Towards Efficient and Effective Diffusion Language Model Inference via Semantic-Aware Adaptive Denoising

Fan LI [1], Yu GU [1]*, Zhigang WANG [2]*, Fangling LENG [1], Zhenghao LIU [1], Ge YU [1]
[1] School of Computer Science and Engineering, Northeastern University
[2] Cyberspace Institute of Advanced Technology, Guangzhou University
lif13@mails.neu.edu.cn
{guyu, lengfangling, liuzhenghao, yuge}@mail.neu.edu.cn
wangzhigang@gzhu.edu.cn

Abstract

Diffusion language models (DLMs) have emerged as a powerful non-autoregressive alternative to GPT-style sequential generation, but suffer from substantial computational overhead due to their iterative parallel denoising. Existing acceleration works cannot accurately detect semantically stabilized tokens and then skip computation, leading to sub-optimal speedup in practice. This paper presents the first systematic study of convergence dynamics in DLMs. Innovative observations include the misalignment between traditionally used scalar detection criterion and the semantic convergence, and the post-peak confidence score, that wastes denoising computation and degrades inference quality. To address these limitations, we propose Ada-DLM, a semantic-aware adaptive denoising framework that encodes the trajectory of scalar confidence scores into an evolution-aware feature vector and then clusters vectors proactively to adaptively identify semantically converged tokens. Furthermore, we incorporate system-level optimizations to maximize runtime efficiency. Experiments show that Ada-DLM consistently outperforms the SOTA competitor, achieving up to 2× speedup and 19% quality improvement. That offers a practical path toward efficient high-quality DLM deployment.

1 Introduction

Different from the GPT-style autoregressive framework that generates output tokens sequentially from left to right (Grattafiori et al., 2024; Liu et al., 2024; Yang et al., 2025), Diffusion Language Models (DLMs) adopt a non-autoregressive, parallel denoising paradigm (Austin et al., 2021; Han et al., 2023). During inference, DLMs first produce a complete set of initial token predictions and then iteratively refine them through a multi-step **denoising** process. Under the global refinement design, DLMs are capable of capturing full-sequence contextual dependencies, which often leads to improved output coherence and quality.

However, this performance advantage comes at a substantial computational cost. Unlike autoregressive generation, where computation scales gradually with sequence length, DLMs must update the hidden states of all output tokens at every denoising step. Each step involves expensive full-sequence self-attention and feed-forward network (FFN) computations, followed by token distribution decoding. Consequently, DLM inference is significantly more demanding in compute, posing a major challenge for practical deployment.

Recent work, such as dKV-Cache (Ma et al., 2025) and Fast-dLLM (Wu et al., 2025), has explored mitigating this inefficiency by early-stopping computation for converged tokens. dKV-Cache identifies these tokens using a mechanism originally designed for controlling output display. Fast-dLLM, in contrast, employs a pre-defined, fixed confidence threshold. However, dKV-Cache's unmasking mechanism is conservatively tuned, prioritizing user experience over computational efficiency. The latter, relying on a single scalar threshold, often cannot capture token-wise semantic convergence accurately. Consequently, the potential speedup is severely limited by the inability to detect a sufficient number of converged tokens.

In this work, we present the first systematic investigation into the complex, dynamic convergence behaviors of tokens during denoising, revealing a critical limitation in existing acceleration approaches. We show both empirically and theoretically that the first-order scalar of confidence probability cannot reliably track the underlying semantic convergence. As a result, even advanced methods such as Fast-dLLM fail to timely detect up to 43.75% of semantically stabilized tokens, which fundamentally caps the achievable speedup. More importantly, we identify a previously unreported phenomenon. When converged tokens are not de-

* Corresponding authors.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 17988–18002

coded promptly, their confidence scores can decline in subsequent iterations. This not only wastes computational resources but also degrades output quality, as sub-optimal tokens with post-peak confidence are selected as the final inference result.

Inspired by the new observation, we propose Ada-DLM, a semantic-aware adaptive denoising framework that accelerates inference and boosts its quality by proactively identifying converged tokens. Unlike prior static-threshold-based passive methods, Ada-DLM actively detects semantically stable tokens in advance via a novel Rank-Adaptive Information Gain criterion. It encodes each potentially stable token as an evolution-aware feature vector formed by stacking its confidence scores from recent steps. These vectors are then clustered via a semantics-aware K-means algorithm to separate converged from non-converged tokens. A key innovation lies in our rank-aware centroid initialization. Tokens are first ranked by confidence, separated into top- and bottom-tier subsets by a binary classifier based on rank gap. Representative vectors from each subset are used as initial centroids. The resulting top-tier cluster corresponds to tokens that have semantically stabilized and then are early-stopped.

2 Related Work

Optimization strategies for Diffusion Language Models (DLMs) primarily target three dimensions, inference steps, memory access, and computation redundancy.

Orthogonal Optimizations: Trajectory & Memory. Research on *trajectory compression* minimizes denoising steps T using advanced ODE solvers (Lu et al., 2022; Karras et al., 2022; Song et al., 2023) or schedulers (Austin et al., 2021; Gong et al., 2023; Liu et al., 2025; Wang et al., 2025; Jin et al., 2025; Lou et al., 2024). Simultaneously, *memory-centric* works address Attention's quadratic cost via dynamic KV-caching (Ma et al., 2025; Jiang et al., 2025; Nguyen-Tri et al., 2025; Hu et al., 2025; Huang et al., 2025; Ma et al., 2024). Crucially, these approaches are *orthogonal* to our focus, they reduce temporal resolution or IO overhead but leave the per-step spatial computation ($L \times D$, dominated by FFNs) unoptimized. Ada-DLM can be seamlessly integrated with these methods for compound acceleration.

Token-Level Computation Pruning. Directly targeting FFN redundancy, methods like FastdLLM (Wu et al., 2025) and others (Kong et al., 2025; Chen et al., 2025; Agrawal et al., 2025) employ passive static scalar thresholds to skip tokens. However, numerical confidence often lags behind semantic convergence (Statistical Stagnation). Unlike passive static baselines, Ada-DLM reformulates pruning as a Semantics-Aware Dynamic Clustering problem. By analyzing geometric stability via Semantics-Aware Clustering, we actively recover sparse computation opportunities that rigid thresholds miss without compromising generation quality.

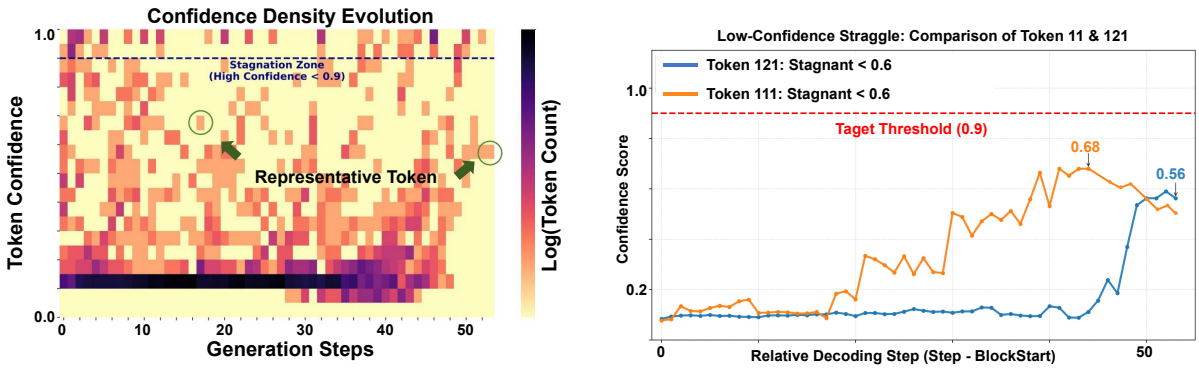
Concurrent Adaptive Inference for DLMs. Recent concurrent works have actively explored accelerating DLMs via adaptive caching or confidence thresholding. While sharing similar motivations, these methods often rely on static percentiles, contiguous left-aligned token freezing, or computationally expensive backtracking and remasking mechanisms. In contrast, Ada-DLM distinguishes itself by moving beyond instantaneous states. By leveraging multi-step temporal evolution trajectories, our framework searches globally to dynamically freeze tokens at any arbitrary position. Furthermore, the unidirectional state freezing effectively circumvents the computational overhead introduced by remasking and statistical stagnation.

3 Motivation

3.1 Diffusion Inference with Transformer Backbone

Diffusion Language Models (DLMs), such as LLaDA (Nie et al., 2025) and Dream-7B (Ye et al., 2025), generate text sequences via iterative denoising in a continuous latent space. Consider a sequence of length L at diffusion step t, denoted as $\mathbf{X}_t \in \mathbb{R}^{L \times D}$, where D is the hidden dimension. The denoiser $\mathcal{M}_\theta$ is typically implemented as a Transformer composed of N stacked layers.

For the l-th layer and the i-th token, the hidden state update consists of a Self-Attention (Attn) module and a Feed-Forward Network (FFN). Crucially, the computational cost is heavily skewed towards the FFN due to the dimension expansion (typically $4\times$ or higher) in the intermediate MLP



(a) Confidence Density Evolution

(b) Micro-Trajectory Analysis

Figure 1: The Inefficiency of Static Thresholds. (a) A heatmap shows the distribution of token confidence scores. The *Stagnation Zone* highlights a significant mass of semantically stable tokens that fail to cross the $\tau = 0.9$ threshold. The green circles mark the two sample tokens analyzed in (b). (b) Trajectories of the representative tokens. Token 121 (Blue) exhibits *Stubborn Stagnation*, remaining numerically low despite stability. Token 111 (Orange) shows *Missed Opportunity* with clear information gain ($0.08 \rightarrow 0.68$) yet is treated identically by the static strategy, causing computational waste and miss best unmasked step.

layers as Eq. 1 and Eq. 2.

$$\mathbf{h}_i^{(l,\text{mid})} = \mathbf{h}_i^{(l-1)} + \text{Attn}(\mathbf{h}_i^{(l-1)}, \mathbf{H}^{(l-1)}) \quad (1)$$

$$\mathbf{h}_i^{(l)} = \mathbf{h}_i^{(l,\text{mid})} + \underbrace{\text{FFN}(\mathbf{h}_i^{(l,\text{mid})})}_{\substack{\text{Dominant Computation} \\ (\approx \frac{2}{3} \text{ FLOPs})}} \quad (2)$$

where $\mathbf{h}_i^{(l-1)} \in \mathbb{R}^D$ denotes the input state of the i-th token, and $\mathbf{H}^{(l-1)} \in \mathbb{R}^{L \times D}$ represents the full sequence states required for the attention mechanism. The term $\mathbf{h}_i^{(l,\text{mid})}$ signifies the intermediate state following the attention residual connection. While Attention complexity scales quadratically with sequence length ($\mathcal{O}(L^2 \cdot D)$), FFNs dominate the inference cost ($\mathcal{O}(L \cdot D^2)$) for typical sequence lengths, making them the primary target for acceleration.

Finally, a projection head maps the final hidden state to the vocabulary size V as Eq. 3.

$$\mathbf{p}_{0,i}^{(t)} = \text{softmax}\left(\text{Head}(\mathbf{h}_i^{(N)})\right) \in \mathbb{R}^V \quad (3)$$

where $\mathbf{p}_{0,i}^{(t)}$ represents the predicted probability distribution over the vocabulary for the clean token $\mathbf{x}_0$ at step t.

3.2 The Bottleneck of Static Acceleration

To mitigate the computational redundancy in Eq. 2, Fast-dLLM (Wu et al., 2025) introduces a *Predict-then-Skip* paradigm. The core hypothesis is that if a token's predicted distribution $\mathbf{p}_{0,i}^{(t)}$ is sufficiently confident, its semantic representation is deemed stable. Formally, a binary computation mask $M_{t,i} \in \{0, 1\}$ is determined by a fixed threshold τ (e.g., 0.9) as Eq. 4.

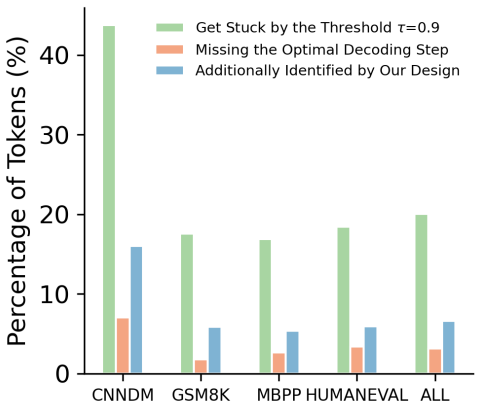


Figure 2: Quantifying Failure Modes of Static Thresholding.

$$M_{t,i} = \mathbb{I}\left(\max(\mathbf{p}_{0,i}^{(t)}) < \tau\right) \quad (4)$$

where $\mathbb{I}(\cdot)$ is the indicator function. Here, $M_{t,i} = 1$ indicates that the confidence is below τ, necessitating further refinement (Active), while $M_{t,i} = 0$ implies high confidence, allowing the block to be skipped (Inactive). In the subsequent layer, this mask gates the FFN execution as Eq. 5.

$$\mathbf{h}_i^{(l)} = \mathbf{h}_i^{(l,\text{mid})} + M_{t,i} \cdot \text{FFN}(\mathbf{h}_i^{(l,\text{mid})}) \quad (5)$$

However, we argue that relying on a *static* scalar τ creates a misalignment between numerical confidence and semantic convergence, leading to two

distinct failure modes, *Redundancy (Stagnation)*. Tokens that have converged semantically but fail to cross τ (e.g., peaking at 0.85) force the model to execute Eq. 5 with $M_{t,i} = 1$ unnecessarily. *Regret (Missed Opportunity)*. Tokens that transiently peak (e.g., 0.88) but are rejected by the rigid τ may suffer from confidence collapse in later steps, causing the optimal decoding window to be missed.

3.3 Empirical Analysis: Statistical Stagnation

The Semantic-Numerical Gap. We identify a fundamental inefficiency termed *Statistical Stagnation*, where tokens lock onto correct targets long before crossing the rigid threshold τ. Formally, let $v_{t,i}$ denote the predicted token index. We define the *Stagnation Set* as $\mathcal{S}_{stag}^{(t)} = \{i \mid v_{t,i} = v_{t-1,i} \wedge \max(\mathbf{p}_{0,i}^{(t)}) < \tau\}$. For tokens in $\mathcal{S}_{stag}^{(t)}$, the static mechanism ($M_{t,i} = 1$) forces expensive FFN execution solely to push scalar confidence (e.g., $0.7 \rightarrow 0.9$), yielding zero semantic gain. Our quantitative analysis reveals that approximately 15–20% of active computation steps reside within this *Stagnation Zone* (Figure 1a). As detailed in Figure 2, this misalignment is systemic. Micro-trajectory analysis further confirms that stubborn tokens often remain trapped in low-confidence states or suffer from *post-peak confidence decline*, degrading parallel decoding into a *quasi-serial* mode in long-context regimes.

Motivation for Rank-Adaptive Information Gain. To rectify this, Ada-DLM transitions from passive static thresholds to an active, semantics-aware approach. We reformulate token selection using a Rank-Adaptive Information Gain criterion. By encoding tokens into evolution-aware feature vectors and applying semantics-aware clustering, we effectively target $\mathcal{S}_{stag}$. This approach actively distinguishes *semantically stabilized* tokens from those with *True Uncertainty*, ensuring resources are allocated only to drive meaningful semantic evolution.

4 Methodology

This section details Ada-DLM, a mechanism-aware inference framework designed to eliminate the *Statistical Stagnation* identified in Section 3.3. Unlike existing methods that treat token pruning as a static scalar filtering task, we reformulate the parallel decoding process as a Semantics-Aware Dynamic Clustering problem. We propose an active pipeline that adapts to the evolving confidence distribution of diffusion steps. Specifically, our framework consists of two stages: (1) Rank-Aware Centroid Initialization, which acts as a binary classifier based on rank gap to isolate high-potential candidates; and (2) Semantics-Aware Adaptive Clustering, which employs a K-means mechanism to accurately separate active tokens from converged ones based on evolution-aware feature distances rather than absolute scalar values.

4.1 Problem Formulation: From Static Scalar to Dynamic Clustering

To strictly formalize the inefficiency of static baselines, we first analyze the decision boundary of current state-of-the-art methods. Let $x_{t,i}$ denote the i-th token at diffusion step t, and $C_{t,i}$ be its associated confidence score. Static paradigms implicitly model the decision boundary as a Heaviside Step Function, $\mathcal{H}$, parameterized by a fixed scalar τ (typically 0.9) as Eq. 4. We argue that this rigid formulation is mathematically suboptimal because it is *distribution-agnostic*. In diffusion models, the distribution of confidence scores shifts dramatically from high-entropy states (at $t = N$) to low-entropy states (at $t = 0$). A fixed τ fails to adapt to these shifts, forcing the model to compute tokens that are semantically stabilized relative to the current step's distribution but strictly below τ (i.e., Statistical Stagnation).

To address this, we reframe the selection process as finding an optimal partition of tokens into two latent clusters, $\mathcal{S}_{\text{active}}$ (Active) and $\mathcal{S}_{\text{frozen}}$ (Frozen). Instead of a scalar cut-off, we aim to minimize the clustering objective in a feature space $\mathcal{F}$ as Eq. 6.

$$\min_{\mu} \sum_{z \in \{\text{active, frozen}\}} \sum_{x_i \in \mathcal{S}_z} ||\mathbf{v}_{t,i} - \mu_z||^2 \quad (6)$$

where $\mathbf{v}_{t,i}$ represents the evolution-aware feature vector of token i, and μ_z denotes the dynamic centroid of cluster z. By solving this optimization problem, the decision boundary becomes a dynamic hyperplane determined by the relative distances to the High-Confidence Centroid (μ_{high}) and the Boundary Centroid (μ_{bound}).

Geometric Stability Hypothesis. Rather than relying on absolute numerical saturation, we ground our formulation in the concept of *Geometric Stability*. We posit that a token's convergence status

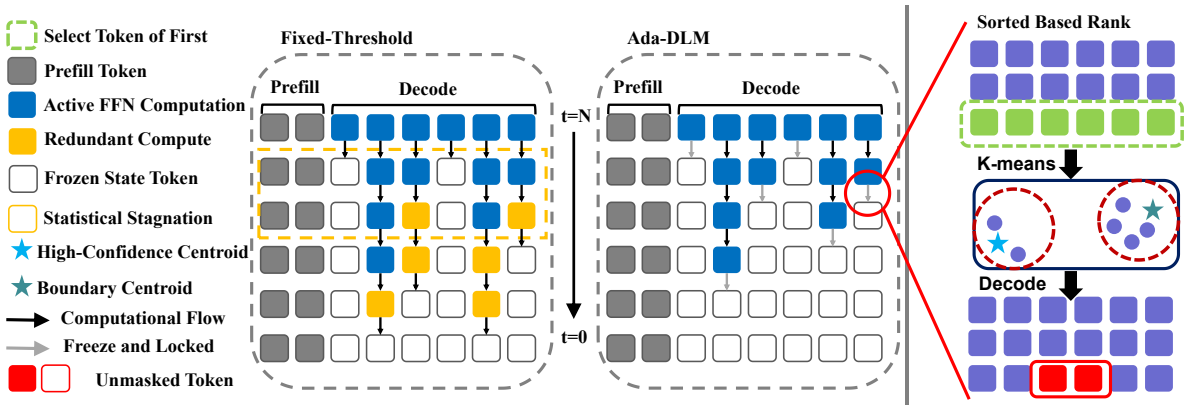


Figure 3: Schematic Comparison of Computational Flows: Static Rigidity vs. Cluster-Aware Adaptive Inference. (Left) Fixed-Threshold Paradigm: Represents SOTA static methods bounded by a rigid scalar τ. The *Yellow Blocks* highlight *Statistical Stagnation*, where semantically stable tokens are redundantly computed solely because they fail to cross the absolute confidence threshold. (Right) Our Proposed Framework (Ada-DLM): We dismantle this rigidity via a mechanism-aware two-stage strategy. (1) Rank-Based Sorting: Tokens are prioritized by confidence to isolate candidates. (2) Temporal K-Means Clustering: As shown in the zoom-in view, instead of a fixed scalar, we employ a dynamic clustering mechanism. Using the highest-confidence token (from the head) and the boundary candidate (from the tail) as dynamic centroids, and incorporating temporal consistency (confidence history over 3 steps), the model adaptively groups tokens into *Active* (Red Box) or *Frozen* states. This allows the model to skip computation for tokens clustering with the stable centroid, regardless of their absolute values.

is defined by its relative position within the confidence distribution's geometry, rather than its absolute probability magnitude. Consequently, tokens that gravitate towards the high-confidence mode of the distribution (i.e., clustering near μ_{high}) have achieved semantic stability, even if their numerical values remain unsaturated (e.g., $C_i \approx 0.7$). In this state, further refinement steps yield negligible semantic shifts. Therefore, an optimal policy π^* must decouple absolute confidence from relative geometric position, allocating computational budget only to tokens that exhibit *True Uncertainty* (i.e., deviation from the stable centroid).

4.2 Ada-DLM

We operationalize the theoretical framework established in Section 4.1 into the Ada-DLM algorithm. Functioning as a post-hoc decision module, it dynamically determines the *Active Set* $\mathcal{S}^*_{t-1}$ for the next denoising step via a three-stage pipeline.

Mechanism Overview. Figure 3 contrasts our approach with standard paradigms. While fixed-threshold policies fall into the *Stagnation Trap* (computing semantically stabilized but low-confidence tokens), Ada-DLM employs a Semantics-Aware Freezing logic. By dynamically anchoring tokens to either a high-confidence centroid or a boundary centroid, our engine issues an explicit *State Freezing* command to semantically stabilized tokens, transitioning them into a zero-cost state for all subsequent steps.

4.2.1 Rank-Aware Centroid Initialization

To efficiently initialize the dynamic clusters defined in Eq. 6, we employ a deterministic ranking strategy acting as a binary classifier. Instead of expensive iterative convergence (e.g., Lloyd's algorithm), we leverage the heavy-tailed nature of the confidence distribution to identify cluster prototypes directly. Given the token set $\mathcal{X}_t$ at step t, we first sort all tokens by their confidence scores. Let ρ be the permutation index such that $C_{t,\rho(1)} \geq C_{t,\rho(2)} \geq \cdots \geq C_{t,\rho(L)}$. We partition the sequence into a Top-Tier Subset ($\mathcal{S}_{\text{cand}}$) and a Bottom-Tier Subset ($\mathcal{S}_{\text{res}}$) based on a pre-estimated capacity K as Eq. 7 as Eq. 7.

$$\mathcal{S}_{\text{cand}} = \{x_{\rho(i)} \mid i \leq K\}, \quad \mathcal{S}_{\text{res}} = \{x_{\rho(i)} \mid i > K\} \tag{7}$$

This rank-based separation serves two critical functions utilized in the subsequent clustering phase. First, it acts as a Binary Classifier based on Rank Gap, implying a prior belief that active tokens are concentrated in the head of the distribution ($\mathcal{S}_{\text{cand}}$). Second, and most crucially, it provides instantaneous Rank-Aware Centroid Initialization. We deterministically designate the top-ranked token ($x_{\rho(1)}$) as the prototype for the High-

Confidence Centroid (μ_{high}), and the leading token of the bottom-tier subset ($x_{\rho(K+1)}$) as the prototype for the Boundary Centroid (μ_{bound}). By leveraging sorting, we transform the initialization of cluster centroids from a random process into a deterministic, data-driven selection, reducing the initialization complexity to $\mathcal{O}(L \log L)$

4.2.2 Semantics-Aware Adaptive Clustering

Relying solely on the snapshot confidence $C_{t,i}$ is susceptible to numerical instability (e.g., flashing scores), where a converged token transiently drops below the threshold due to local noise. To enforce the *Geometric Stability Hypothesis*, we introduce a temporal dimension into the feature space to capture the recent convergence trajectory.

Evolution-Aware Feature Construction. We construct a 3-dimensional evolution-aware feature vector $\mathbf{v}_{t,i} \in \mathbb{R}^3$ for each token i, formed by stacking the confidence scores from the current step and the two immediate historical steps. Formally, the vector is defined as Eq. 8.

$$\mathbf{v}_{t,i} = [C_{t,i}, C_{t-1,i}, C_{t-2,i}]^\top \tag{8}$$

where $C_{t-1,i}$ and $C_{t-2,i}$ denote the confidence scores from the preceding inference steps. Correspondingly, the centroids μ_{high} and μ_{bound} (initialized in Section 4.2.1) are expanded into this 3D space by tracking their respective history traces.

Semantics-Aware Clustering Decision. This formulation transforms the selection process into a robust 3D Clustering Problem. We solve this via a single-pass Nearest Centroid Classification, effectively implementing the Rank-Adaptive Information Gain criterion. We define the discriminant function $\mathcal{D}(i)$ as the ratio of Squared Euclidean Distances in the feature space as Eq. 9.

$$\mathcal{D}(i) = \frac{\|\mathbf{v}_{t,i} - \mu_{\text{high}}\|_2^2}{\|\mathbf{v}_{t,i} - \mu_{\text{bound}}\|_2^2} \tag{9}$$

A token is identified as semantically stabilized if it lies closer to the High-Confidence Centroid ($\mathcal{D}(i) < 1$). Geometrically, this implies that the token's 3-step trajectory aligns with the stable mode of the distribution, triggering an explicit State Freezing (removing it from $\mathcal{S}_{\text{next}}$). Conversely, tokens satisfying $\mathcal{D}(i) \geq 1$ exhibit *True Uncertainty* (clustering with the boundary) and remain Active.

Crucially, the cardinality of the final active set $|\mathcal{S}_{\text{next}}|$ is decoupled from the initialization capacity K. Unlike static Top-K pruning, our semantics-aware mechanism allows the computational budget to dynamically expand or contract based on the intrinsic distributional geometry, ensuring that the selection is driven by data characteristics rather than rigid hyperparameters.

Theoretical Justification of Semantic Evolution. To formalize the stability of a token's semantic drift, we introduce a geometric measure based on its Squared Euclidean Distance to a local stable high-confidence centroid $\mu \approx [c, c, c]^\top$ within a historical window $m = 3$. The distance can be rigorously decomposed via the sum-of-squares identity as Eq 10.

$$D^2(v_{t,i}, \mu) = 3 \cdot (\bar{C} - c)^2 + \sum_{k=0}^{2} (C_{t-k,i} - \bar{C})^2 \tag{10}$$

where $\bar{C}$ is the local average confidence of the token within the time window. This decomposition reveals the theoretical mechanism of our metric in Eq. 10: the first term, $3 \cdot (\bar{C} - c)^2$, measures and penalizes the *Magnitude Gap* from the high-confidence target. The second term strictly calculates the local variance, penalizing *Temporal Fluctuation* (approximating the first-order velocity of confidence).

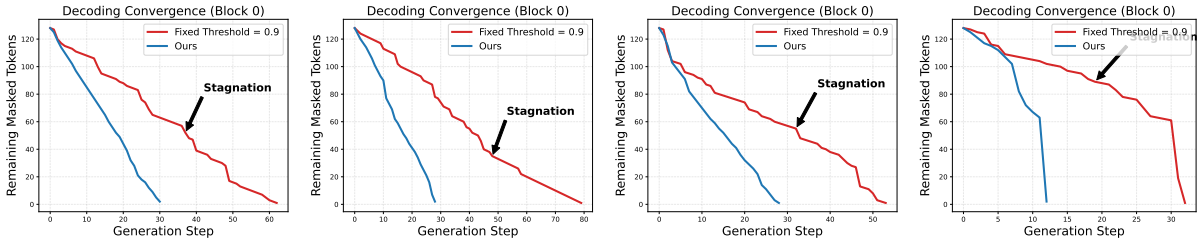
By optimizing this objective distance, Ada-DLM transforms rigid scalar thresholds into an adaptive parabolic decision boundary within the 3D feature space. It theoretically guarantees that a converged token has not only accumulated sufficient probability mass but also that its semantic drift rate has approached zero, effectively resolving the *statistical stagnation* issue.

4.2.3 Hardware-Efficient Realization

The final execution phase operationalizes the active set $\mathcal{S}_{\text{next}}$ derived from the semantics-aware clustering decision (Eq. 9). To translate the theoretical algorithmic sparsity into tangible wall-clock acceleration, we bypass naive boolean masking, which often suffers from memory fragmentation and kernel launch overheads. Instead, we implement a hardware-friendly Gather-GEMM-Scatter pipeline. First, the Gather operation compacts active tokens in $\mathcal{S}_{\text{next}}$ into a contiguous tensor via index selection, scaling linearly as $\mathcal{O}(|\mathcal{S}_{\text{next}}| \cdot D)$. Next, the GEMM step executes the Feed-Forward Network (FFN) layers solely on this compacted batch, restoring dense matrix multiplication efficiency and fully

Table 1: Performance on Standard Short-Context Benchmarks. We evaluate generation quality using task-specific metrics, Accuracy (Acc) for Reasoning (GSM8K), Pass@1 for Coding (MBPP, HumanEval), and ROUGE-L (R-L) for Summarization (CNN/DM). Ada-DLM achieves significant acceleration (up to 8.09×) while maintaining performance parity with the vanilla baseline.

Backbone	Method	Generation Quality				Efficiency (Speedup ↑)					
		GSM8K (Acc)	CNN/DM (R-L)	MBPP (Pass@1)	HEval (Pass@1)	Speed (TPS)	GSM	CNN	MBPP	HEval	Avg. Speedup
Dream-7B	Vanilla	75.40	25.38	15.00	14.70	14.7	1.00	1.00	1.00	1.00	1.00
	dKV-Cache	74.40	25.30	14.00	14.00	13.0	0.88	1.33	1.03	1.09	1.08
	Fast-dLLM	73.50	24.14	14.20	14.20	47.9	3.27	2.65	1.16	0.99	1.87
	Ours	73.50	24.25	14.80	14.90	**97.7**	**6.67**	**5.27**	**2.47**	**2.04**	**3.87**
LLaDA-8B	Vanilla	72.60	16.88	14.60	14.30	16.2	1.00	1.00	1.00	1.00	1.00
	dKV-Cache	72.90	19.25	14.70	14.20	13.3	0.82	1.54	1.06	1.15	1.13
	Fast-dLLM	72.50	19.31	14.30	14.10	94.4	5.82	3.73	1.09	1.40	2.71
	Ours	72.50	21.82	14.60	14.40	**131.1**	**8.09**	**7.13**	**2.25**	**2.62**	**4.65**



(a) MBPP (b) CNNDM (c) GSM8K (d) HUMANEVAL

Figure 4: Comparison of decoding convergence between the static baseline (Fast-dLLM, red) and our method (blue) on four distinct tasks: CNN/DM, GSM8K, MBPP, HumanEval.

leveraging Tensor Cores for high arithmetic intensity. Finally, Scatter writes the computed residuals back to the full sequence buffer to maintain positional consistency for subsequent attention layers.

Complexity Analysis. A critical requirement for acceleration is that the decision overhead must be strictly less than the computational savings. The complexity of our decision module is dominated by the Rank-Aware Initialization (Stage 1), i.e., $\mathcal{O}(L \log L)$. The subsequent feature construction and centroid distance calculation (Stage 2) operate in $\mathcal{O}(L \cdot m)$, which simplifies to $\mathcal{O}(L)$ given that the evolution-aware window $m = 3$ is a small constant. In contrast, the skipped FFN computation scales quadratically with model width, i.e., $\mathcal{O}(L \cdot D^2)$. Given that modern LLMs operate in a regime where $D \gg \log L$ (e.g., $D = 4096$), the algorithmic overhead is asymptotically negligible ($< 0.2\%$ in our profiling). Consequently, the efficiency gains scale linearly with the sparsity rate, robustly translating token reduction into latency reduction.

5 Experiments

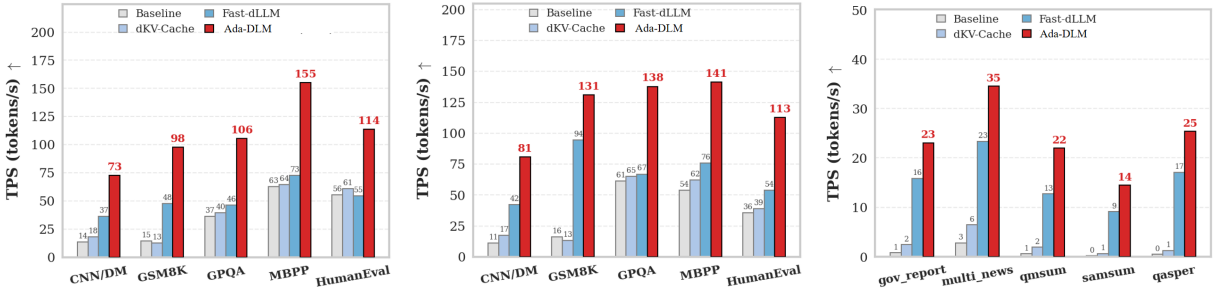
In this section, we empirically validate Ada-DLM. Beyond standard efficiency metrics, we aim to verify our core hypothesis, *does transitioning from passive scalar thresholds to active semantics-aware detection effectively identify semantically stabilized tokens that prior methods miss?* Furthermore, we investigate whether our Rank-Adaptive Information Gain criterion can prevent the generation degradation caused by the post-peak confidence decline phenomenon identified in our Introduction.

5.1 Experimental Setup

We benchmark Ada-DLM on three backbones, LLaDA-8B (Nie et al., 2025), Dream-7B (Ye et al., 2025), and UltraLLaDA (He et al., 2025). Evaluations span diverse domains including GSM8K (Cobbe et al., 2021) (Reasoning), MBPP (Chen et al., 2021) and HumanEval (Chen et al., 2021) (Coding), CNN/DM (Hermann et al., 2015) (Summarization), and the LongBench (Bai et al., 2024) suite ($L > 4$K) for long-context generalization. To ensure rigorous comparison, we

Table 2: Performance on Long-Context Benchmarks ($L \approx 4K - 32K$). We evaluate Ada-DLM using the UltraLLaDA backbone on the LongBench suite. Compared to dKV-Cache and Fast-dLLM, our method achieves a massive leap in efficiency (avg. 41.3× speedup) while maintaining or even enhancing generation quality.

Backbone	Method	Generation Quality (ROUGE-L)					Efficiency (Speedup ↑)						
		GovRep	MNews	QMSum	SAMSum	Qasper	SPS (s/sample)	Gov	News	QMS	SAM	Qasp	Avg. Speedup
UltraLLaDA	Vanilla	22.91	18.49	15.89	12.73	8.60	341.55	1.00	1.00	1.00	1.00	1.00	1.00
	dKV-Cache	23.05	18.64	16.03	14.69	9.17	119.00	2.98	2.31	3.21	3.14	2.72	2.87
	Fast-dLLM	22.79	16.95	14.20	14.58	9.83	12.81	19.66	8.41	21.09	46.30	37.80	26.65
	Ours	22.84	16.97	15.38	17.37	10.62	**8.26**	**28.60**	**12.43**	**36.47**	**73.12**	**56.07**	**41.34**



(a) Dream-7B (b) LLaDA-8B (c) UltraLLaDA (Long)

Figure 5: End-to-End Throughput Comparison. We report Token-Per-Second (TPS) across three backbones. Ada-DLM (Red) consistently achieves the highest throughput, surpassing the static method Fast-dLLM (Blue) by a significant margin. Notably, on the long-context UltraLLaDA benchmark (c), our method maintains high usability (≈ 23-35 TPS) while the vanilla baseline performance degrades to near zero.

evaluate against: (1) Vanilla Sampling; (2) dKV-Cache (Ma et al., 2025) (Infrastructure-level optimization); and (3) Fast-dLLM (Wu et al., 2025) (SOTA passive static thresholding). All methods are evaluated on NVIDIA A800 GPUs with unified KV-caching. Detailed configurations are provided in Appendix A.

5.2 Standard Short-Context Evaluation

We evaluate performance on standard benchmarks (Table 1). Ada-DLM demonstrates superior efficiency over passive baselines while preserving fidelity. Infrastructure optimization yields limited speedup (≈ 1.1×), confirming FFN layers as the primary bottleneck. By actively addressing this, Ada-DLM achieves substantial acceleration, outperforming the SOTA passive static method (Fast-dLLM) by an additional 1.55×–2.12×. Absolute acceleration averages 3.87× on Dream-7B and 4.65× on LLaDA-8B.

Analysis of Active Detection. Crucially, relative gains over Fast-dLLM are maximized on reasoning tasks (e.g., GSM8K) where confidence scores fluctuate significantly. This indicates that our Rank-Adaptive Information Gain criterion effectively identifies semantically stabilized tokens (via evolution-aware feature proximity) that passive scalars fail to capture. Convergence analysis (Figure 4) further confirms that Ada-DLM induces a steeper drop in active tokens. This validates that our geometric criterion enables earlier exit for stable regions, eliminating the redundancy inherent in rigid thresholding.

Prevention of Post-Peak Decline. Regarding quality, Ada-DLM maintains statistical parity with the full-computation baseline (e.g., 72.5% vs 72.6% accuracy on GSM8K). Notably, on open-ended tasks like CNN/DM, we observe a slight increase in ROUGE-L scores. This empirical evidence supports our hypothesis that active detection locks in optimal tokens before they suffer from post-peak confidence decline, effectively filtering out noisy tail-distribution signals that passive methods might erroneously retain.

5.3 Evaluation on Long-Context Tasks

We extend our evaluation to the LongBench suite using the UltraLLaDA backbone. As sequence length expands, the distribution-agnostic nature of passive static thresholds becomes a critical bottleneck. While Fast-dLLM provides acceleration, it exhibits diminishing returns in highly sparse regions due to its inability to adapt to shifting confidence distributions. In contrast, Ada-DLM demonstrates superior scalability, achieving a 1.93× relative speedup over Fast-dLLM on the sparse SAM-

Sum dataset.

Regarding absolute metrics, the unoptimized baseline exhibits quadratic latency growth (73× speedup on SAMSum); we emphasize this reflects Ada-DLM's capability to prevent latency collapse rather than simple linear acceleration. To assess semantic coherence in long-range dependency tasks, we evaluate GovReport. Ada-DLM yields a ROUGE-L score of 22.84, statistically comparable to the full-computation baseline. This confirms that even with aggressive semantics-aware active skipping, the model retains the critical semantic structure required for global context modeling.

Finally, regarding end-to-end throughput (Figure 5), the Vanilla baseline suffers from excessive latency in long-context regimes (SPS $\approx$ 341s). Ada-DLM maintains practical inference speeds (SPS $\approx$ 8.26s), effectively converting algorithmic sparsity into deployment-ready throughput.

K	**5%**	**10%**	**15%**	**20%**	**25%**
Acc	72.80	72.66	72.51	72.04	71.55
Speed	3.82×	4.11×	4.65×	4.95×	5.21×

Table 3: Sensitivity analysis of capacity ratio (K) on LLaDA-8B (GSM8K). The default setting ($K = 15\%$) represents the Pareto optimal point in the speed-quality trade-off space.

m	**1**	**2**	**3**	**4**	**5**
Acc	71.80	72.26	72.50	72.68	72.66
Speed	5.09×	4.88×	4.65×	4.15×	3.96×

Table 4: Sensitivity analysis of history window size (m). A window of $m = 3$ provides the minimum effective receptive field to filter out transient noise.

Metric	**Cosine**	**Manhattan**	**Sq. Euclidean**
Accuracy	64.20	70.34	72.50
Speedup	6.08×	4.21×	4.65×

Table 5: Ablation study on distance metrics. Squared Euclidean distance optimally penalizes transient deviations (temporal fluctuation) without the severe accuracy degradation seen in Cosine Similarity.

Robustness and Scalability. To ensure the statistical significance of our findings and rule out the influence of variance, we rigorously evaluated Ada-DLM across 5 random seeds on the reasoning-heavy GSM8K dataset. We observed highly stable performance (72.50 ± 0.18), confirming that our semantics-aware clustering robustly preserves generation quality without suffering from the high variance often seen in passive thresholding. Furthermore, detailed hyperparameter sensitivity analyses confirm that $K = 15\%$ (Table 3) and $m = 3$ (Table 4) act as the Pareto optimal point, providing the minimum effective receptive field to filter out transient noise. Ablation on distance metrics (Table 5) further validates that Squared Euclidean optimally penalizes temporal deviations. Finally, Ada-DLM demonstrates remarkable scalability beyond text-only architectures. Preliminary evaluations on diffusion-based Multimodal Large Language Models (MLLMs), such as LLaDA-V, yield up to 5.4× inference speedups with negligible accuracy loss, demonstrating its immense potential for handling heterogeneous token representations. Comprehensive memory footprint profiling and detailed MLLM evaluation results are provided in the Appendix.

6 Conclusion

In this work, we identified Statistical Stagnation and the associated Post-Peak Confidence Decline as structural inefficiencies inherent in passive static confidence gating. To address these, we proposed Ada-DLM, a mechanism-aware framework that reformulates parallel decoding as a Semantics-Aware Dynamic Clustering problem. By leveraging Rank-Aware Centroid Initialization and Semantics-Aware Clustering, our approach transitions diffusion inference from a passive filtering paradigm to an active detection paradigm. This effectively decouples semantic stability from absolute numerical saturation. Empirical evaluations demonstrate that Ada-DLM outperforms state-of-the-art passive baselines, achieving relative speedups of $1.55\times$–$2.12\times$ while preventing latency collapse in long-context regimes. Beyond algorithmic gains, our method contributes to sustainable computing by significantly reducing the energy footprint of generative inference. Future work will focus on developing fused CUDA kernels to further optimize the Gather-GEMM-Scatter pipeline. Additionally, we plan to explore integrating Ada-DLM with linear-attention backbones, aiming to achieve end-to-end linear scalability for ultra-long sequence generation. We plan to extend our step-level clustering mechanism to explore layer-by-layer dynamic convergence for further fine-grained acceleration.

Limitations

Currently, Ada-DLM operates as a research prototype implemented in high-level PyTorch. The absence of custom kernel fusion (e.g., integration with vLLM (Kwon et al., 2023)) implies that our reported throughput represents a *conservative lower bound* on attainable efficiency. Furthermore, while our method effectively minimizes FFN redundancy ($\mathcal{O}(LD^2)$), the backbone model's intrinsic quadratic attention complexity ($\mathcal{O}(L^2)$) remains a computational constraint. Future work will focus on integrating Ada-DLM with linear-attention architectures and developing fused CUDA kernels to achieve end-to-end linear scalability for ultra-long sequences.

Ethics Statement

This work aims to accelerate the inference of Diffusion Language Models (DLMs), thereby reducing the computational energy required for deployment. By improving the throughput-to-energy ratio, our approach aligns with the goals of Green AI and sustainable computing.

However, we acknowledge that accelerating generative models may inadvertently facilitate the rapid generation of harmful or biased content if the underlying backbone models are not properly aligned. Since Ada-DLM is a training-free inference framework that does not modify the model weights, the safety properties and biases of the generated text remain inherent to the pre-trained backbones (e.g., LLaDA, Dream-7B). We advocate for the responsible deployment of such accelerated models, accompanied by robust safety guardrails and content filtering mechanisms.

Disclosure on AI Assistance. We utilized AI-based tools (e.g., ChatGPT) solely for grammatical polishing, rephrasing, and improving textual fluency. All scientific claims, experimental designs, and empirical results presented in this paper are the original work of the authors and have been manually verified.

Acknowledgements

This work was supported by the National Key R&D Program of China (Grant No. 2023YFB4503600), the Guangdong S&T Program (Grant No. 2026B0101100002), the National Natural Science Foundation of China (Grant No. 62572108), the Key R&D Program of Shandong Province, China (No. 2023CXPT020), and the Guangdong Basic and Applied Basic Research Foundation (Grant No. 2026A1515010459).

References

Sudhanshu Agrawal, Risheek Garrepalli, Raghavv Goel, Mingu Lee, Christopher Lott, and Fatih Porikli. 2025. Spiffy: Multiplying diffusion llm acceleration via lossless speculative decoding. *arXiv preprint arXiv:2509.18085*.

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

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, and 1 others. 2024. Longbench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 3119–3137.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Zigeng Chen, Gongfan Fang, Xinyin Ma, Ruonan Yu, and Xinchao Wang. 2025. dparallel: Learnable parallel decoding for dllms. *arXiv preprint arXiv:2509.26488*.

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

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. 2024. Scaling rectified flow transformers for high-resolution image synthesis. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. 2023. Diffuseq-v2: Bridging discrete and continuous text spaces for accelerated seq2seq diffusion models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9868–9875.

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

Xiaochuang Han, Sachin Kumar, and Yulia Tsvetkov. 2023. Ssd-lm: Semi-autoregressive simplex-based diffusion language model for text generation and modular control. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11575–11596.

Guangxin He, Shen Nie, Fengqi Zhu, Yuankang Zhao, Tianyi Bai, Ran Yan, Jie Fu, Chongxuan Li, and Binhang Yuan. 2025. Ultrallada: Scaling the context length to 128k for diffusion large language models. *arXiv preprint arXiv:2510.10481*.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NIPS*, pages 1693–1701.

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

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. 2021. Argmax flows and multinomial diffusion: learning categorical distributions. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA. Curran Associates Inc.

Zhanqiu Hu, Jian Meng, Yash Akhauri, Mohamed S Abdelfattah, Jae-sun Seo, Zhiru Zhang, and Udit Gupta. 2025. Accelerating diffusion language model inference via efficient kv caching and guided diffusion. *arXiv preprint arXiv:2505.21467*.

Jianuo Huang, Yaojie Zhang, Yicun Yang, Benhao Huang, Biqing Qi, Dongrui Liu, and Linfeng Zhang. 2025. Mask tokens as prophet: Fine-grained cache eviction for efficient dllm inference. *arXiv preprint arXiv:2510.09309*.

Yuchu Jiang, Yue Cai, Xiangzhong Luo, Jiale Fu, Jiarui Wang, Chonghan Liu, and Xu Yang. 2025. d2cache: Accelerating diffusion-based llms via dual adaptive caching. *arXiv preprint arXiv:2509.23094*.

Xiangqi Jin, Yuxuan Wang, Yifeng Gao, Zichen Wen, Biqing Qi, Dongrui Liu, and Linfeng Zhang. 2025. Thinking inside the mask: In-place prompting in diffusion llms. *arXiv preprint arXiv:2508.10736*.

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

Fanheng Kong, Jingyuan Zhang, Yahui Liu, Zirui Wu, Yu Tian, Guorui Zhou, and 1 others. 2025. Accelerating diffusion llm inference via local determinism propagation. *arXiv preprint arXiv:2510.07081*.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, and 1 others. 2024. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B. Hashimoto. 2022. Diffusion-lm improves controllable text generation. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. 2022. Pseudo numerical methods for diffusion models on manifolds. In *International Conference on Learning Representations*.

Yangzhou Liu, Yue Cao, Hao Li, Gen Luo, Zhe Chen, Weiyun Wang, Xiaobo Liang, Biqing Qi, Lijun Wu, Changyao Tian, and 1 others. 2025. Sequential diffusion language models. *arXiv preprint arXiv:2509.24007*.

Aaron Lou, Chenlin Meng, and Stefano Ermon. 2024. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Proceedings of the 41st International Conference on Machine Learning*, pages 32819–32848.

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

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2024. Deepcache: Accelerating diffusion models for free. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15762–15772.

Xinyin Ma, Runpeng Yu, Gongfan Fang, and Xinchao Wang. 2025. dkv-cache: The cache for diffusion language models. *arXiv preprint arXiv:2505.15781*.

Quan Nguyen-Tri, Mukul Ranjan, and Zhiqiang Shen. 2025. Attention is all you need for kv cache in diffusion llms. *arXiv preprint arXiv:2510.14973*.

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

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

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021a. Denoising diffusion implicit models. In *International Conference on Learning Representations*.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. 2023. Consistency models. In *International Conference on Machine Learning*, pages 32211–32252. PMLR.

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

Xu Wang, Chenkai Xu, Yijie Jin, Jiachun Jin, Hao Zhang, and Zhijie Deng. 2025. Diffusion llms can do faster-than-ar inference via discrete diffusion forcing. *arXiv preprint arXiv:2508.09192*.

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. 2025. Fast-dllm: Training-free acceleration of diffusion llm by enabling kv cache and parallel decoding. *arXiv preprint arXiv:2505.22618*.

Enze Xie, Junsong Chen, Junyu Chen, Han Cai, Haotian Tang, Yujun Lin, Zhekai Zhang, Muyang Li, Ligeng Zhu, Yao Lu, and 1 others. 2024. Sana: Efficient high-resolution image synthesis with linear diffusion transformers. *arXiv preprint arXiv:2410.10629*.

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

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

A Detailed Experimental Setup

In this section, we provide granular details regarding the experimental environment, hyperparameter configurations, and baseline implementations to facilitate reproducibility.

A.1 System Environment

All experiments were conducted on a high-performance cluster equipped with NVIDIA A800 (80GB) GPUs. The software environment includes PyTorch 2.1.0 and CUDA 11.8. To simulate a realistic streaming inference scenario, all latency and throughput benchmarks report end-to-end wall-clock time (encompassing both the prefill and decode phases) with a strict batch size of 1.

A.2 Hyperparameter Configuration

For all experiments involving Ada-DLM, we utilized a consistent set of hyperparameters derived from our dynamic clustering formulation:

- History Window (m): Set to 3. This determines the dimensionality of the evolution-aware feature vector $\mathbf{v}_{t,i} = [C_{t,i}, C_{t-1,i}, C_{t-2,i}]^\top$. A window of 3 steps was empirically found to effectively smooth out numerical instability (e.g., flashing scores) without incurring significant memory overhead.

- Initialization Capacity (K): Set to approximately 15% of the sequence length. This parameter controls the Rank-Aware Centroid Initialization for the High-Confidence and Boundary Centroids. Note that K serves solely as an initialization anchor for the binary classifier; the final size of the active set is dynamic and decoupled from K.

- Distance Metric: We employ Squared Euclidean Distance for the Semantics-Aware Clustering Decision. This metric serves as a computational proxy for our Rank-Adaptive Information Gain criterion, optimized for GPU efficiency.

The total denoising steps T were configured according to the official optimal settings for each backbone model, 64 steps for Dream-7B and 100 steps for LLaDA-8B-Instruct.

A.3 Baseline Implementation Details

To ensure a fair comparison, all baselines were implemented within the same codebase and runtime environment:

- dKV-Cache: Implemented following the protocol in Ma et al. (2025). We enabled static prompt caching and dynamic decode caching. Crucially, this served as the memory infrastructure for *all* methods (including Fast-dLLM and Ada-DLM) to isolate the gains purely to FFN pruning.

- Fast-dLLM: Implemented based on Wu et al. (2025). We used the recommended passive static threshold $\tau_{\text{static}} = 0.9$. Tokens with confidence $C > 0.9$ were considered converged and skipped in subsequent steps.

B Algorithm Pseudocode

This section details the implementation of Ada-DLM. Algorithm 1 presents the unified inference pipeline, integrating the hardware-efficient sparse execution with the semantics-aware decision logic.

Algorithm 1 Ada-DLM: Semantics-Aware Adaptive Inference

Require: Sequence $\mathbf{X}_N$, Total steps T, Model $\mathcal{M}_\theta$, Capacity K, History Window $m = 3$.
Ensure: Generated token x_0.
$\mathcal{I}_{\text{active}} \leftarrow \{1, \dots, L\}$
Initialize History Buffer $\mathbf{B} \in \mathbb{R}^{L \times m}$ with zeros
$\mathbf{h} \leftarrow \text{Embed}(\mathbf{X}_N)$
for $t = T$ **downto** 1 **do**
  // Phase 1: Sparse Execution (Gather-GEMM-Scatter)
  $\mathbf{h}_{\text{active}} \leftarrow \text{Gather}(\mathbf{h}, \mathcal{I}_{\text{active}})$
  $\mathcal{L}_t, \mathbf{h}_{\text{new}} \leftarrow \mathcal{M}_\theta.\text{FFN}(\mathbf{h}_{\text{active}})$
  $\mathbf{h} \leftarrow \text{Scatter}(\mathbf{h}_{\text{new}}, \mathcal{I}_{\text{active}})$
  // Phase 2: Semantics-Aware Decision Logic
  $\mathbf{C}_t \leftarrow \max(\text{Softmax}(\mathcal{L}_t))$ ▷ Snapshot confidence
  $\mathbf{B} \leftarrow \text{UpdateBuffer}(\mathbf{B}, \mathbf{C}_t)$ ▷ Update history
  2.1 Rank-Aware Centroid Initialization (Stage 1)
  $\rho \leftarrow \text{Argsort}(\mathbf{C}_t, \text{descending})$
  $\mu_{\text{high}} \leftarrow \mathbf{B}[\rho[1]]$; $\mu_{\text{bound}} \leftarrow \mathbf{B}[\rho[K+1]]$
  2.2 Semantics-Aware Clustering (Stage 2)
  $\mathcal{I}_{\text{next}} \leftarrow \emptyset$
  for $i \in \mathcal{I}_{\text{active}}$ **do**
    $\mathbf{v}_i \leftarrow \mathbf{B}[i]$ ▷ Get evolution-aware feature
    $d_{\text{high}} \leftarrow \|\mathbf{v}_i - \mu_{\text{high}}\|^2$; $d_{\text{bound}} \leftarrow \|\mathbf{v}_i - \mu_{\text{bound}}\|^2$
    if $d_{\text{high}} \geq d_{\text{bound}}$ **then**
      $\mathcal{I}_{\text{next}}.\text{add}(i)$ ▷ Closer to boundary → Active
    end if
  end for
  // Phase 3: Update & Early Exit
  $\mathcal{I}_{\text{active}} \leftarrow \mathcal{I}_{\text{next}}$ ▷ Tokens closer to μ_{high} are frozen
  if $\mathcal{I}_{\text{active}} = \emptyset$ **then break**
  end if
end for
return $\text{Argmax}(\mathcal{L}_{\text{final}})$

C Semantic Fidelity on Complex Domains

While standard benchmarks like GSM8K measure rigorous reasoning paths, it is equally important to assess whether the accelerated decoding process preserves the semantic richness of the generated text in highly complex domains. To this end, we conduct an additional evaluation on GPQA (Rein et al., 2024), a graduate-level dataset characterized by domain-specific terminology and complex sentence structures.

Instead of treating this solely as a QA task (which primarily measures accuracy), we utilize ROUGE-1 to quantify the *Semantic Fidelity* (i.e., the extent to which the accelerated generation retains the lexical and semantic content of the ground truth compared to the vanilla baseline). This metric serves as a proxy for measuring information loss during token pruning.

Analysis. As shown in Table 6, Ada-DLM demonstrates robust semantic stability. On the Dream-7B backbone, our method achieves a ROUGE-1 score of 25.25, statistically comparable to the Vanilla baseline (25.53). Notably, on the LLaDA-8B backbone, Ada-DLM attains a score of 16.51, outperforming both the Vanilla baseline (16.28) and the passive Fast-dLLM (14.50). This result suggests that our Semantics-Aware Clustering mechanism is particularly effective at differentiating between semantically stabilized cores and necessary refinements. Unlike passive static thresholding (Fast-dLLM) which blindly truncates based on scalar values (leading to over-pruning), our geometric criterion preserves structurally critical tokens while filtering out noisy tail-distribution signals caused by post-peak confidence decline, thereby enhancing lexical precision in complex sequences.

D Comprehensive Memory Footprint Analysis

As discussed in the main text, Ada-DLM introduces an evolution-aware feature vector to track the temporal trajectory of token confidence. To address potential concerns regarding the space complexity of this mechanism, we theoretically and empirically analyzed its memory footprint.

Theoretically, given an input sequence of length L and a constant historical window size $m = 3$, the spatial complexity of the additional tracking

Table 6: Semantic Fidelity Analysis on GPQA. We report ROUGE-1 scores to evaluate the preservation of semantic content. Ada-DLM maintains high semantic overlap with the ground truth, particularly on LLaDA-8B (16.51), surpassing both baselines.

Method	**Dream-7B**	**LLaDA-8B**	**Avg. Speedup**
Baselines			
Vanilla Sampling	25.53	16.28	1.00×
dKV-Cache	25.08	15.75	1.10×
Fast-dLLM	25.62	14.50	2.29×
Ada-DLM (Ours)	**25.25**	**16.51**	**4.26×**

vectors strictly converges to an extremely low value of $\mathcal{O}(L)$.

Empirically, we measured the KV cache memory and peak memory consumption during inference across different backbone models and context lengths. As shown in Table 7, the additional memory required by Ada-DLM's tracking mechanism is negligible (e.g., less than 0.02 MB for 8B models with short contexts, and only +0.19 MB even when the prompt length scales to 16K in UltraLLaDA). The relative memory increase compared to the vanilla and static baseline (Fast-dLLM) is strictly below 0.01%, confirming that our significant acceleration does not come at the expense of VRAM consumption.

E Extension to Diffusion-Based MLLMs

To demonstrate the generalizability of our framework beyond text-only architectures, we conducted extensive evaluations on several state-of-the-art diffusion-based Multimodal Large Language Models (MLLMs), including LLaDA-V (8B), Dimple-7B, and MMAda-8B.

We evaluated the generation quality and acceleration ratios across three highly complex multimodal reasoning benchmarks: MathVista, LLaVA-Bench, and MathVerse. For a more rigorous evaluation, we additionally included the recently proposed *dLLM-Cache* as a strong concurrent baseline.

As reported in Table 8, Ada-DLM consistently outperforms all baselines, achieving significant acceleration rates (up to 5.44× on Dimple-7B and 5.40× on MMAda-8B) while preserving high evaluation scores across all multimodal benchmarks. This confirms that our semantic-aware clustering approach is highly robust and can effectively manage the heterogeneous token representations inherent in vision-language tasks.

F Implementation Details of Non-Iterative Clustering

To ensure the decision overhead does not eclipse the computational savings, Ada-DLM strictly bounds the time complexity of the clustering algorithm. The standard K-means algorithm is computationally expensive, exhibiting a time complexity of $\mathcal{O}(I \cdot K \cdot L)$ for I iterations.

Our framework adopts a non-iterative variant where the centroids (the high-confidence centroid and the boundary centroid) are deterministically initialized via the rank-aware sorting mechanism, meaning only a single pass of distance calculation is required. This drastically reduces the time complexity to $\mathcal{O}(L \log L)$. Coupled with the hardware-specific `Gather-GEMM-Scatter` operators, the runtime overhead of the non-iterative clustering accounts for less than 0.2% of the total inference cost.

Furthermore, the Ada-DLM technique is highly decoupled from the underlying Transformer architecture. It is encapsulated as a set of modular API functions—namely, `rank_aware_init`, `semantic_clustering`, and `gather_gemm_scatter`. These high-level functions respectively enable deterministic prototype initialization without iterative overhead, dynamic separation of active tokens based on evolution trajectories, and hardware-efficient dense execution that circumvents memory fragmentation. This modular design insulates programmers from tedious low-level details, allowing ready integration with future diffusion-based architectures.

Model (Size)	Avg. Prompt	Method	KV Cache (MB)	Peak Memory (MB)
Dream (7B)	512	Vanilla	0.00	14950.40
		dKV-Cache	28.67	14979.07
		Fast-dLLM	28.67	14979.07
		Ada-DLM (Ours)	**28.68** ***(+0.01)***	**14979.08**
LLaDA (8B)	512	Vanilla	0.00	17059.84
		dKV-Cache	266.24	17305.60
		Fast-dLLM	266.24	17305.60
		Ada-DLM (Ours)	**266.25** ***(+0.01)***	**17305.61**
LLaDA-V (8B)	864	Vanilla	0.00	19671.04
		dKV-Cache	522.24	20162.56
		Fast-dLLM	522.24	20162.56
		Ada-DLM (Ours)	**522.25** ***(+0.02)***	**20162.58**
UltraLLaDA	16K	Vanilla	0.00	40376.32
		dKV-Cache	8192.00	48128.00
		Fast-dLLM	8192.00	48128.00
		Ada-DLM (Ours)	**8192.19** ***(+0.19)***	**48128.19**

Table 7: Detailed memory cost comparisons during inference. The extra memory required by Ada-DLM is strictly bounded and negligible across all tested context lengths.

Backbone	Method	MathVista	LLaVA-Bench	MathVerse	Avg Speedup
LLaDA-V	Vanilla	59.76 (1.00×)	70.45 (1.00×)	28.54 (1.00×)	1.00×
	dKV-Cache	58.91 (1.33×)	70.15 (1.28×)	28.33 (1.42×)	1.34×
	Fast-dLLM	57.60 (2.43×)	69.38 (2.42×)	27.91 (2.36×)	2.40×
	dLLM-Cache	58.21 (4.43×)	69.58 (4.42×)	28.41 (3.36×)	4.07×
	Ada-DLM	**59.39 (5.01×)**	**70.13 (4.66×)**	**28.57 (3.59×)**	**4.42×**
Dimple-7B	Vanilla	42.17 (1.00×)	64.94 (1.00×)	23.82 (1.00×)	1.00×
	dKV-Cache	42.05 (1.45×)	64.53 (0.95×)	23.64 (1.18×)	1.19×
	Fast-dLLM	42.00 (2.16×)	63.98 (3.06×)	22.99 (2.99×)	2.73×
	dLLM-Cache	42.09 (4.43×)	64.37 (5.42×)	23.60 (4.36×)	4.74×
	Ada-DLM	**42.09 (4.85×)**	**64.88 (6.13×)**	**23.71 (5.34×)**	**5.44×**
MMAda-8B	Vanilla	47.32 (1.00×)	69.78 (1.00×)	25.45 (1.00×)	1.00×
	dKV-Cache	47.28 (1.09×)	69.03 (0.88×)	25.11 (1.23×)	1.06×
	Fast-dLLM	46.65 (3.15×)	68.56 (2.96×)	24.68 (2.48×)	2.86×
	dLLM-Cache	47.14 (4.83×)	69.55 (5.36×)	24.91 (4.80×)	4.99×
	Ada-DLM	**47.20 (5.23×)**	**69.56 (6.07×)**	**25.32 (4.92×)**	**5.40×**

Table 8: Performance and acceleration comparison on diffusion-based MLLMs. The reported results are in the format of "Accuracy (Speedup)".