethical risks beyond those inherent to the underlying language models.

Acknowledgements

This research was conducted using the computing resources provided by the Research Centre for the Mathematical Foundations of Generative AI (PO046811) at The Hong Kong Polytechnic University.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Marianne Arriola, Aaron Gokaslan, Justin T Chiu, Jiaqi Han, Zhihan Yang, Zhixuan Qi, Subham Sekhar Sahoo, and Volodymyr Kuleshov. 2025. Block diffusion: Interpolating between autoregressive and diffusion language models. In *The Thirteenth International Conference on Learning Representations*.

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. 2021a. Structured denoising diffusion models in discrete state-spaces. *Advances in neural information processing systems*, 34:17981–17993.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and 1 others. 2021b. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas Nolte, and Brian Karrer. 2025. Accelerated sampling from masked diffusion models via entropy bounded unmasking. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are

few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Andrew Campbell, Joe Benton, Valentin De Bortoli, Thomas Rainforth, George Deligiannidis, and Arnaud Doucet. 2022. A continuous time framework for discrete denoising models. *Advances in Neural Information Processing Systems*, 35:28266–28279.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. 2022. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11315–11325.

Mark Chen. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does bert look at? an analysis of bert's attention. In *Proceedings of the 2019 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, and 1 others. 2024. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

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

Fei Huang, Pei Ke, and Minlie Huang. 2023. Directed acyclic transformer pre-training for high-quality non-autoregressive text generation. *Transactions of the Association for Computational Linguistics*, 11:941–959.

Fei Huang, Hao Zhou, Yang Liu, Hang Li, and Minlie Huang. 2022. Directed acyclic transformer for non-autoregressive machine translation. In *Proceedings of the 39th International Conference on Machine Learning, ICML 2022*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*.

Bowen Jing, Gabriele Corso, Jeffrey Chang, Regina Barzilay, and Tommi Jaakkola. 2022. Torsional diffusion for molecular conformer generation. *Advances in neural information processing systems*, 35:24240–24253.

Jaeyeon Kim, Kulin Shah, Vasilis Kontonis, Sham M. Kakade, and Sitan Chen. 2025a. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. In *Forty-second International Conference on Machine Learning*.

Seo Hyun Kim, Sunwoo Hong, Hojung Jung, Youngrok Park, and Se-Young Yun. 2025b. KLASS: KL-guided fast inference in masked diffusion models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Hyukhun Koh, Minha Jhang, Dohyung Kim, Sangmook Lee, and Kyomin Jung. 2024. Plm-based discrete diffusion language models with entropy-adaptive gibbs sampling. *arXiv e-prints*, pages arXiv–2411.

Jia-Nan Li, Jian Guan, Wei Wu, and Chongxuan Li. 2026. Refusion: A diffusion large language model with parallel autoregressive decoding. In *The Fourteenth International Conference on Learning Representations*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Aaron Lou, Chenlin Meng, and Stefano Ermon. 2024. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Forty-first International Conference on Machine Learning*.

Chenlin Meng, Kristy Choi, Jiaming Song, and Stefano Ermon. 2022. Concrete score matching: Generalized score matching for discrete data. *Advances in Neural Information Processing Systems*, 35:34532–34545.

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

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. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. 2025. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. In *The Thirteenth International Conference on Learning Representations*.

Tian Qin, David Alvarez-Melis, Samy Jelassi, and Eran Malach. 2025. To backtrack or not to backtrack: When sequential search limits model reasoning. In *Second Conference on Language Modeling*.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, and 1 others. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Alessandro Raganato and Jörg Tiedemann. 2018. An analysis of encoder representations in transformer-based machine translation. In *Proceedings of the 2018 EMNLP workshop BlackboxNLP: analyzing and interpreting neural networks for NLP*, pages 287–297.

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

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. 2024. Simplified and generalized masked diffusion for discrete data. *Advances in neural information processing systems*, 37:103131–103167.

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

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601.

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

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808. ACL Anthology.

Xu Wang, Chenkai Xu, Yijie Jin, Jiachun Jin, Hao Zhang, Kai Yu, and Zhijie Deng. 2026. Diffusion LLMs can do faster-than-AR inference via discrete diffusion forcing. In *The Fourteenth International Conference on Learning Representations*.

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. 2026. Fast-dLLM: Training-free acceleration of diffusion LLM by enabling KV cache and parallel decoding. In *The Fourteenth International Conference on Learning Representations*.

Heming Xia, Zhe Yang, Qingxiu Dong, Peiyi Wang, Yongqi Li, Tao Ge, Tianyu Liu, Wenjie Li, and Zhifang Sui. 2024. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 7655–7671, Bangkok, Thailand. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Jingyi Yang, Guanxu Chen, Xuhao Hu, and Jing Shao. 2025b. Taming masked diffusion language models via consistency trajectory reinforcement learning with fewer decoding step. *arXiv preprint arXiv:2509.23924*.

Yicun Yang, Cong Wang, Shaobo Wang, Zichen Wen, Biqing Qi, Hanlin Xu, and Linfeng Zhang. 2025c. Diffusion llm with native variable generation lengths: Let [eos] lead the way. *arXiv preprint arXiv:2510.24605*.

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

A Experiment Details

A.1 Experimental Setup

We conduct all experiments with LLaDA-Instruct-8B (Nie et al., 2025) and Dream-v0-Instruct-7B (Ye et al., 2025). We run inference with batch size of 1 on a single **NVIDIA L40** GPU with 48 GB memory. For deterministic decoding, we set temperature to 0 and random seed to **42** for all runs.

A.2 Baseline Methods

This subsection provides implementation details of the key baseline methods used in our experiments, including uncertainty-based methods and accelerated samplers. Denote the discrete distribution induced by the MDLM f_θ at time t with input x_t for position i as $p_t^i = f_\theta^i(x_t)$.

Confidence (Chang et al., 2022; Nie et al., 2025) For MDLM f_θ at time t and position i, **Confidence** selects tokens based on the maximum value of the discrete distribution p_t^i over the vocabulary $\mathcal{V}$, defined as

$$\mathit{conf}_t^i = \max_{v \in \mathcal{V}} p_t^i(v). \tag{8}$$

Entropy (Koh et al., 2024) For MDLM f_θ at time t and position i, **Entropy** measures the uncertainty of the discrete distribution p_t^i over the vocabulary $\mathcal{V}$, defined as

$$\mathit{ent}_t^i = -\sum_{v \in \mathcal{V}} p_t^i(v) \log p_t^i(v). \tag{9}$$

Margin confidence (Kim et al., 2025a) For MDLM f_θ at time t and position i, **Margin confidence** considers the difference between the highest and the second-highest probabilities in the discrete distribution p_t^i,, defined as

$$\mathit{margin}_t^i = p_t^i(v^*) - \max_{v \in \mathcal{V} \setminus \{v^*\}} p_t^i(v), \tag{10}$$

where $v^* = \arg\max_{v \in \mathcal{V}} p_t^i(v)$.

Fast-dLLM Fast-dLLM performs confidence-based parallel decoding by unmasking all positions whose confidence exceeds a confidence threshold ϵ. Concretely, at each step t, let $\mathcal{M}_t$ denote the set of masked positions. It computes the confidence score conf_t^i and masks the subset of positions

$$\mathcal{S}_t = \left\{ i \in \mathcal{M}_t \,\middle|\, \mathrm{conf}_t^i = \max_{v \in \mathcal{V}} p_t^i(v) > \epsilon \right\}. \tag{11}$$

Compared to fixed-budget top-K unmasking, this approach yields a dynamic number of updates per iteration and can be integrated with semi-autoregressive block decoding for enhanced efficiency. Furthermore, while Fast-dLLM supports block-wise KV cache to eliminate redundant computations of key and value vectors across previous blocks, our evaluation focuses on decoding strategy rather than hardware-specific wall-clock speed; consequently, we do not utilize block-wise KV cache in our experiments.

KLASS. KLASS performs stable-token selection by identifying positions where the predictive distribution is stable across steps and reaches high confidence. At each step t, let $\mathcal{M}_t$ denote the set of masked positions. KLASS selects a subset of stable tokens $\mathcal{S}_t \subseteq \mathcal{M}_t$ to be unmasked based on a history length n, a KL-divergence threshold ϵ_{KL}, and a confidence threshold τ:

$$\mathcal{S}_t = \Big\{ i \in \mathcal{M}_t \,\Big|\, \forall k \in \{1, \dots, n\}, \; D_{\mathrm{KL}}(p_{t-k}^i \,\|\, p_{t-k+1}^i) < \epsilon_{\mathrm{KL}} \wedge \; \mathrm{conf}_t^i > \tau \Big\}, \tag{12}$$

where D_{KL} denotes the Kullback-Leibler divergence between the predictive distributions of consecutive steps at position i, and $\mathrm{conf}_t^i = \max_{v \in \mathcal{V}} p_t^i(v)$ is the confidence score. By requiring predictions to remain consistent over n steps, KLASS ensures that only stable tokens are updated in parallel, thereby improving generation reliability.

EB-Sampler. The entropy-bounded sampler accelerates generation by unmasking a number of tokens per step that satisfy the entropy constraint. Concretely, at each step t, it first ranks all masked positions in $\mathcal{M}_t$ based on a specific score like confidence *conf* or our proposed dependency *dep*. Then, starting from the highest-ranked token, it selects the largest subset $\mathcal{S}_t \subseteq \mathcal{M}_t$ that satisfies the cumulative entropy constraint:

$$\sum_{i \in \mathcal{S}_t} \mathrm{ent}_t^i - \max_{j \in \mathcal{S}_t} \mathrm{ent}_t^j \le \gamma, \tag{13}$$

where $\mathrm{ent}_t^i = -\sum_{v \in \mathcal{V}} p_t^i(v) \log p_t^i(v)$ denotes the entropy of the predictive distribution at position i.

A.3 Hyperparameter Choices

We follow prior work whenever the recommended settings are available, and otherwise tune on a small held-out subset. The main hyperparameters for accelerated baselines are:

- **EB-Sampler**: We set the entropy tolerance to $\gamma = 0.01$ for all datasets.
- **Fast-dLLM**: We set the confidence threshold to $\epsilon = 0.95$ for all datasets.
- **KLASS**: We follow the threshold configurations in prior work (Kim et al., 2025b). KLASS uses two thresholds: a confidence threshold (Conf) and a KL-based threshold (KL). The history length n is set to 2 for all datasets. We report the exact values used for each dataset and model in Table 4.
- **DOS**: We report the optimal layer for DOS and DOS+EB in Table 5.

  Entries indicate the transformer layer indices from which the attention matrices are extracted to achieve the best accuracy under each setting.

Task	LLaDA		Dream	
	Conf	KL	Conf	KL
MATH	0.6	0.010	0.9	0.005
GSM8K	0.6	0.015	0.9	0.001
HumanEval	0.9	0.010	0.8	0.001
MBPP	0.7	0.010	0.9	0.001

Table 4: Threshold configurations for KLASS.

	HumanEval	MBPP	GSM8K	MATH500
LLaDA-Instruct-8B				
DOS (w/o block)	13	9	7	7
DOS (w/ block)	13	9	16	30
DOS+EB (w/o block)	29	5	4	3
DOS+EB (w/ block)	13	1	1	7
Dream-vo-Instruct-7B				
DOS (w/o block)	1	22	6	7
DOS(w/ block)	6	6	6	7
DOS+EB (w/o block)	6	22	3	3
DOS+EB (w/ block)	6	26	3	1

Table 5: Transformer layers that yield the best performance for DOS under different settings. For **DOS**, *w/o block* corresponds to single-block decoding where the block length equals the generation length, while *w/ block* applies block partitioning with a fixed block size of 32.

B Additional Experiment Results

B.1 Experiments on LLaDA

Table 6 presents the comparative results on HumanEval, MBPP, GSM8K, and MATH500 using the LLaDA-Instruct-8B model with a fixed block size of 32. DOS consistently demonstrates superior generation quality across both Top-1 and parallel sampling settings, proving effective regardless of whether block partitioning is applied. Specifically, in the Top-1 regime, DOS achieves the highest accuracy on all four benchmarks, significantly outperforming standard scoring methods such as Confidence and Entropy. Furthermore, when integrated with the EB-sampler for parallel decoding, DOS maintains this performance advantage, validating the robustness of our scoring strategy in guiding the model toward high-quality outputs.

In terms of computational efficiency, Fast-dLLM exhibits a substantial reduction in the number of function evaluations (NFE), achieving the lowest inference cost across tasks. This efficiency is largely attributed to its threshold-based mechanism. However, this speed comes with a slight trade-off in accuracy compared to our method. Consequently, a promising avenue for future work is to synergize the precise scoring capability of DOS with dynamic threshold-based strategies similar to Fast-dLLM. Such an integration could potentially yield a more optimal generation process, maintaining the high accuracy of DOS while significantly accelerating inference.

B.2 Experiments on Dream

Table 7 presents the performance of the Dream-v0-Instruct-7B model across four datasets using both Top-1 and parallel sampling strategies with a fixed block size of 32. Consistent with our previous observations, the proposed DOS method demonstrates superior performance on the majority of tasks with and without block partitioning, particularly excelling in code generation and complex mathematical reasoning. In the Top-1 setting, DOS (w/ block) achieves the highest accuracy on HumanEval and MATH500, significantly outperforming standard baselines like Confidence and Entropy. This advantage extends to the parallel sampling setting, where DOS combined with the EB-sampler attains 45.40% accuracy on MATH500, surpassing robust competitors such as KLASS and Fast-dLLM.

While DOS remains highly competitive overall, we observe a slight performance gap on the GSM8K benchmark compared to KLASS and uncertainty-based methods, suggesting that our scoring mechanism could be further refined for specific arithmetic reasoning patterns. Furthermore, we note that Fast-dLLM achieves the low-

Methods	HumanEval		MBPP		GSM8K		MATH500	
	Acc ↑	NFE ↓	Acc ↑	NFE ↓	Acc ↑	NFE ↓	Acc ↑	NFE ↓
Top-1								
Confidence	40.85	256.00	38.20	256.00	82.56	512.00	38.80	512.00
Entropy	41.46	256.00	37.80	256.00	82.79	512.00	40.00	512.00
Margin	40.85	256.00	37.40	256.00	82.64	512.00	38.80	512.00
DOS (w/o block)	42.68	256.00	38.40	256.00	84.31	512.00	41.60	512.00
DOS (w/ block)	44.51	256.00	38.80	256.00	84.61	512.00	42.40	512.00
Parallel								
Fast-dLLM	41.46	77.54	38.80	54.12	83.40	120.56	39.60	176.07
KLASS	42.07	103.06	38.60	74.18	83.02	178.26	40.20	252.47
Confidence+EB	41.46	104.28	38.60	73.39	82.71	165.33	39.20	249.76
Entropy+EB	42.68	107.88	37.00	77.88	82.64	176.26	38.60	265.76
Margin+EB	40.24	102.44	38.80	71.31	82.49	161.24	37.80	243.58
DOS+EB (w/o block)	45.12	115.84	38.80	76.09	84.61	163.48	41.00	244.24
DOS+EB (w/ block)	43.90	103.05	39.00	80.90	84.00	170.06	41.40	257.55

Table 6: Additional results through top-1 sampling and parallel sampling on HumanEval, MBPP, GSM8K, and MATH500 using **LLaDA-Instruct-8B** model (Nie et al., 2025) with a fixed block size of 32. **EB** denotes the EB-sampler (Ben-Hamu et al., 2025), a parallel decoding method that can be combined with different scoring strategies. **Acc** denotes task accuracy, and **NFE** denotes the number of model forward evaluations. For HumanEval and MBPP, the generation length is fixed to 256 tokens, while for GSM8K and MATH500, the generation length is 512 tokens. For **DOS**, *w/o block* corresponds to single-block decoding, where the block length equals the generation length, while *w/ block* applies block partitioning with block size 32.

Methods	HumanEval		MBPP		GSM8K		MATH500	
	Acc ↑	NFE ↓	Acc ↑	NFE ↓	Acc ↑	NFE ↓	Acc ↑	NFE ↓
Top-1								
Confidence	58.54	256.00	50.02	256.00	81.50	512.00	40.08	512.00
Entropy	57.32	256.00	50.80	256.00	83.93	512.00	42.20	512.00
Margin	56.71	256.00	50.40	256.00	81.05	512.00	41.80	512.00
DOS (w/o block)	59.15	256.00	50.60	256.00	80.21	512.00	45.00	512.00
DOS (w/ block)	61.59	256.00	50.20	256.00	80.29	512.00	45.00	512.00
Parallel								
Fast-dLLM	58.54	62.68	50.20	50.04	81.50	134.17	39.80	154.46
KLASS	59.15	78.11	52.20	72.69	84.08	200.85	43.40	277.11
Confidence+EB	57.93	78.86	50.20	71.90	81.96	182.38	39.80	227.54
Entropy+EB	56.10	88.29	50.20	73.00	83.17	192.83	44.20	239.13
Margin+EB	56.10	80.34	50.20	66.90	81.20	179.19	39.40	226.74
DOS+EB (w/o block)	58.54	72.47	51.00	68.33	79.98	182.43	45.40	229.36
DOS+EB (w/ block)	60.37	77.07	50.60	71.88	79.83	190.20	45.00	236.81

Table 7: Additional results through top-1 sampling and parallel sampling on HumanEval, MBPP, GSM8K, and MATH500 using **Dream-v0-Instruct-7B** model (Ye et al., 2025) with a fixed block size of 32. **EB** denotes the EB-sampler (Ben-Hamu et al., 2025), a parallel decoding method that can be combined with different scoring strategies. **Acc** denotes task accuracy, and **NFE** denotes the number of model forward evaluations. For HumanEval and MBPP, the generation length is fixed to 256 tokens, while for GSM8K and MATH500, the generation length is 512 tokens. For **DOS**, *w/o block* corresponds to single-block decoding, where the block length equals the generation length, while *w/ block* applies block partitioning with block size 32.

Methods	HumanEval		MBPP		GSM8K		MATH500	
	Acc ↑	NFE ↓	Acc ↑	NFE ↓	Acc ↑	NFE ↓	Acc ↑	NFE ↓
LLaDA-Instruct-8B								
DOS (w/o block)	41.46	256.00	38.20	256.00	84.23	512.00	41.00	512.00
DOS (w/ block)	42.07	256.00	37.80	256.00	84.23	512.00	40.40	512.00
DOS+EB (w/o block)	43.90	104.93	38.60	76.34	83.55	164.26	39.60	258.37
DOS+EB (w/ block)	40.85	107.11	39.00	80.90	84.00	170.06	40.60	260.90
Dream-vo-Instruct-7B								
DOS (w/o block)	59.15	256.00	49.80	256.00	79.53	512.00	44.20	512.00
DOS (w/ block)	59.15	256.00	49.80	256.00	79.53	512.00	44.20	512.00
DOS+EB (w/o block)	57.32	75.49	50.00	69.05	79.83	181.63	44.80	230.09
DOS+EB (w/ block)	57.32	79.44	50.00	73.34	79.83	190.81	45.00	236.81

Table 8: Results of DOS and DOS+EB with attention matrices extracted from the first transformer layer. Experiments are conducted on LLaDA-Instruct-8B and Dream-v0-Instruct-7B under both single-block and block-based decoding settings. For HumanEval and MBPP, the generation length is fixed to 256 tokens, while for GSM8K and MATH500, the generation length is 512 tokens. **DOS** *w/o block* denotes single-block decoding where the block length equals the generation length, while **DOS** *w/ block* applies block partitioning with a fixed block size of 32. **Acc** denotes task accuracy, and **NFE** denotes the number of model forward evaluations, reflecting decoding efficiency.

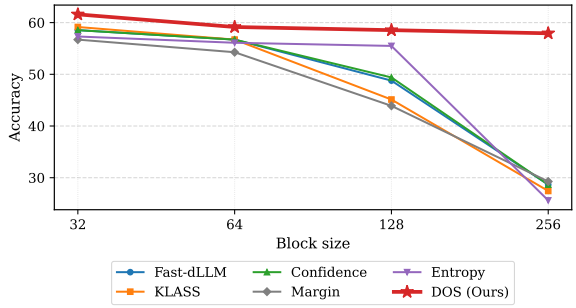

Figure 5: Accuracy on HumanEval using Dream-v0-Instruct-7B with a fixed generation length of 256 under varying block sizes. Existing decoding strategies are sensitive to block size and degrade as the block size increases, whereas DOS (ours) demonstrates strong robustness to block size variation and maintains superior consistency across all settings.

est NFE, demonstrating impressive inference efficiency, while KLASS shows strong adaptability on MBPP; these baselines offer valuable insights into efficiency optimization and task-specific selection, pointing toward promising directions for further enhancing the DOS framework in future work.

Figure 5 shows the performance of different decoding strategies under varying block sizes on the Dream model. Consistent with the main results, existing uncertainty-based methods exhibit noticeable performance degradation as the block size increases. In contrast, DOS maintains stable performance across a wide range of block sizes and consistently outperforms competing methods. These results further confirm the robustness of DOS to block size choices.

B.3 Performance with a Fixed Attention Layer

To examine whether DOS critically relies on careful attention-layer selection, we conduct an additional experiment in which the attention matrices are *fixed to the first transformer layer* across all settings. The results are summarized in Table 8.

Across both LLaDA-Instruct-8B and Dream-v0-Instruct-7B, DOS with the first-layer attention consistently achieves performance comparable to the corresponding baselines under all different settings. This observation holds for both top-1 sampling and parallel decoding with EB-sampler. With a fixed early-layer attention signal, the proposed dependency-oriented scoring remains sufficient to guide decoding, demonstrating the robustness and practical applicability of DOS.

B.4 Performance of DOS Using Single Attention Head

To examine whether DOS critically relies on specific attention heads, we conduct an additional experiment in which each attention head is used *individually* as the dependency signal. The detailed results are presented in Table 9 and Table 10.

Across both *w/o block* and *w/ block* settings, the performance varies significantly when using different heads. Moreover, using a single attention head consistently leads to significantly worse

Head	w/o block	w/ block	Head	w/o block	w/ block
1	28.66	36.58	17	28.05	31.71
2	28.66	27.44	18	22.56	28.66
3	19.51	24.39	19	17.07	30.49
4	17.68	25.61	20	29.88	32.32
5	18.29	35.98	21	14.63	25.00
6	21.34	23.17	22	28.05	31.10
7	10.96	26.83	23	17.07	25.00
8	23.17	25.61	24	18.90	28.05
9	25.61	29.88	25	25.00	25.61
10	28.66	29.88	26	19.51	31.71
11	26.83	29.27	27	15.24	28.66
12	31.10	34.76	28	23.17	26.83
13	33.54	37.80	29	21.34	26.22
14	16.46	21.95	30	10.98	20.12
15	33.54	34.15	31	18.90	27.44
16	21.95	26.83	32	12.80	23.78

Table 9: Head-wise results on HumanEval using the **LLaDA-Instruct-8B** model (Nie et al., 2025) with a fixed generation length of 256 tokens. We report the accuracy of DOS when using individual attention heads as the dependency signal, under both *w/o block* (single-block) and *w/ block* (block size = 32) settings. Each row corresponds to a single attention head. The results show significant performance variance across different heads, highlighting the importance of aggregation for stable dependency estimation.

Setting	Mean (Std)	Best Head	Worst Head	DOS
w/o block	22.95 (6.93)	33.54	10.96	**42.68**
w/ block	28.93 (5.08)	37.80	20.12	**44.51**

Table 10: Summary of head-wise ablation results on HumanEval using the **LLaDA-Instruct-8B** model (Nie et al., 2025). Using a single head consistently underperforms DOS with head aggregation, demonstrating the effectiveness of aggregating information across multiple attention heads.

performance compared to DOS, which aggregates information from all heads. These results demonstrate that reliable dependency estimation cannot be obtained from individual heads alone, and instead requires aggregating complementary signals across multiple heads, validating the design choice of head averaging in DOS.