STDec: Spatio-Temporal Stability Guided Decoding for dLLMs

Yuzhe Chen[1] **Jiale Cao**[1*] **Xuyang Liu**[2] **Jin Xie**[3] **Aiping Yang**[1] **Yanwei Pang**[1]

[1] Tianjin University [2] The Hong Kong Polytechnic University [3] Chongqing University
{yzchen,connor,yangaiping,pyw}@tju.edu.cn
xu-yang.liu@connect.polyu.hk xiejin@cqu.edu.cn

Abstract

Diffusion Large Language Models (dLLMs) have achieved rapid progress, viewed as a promising alternative to the autoregressive paradigm. However, most dLLM decoders still adopt a global confidence threshold, and do not explicitly model local context from neighboring decoded states or temporal consistency of predicted token IDs across steps. To address this issue, we propose a spatio-temporal stability guided decoding approach, named STDec. We observe strong spatio-temporal stability in dLLM decoding: newly decoded tokens tend to lie near decoded neighbors, and their predicted IDs often remain consistent across several denoising steps. Inspired by this stability, our STDec introduces spatial-aware decoding and temporal-aware decoding. The spatial-aware decoding dynamically generates the token-adaptive thresholds by aggregating the decoded states of nearby tokens. The temporal-aware decoding relaxes the decoding thresholds for tokens whose predicted token IDs remain consistent over denoising steps. Our STDec is training-free and remains compatible with cache-based acceleration methods. Across textual reasoning and multimodal understanding benchmarks, STDec achieves improved inference throughput while maintaining comparable generation score. Notably, on MBPP with LLaDA, STDec achieves up to 14.17× speedup with a comparable score. The homepage is available at https://yzchen02.github.io/STDec.

1 Introduction

Most current LLMs [19, 13, 49] adopt the autoregressive paradigm, whose left-to-right generation limits inference-time parallelism. Diffusion language models (dLLMs) [35, 51] offer a compelling alternative to this left-to-right paradigm. dLLMs can exploit bidirectional context at each refinement step and decode multiple token positions in parallel. Recent dLLMs have shown competitive performance on both textual reasoning [35, 54, 51] and multimodal understanding tasks [50, 48, 27].

Although dLLMs are inherently parallel, inference in practical dLLMs [35, 27, 48] remains bottlenecked by conservative decoding policies. Most existing methods finalize a fixed number of high-confidence tokens at each denoising step through a global confidence rule, such as top-k or fixed-threshold decoding. For example, top-1 strategy performs token decoding according to instantaneous confidence, failing to leverage rich spatio-temporal contextual signals. Recent decoding strategies [24, 3, 46] have considered to incorporate various confidence information of masked tokens for token decoding. However, they still do not jointly exploit the local decoded context and cross-step ID stability as a unified token-level threshold.

We conduct an empirical study to reveal spatio-temporal patterns in dLLM decoding. Spatially, newly decoded tokens tend to emerge near already decoded regions, suggesting that decoded neighbors

*Corresponding Author

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

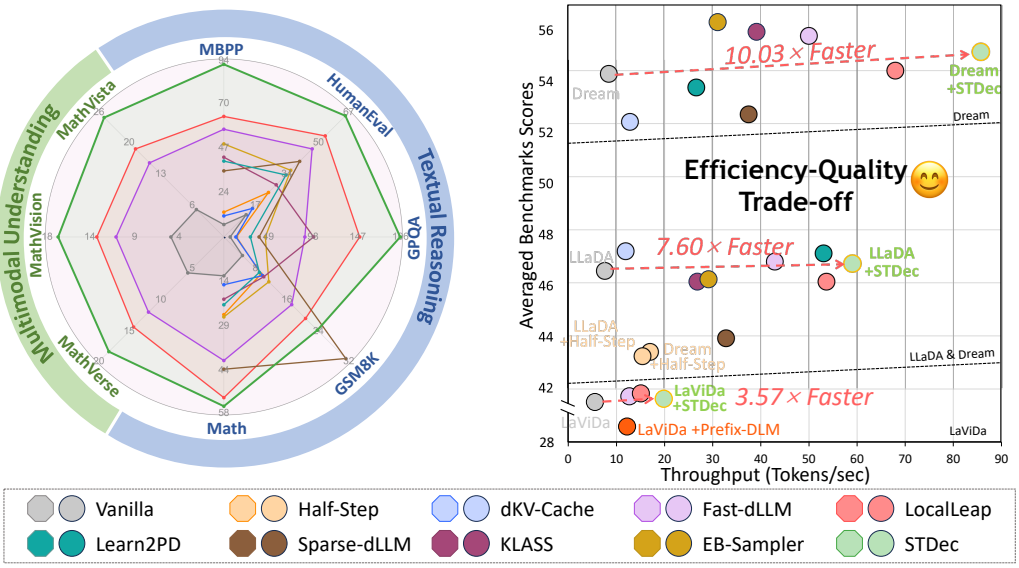


Figure 1: **Efficiency-quality trade-off compared with existing acceleration approaches.** Across eight benchmarks (left) and three models (right), STDec achieves a strong trade-off between throughput and quality, consistently delivering higher generation efficiency while preserving competitive overall performance on both textual reasoning and multimodal understanding tasks.

provide useful cues to perform token decoding. Temporally, the tokens tend to preserve the same predicted IDs at multiple denoising steps before they are finally decoded, indicating that cross-step ID consistency is a common phenomenon before decoded. This motivates us a natural question: *how can we perform token decoding to achieve a better trade-off between efficiency and quality according to the spatio-temporal state?*

In this paper, we propose STDec, a spatio-temporal stability guided decoding method for dLLMs. Our STDec replaces the single global threshold with token-adaptive thresholds according to spatio-temporal stability cues, aiming to improve the efficiency-quality trade-off. Our STDec consists of two complementary components: spatial-aware decoding and temporal-aware decoding. Spatial-aware decoding leverages the local decoded states to generate token-adaptive thresholds. Specifically, initializing from a binary map over decoded and masked positions, we apply a smoothing operation to propagate local decoded-state information, resulting in that masked token closer to more decoded tokens has lower threshold. Temporal-aware decoding further adjusts these thresholds according to historical ID information. When a masked token consistently predicts the same ID across previous denoising steps, we apply a relaxation factor to further lower its threshold. In this way, we can employ spatio-temporal state to jointly guide token-adaptive thresholding for efficient decoding.

We conduct extensive experiments on different textual reasoning and multimodal understanding tasks to validate the effectiveness and efficiency of our STDec. Extensive results indicate that our token-adaptive decoding yields a stronger efficiency-quality trade-off than confidence-only decoding rules. As shown in Fig. 1, our STDec consistently improves decoding throughput while maintaining competitive generation performance on various benchmarks and models. Further, our STDec remains compatible with cache-based acceleration methods. The contributions can be summarized as follows:

- **Spatio-temporal stability characterization.** We conduct empirical study and show that dLLM decoding exhibits spatio-temporal stability: newly decoded tokens tend to emerge near already decoded regions, and many tokens preserve the same predicted ID across multiple denoising steps before final decoding. These regularities reveal that different tokens can have substantially different decoding states during iterative denoising.
- **Token-adaptive decoding via spatio-temporal cues.** We propose STDec, a training-free spatio-temporal stability guided decoding method that makes token-adaptive decoding based on local decoded states and cross-step ID consistency, enabling faster decoding while maintaining quality.
- **Strong efficiency-quality trade-off.** Extensive experiments on diverse textual reasoning and multimodal understanding benchmarks demonstrate that STDec achieves substantial decoding

acceleration while preserving generation quality. Compared with vanilla LLaDA [35], it delivers an average 7.60× speedup on five textual reasoning benchmarks with no loss in average score.

2 Related Work

2.1 Diffusion Large Language Models

Discrete diffusion language models (dLLMs) generate text through iterative denoising over masked token sequences, enabling bidirectional context integration and parallel token refinement. Recent dLLMs [35, 51, 1, 40] scale this paradigm and achieve competitive results on general language, mathematical reasoning, and code generation benchmarks. Related work explores interpretable latent representations for reasoning [6, 5]. Beyond model scaling, recent studies [34, 21, 12, 29] have also begun to examine inference-time decoding behavior in dLLMs, including how token commitment order and denoising trajectories affect the quality-efficiency trade-off. In parallel, multimodal dLLMs incorporate visual encoders for vision-language understanding [27, 52], while newer architectures further unify discrete diffusion for multimodal understanding and generation [50, 48]. In video understanding, structured temporal reasoning also supports event prediction [39, 38]. Despite progress in scaling, most dLLM inference pipelines decode tokens by a position-agnostic threshold, leaving local decoded context and cross-step ID stability under-exploited.

2.2 Efficient Inference for dLLMs

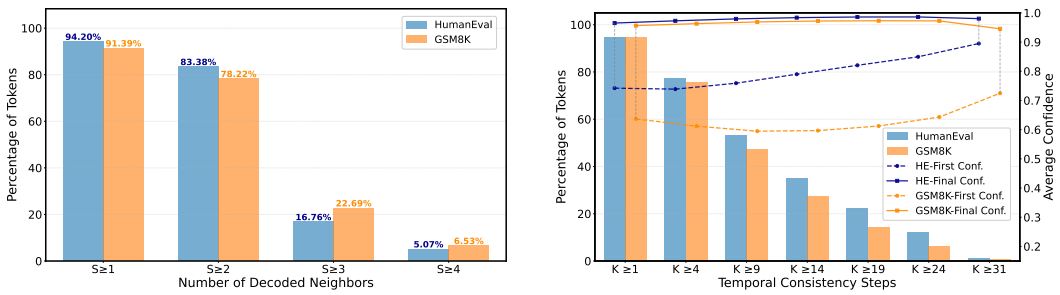
Existing acceleration methods for dLLMs mainly follow two directions: reducing per-step cost via caching and increasing per-step progress via decoding policies. Cache-based approaches adapt the KV-cache mechanism to bidirectional diffusion transformers. These methods reuse attention states or intermediate token features across denoising steps in a training-free manner [30, 33, 32, 46, 55], with further refinements on cache organization, update frequency, feature reuse, and eviction strategies [16, 37, 20, 17]. They effectively reduce redundant computation within each denoising step, but generally do not change the decoding policy that determines which masked tokens are finalized earlier. Other approaches reduce computation through pruning and routing in video models [44, 28]. Complementarily, decoding-policy methods accelerate inference by increasing the number of tokens decoded per step or by reducing unnecessary denoising iterations. Some methods exploit local propagation or adaptive parallel decoding to expand decoded regions from confident tokens [24], while others use phase-wise or slow-fast schedules to allocate different decoding budgets across early and late denoising stages [45, 14]. Beyond these strategies, other approaches employ convergence-based early stopping [26] or modify denoising and remasking schedules [41, 43, 3, 23]. Related methods further incorporate inference-time calibration and remasking to mitigate decoding bias [18, 9, 7]. Although these methods improve the efficiency of dLLM inference, many of them still rely on pre-defined schedules, global confidence rules, or single-source trajectory signals, rather than explicitly designing token-level adaptive thresholds from complementary spatial and temporal decoding cues.

Unlike existing dLLM acceleration methods, LocalLeap [24] performs anchor-triggered regional relaxation with a single threshold, while KLASS [22] uses cross-step distributional stability as a decode-or-defer criterion. In contrast, our STDec constructs token-specific spatial thresholds from the decoded/masked state pattern and further adjusts it using cross-step ID consistency, thereby integrating spatial and temporal cues within a unified token-level threshold. This provides a training-free and model-agnostic decoding rule that increases per-step decoding progress and remains complementary to cache-based acceleration.

3 Method

3.1 Preliminaries

Diffusion Large Language Models (dLLMs). dLLMs [35, 48] generate sequences by iteratively denoising masked token sequences. dLLMs first encode inputs into a discrete token sequence $X_0 = \{x_0(i)\}_{i=1}^{L}$, where L is the sequence length and each token belongs to a discrete vocabulary $\mathcal{V}$. A T-step Markov masking process gradually corrupts X_0 into X_T filled with `[MASK]` tokens. During inference, a bidirectional Transformer denoiser p_θ reverses this process. Assuming that $M_t \subseteq \{1, \ldots, L\}$ is the set of masked positions at step t, $\mathcal{D}_t = \{1, \ldots, L\} \setminus M_t$ denotes the decoded



(a)Spatial stability (b)Temporal stability

Figure 2: **Empirical evidence of spatio-temporal regularities in dLLM decoding.** (a) *Spatial stability.* For tokens decoded at the current step, we report the percentage whose previously decoded neighbors within a radius of 2 are at least S. (b) *Temporal stability.* For decoded tokens, we report the percentage whose top-1 predictions have remained unchanged for at least K previous consecutive steps, together with the average confidence at the first ID-stable step and at the final decoded step. Across HumanEval [8] and GSM8K [10], newly decoded tokens are often located near previously decoded tokens, and many token identities remain consistent before final decoding.

position, and P are prompt tokens. For each masked position $i \in M_t$, the denoiser predicts a categorical distribution over $\mathcal{V}$:

$$p_t(i)[v] = p_\theta\big(x_t(i) = v \,\big|\, X_t, P\big), \quad i \in M_t, v \in \mathcal{V}. \tag{1}$$

Then top-1 token can be predicted as $\hat{x}_t(i) = \arg\max_{v \in \mathcal{V}} p_t(i)[v]$ and its confidence is $c_t(i) = p_t(i)\big[\hat{x}_t(i)\big]$. At each denoising step, the decoder faces a token-level selective decoding decision: whether to decode a masked token now or defer it for further refinement.

Global Threshold based Decoding. Most existing dLLMs [35, 48] adopt a global threshold τ_t to determine which masked tokens are decoded at step t:

$$\mathcal{C}_t = \big\{\, i \in M_t \,\big|\, c_t(i) \geq \tau_t \big\}, \tag{2}$$

where global threshold τ_t is obtained from top-k strategy [48] or fixed threshold [46]. The masked set is then updated as $M_{t-1} = M_t \setminus \mathcal{C}_t$, and decoded tokens are fixed as $x(i) = \hat{x}_t(i)$ thereafter. The global-threshold strategy is simple and robust, but it reduces token selection to a confidence-only criterion shared by all masked positions. In iterative denoising, however, global confidence rules do not explicitly use local decoded-state patterns or cross-step prediction consistency, motivating a spatio-temporal reformulation of token-level thresholding.

3.2 Empirical Analysis

Here we perform empirical study to reveal there exists spatio-temporal stability in dLLM token decoding, which is used by STDec for token-adaptive thresholding. Specifically, we analyze decoding trajectories on HumanEval [8] and GSM8K [10], as shown in Fig. 2.

Spatial Stability. We first reveal where newly decoded tokens appear relative to tokens already decoded at previous steps. For each token decoded at the current step, we count the number of previously decoded neighbors within a radius of 2. As shown in Fig. 2a, more than 90% of newly decoded tokens have at least one decoded neighbor nearby. Further, among these decoded tokens, a large fraction have multiple decoded neighbors. This indicates that decoded tokens are spatially around already decoded tokens, suggesting that nearby decoded tokens provide a useful local cue for decoding, which can be used to set token-adaptive thresholds.

Temporal Stability. We then reveal whether token identities tend to persist before they are decoded. For each decoded token, we measure whether its top-1 prediction has remained unchanged for at least K previous consecutive steps. As shown in Fig. 2b, the fraction is close to 95% for $K \geq 1$ on both HumanEval and GSM8K. Moreover, the confidence at the first ID-stable step is consistently lower than the confidence at the eventual decoding step. This suggests that cross-step ID consistency

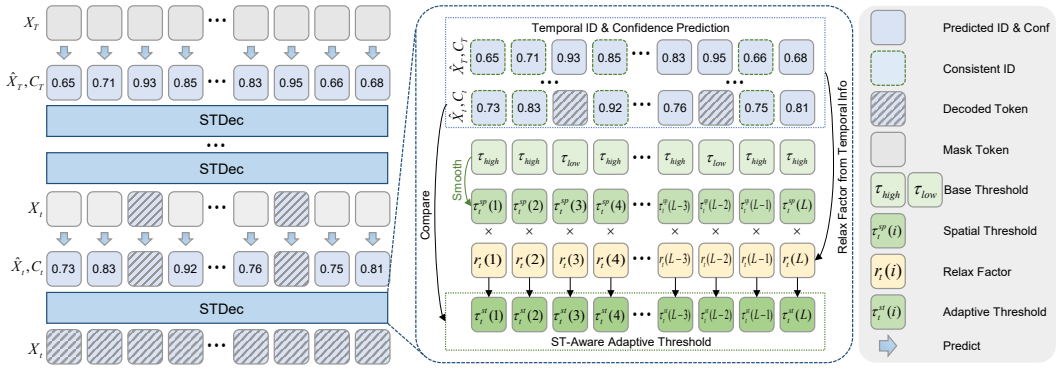


Figure 3: **Overview of our STDec.** The left gives the overall decoding with our STDec in the dLLM, where STDec progressively decodes tokens across denoising steps. The middle gives the details of our STDec that calculates the token-adaptive thresholds by combining spatial and temporal stability.

captures a complementary temporal signal that is not fully reflected by instantaneous confidence alone, motivating its use as an additional cue for threshold relaxation.

3.3 Spatio-Temporal Stability Guided Decoding

Motivated by the empirical study in Sec. 3.2, we introduce a spatio-temporal stability guided decoding strategy that replaces the global threshold with token-adaptive thresholds, named STDec. When designing token-adaptive thresholds, there are two key factors to be considered: *(1) masked tokens around more decoded tokens should receive lower thresholds.* and *(2) cross-step ID consistency can be used to relax threshold.* To achieve this goal, we introduce two complementary modules: spatial-aware decoding and temporal-aware decoding, which adjust the thresholds according to the local decoded state and cross-step ID consistency respectively.

Spatial-aware Decoding. Spatial-aware decoding converts the current decoded/masked state pattern into a spatial threshold for each masked token. We first choose two seed values $\tau_{\text{low}} < \tau_{\text{high}}$ and construct an auxiliary threshold seed map from the token states:

$$\tau_t^{\text{init}}(i) = \begin{cases} \tau_{\text{low}}, & i \notin M_t, \\ \tau_{\text{high}}, & i \in M_t. \end{cases} \tag{3}$$

The low seed values assigned to decoded positions are not used for decoding decisions on those positions; instead, they are propagated to nearby masked positions through smoothing. We apply a Gaussian smoothing operator to the seed map and obtain the spatial threshold for each masked token:

$$\tau_t^s(i) = f_G(\tau_t^{\text{init}}, \sigma)(i), \quad i \in M_t, \tag{4}$$

where σ controls the smoothing scale and f_G denotes the Gaussian smoothing operation. As a result, masked tokens closer to decoded positions receive lower spatial thresholds, whereas tokens in largely masked regions remain closer to τ_{high}.

Temporal-aware Decoding. The temporal-aware decoding relaxes token thresholds according to historical ID consistency, which serves as an observable proxy for reduced future flip risk during iterative refinement. Let $K_t(i)$ denote the number of consecutive previous steps for which token i keeps the same top-1 predicted ID:

$$K_t(i) = \begin{cases} K_{t+1}(i) + 1, & \hat{x}_t(i) = \hat{x}_{t+1}(i), \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

We then define a temporal relaxation factor $r_t(i) \in (0, 1]$ as

$$r_t(i) = \begin{cases} 1, & K_t(i) \le 1, \\ \alpha, & K_t(i) > 1, \end{cases} \tag{6}$$

where $\alpha < 1$ is a constant relaxation factor. We apply temporal relaxation only when the current top-1 prediction matches those from the two preceding denoising steps, which acts as a conservative guard against rewarding transient or noisy predictions. In practice, we further require the previous-step confidence to exceed τ_{low}, so that clearly uncertain tokens are excluded from temporal relaxation.

Final Token-Adaptive Decoding. We combine spatial-aware and temporal-aware decoding into a unified token-adaptive decoding strategy:

$$\tau_t^{st}(i) = \tau_t^s(i) \cdot r_t(i), \tag{7}$$

where $\tau_t^s(i)$ denotes the spatial-aware threshold, and $r_t(i)$ is temporal relaxation factor. For the fully masked tokens X_T, we first predict top-1 IDs $\hat{X_T} = \{\hat{x_T}(i)\}_{i=1}^L$ and corresponding confidence scores $C_T = \{c_T(i)\}_{i=1}^L$. Based on these confidence scores C_T, we employ our proposed STDec to generate token-adaptive thresholds $\tau_T^{st} = \{\tau_T^{st}(i)\}_{i=1}^L$, and then perform token decoding according to these token-adaptive thresholds. The masked tokens are fed into the following step to predict confidence scores and perform decoding again. In this way, we can decode all the masked tokens after T steps. At step t, we can decode the masked tokens as

$$\mathcal{C}_t = \left\{ i \in M_t \;\middle|\; c_t(i) \geq \tau_t^{\text{st}}(i) \right\}, \tag{8}$$

which allows different masked tokens to be decoded under different thresholds at the same step. If no masked token meets its adaptive threshold, we decode the masked token with the highest confidence to ensure progress. We also present the pseudo code of our STDec in Appendix C.

Decoding Safety Analysis. Here we present theoretical analysis on the decoding safety of token-adaptive decoding. Following Fast-DLLM [46], we analyze the decoded positions selected by our STDec under the joint-PMF idealization. Let $C_t = \{i_1, \ldots, i_n\}$ be the positions decoded at step t, and define $\varepsilon_j := 1 - c_t(i_j)$ for each $i_j \in C_t$. The token-adaptive decoding on the selected set C_t yields the same assignment as a greedy sequential decoder operating on C_t when

$$\sum_{j=1}^{n} \varepsilon_j + \max_{1 \leq j \leq n} \varepsilon_j < 1. \tag{9}$$

This gives a sufficient condition under which the token-adaptive parallel decoding induced by our spatial-aware and temporal-aware modules agrees with ideal greedy sequential decoding on the selected positions. Appendix B shows the detailed analysis and empirical experiment that about 90% of STDec decoding steps fall within the safe regime.

4 Experiments

4.1 Experiment Setup

Baselines. For textual reasoning, we compare our STDec with the original decoding strategy in Dream [51] and LLaDA [35], a cache-based method [33, 37], and decoding-policy acceleration methods [46, 24, 3, 22, 4]. For Learn2PD [3], we only use its policy-only setting. In addition, we include a simple half-step decoding baseline, which directly halves the number of denoising step used. For multimodal understanding, we report a cache-based strategy Prefix-DLM, and some decoding-policy acceleration methods based on LaViDa-Reason [27].

Implementation Details. For textual reasoning with Dream [51] and LLaDA [35], we adopt the semi-autoregressive decoding strategy. For all baselines, we use the hyperparameters recommended in their respective papers. Detailed settings, including generation length, denoising steps, and block size, are provided in Appendix A. For our STDec, we use a single fixed set of stability hyperparameters across all experiments: τ_{high}=0.9, τ_{low}=0.3, relaxation factor α=0.85, and Gaussian smoothing with σ=1 and radius 2. These global hyperparameters remain fixed, while the token-level thresholds are dynamically recomputed at each denoising step from the evolving decoding states. All experiments are conducted on NVIDIA RTX 4090D GPUs.

Evaluation Metrics. We report task scores (accuracy or Pass@1) and inference throughput, where throughput is measured as the averaged number of output tokens generated per second (TPS). We additionally report Tokens per Forward (TPF), measured as the average number of tokens decoded per model forward pass, and TPS speedup relative to the default decoding strategy in dLLMs.

4.2 Main Results

Performance on Dream. Table 1 gives the results on five different benchmarks based on Dream [51]. On these benchmarks, our STDec achieves higher TPS while maintaining comparable scores with

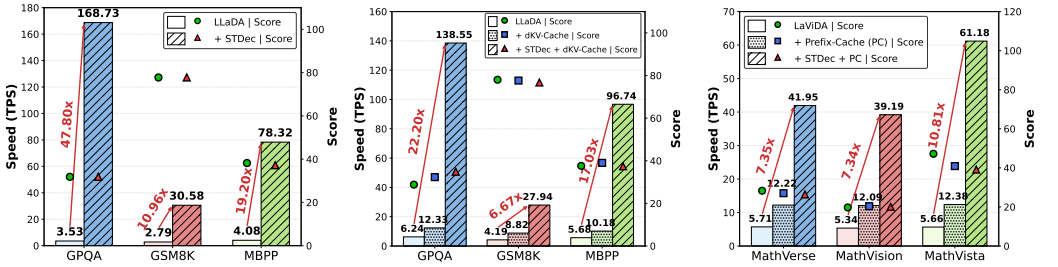
Table 1: **Comparison on textual reasoning benchmarks.** We report TPS, speed, TPF, and score for Dream-7B and LLaDA-8B. Bold values indicate the best TPS, speed, and TPF within each task group, while grayed scores indicate performance drops relative to the corresponding vanilla model.

Task	Method	Dream-7B-Instruct				LLaDA-8B-Instruct			
		TPS↑	Speed↑	TPF↑	Score↑	TPS↑	Speed↑	TPF↑	Score↑
MBPP	Vanilla	6.57	1.00×	1.00	51.40	5.68	1.00×	1.00	37.60
	+ Half-Step	13.11	2.00×	2.00	35.80	11.51	2.03×	2.00	34.80
	+ dKV-Cache [33]	11.12	1.69×	1.00	49.60	10.18	1.79×	1.00	39.00
	+ Fast-dLLM [46]	56.88	8.66×	8.74	54.80	62.68	11.04×	11.42	37.40
	+ LocalLeap [24]	63.65	9.69×	8.96	53.60	76.48	13.46×	11.83	37.60
	+ EB-Sampler [4]	49.14	7.48×	7.49	55.40	35.58	6.26×	6.29	38.40
	+ KLASS [22]	42.11	6.41×	7.48	54.80	36.03	6.34×	7.02	37.40
	+ Sparse-dLLM [37]	34.90	5.31×	1.00	48.80	29.83	5.25×	1.00	35.20
	+ Learn2PD [3]	40.07	6.10×	6.08	50.20	76.76	13.51×	13.35	38.00
	+ STDec	**91.16**	**13.88×**	**14.16**	55.60	**80.47**	**14.17×**	**14.03**	38.40
HumanEval	Vanilla	11.82	1.00×	1.00	59.15	10.82	1.00×	1.00	48.17
	+ Half-Step	23.69	2.00×	2.00	35.37	21.67	2.00×	2.00	35.37
	+ dKV-Cache [33]	15.21	1.29×	1.00	56.10	14.23	1.32×	1.00	46.95
	+ Fast-dLLM [46]	46.87	3.97×	4.03	62.20	38.49	3.56×	3.95	48.78
	+ LocalLeap [24]	53.80	4.55×	4.46	58.54	50.33	4.65×	**4.85**	46.34
	+ EB-Sampler [4]	35.48	3.00×	3.01	63.41	24.47	2.26×	2.27	43.90
	+ KLASS [22]	27.77	2.35×	3.07	62.20	20.44	1.89×	2.31	48.17
	+ Sparse-dLLM [37]	40.23	3.40×	1.00	58.54	35.93	3.32×	1.00	39.63
	+ Learn2PD [3]	32.99	2.79×	2.52	58.54	51.17	4.73×	4.77	46.95
	+ STDec	**64.50**	**5.46×**	**5.48**	60.37	**52.92**	**4.89×**	**4.85**	48.78
GPQA	Vanilla	6.95	1.00×	1.00	32.83	6.24	1.00×	1.00	28.79
	+ Half-Step	13.88	2.00×	2.00	32.32	12.66	2.03×	2.00	30.81
	+ dKV-Cache [33]	13.21	1.90×	1.00	33.33	12.33	1.98×	1.00	32.32
	+ Fast-dLLM [46]	88.95	12.80×	12.72	34.85	62.61	10.03×	12.90	28.79
	+ LocalLeap [24]	149.15	21.46×	21.83	33.33	74.60	11.96×	14.71	29.29
	+ EB-Sampler [4]	45.54	6.55×	6.46	33.33	57.69	9.24×	9.43	30.30
	+ KLASS [22]	98.54	14.18×	14.93	32.83	54.34	8.71×	9.00	29.29
	+ Sparse-dLLM [37]	38.66	5.56×	1.00	30.30	34.06	5.46×	1.00	32.83
	+ Learn2PD [3]	29.13	4.19×	4.11	33.33	71.03	11.38×	11.48	33.33
	+ STDec	**193.62**	**27.86×**	**27.94**	32.83	**92.08**	**14.76×**	**14.72**	29.29
GSM8K	Vanilla	4.71	1.00×	1.00	83.47	4.19	1.00×	1.00	78.01
	+ Half-Step	9.41	2.00×	2.00	74.22	8.25	1.97×	2.00	75.82
	+ dKV-Cache [33]	9.83	2.09×	1.00	79.08	8.82	2.11×	1.00	77.63
	+ Fast-dLLM [46]	17.20	3.65×	3.67	82.94	12.16	2.90×	3.16	78.77
	+ LocalLeap [24]	20.71	4.40×	4.45	82.49	17.01	4.06×	**4.02**	77.98
	+ EB-Sampler [4]	11.40	2.42×	2.42	83.70	7.64	1.82×	1.84	78.77
	+ KLASS [22]	10.12	2.15×	2.28	83.55	8.97	2.14×	2.26	76.80
	+ Sparse-dLLM [37]	**30.90**	**6.56×**	1.00	81.88	**26.11**	**6.23×**	1.00	77.71
	+ Learn2PD [3]	9.01	1.91×	1.83	82.56	16.49	3.94×	3.97	78.09
	+ STDec	23.70	5.03×	**5.01**	82.34	16.94	4.04×	4.00	78.01
MATH	Vanilla	12.67	1.00×	1.00	44.64	11.90	1.00×	1.00	40.38
	+ Half-Step	25.31	2.00×	2.00	39.40	23.64	1.99×	2.00	39.44
	+ dKV-Cache [33]	15.56	1.23×	1.00	44.04	14.73	1.24×	1.00	40.90
	+ Fast-dLLM [46]	40.26	3.18×	3.22	44.16	39.00	3.28×	3.54	41.00
	+ LocalLeap [24]	52.35	4.13×	4.28	44.26	49.93	4.20×	**4.65**	39.66
	+ EB-Sampler [4]	26.05	2.06×	2.06	48.22	22.38	1.88×	1.89	41.14
	+ KLASS [22]	20.28	1.60×	2.13	45.90	23.75	2.00×	2.50	38.94
	+ Sparse-dLLM [37]	43.09	3.40×	1.00	44.08	38.49	3.23×	1.00	34.44
	+ Learn2PD [3]	22.09	1.74×	1.47	44.26	49.88	4.19×	4.23	40.04
	+ STDec	**55.15**	**4.35×**	**4.45**	44.64	**53.05**	**4.46×**	4.58	39.86
Avg.	Vanilla	8.54	1.00×	1.00	54.30	7.77	1.00×	1.00	46.59
	+ Half-Step	17.08	2.00×	2.00	43.42	15.55	2.00×	2.00	43.25
	+ dKV-Cache [33]	12.99	1.52×	1.00	52.43	12.06	1.55×	1.00	47.36
	+ Fast-dLLM [46]	50.03	5.86×	6.47	55.79	42.99	5.53×	6.99	46.95
	+ LocalLeap [24]	67.93	7.95×	8.80	54.44	53.67	6.91×	8.01	46.17
	+ EB-Sampler [4]	33.52	3.93×	4.29	56.81	29.55	3.80×	4.34	46.50
	+ KLASS [22]	39.76	4.66×	5.98	55.86	28.71	3.69×	4.62	46.12
	+ Sparse-dLLM [37]	37.56	4.40×	1.00	52.72	32.88	4.23×	1.00	43.96
	+ Learn2PD [3]	26.66	3.12×	3.20	53.78	53.07	6.83×	7.56	47.28
	+ STDec	**85.63**	**10.03×**	**11.41**	55.16	**59.09**	**7.60×**	**8.43**	46.87

original Dream. Specifically, our STDec improves TPS from 6.57 to 91.16 on MBPP (13.88×) and from 11.82 to 64.50 on HumanEval (5.46×). On mathematics and science benchmarks, our STDec yields substantial gains in decoding speed, such as 5.03× on GSM8K, 4.35× on MATH, and 27.86× on GPQA, while almost maintaining the scores. When simply halving the number of denoising steps,

Table 2: **Comparison on multimodal understanding with LaViDa-Reason.** We report TPS, speedup, and score across benchmarks. Bold values indicate the best efficiency, and grayed scores denote performance drop.

	MathVerse			MathVision			MathVista			*Avg.*		
Method	TPS	Speed	Score	TPS	Speed	Score	TPS	Speed	Score	TPS	Speed	Score
LaViDa [27]	5.71	1.00×	28.30	5.34	1.00×	19.74	5.66	1.00×	47.20	5.57	1.00×	31.75
+ Prefix-DLM [27]	12.22	2.14×	27.03	12.09	2.26×	20.39	12.38	2.19×	40.90	12.23	2.20×	29.44
+ Fast-dLLM [46]	11.94	2.09×	28.68	10.87	2.04×	20.72	15.31	2.70×	47.50	12.71	2.28×	32.30
+ LocalLeap [24]	14.31	2.51×	28.30	12.81	2.40×	21.71	18.19	3.21×	47.70	15.10	2.71×	32.57
+ STDec	**18.22**	**3.19**×	28.30	**16.71**	**3.13**×	21.71	**24.66**	**4.36**×	46.20	**19.86**	**3.57**×	32.07



(a) (b) (c)

Figure 4: **Further scalability of our STDec.** *(a) Long-context textual reasoning.* Our STDec can be adopted for long-context generation (1024 tokens). *(b-c) Combination with cache-based approaches.* Our method can be integrated with cache-based approaches dKV-Cache [33] and Prefix-DLM [27], which can accelerate the inference speed almost without loss of generation quality.

it presents 2× speedup but leads to a significant drop in scores on most benchmarks, demonstrating that simple step reduction cannot simultaneously accelerate and maintain its quality. Compared to other methods [33, 46, 24, 37, 3], our STDec offers a better efficiency–quality trade-off.

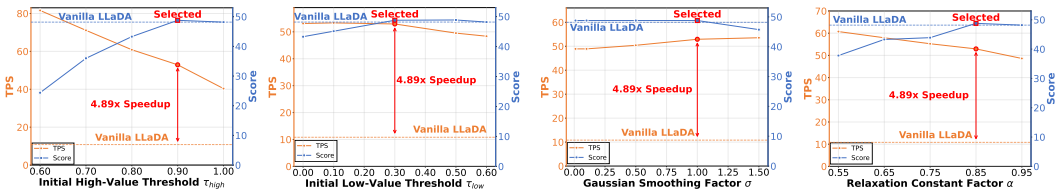
Performance on LLaDA. Table 1 further presents the results on textual reasoning benchmarks with LLaDA [35]. Similarly, our STDec gives the best trade-off between TPS and quality. For code generation, our STDec increases TPS from 5.68 to 80.47 on MBPP and from 10.82 to 52.92 on HumanEval, without loss of quality. For mathematics and science, our STDec obtains 4.04× speedup on GSM8K, 4.46× on MATH, and 14.76× on GPQA, while almost preserving generation scores. Compared to Fast-dLLM [46] and Learn2PD [3], our STDec achieves higher average throughput. In terms of TPF, STDec achieves an average TPF of 11.41 on Dream [51] and 8.43 on LLaDA [35]. These values are substantially higher than those of KLASS [22] (5.98 and 4.62) and EB-Sampler [4] (4.29 and 4.34), while STDec maintains comparable average task performance.

Open-ended generation. We further evaluate STDec on 183 English and Chinese creative-writing prompts from WritingBench [47], using the default hyperparameters without task-specific tuning. STDec improves TPS from 4.36 to 7.99 (1.83×) and TPF from 1.00 to 1.77, while maintaining comparable English and Chinese scores.

Performance on LaViDa. Table 2 reports the results of different acceleration strategies with LaViDa-Reason [27] on multimodal understanding tasks. Our STDec also provides a good efficiency-quality trade-off on multimodal understanding. Specifically, our STDec improves TPS from 5.57 to 19.86 (3.57×) averaged over three benchmarks. Compared to the built-in cache-based and other sampling-based approaches, our STDec consistently achieves higher TPS on all benchmarks maintaining comparable performance.

Further Scalability. To further validate the efficiency of our STDec, we evaluate its scalability along two aspects: long-context textual reasoning and compatibility with cache-based acceleration.

(i) Long-context textual reasoning. Fig. 4a reports 1024-token generation results with LLaDA [35]. Our STDec consistently improves throughput by 10.96×-47.80× with comparable scores; for example, it achieves a 47.80× speedup on GPQA with the same score. These results suggest that STDec scales favorably to long-context decoding, substantially improving throughput while keeping performance broadly comparable.



(a) (b) (c) (d)

Figure 5: **Impact of hyperparameters in our STDec**, including initial high-value thresholds τ_{high}, initial low-value thresholds τ_{low}, Gaussian smoothing factor σ, and the relaxation constant factor α. The red markers denote our default setting, which achieves a $4.89\times$ speedup over vanilla LLaDA.

(ii) Stacking STDec with cache-based acceleration. Fig. 4b and Fig. 4c show that STDec is complementary to caching-based approaches. This complementarity arises because caching reduces the cost of each forward pass, whereas STDec reduces the required number of forward passes. On textual reasoning with LLaDA [35], stacking STDec on top of dKV-Cache [33] consistently brings additional gains over the cache baseline: $11.2\times$ on GPQA, $3.2\times$ on GSM8K, and $9.5\times$ on MBPP in Fig. 4b. On multimodal understanding with LaViDa [27], which natively supports Prefix-DLM caching, adding STDec on top of Prefix-DLM yields similar speed improvements while maintaining comparable performance in Fig. 4c.

We further extend our STDec to image generation, additional scalability results are provided in Appendix D. These additional experiments demonstrate that STDec is robust across sequence lengths and modalities, and can be stacked with cache-based methods.

4.3 Ablation Studies

We conduct extensive ablation studies on HumanEval [8] with LLaDA-8B-Instruct [35]. In addition, we present some additional ablation experiments in Appendix E.

Impact of Spatial-aware and Temporal-aware Decoding. Fig. 6 presents the results of integrating our proposed modules into the baseline LLaDA [35]. When only integrating the temporal-aware decoding into the baseline, it improves TPS from 10.82 to 32.89 without sacrificing score. When integrating these two modules, it provides the fastest speed of 52.92 TPS. Namely, our spatial-aware decoding and temporal-aware decoding are complementary, together enabling faster speed.

Ablation on Spatial-aware Decoding. In spatial-aware decoding, we first introduce an initial high-value threshold τ_{high} and an initial low-value threshold τ_{low} to generate an initial threshold map, and then employ a Gaussian smoothing operation with the factor of σ to generate the final threshold map. (i) We first show the impact of τ_{high} in Fig. 5a. When $\tau_{\text{high}} = 0.9$, it presents best trade-off between generation score and TPS. When τ_{high} is larger than 0.9, it does not improve generation score while reducing speed. When τ_{high} is smaller than 0.9, although TPS shows some improvements, the generation score decreases significantly. (ii) We further validate the impact of τ_{low} in Fig. 5b. It has best trade-off when $\tau_{\text{low}} = 0.3$. (iii) Finally, we show the impact of σ in Fig 5c. It achieves best trade-off when $\sigma = 1.0$. When σ is too large, the influence of surrounding decoded tokens increases, leading to a decline in performance. When σ is too small, it presents insufficient acceleration because the information from surrounding decoded tokens is not fully taken into account.

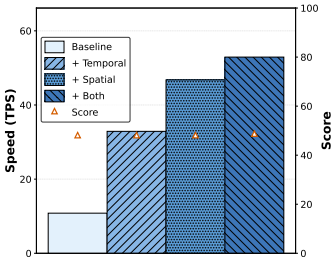


Figure 6: **Impact of integrating our STDec modules.** We integrate our spatial-aware decoding and temporal-aware decoding into the baseline LLaDA [35].

Error Risk Analysis. We assess whether STDec introduces additional decoding errors relative to the vanilla decoder. (1) STDec maintains high semantic consistency with vanilla decoding. Specifically, the cosine similarity between output token embeddings of our STDec and vanilla decoding is 99.1% and 98.9% on HumanEval [8] and GSM8K [10], respectively. (2) We evaluate the aggregate error-budget certificate on decoding steps in Appendix B. Our STDec satisfies the certificate condition in more than 90% of decoding steps, with step-level satisfaction rates of 90.8% on GSM8K [10] and

91.7% on HumanEval [8]. These results suggest that our STDec accelerates decoding while largely preserving the reference decoding behavior.

5 Conclusion

In this paper, we present an analysis on spatial and temporal stability in dLLM decoding, and introduce a spatio-temporal stability guided decoding approach for dLLMs, named STDec. Our STDec includes a spatial-aware decoding and a temporal-aware decoding, which aim to leverage the information of decoded tokens and ID consistency to adjust decoding thresholds for masked tokens. We perform the experiments on textual reasoning and multimodal understanding benchmarks, which significantly demonstrates the efficiency of our STDec.

Limitations. STDec is a training-free inference acceleration method that improves efficiency by decoding more tokens at each denoising step. Therefore, its acceleration is more pronounced for tasks requiring relatively long generations, while the benefit can be limited for tasks with very short outputs, such as simple yes/no questions. For such tasks, reducing the per-step denoising cost may be more important than increasing the number of decoded tokens per step.

Broader Impacts. By reducing inference cost, STDec can make dLLM deployment more efficient and accessible. However, faster generation may also accelerate misuse of the underlying generative models, so deployment should follow the safety policies and safeguards of the base models.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62271346).

References

[1] Marianne Arriola, Aaron Gokaslan, Justin Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. In *International Conference on Learning Representations*, pages 50726–50753, 2025.

[2] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

[3] Wenrui Bao, Zhiben Chen, Dan Xu, and Yuzhang Shang. Learning to parallel: Accelerating diffusion large language models via learnable parallel decoding. In *International Conference on Learning Representations*, pages 83376–83392, 2026.

[4] Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas S Nolte, and Brian Karrer. Accelerated sampling from masked diffusion models via entropy bounded unmasking. In *Advances in Neural Information Processing Systems*, volume 38, pages 55981–56007, 2025.

[5] Shuochen Chang, Tong Bai, Xiaofeng Zhang, Qianli Ma, Qingyang Liu, Zhaohe Liao, Yibo Miao, and Li Niu. Unlocking the black box of latent reasoning: An interpretability-guided approach to intervention. In *Annual Meeting of the Association for Computational Linguistics*, pages 34019–34032, 2026.

[6] Shuochen Chang, Qingyang Liu, Shaobo Wang, Bingjie Gao, Qianli Ma, Haonan Zhao, Yibo Miao, Yulin Sun, Zelin Peng, Jiangtong Li, et al. Why struggle with continuous latents? interpretable discrete latent reasoning via rendered compression. *arXiv preprint arXiv:2606.29712*, 2026.

[7] Shuochen Chang, Xiaofeng Zhang, Qingyang Liu, and Li Niu. D3tom: Decider-guided dynamic token merging for accelerating diffusion mllms. In *AAAI Conference on Artificial Intelligence*, volume 40, pages 19961–19969, 2026.

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

[9] Xinhua Chen, Sitao Huang, Cong Guo, Chiyue Wei, Yintao He, Jianyi Zhang, Hai Li, and Yiran Chen. Dpad: Efficient diffusion language models with suffix dropout. In *International Conference on Learning Representations*, pages 57286–57316, 2026.

[10] 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. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[11] OpenCompass Contributors. Opencompass: A universal evaluation platform for foundation models. https://github.com/open-compass/opencompass, 2023.

[12] Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, Chenxin An, Peilin Zhao, Wei Bi, Jiawei Han, et al. Scaling diffusion language models via adaptation from autoregressive models. In *International Conference on Learning Representations*, 2025.

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

[14] Yufei Guo, Yinan Wu, Haoran Duan, Guiguang Ding, and Jungong Han. Phaselora: Control-regime-conditioned low-rank adaptation for continuous-action vision-language-action policies. *arXiv preprint arXiv:2608.15285*, 2026.

[15] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. In *Advances in Neural Information Processing Systems*, 2021.

[16] Zhanqiu Hu, Jian Meng, Yash Akhauri, Mohamed Abdelfattah, Jae-sun Seo, Zhiru Zhang, and Udit Gupta. Flashdlm: Accelerating diffusion language model inference via efficient kv caching and guided diffusion. In *International Conference on Learning Representations*, pages 153529–153544, 2026.

[17] Jianuo Huang, Yaojie Zhang, Yicun Yang, Benhao Huang, and Linfeng Zhang. Mask tokens as prophet: Fine-grained cache eviction for efficient dllm inference. In *Findings of the Association for Computational Linguistics*, pages 3456–3479, 2026.

[18] Pengcheng Huang, Shuhao Liu, Zhenghao Liu, Yukun Yan, Shuo Wang, Zulong Chen, and Tong Xiao. Pc-sampler: Position-aware calibration of decoding bias in masked diffusion models. *arXiv preprint arXiv:2508.13021*, 2025.

[19] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[20] Yuchu Jiang, Yue Cai, Xiangzhong Luo, Jiale Fu, Jiarui Wang, Chonghan Liu, et al. d^2 cache: Accelerating diffusion-based llms via dual adaptive caching. In *International Conference on Learning Representations*, pages 137511–137531, 2026.

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

[22] Seo Hyun Kim, Sunwoo Hong, Hojung Jung, Youngrok Park, and Se-Young Yun. Klass: Kl-guided fast inference in masked diffusion models. In *Advances in Neural Information Processing Systems*, volume 38, pages 92267–92301, 2025.

[23] Hyukhun Koh, Minha Jhang, Dohyung Kim, Sangmook Lee, and Kyomin Jung. Conditional [mask] discrete diffusion language model. In *Conference on Empirical Methods in Natural Language Processing*, pages 8910–8934, 2025.

[24] Fanheng Kong, Jingyuan Zhang, Yahui Liu, Zirui Wu, Yu Tian, Guorui Zhou, et al. Accelerating diffusion llm inference via local determinism propagation. *arXiv preprint arXiv:2510.07081*, 2025.

[25] Bo Li, Peiyuan Zhang, Kaichen Zhang, Fanyi Pu, Xinrun Du, Yuhao Dong, Haotian Liu, Yuanhan Zhang, Ge Zhang, Chunyuan Li, and Ziwei Liu. Lmms-eval: Accelerating the development of large multimoal models. https://github.com/EvolvingLMMs-Lab/lmms-eval, 2024.

[26] Pengxiang Li, Yefan Zhou, Dilxat Muhtar, Lu Yin, Shilin Yan, Li Shen, Yi Liang, Soroush Vosoughi, and Shiwei Liu. Diffusion language models know the answer before decoding. *arXiv preprint arXiv:2508.19982*, 2025.

[27] Shufan Li, Konstantinos Kallidromitis, Hritik Bansal, Akash Gokul, Yusuke Kato, Kazuki Kozuka, Jason Kuen, Zhe Lin, Kai-Wei Chang, and Aditya Grover. Lavida: A large diffusion language model for multimodal understanding. In *Advances in Neural Information Processing Systems*, volume 38, pages 105101–105134, 2026.

[28] Xinying Lin, Xuyang Liu, Yiyu Wang, Teng Ma, and Wenqi Ren. V-cast: Video curvature-aware spatio-temporal pruning for efficient video large language models. *arXiv preprint arXiv:2603.27650*, 2026.

[29] Sulin Liu, Juno Nam, Andrew Campbell, Hannes Stark, Yilun Xu, Tommi Jaakkola, and Rafael Gomez-Bombarelli. Think while you generate: Discrete diffusion with planned denoising. In *International Conference on Learning Representations*, 2025.

[30] Zhiyuan Liu, Yicun Yang, Yaojie Zhang, Junjie Chen, Chang Zou, Qingyuan Wei, Shaobo Wang, and Linfeng Zhang. dllm-cache: Accelerating diffusion large language models with adaptive caching. *arXiv preprint arXiv:2506.06295*, 2025.

[31] Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. In *International Conference on Learning Representations*, pages 23439–23554, 2024.

[32] Xinyin Ma, Gongfan Fang, and Xinchao Wang. Deepcache: Accelerating diffusion models for free. In *Conference on Computer Vision and Pattern Recognition*, pages 15762–15772, 2024.

[33] Xinyin Ma, Runpeng Yu, Gongfan Fang, and Xinchao Wang. dkv-cache: The cache for diffusion language models. In *Advances in Neural Information Processing Systems*, volume 38, pages 149009–149033, 2026.

[34] Shen Nie, Fengqi Zhu, Chao Du, Tianyu Pang, Qian Liu, Guangtao Zeng, Min Lin, and Chongxuan Li. Scaling up masked diffusion models on text. In *International Conference on Learning Representations*, pages 82974–82997, 2025.

[35] Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. In *Advances in Neural Information Processing Systems*, volume 38, pages 50608–50646, 2026.

[36] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

[37] Yuerong Song, Xiaoran Liu, Ruixiao Li, Zhigeng Liu, Zengfeng Huang, Qipeng Guo, Ziwei He, and Xipeng Qiu. Sparse-dllm: Accelerating diffusion llms with dynamic cache eviction. In *AAAI Conference on Artificial Intelligence*, volume 40, pages 33038–33046, 2026.

[38] Qile Su, Shoutai Zhu, Shuai Zhang, Baoyu Liang, and Chao Tong. Eventformer: A node-graph hierarchical attention transformer for action-centric video event prediction. In *ACM International Conference on Multimedia*, page 4698–4707, 2025.

[39] Qile Su, Jing Tang, Rui Chen, Lei Sun, and Xiangxiang Chu. Video-coe: Reinforcing video event prediction via chain of events. *arXiv preprint arXiv:2603.14935*, 2026.

[40] Jaesung Tae, Hamish Ivison, Sachin Kumar, and Arman Cohan. Tess 2: A large-scale generalist diffusion language model. In *Annual Meeting of the Association for Computational Linguistics*, pages 21171–21188, 2025.

[41] Guanghan Wang, Yair Schiff, Subham Sahoo, and Volodymyr Kuleshov. Remasking discrete diffusion models with inference-time scaling. In *Advances in Neural Information Processing Systems*, volume 38, pages 147282–147339, 2026.

[42] Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. Measuring multimodal mathematical reasoning with math-vision dataset. In *Advances in Neural Information Processing Systems*, volume 37, pages 95095–95169, 2024.

[43] Xu Wang, Chenkai Xu, Yijie Jin, Jiachun Jin, Hao Zhang, Kai Yu, and Zhijie Deng. Diffusion llms can do faster-than-ar inference via discrete diffusion forcing. In *International Conference on Learning Representations*, pages 2282–2299, 2026.

[44] Yutong Wang, Yunke Wang, Tianfan Xue, Yu Qiao, Yaohui Wang, Xinyuan Chen, and Chang Xu. Pare: Pruning and adaptive routing for efficient video generation. *arXiv preprint arXiv:2605.27336*, 2026.

[45] Qingyan Wei, Yaojie Zhang, Zhiyuan Liu, Puyu Zeng, Yuxuan Wang, Biqing Qi, Dongrui Liu, and Linfeng Zhang. Accelerating diffusion large language models with slowfast sampling: The three golden principles. In *International Conference on Learning Representations*, pages 4340–4354, 2026.

[46] Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dllm: Training-free acceleration of diffusion llm by enabling kv cache and parallel decoding. In *International Conference on Learning Representations*, pages 57027–57051, 2026.

[47] Yuning Wu, Jiahao Mei, Ming Yan, Chenliang Li, Shaopeng Lai, Yuran Ren, Zijia Wang, Ji Zhang, Mengyue Wu, Qin Jin, and Fei Huang. Writingbench: A comprehensive benchmark for generative writing. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

[48] Yi Xin, Qi Qin, Siqi Luo, Kaiwen Zhu, Juncheng Yan, Yan Tai, Jiayi Lei, Yuewen Cao, Keqi Wang, Yibin Wang, et al. Lumina-dimoo: An omni diffusion large language model for multi-modal generation and understanding. *arXiv preprint arXiv:2510.06308*, 2025.

[49] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[50] Ling Yang, Ye Tian, Bowen Li, Xinchen Zhang, Ke Shen, Yunhai Tong, and Mengdi Wang. Mmada: Multimodal large diffusion language models. In *Advances in Neural Information Processing Systems*, volume 38, pages 138867–138907, 2026.

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

[52] Runpeng Yu, Xinyin Ma, and Xinchao Wang. Dimple: Discrete diffusion multimodal large language model with parallel decoding. *arXiv preprint arXiv:2505.16990*, 2025.

[53] Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Yu Qiao, et al. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems? In *European Conference on Computer Vision*, pages 169–186, 2024.

[54] Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Llada 1.5: Variance-reduced preference optimization for large language diffusion models. In *Annual Meeting of the Association for Computational Linguistics*, pages 11425–11460, 2026.

[55] Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating diffusion transformers with token-wise feature caching. *arXiv preprint arXiv:2410.05317*, 2024.

In the appendix, we provide model experiment details in Section A, further theoretical analysis in Section B, algorithm details in Section C, extension to multi-modal dLLMs details in Section D, more ablation experiments in Section E, and more case studies in Section F.

A Experiment Details

A.1 Benchmark Details

We conduct experiments on textual reasoning and multimodal understanding benchmarks. In textual reasoning tasks, a detailed overview of benchmarks, as described below:

- **HumanEval** [8] is a hand-written evaluation set of 164 Python programming problems for docstring-conditioned function synthesis. The problems assess language comprehension, algorithms, and mathematics.
- **MBPP** [2] is a Python code-generation benchmark of entry-level tasks with natural-language prompts and unit-test evaluation. Following the standard split used by common evaluation suites, we evaluate on its problem test set, where each prediction is judged by passing the provided tests.
- **GPQA** [36] is a graduate-level science QA benchmark with expert-written 4-choice multiple-choice questions spanning biology, physics, and chemistry.
- **GSM8K** [10] is a curated dataset of 8.5K human-written grade-school math word problems with step-by-step natural-language solutions, split into 7.5K training and 1K test examples.
- **MATH** [15] is a problem evaluation subset of the MATH benchmark. It is formed by uniformly sampling held-out problems from the MATH test split, and its subject and difficulty distributions are designed to remain representative of the original test set.

In multimodal understanding tasks, we use three math reasoning benchmarks as:

- **MathVerse** [53] is a visual math benchmark with 2,612 diagram-based problems spanning plane geometry, solid geometry, and functions; each problem is rewritten into six multimodal variants, yielding about 15K test samples to better probe whether MLLMs truly use the diagrams.
- **MathVision** [42] is a multimodal math reasoning benchmark of 3,040 competition-style problems paired with visual contexts; it spans 16 mathematical disciplines and 5 difficulty levels to enable diverse, fine-grained evaluation of LMMs.
- **MathVista** [31] is a visual math reasoning benchmark with 6,141 examples, aggregated from 28 existing multimodal math-related datasets plus three new sets (IQTest, FunctionQA, and PaperQA).

A.2 Evaluation Toolkits

For textual reasoning tasks, we conduct evaluations with the OpenCompass [11] toolkit on two recent dLLMs, Dream-7B-Instruct [51] and LLaDA-8B-Instruct [35]. For multimodal understanding tasks, we use LMMS-Eval [25] toolkit and report results on LaViDa-Reason [27].

A.3 Evaluation Setup

Textual Reasoning Task. We follow the original evaluation pipeline in LLaDA [35] using OpenCompass [11] for both LLaDA [35] and Dream [51]. We show the generation length, denoising steps and block size in Table 3.

Multimodal Understanding Task. We re-implemented the evaluation using LMMS-Eval [25] for LaViDa-Reason [27]. We follow all the hyperparameters set in the evaluation script, including the schedule and step ratio. We show the generation length, denoising steps and block size in Table 3.

Table 3: **Evaluation Setup Across All Benchmarks.** We report the detailed configuration, including generation length L, denoising steps T, and block size B, used in the main experiments.

Benchmark	Length L	Steps T	Block Size B	Benchmark	Length L	Steps T	Block Size B
MathVerse	512	512	512	HumanEval	512	512	32
MathVision	512	512	512	GPQA	128	128	64
MathVista	512	512	512	GSM8K	256	256	32
MBPP	512	512	32	MATH	512	512	64

B Decoding Safety Analysis

In this section, we provide a detailed analysis of conservative single-step for token-adaptive decoding rule. Similar to Fast-DLLM [46], we adopt the joint-PMF idealization to analyze parallel decoding in MDMs. For the decoded positions selected by our STDec at one denoising step, we show that under an aggregate error-budget condition, the token-adaptive parallel decoding result agrees with a greedy sequential decoding operating on the same selected positions. This result should be understood as a sufficient analysis evaluated on the realized decoded set, rather than an additional constraint enforced during decoding.

Setup. At denoising step t, let $C_t = \{i_1, \dots, i_n\}$ be the decoded set produced by the token-adaptive rule in Eq. 8. For each decoded position $i_j \in C_t$, the model predicts a marginal distribution $p_t(i_j)$ over the vocabulary. Let $\hat{x}_t(i_j) = \arg\max_v p_t(i_j)[v]$, $c_t(i_j) = p_t(i_j)[\hat{x}_t(i_j)]$ be the top-1 prediction and its confidence. We define the per-token error budget as $\varepsilon_j := 1 - c_t(i_j)$. The decoding rule $c_t(i) \geq \tau_t^{\text{st}}(i)$ can equivalently be written as $\varepsilon_t(i) \leq 1 - \tau_t^{\text{st}}(i)$. Following the joint-PMF idealization, assume that there exists a joint conditional distribution $p(X_{i_1}, \dots, X_{i_n} \mid E_t)$ over the decoded positions whose marginals are the model-predicted distributions $p_t(i_j)$. Here E_t denotes the current decoded context and prompt. We consider a greedy sequential decoder under the same joint PMF, and assume token-wise top-1 predictions as $x_t^\star = (\hat{x}_t(i_1), \dots, \hat{x}_t(i_n))$ obtained by independently taking the top-1 prediction at each selected position.

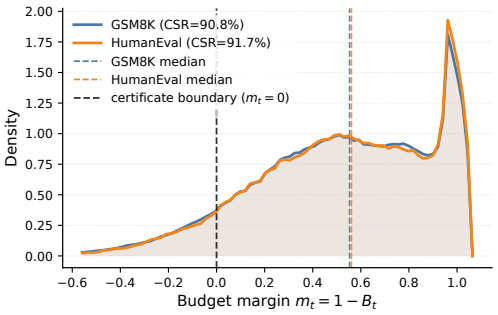


Figure 7: Budget-margin distribution over STDec decoding steps. Positive margins, computed as m_t, indicate satisfaction of the aggregate certificate condition in Eq. (B.1). Most steps on both datasets fall in the positive region.

Theorem 1 (Conservative single-step analysis)**.** *Consider the decoded set $C_t = \{i_1, \dots, i_n\}$ at step t, with the token-wise top-1 assignment $x_t^\star = (\hat{x}_t(i_1), \dots, \hat{x}_t(i_n))$ defined above. If the aggregate error budget satisfies*

$$\sum_{j=1}^{n} \varepsilon_j + \max_{1 \leq j \leq n} \varepsilon_j < 1, \tag{B.1}$$

then $x_t^\star$ is the unique mode of the joint distribution over the decoded positions. Therefore, under Eq. (B.1), the token-adaptive parallel decoding result produced by STDec agrees with greedy sequential decoding on the selected decoded set.

Proof. Let $S := \sum_{j=1}^{n} \varepsilon_j, \varepsilon_{\max} := \max_{1 \leq j \leq n} \varepsilon_j$. For each decoded position i_j, define the event $A_j := \{X_{i_j} = \hat{x}_t(i_j)\}$. Since $\Pr(A_j^c \mid E_t) = 1 - c_t(i_j) = \varepsilon_j$, the union bound gives

$$\Pr(X = x_t^\star \mid E_t) = \Pr\left(\bigcap_{j=1}^{n} A_j \,\middle|\, E_t \right) \geq 1 - \sum_{j=1}^{n} \varepsilon_j. \tag{B.2}$$

Now consider any assignment $z \neq x_t^\star$. There exists at least one coordinate k such that $z_k \neq \hat{x}_t(i_k)$. Therefore, $\{X = z\} \subseteq \{X_{i_k} = z_k\}$, and hence

$$\Pr(X = z \mid E_t) \leq \Pr(X_{i_k} = z_k \mid E_t) \leq \Pr(X_{i_k} \neq \hat{x}_t(i_k) \mid E_t) = \varepsilon_k \leq \varepsilon_{\max}. \tag{B.3}$$

Table 4: **Comparison on image generation.** We integrate our method into Lumina-DiMOO [48] for text-to-image generation.

Method	GenEval			DPGBench		
	TPS↑	Speed↑	Score↑	TPS↑	Speed↑	Score↑
Lumina-DiMOO	50.47	1.00×	0.85	49.50	1.00×	83.38
+ STDec	**178.41**	**3.53**×	0.87	**121.92**	**2.46**×	85.28

If Eq. (B.1) holds, then $1 - S > \varepsilon_{\max}$. Combining Eq. (B.2) and Eq. (B.3), we obtain $p(x_t^\star \mid E_t) > p(z \mid E_t)$ for $z \neq x_t^\star$. Thus, $x_t^\star$ is the unique mode of the joint distribution over the decoded tokens.

We prove by induction that the sequential decoder selects $\hat{x}_t(i_{\pi_k})$ at every step k. For greedy sequential decoding under the same joint PMF, assume that before step k, the decoder has selected the corresponding top-1 entries of $x_t^\star$ for the previous positions. Let $P_{k-1} := \bigcap_{\ell=1}^{k-1} \{X_{i_{\pi_\ell}} = \hat{x}_t(i_{\pi_\ell})\}$ denote this prefix event, with P_0 being the whole sample space. Since $\{X = x_t^\star\} \subseteq P_{k-1}$, we have $\Pr(P_{k-1} \mid E_t) \geq 1 - S > 0$. For the correct token, $\Pr(P_{k-1} \cap \{X_{i_{\pi_k}} = \hat{x}_t(i_{\pi_k})\} \mid E_t) \geq 1 - S$. For any competing token $v \neq \hat{x}_t(i_{\pi_k})$, $\Pr(P_{k-1} \cap \{X_{i_{\pi_k}} = v\} \mid E_t) \leq \varepsilon_{\max}$. Because Eq. (B.1) implies $1 - S > \varepsilon_{\max}$, after dividing by the same positive denominator $\Pr(P_{k-1} \mid E_t)$, the conditional probability of $\hat{x}_t(i_{\pi_k})$ is strictly larger than that of any competing token. Thus, greedy sequential decoding selects $\hat{x}_t(i_{\pi_k})$ at step k. By induction, it returns $x_t^\star$. Hence, the token-adaptive parallel decoding result produced by STDec agrees with greedy sequential decoding on C_t. □

Empirical coverage. To measure how often realized STDec decoding steps fall into this certified regime, we compute, for each decoded set,

$$B_t = \sum_{i \in C_t} (1 - c_t(i)) + \max_{i \in C_t} (1 - c_t(i)), \qquad m_t = 1 - B_t.$$

A positive margin $m_t > 0$ indicates that the aggregate condition in Eq. (B.1) is satisfied. In our experiments, STDec reaches step-level certificate satisfaction rates of 90.8% on GSM8K and 91.7% on HumanEval, showing that most decoding steps fall within the certificate-covered regime.

C Algorithm Details of STDec

Algorithm 1 outlines the overall decoding pipeline. At each denoising step, our STDec generates token-adaptive thresholds from spatial and temporal stability cues, enabling tokens with sufficiently stable predictions to be decoded earlier while preserving generation quality. To avoid rare deadlock steps where no token meets the threshold, we additionally force decoding of the most confident masked token (Line 17–19 in Algorithm 1), ensuring monotonic progress.

D Extension to multi-modal dLLMs

Our STDec extends from 1D text sequences to 2D image lattices by leveraging the same spatio-temporal stability prior. Concretely, at each denoising step, we build a pixel-wise adaptive threshold map by combining spatial cues from neighboring resolved pixels with temporal consistency of predicted pixel tokens across steps, and decode grid tokens accordingly. We plug STDec into the decoding procedure of Lumina-DiMOO [48] and evaluate on standard text-to-image benchmarks (Table 4). STDec speeds up generation by $3.53\times$ on GenEval and $2.46\times$ on DPGBench, while keeping quality scores comparable, demonstrating its effectiveness for text-to-image generation.

E Additional Ablation

E.1 Ablation of Components in our STDec on GSM8K

We conduct additional ablation studies on GSM8K with LLaDA-8B-Instruct to further validate the robustness of our STDec.

Algorithm 1 STDec: Spatio-Temporal Stability Guided Decoding

Require: model p_θ, prompt P, generation length L, steps T

Initialize masked sequence $X_T \leftarrow \{\texttt{[MASK]}\}^L$, mask set $M_T \leftarrow \{1, \ldots, L\}$
Initialize ID history $\hat{x}_{T+1}(i) \leftarrow \texttt{NULL}$, temporal stability streak $K_{T+1}(i) \leftarrow 0$ for all i
for $t = T$ **to** 1 **do**
  if $M_t = \emptyset$ **then**
    $X_0 \leftarrow X_t$
    break
  end if
  Run p_θ on (X_t, P) to obtain $\{p_t(i)\}_{i \in M_t}$
  for each $i \in M_t$ **do**
    $\hat{x}_t(i) \leftarrow \arg\max_v p_t(i)[v]$; $c_t(i) \leftarrow p_t(i)\big[\hat{x}_t(i)\big]$
    $K_t(i) \leftarrow K_{t+1}(i) + 1$ if $\hat{x}_t(i) = \hat{x}_{t+1}(i)$, otherwise $K_t(i) \leftarrow 0$
  end for
  $\tau_t^{\text{init}}(i) \leftarrow \tau_{\text{high}}$ for $i \in M_t$, and $\tau_t^{\text{init}}(i) \leftarrow \tau_{\text{low}}$ for $i \notin M_t$
  $\tau_t^{s}(i) \leftarrow f_G(\tau_t^{\text{init}}, \sigma)(i)$ for all $i \in M_t$
  $r_t(i) \leftarrow \alpha$ if $K_t(i) > 1$, $c_{t+1}(i) \geq \tau_{\text{low}}$, otherwise $r_t(i) \leftarrow 1$
  $\tau_t^{\text{st}}(i) \leftarrow r_t(i)\, \tau_t^{s}(i)$ for all $i \in M_t$
  $\mathcal{C}_t \leftarrow \{\, i \in M_t \mid c_t(i) \geq \tau_t^{\text{st}}(i) \,\}$
  if $|\mathcal{C}_t| = 0$ **then**
    $\mathcal{C}_t \leftarrow \{\arg\max_{i \in M_t} c_t(i)\}$
  end if
  $M_{t-1} \leftarrow M_t \setminus \mathcal{C}_t$, $X_{t-1} \leftarrow X_t$
  Set $X_{t-1}(i) \leftarrow \hat{x}_t(i)$ for $i \in \mathcal{C}_t$, and keep $X_{t-1}(i) \leftarrow \texttt{[MASK]}$ for $i \in M_{t-1}$
end for
return X_0

Table 5: **Smoothing function ablation for spatial smoothing in STDec.** We compare different local smoothing kernels on GSM8K and HumanEval with LLaDA-8B-Instruct, reporting throughput (TPS), TPS speedup over the baseline, and task score.

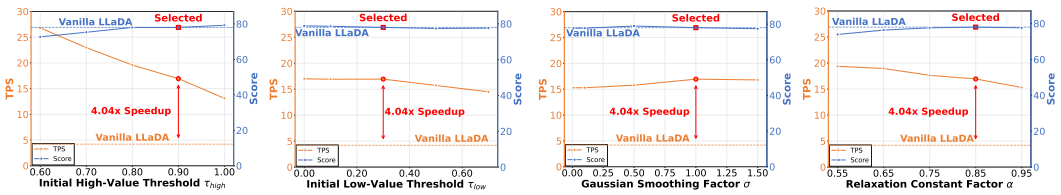
Smoothing Function	GSM8K			HumanEval		
	TPS↑	Speed↑	Score↑	TPS↑	Speed↑	Score↑
Baseline	4.19	1.00×	78.01	10.82	1.00×	48.17
Mean	18.45	4.40×	77.10	58.55	5.41×	40.85
Triangular	17.98	4.29×	77.33	57.13	5.28×	47.56
Gaussian	16.94	4.04×	78.01	52.92	4.89×	48.78

Module Ablation. Fig. 8 shows that both components contribute to acceleration on GSM8K. Temporal-aware decoding increases throughput from 4.19 to 10.1 TPS (2.41×) with a small score gain (78.01→79.08), while spatial-aware decoding reaches 14.61 TPS (3.49×) with comparable score (78.7). Combining both yields the best throughput (16.94 TPS, 4.04×) while preserving the baseline score (78.01), confirming their complementarity.

Hyperparameter Robustness. Fig. 9 reports the same hyper-parameter sweeps as in the main text. The trends closely match our primary results: moderate τ_{high} and τ_{low} balance speed and quality, σ=1.0 remains a stable default, and α=0.85 provides the strongest speed-quality trade-off. Overall, STDec is not sensitive to a narrow set of tuning choices, and the default setting transfers well to GSM8K.

E.2 Smoothing Function for Spatial Decoding

In fact, we can employ different smoothing functions used to smooth the initial threshold map in the spatial-aware module. We consider (i) a mean smoothing function that treats all neighbors within a



(a) (b) (c) (d)

Figure 9: **Impact of different hyperparameters in our STDec on GSM8K,** including masked-token factor τ_{high}, decoded-token factor τ_{low}, Gaussian smoothing factor σ, and the relaxation constant factor α. The red markers denote our default setting, which achieves a $4.04\times$ speedup over vanilla LLaDA.

radius r equally. For each position i, we compute the spatial threshold $\tau^{\text{s}}(i)$ by averaging the initial thresholds in its local neighborhood as

$$\tau^{\text{s}}(i) \;=\; \frac{1}{2r+1} \sum_{u=-r}^{r} \tau^{\text{init}}(i+u). \tag{E.1}$$

Here, r denotes the smoothing radius (window size $k = 2r + 1$), and u indexes the discrete offset within the window. We also consider (ii) a triangular smoothing function that emphasizes closer neighbors with larger weights, which can be written as

$$\begin{aligned} \tau^{\text{s}}(i) \;&=\; \sum_{u=-r}^{r} \alpha_u \, \tau^{\text{init}}(i+u), \\ \alpha_u \;&=\; \frac{r+1-|u|}{\sum_{v=-r}^{r}(r+1-|v|)}. \end{aligned} \tag{E.2}$$

Table 5 reports the results of using different smoothing functions (Mean, Triangular and Gaussian). We observe that all three smoothing functions can accelerate inference speed, demonstrating that it is useful to leverage the neighboring information for efficient decoding. However, mean smoothing (distance-agnostic) leads to a notably larger score drop than triangular or Gaussian smoothing, suggesting that distance-aware weighting is important for preserving quality. Namely, to enable more reliable decoding of tokens, smoothing operations need to consider distance information.

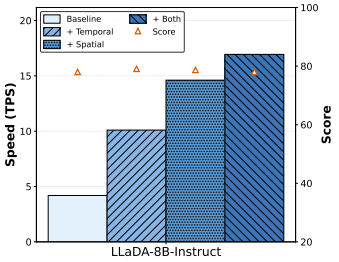


Figure 8: **Impact of integrating our STDec modules on GSM8K.** We integrate our spatial-aware decoding and temporal-aware decoding into the baseline LLaDA.

F Case Study

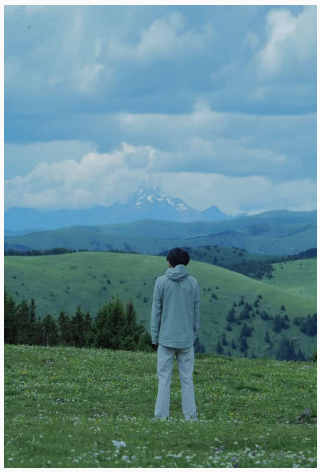
We conduct case studies on both textual reasoning and multimodal understanding tasks. In both cases, using our STDec preserves the semantic content of the generated responses while substantially reducing decoding time.

F.1 Textual Reasoning Task

Table 6 and Table 7 present examples on Dream [51] and LLaDA [35]. Compared with the vanilla model, our STDec produces consistent final answers with similar reasoning content, but decodes noticeably faster by decoding locally stable tokens earlier.

F.2 Multimodal Understanding Task

Fig. 10 shows a multimodal understanding example on LaViDa-Reason. Using our STDec keeps the same key objects and their relative relations described by the baseline, while achieving a $2.88\times$ speedup (99.17 s $\rightarrow$ 34.41 s).



User: Please describe the image in detail.
Baseline (99.17 secs): The image captures a moment of tranquility in a vast landscape. A person, clad in a gray jacket and a beanie, stands in a field of tall, green grass. The field is dotted with yellow flowers, adding a touch of color to the scene. In the distance, a mountain range stretches out, its peaks dusted with snow. The sky above is a clear blue, with a few clouds scattered across it. The perspective of the image is from the ground, looking up at the mountains, giving a sense of their grandeur and majesty. The image does not contain any discernible text. The relative positions of the objects remain constant, with the person in the foreground, the mountains in the background, and the sky at the top.
+ STDec (34.41 secs): The image captures a moment of tranquility in a vast landscape. A person, clad in a gray jacket and a beanie, stands in a field of tall, green grass. The field is dotted with white flowers, adding a touch of color to the scene. In the distance, a mountain range stretches out, its peaks dusted with snow. The sky above is a clear blue, with a few clouds scattered across it. The perspective of the image is from the ground, looking up at the mountains, giving a sense of their grandeur and majesty. The image does not contain any discernible text. The relative positions of the objects remain constant, with the person in the foreground, the field in the middle, and the mountains in the background.

Figure 10: **Case study on multimodal understanding with LaViDa-Reason [27].** Given an input image, LaViDa-Reason generates a detailed scene description. Applying our STDec preserves the semantic content and relative object relations in the response, while reducing decoding time from 99.17 s to 34.41 s (2.88× speedup).

Table 6: **Case study on GSM8K [10] with Dream-7B-Instruct [51].** Our STDec substantially reduces decoding latency while keeping the generated solution consistent with the baseline output.

Model	**Question 1:** Lily can run 12 kilometers per hour for 4 hours. After that, she runs 6 kilometers per hour. How many kilometers can she run in 8 hours?
Dream-7B-Instruct 11.59 secs	Lily can run 12 kilometers per hour for 4 hours, so she can run 12 * 4 = 48 kilometers in those 4 hours. After that, she runs 6 kilometers per hour for the next 4 hours, so she can run 6 * 4 = 24 kilometers in those 4 hours. Thus, in 8 hours, she can run 48 + 24 = 72 kilometers. So Lily can run $\boxed{72}$ kilometers in 8 hours.
+ STDec **2.30 secs** (5.03×)	Lily can run 12 kilometers per hour for 4 hours, so she runs 12 * 4 = 48 kilometers in the first 4 hours. After that, she runs 6 kilometers per hour for the next 4 hours, so she runs 6 * 4 = 24 kilometers in the next 4 hours. Thus, in 8 hours, she can run 48 + 24 = 72 kilometers. So Lily can run $\boxed{72}$ kilometers in 8 hours.
Question 2: Joy can read 8 pages of a book in 20 minutes. How many hours will it take her to read 120 pages?	
Dream-7B-Instruct 11.25 secs	To find out how many hours it will take Joy to read 120 pages, we first need to determine her reading speed in pages per minute. Joy can read 8 pages in 20 minutes, so her reading speed is 8 pages / 20 minutes = 0.4 pages per minute. Now, we need to find out how many minutes it will take her to read 120 pages. Since her reading speed is 0.4 pages per minute, it will take her 120 pages / 0.4 pages per minute = 300 minutes to read 120 pages. Finally, we need to convert the time from minutes to hours. There are 60 minutes in an hour, so 300 minutes is equal to 300 minutes / 60 minutes per hour = 5 hours. Thus, it will take Joy $\boxed{5}$ hours to read 120 pages.
+ STDec **3.59 secs** (3.13×)	To find out how many hours it will take Joy to read 120 pages, we first need to determine her reading speed in pages per minute. Joy can read 8 pages in 20 minutes, so her reading speed is 8 pages / 20 minutes = 0.4 pages per minute. Now, we need to find out how many minutes it will take her to read 120 pages. We can do this by dividing the total number of pages by her reading speed: 120 pages / 0.4 pages per minute = 300 minutes. Finally, we need to convert the time from minutes to hours. Since there are 60 minutes in an hour, we divide the number of minutes by 60: 300 minutes / 60 minutes per hour = 5 hours. So, it will take Joy $\boxed{5}$ hours to read 120 pages.

Table 7: **Case study on GSM8K [10] with LLaDA-8B-Instruct [35].** We apply our STDec to speed up decoding while preserving the original reasoning and final answer.

Model	**Question 1:** Lily can run 12 kilometers per hour for 4 hours. After that, she runs 6 kilometers per hour. How many kilometers can she run in 8 hours?
LLaDA-8B-Instruct 12.04 secs	To determine how many kilometers Lily can run in 8 hours, we need to break down the problem into two parts: the distance she runs in the first 4 hours and the distance she runs in the next 4 hours. First, let's calculate the distance she runs in the first 4 hours: - She runs 12 kilometers per hour for 4 hours. - Distance = Speed × Time - Distance in the first 4 hours = 12 km/h × 4 h = 48 km Next, let's calculate the distance she runs in the next 4 hours: - She runs 6 kilometers per hour for 4 hours. - Distance = Speed × Time - Distance in the next 4 hours = 6 km/h × 4 h = 24 km Now, we add the distances from both parts to find the total distance she can run in 8 hours: - Total distance = Distance in the first 4 hours + Distance in the next 4 hours - Total distance = 48 km + 24 km = 72 km Therefore, Lily can run a total of 72 kilometers in 8 hours.
+ STDec **2.55 secs** (4.72×)	To determine how many kilometers Lily can run in 8 hours, we need to break down the problem into two parts: the distance she runs in the first 4 hours and the distance she runs in the next 4 hours. First, let's calculate the distance she runs in the first 4 hours: - Speed: 12 kilometers per hour - Time: 4 hours - Distance = Speed × Time - Distance = 12 km/h × 4 h - Distance = 48 kilometers Next, let's calculate the distance she runs in the next 4 hours: - Speed: 6 kilometers per hour - Time: 4 hours - Distance = Speed × Time - Distance = 6 km/h × 4 h - Distance = 24 kilometers Now, we add the distances from both parts to find the total distance she can run in 8 hours: - Total distance = Distance in the first 4 hours + Distance in the next 4 hours - Total distance = 48 km + 24 km - Total distance = 72 kilometers Therefore, Lily can run **72 kilometers** in 8 hours.
Question 2: Joy can read 8 pages of a book in 20 minutes. How many hours will it take her to read 120 pages?	
LLaDA-8B-Instruct 11.94 secs	To determine how many hours it will take Joy to read 120 pages, we need to follow these steps: Step 1: Calculate Joy's reading rate in pages per minute. Joy reads 8 pages in 20 minutes. Therefore, her reading rate is: $$\frac{8 \text{ pages}}{20 \text{ minutes}} = 0.4 \text{ pages per minute}$$ Step 2: Determine the total time required to read 120 pages. Since Joy's reading rate is 0.4 pages per minute, the time required to read 120 pages is: $$\frac{120 \text{ pages}}{0.4 \text{ pages per minute}} = 300 \text{ minutes}$$ Step 3: Convert the total time from minutes to hours. There are 60 minutes in an hour, so: $$\frac{300 \text{ minutes}}{60 \text{ minutes per hour}} = 5 \text{ hours}$$ Therefore, it will take Joy $\boxed{5}$ hours to read 120 pages.
+ STDec **2.60 secs** (4.59×)	To determine how many hours it will take Joy to read 120 pages, we need to follow these steps: Step 1: Calculate Joy's reading rate in pages per minute. Joy reads 8 pages in 20 minutes. Therefore, her reading rate is: $$\text{Reading rate} = \frac{8 \text{ pages}}{20 \text{ minutes}} = 0.4 \text{ pages per minute}$$ Step 2: Determine the total time required to read 120 pages. Since Joy reads at a rate of 0.4 pages per minute, the total time required to read 120 pages is: $$\text{Total time} = \frac{120 \text{ pages}}{0.4 \text{ pages per minute}} = 300 \text{ minutes}$$ Step 3: Convert the total time from minutes to hours. $$\text{Total time in hours} = \frac{300 \text{ minutes}}{60 \text{ minutes per hour}} = 5 \text{ hours}$$ Final answer: $\boxed{5}$

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the main contributions of STDec, including spatio-temporal stability characterization, token-adaptive decoding, and efficiency–quality improvements. The claims are supported by the method, theoretical analysis, and experiments on textual reasoning, multimodal understanding, and image generation benchmarks.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses the scope of the proposed method, including its reliance on observable decoding trajectories and the fact that the theoretical certificate is sufficient rather than necessary. Empirical limitations are also reflected through ablations, risk analysis, and benchmark-specific evaluations.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The theoretical result is stated with its assumptions under the joint-PMF idealization, and the complete proof is provided in Appendix B. The theorem is explicitly presented as a conservative single-step certificate for the realized decoded set.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper specifies the evaluated models, benchmarks, decoding settings, hyperparameters, evaluation toolkits, and implementation details in Sec. 4 and Appendix A. The proposed method is training-free and is fully specified by the algorithm and thresholding rules.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes].

   Justification: The experiments use public benchmarks and publicly available model checkpoints. We publicly release the implementation code and reproduction instructions.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: STDec is training-free and does not involve optimizer settings. The paper reports evaluation benchmarks, model backbones, generation length, denoising steps, block size, decoding hyperparameters, and evaluation toolkits in Sec. 4 and Appendix A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The paper does not report error bars or confidence intervals because the main experiments follow standard deterministic benchmark evaluation protocols and repeated large-scale evaluations would be computationally expensive. Instead, we report results across multiple models, tasks, modalities, and ablation settings to assess consistency.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper reports the GPU type used for experiments and the throughput metric used for efficiency comparison. Since STDec is training-free, the main computational cost comes from inference evaluation rather than model training.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes].

   Justification: The work focuses on inference acceleration for existing dLLMs and uses public benchmarks and publicly available models. We do not collect private data, perform human-subject studies, or release unsafe datasets.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses that improving dLLM inference efficiency can reduce computational cost and broaden accessibility. It also acknowledges that faster generation may accelerate both beneficial and potentially harmful uses of generative models, so the method should be deployed with the safeguards of the underlying model.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release a new pretrained language model, image generator, or scraped dataset. We release only an inference-time acceleration method, which should be used together with the safety mechanisms of the underlying models.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The paper cites the original works for all evaluated models, benchmarks, and evaluation toolkits. We use public assets according to their intended research and evaluation settings.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: The paper does not introduce a new dataset or pretrained model. The main new asset is the implementation of STDec, which will be documented in the released code.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing experiments or research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing, user studies, or human-subject research, so IRB approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method does not use LLMs as an important, original, or non-standard component beyond evaluating existing dLLM backbones. Any use of LLMs for writing or editing does not affect the methodology, scientific claims, or experimental results.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.