SKIP TO THE GOOD PART: REPRESENTATION STRUCTURE & INFERENCE-TIME LAYER SKIPPING IN DIFFUSION VS. AUTOREGRESSIVE LLM

Anonymous authors
Paper under double-blind review

ABSTRACT

Autoregressive (AR) language models form representations incrementally through left-to-right prediction, whereas diffusion language models (dLLMs) are trained via full-sequence denoising. Although recent dLLMs match AR performance, it remains unclear whether diffusion objectives fundamentally reshape internal representations across depth. We perform the first layer- and token-wise representational analysis comparing native dLLMs (LLaDA), native AR models (Qwen2.5), and AR-initialized dLLMs (Dream-7B). We find that diffusion objectives result in different, more hierarchical abstraction with substantial early-layer redundancy and reduced recency bias, while AR objectives produce tightly coupled, depth-dependent representations. Critically, AR-initialized dLLMs retain AR-like representational dynamics despite diffusion training, revealing persistent initialization bias. Leveraging this observed representational redundancy, we introduce a static, task-agnostic inference-time layer-skipping method requiring no architectural changes or KV-cache sharing. Native dLLMs achieve up to 18.75% FLOPs reduction while preserving over 90% performance on reasoning and code generation benchmarks, whereas AR models degrade sharply under comparable skipping. These results link training objectives to representational structure and enable practical, cache-orthogonal efficiency gains.

1 INTRODUCTION

Transition from Next-Token Prediction (NTP) to diffusion changes how LLMs learn: AR models trained with NTP refine predictions left-to-right, whereas diffusion LLMs (dLLMs) iteratively denoise full-sequence states. Recent discrete diffusion formulations replace AR factorization with stepwise refinement over the entire sequence. Although dLLMs such as LLaDA Nie et al. (2025) and DiffuCoder Shansan Gong (2025) now approach AR-level performance, we still lack a clear understanding of how their internal representations differ. In particular, it is unknown whether full-sequence training fundamentally alters where and how abstraction emerges across depth.

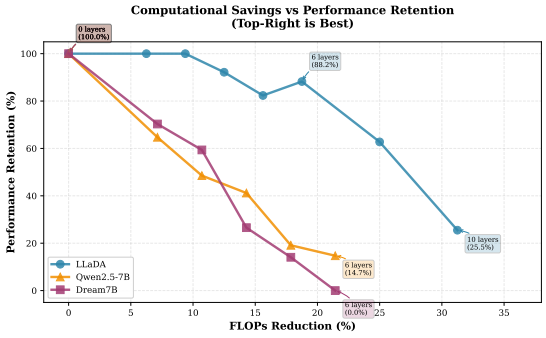


Figure 1: **Representational redundancy enables efficient inference-time layer skipping.** We evaluate zero-shot layer pruning across three architectures: LLaDA (diffusion LLM), Dream7B (dLLM initialized from Qwen2.5), and Qwen2.5-7B (AR-LLM). The diffusion-based LLaDA exhibits remarkable robustness, retaining 88.24% performance at 18.75% FLOPs reduction (6 layers skipped), demonstrating significant representational redundancy. Conversely, autoregressive models show brittle behavior with only 64.71% retention at 7.14% FLOPs reduction (2 layers), revealing concentrated, non-redundant representations. Top-right region indicates optimal performance-efficiency trade-off.

Most prior work on dLLMs emphasizes inference-time efficiency—parallel decoding, KV-cache reuse, and architectural optimizations—while their representational geometry remains underexplored.

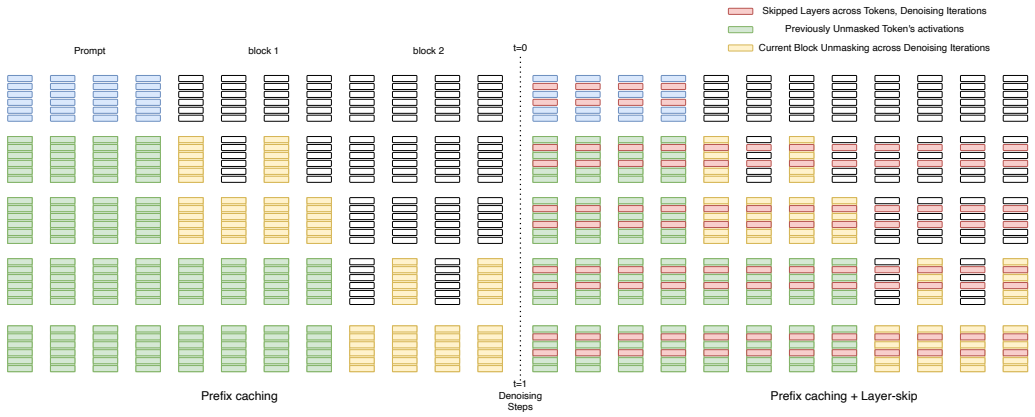


Figure 2: **Layer-skip mechanism for dLLMs.** At each denoising step, high-similarity layers (shaded) are bypassed, with hidden states passed directly to the next active layer. This reduces per-step FLOPs while preserving the coarse-to-fine abstraction hierarchy.

Initial studies Shansan Gong (2025) have begun probing local versus global features in dLLMs, but a systematic comparison of how diffusion objectives shape layer-wise abstraction relative to AR training is still missing. Such understanding is important: training objectives, like architectural choices in ResNets, can materially reshape optimization and information flow.

We hypothesize that training objective and initialization jointly shape an LLM's internal geometry. Full-sequence diffusion feedback may promote earlier or more global abstraction, potentially yielding inference-time redundancy. Unlike cache-centric methods such as YOCO, our focus is orthogonal: does objective-induced redundancy alone enable inference-only layer skipping, without KV-cache sharing or parameter tying?

To isolate objective and initialization effects, we compare three families: a native dLLM (LLaDA), a native AR model (Qwen2.5), and an AR-initialized dLLM (Dream-7B). This setup tests whether diffusion training can override AR-induced structure. Our layer- and token-level analysis shows that Dream-7B remains closer to Qwen2.5 than to LLaDA, suggesting persistent initialization bias. Motivated by this, we evaluate whether diffusion-induced redundancy can support static, task-agnostic layer skipping—without KV sharing—achieving measurable speedups with minimal degradation. Below we summarize our contributions.

- *Representational analysis revealing objective-induced redundancy:* We present the first systematic layer-wise and token-wise similarity analysis comparing native dLLMs, AR models, and AR-initialized dLLMs. We demonstrate that diffusion objectives induce stronger hierarchical abstraction with concentrated early-layer redundancy and minimal recency bias, while AR objectives maintain incremental token-by-token refinement throughout depth with strong recency bias. We further reveal strong initialization bias: AR-initialized dLLMs (Dream-7B) retain AR-like representational patterns despite diffusion training.

- *Inference-time layer skipping:* Leveraging objective-induced representational redundancy, we introduce a static, task-agnostic layer-skip policy that requires no KV-cache sharing and no architectural modifications. Native dLLMs achieve up to 18.75% FLOPs reduction with minimal accuracy loss ($< 10\%$ degradation on average), while AR models show brittleness.

- *Cross-domain benchmarking* We evaluate across reasoning (GSM8K, MATH-500) and code synthesis (HumanEval, MBPP) tasks, demonstrating consistent patterns: native dLLMs tolerate aggressive layer skipping (6+ layers with $> 90\%$ retention), AR-initialized dLLMs show intermediate robustness (2-4 layers), and native AR models are brittle (substantial degradation at 2 layers).

2 LAYER-WISE AND TOKEN-WISE SIMILARITY ANALYSIS

Motivation: To understand how training objectives shape internal representations and induce redundancy patterns, we examine layer-to-layer similarity across dLLMs and AR models. Unlike AR models that build representations incrementally through left-to-right token prediction, dLLMs receive full-sequence gradient feedback during training, potentially leading to different abstraction pathways and redundancy structures across depth.

We hypothesize that this objective-level difference manifests as measurable representational redundancy that can be exploited for inference-time efficiency gains without architectural modifications or KV-cache sharing. Given that 'coarse-to-fine' generation is a common motivation and hypothesis for diffusion models, we take a first step in analyzing the rate of change of representations (within the shared embedding space) across both layers and tokens as illustrated in Fig. 4 and Fig. 5 for cosine similarity across tokens.

Methodology: We track the cosine similarity between consecutive layer representations $\mathbf{h}_\ell$ and $\mathbf{h}_{\ell+1}$ across all tokens in a sequence. For dLLMs, we compute this similarity at multiple denoising steps $t \in \{1, \ldots, T\}$; for AR models, we compute it during standard forward passes. We aggregate statistics across diverse prompts from our evaluation benchmarks.

1) Coarse-to-fine abstraction in native dLLMs: For LLaDA, from Fig. 4 the layer-wise similarity pattern remains largely consistent across denoising steps, suggesting a pronounced representational abstraction hierarchy. Early layers establish coarse representations with high inter-layer similarity (plateau regions with cosine similarity > 0.95), while later layers and denoising steps perform iterative refinement rather than restructuring the representation at each step. This hierarchical organization indicates potential redundancy in early layers that can be exploited at inference time.

2) Recency bias and Global vs. Local representations: Token-wise analysis reveals striking differences in representational dynamics as shown in Figure 5. LLaDA exhibits minimal recency bias with smooth, high-similarity transitions across all tokens and layers, indicating global representational abstraction.

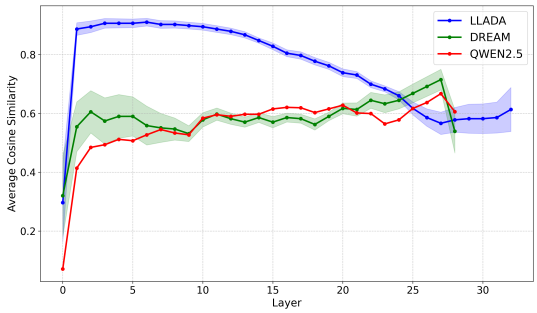
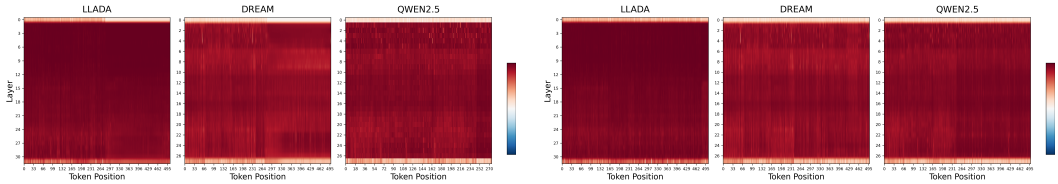


Figure 3: **Average token-wise cosine similarity across layers and denoising steps.** LLaDA (native dLLM) exhibits high similarity (> 0.9) in early layers with smooth transitions, followed by lower similarity in later layers where refinement occurs. Dream-7B closely follows Qwen2.5's pattern despite diffusion training, revealing persistent initialization bias. Shaded regions show standard deviation across denoising steps for LLaDA and Dream-7B.

In contrast, both Dream-7B and Qwen2.5 demonstrate significant recency bias—representations change substantially for each new token across all layers. Notably, in LLaDA, recency bias emerges primarily in later layers (which begin to act more like decoder layers), whereas in Dream-7B and Qwen2.5, recency bias is prominent across all layers and tokens. This suggests that AR-style models maintain consistent token-by-token representational updates throughout depth, indicating less hierarchical abstraction compared to native dLLMs.

Our representational similarity analysis complements the behavioral analysis in Shansan Gong (2025), which measures AR-ness through generation patterns (local consecutive next-token prediction and global earliest-mask selection). While their metrics capture *output-level* generation strategies—whether models follow left-to-right filling patterns—our layer-wise and token-wise cosine similarity analysis reveals *internal representational dynamics*: how hidden states evolve across depth and tokens.

Critically, we find that Dream-7B exhibits AR-like recency bias in its *representations* (mirroring Qwen2.5's token-by-token updates) despite being trained with diffusion objectives, providing mechanistic evidence for initialization bias that persists beyond surface-level generation behavior. This



(a) 32 tokens decoded

(b) All tokens decoded

Figure 4: **Layer-wise cosine similarity across models.** Each row shows similarity between consecutive layers for (top) LLaDA, (middle) Qwen2.5, and (bottom) Dream-7B. High-similarity regions (yellow) indicate representational redundancy. Dream-7B's pattern closely resembles Qwen2.5 despite diffusion training, revealing strong initialization bias.

representational perspective explains *why* certain models exhibit AR-like generation patterns and reveals that initialization effects run deeper than decoding strategies alone.

Recency Bias & Representational Abstraction

Diffusion objectives reduce recency bias and promote hierarchical abstraction: LLaDA shows minimal recency bias with global representations across tokens, while AR models (Qwen2.5, Dream-7B) exhibit strong recency bias at all layers. This suggests diffusion training encourages coarse-to-fine abstraction, whereas AR training maintains incremental, token-by-token updates throughout network depth.

3) Strong initialization bias in AR-adopted dLLMs: Despite being trained with a diffusion objective, Dream-7B's similarity profile—both layer-wise and token-wise—closely mirrors that of its AR initialization (Qwen2.5), with high-similarity regions and recency patterns appearing in nearly identical layer ranges.

As shown in Figure 3, Dream-7B's average token-wise cosine similarity across layers follows Qwen2.5's pattern remarkably closely throughout the network depth, despite undergoing diffusion training. In contrast, LLaDA exhibits a distinctly different profile: it begins with very high similarity (> 0.95) in early layers, indicating redundant representations with smooth transitions, then transitions to lower similarity in later layers where refinement occurs. This hierarchical pattern—high redundancy in early layers, active refinement in later layers—is characteristic of native diffusion training and absent in AR-initialized models. This highlights the strong regularization effect of initialization on resulting representations and abstractions, persisting even after significant fine-tuning with diffusion-based objectives.

Initialization Effect

AR initialization creates persistent representational structure: Dream-7B, despite diffusion training, retains Qwen2.5's similarity patterns and recency bias, demonstrating that pre-trained AR representations strongly regularize subsequent diffusion fine-tuning. Native dLLMs (LLaDA) develop fundamentally different abstraction hierarchies.

Magnitude Evolution One potential limitation of cosine similarity is its invariance to magnitude. To ensure our redundancy findings are not artifacts of magnitude collapse, we analyze the ℓ_2 norm of hidden states across layers as observed in Fig. 6.We observed that the magnitude evolution is small for initial 60-70% layers and then rises steeply. There is also presence of sink tokens – super high magnitude than the rest of the tokens, as discussed in Rulli et al. (2025).

Together, these analyses validate cosine similarity as a meaningful proxy which demonstrate potential redundancy in representations and motivate our inference-time layer-skipping strategy.

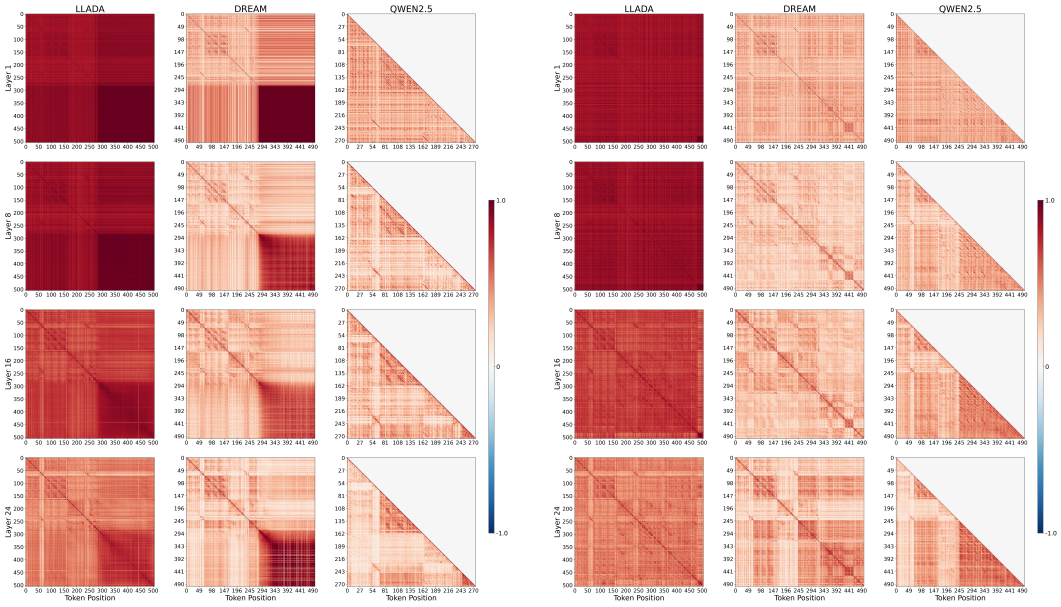


Figure 5: **Token-wise cosine similarity across layers and models.** Rows correspond to layers $(1, 8, 16, 24)$; columns show (left) LLaDA, (middle) Dream-7B, and (right) Qwen. Left: decoding limited to 32 tokens highlights early representational stabilization in native diffusion models. Right: full-sequence decoding emphasizes global context integration and architectural differences across objectives.

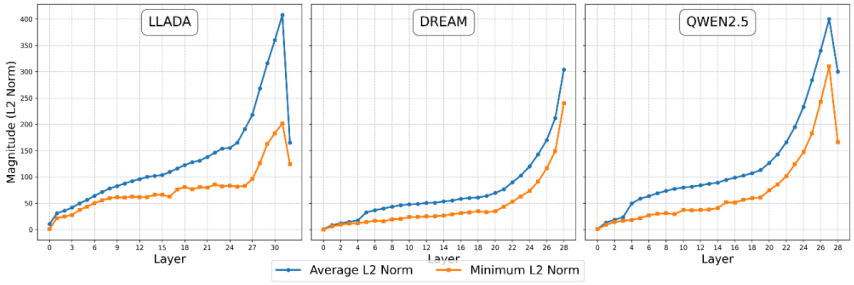


Figure 6: **Hidden-state magnitude across depth.** Layer-wise evolution of the ℓ_2 norm of token hidden states for LLaDA, Dream, and Qwen. Norms remain relatively stable through the first $\sim$60–70% of layers and increase sharply near the top of the network. The *maximum* norm is dominated by rare *sink tokens* (spikes; often $\geq 10^3$), so max values should be interpreted as outliers rather than typical token magnitudes.

3 Layer-skip at Inference

The observed high-similarity plateaus suggest that certain layers contribute minimally to representational transformation. We hypothesize that *skipping* these layers at inference time can reduce computational cost with minimal impact on task performance. We demonstrate with and without layer-skipping and prefix caching in Fig. 2

Crucially, our approach is:1) *Static and task-agnostic:* We identify skip-eligible layers based on training-time similarity analysis, without per-task tuning or dynamic routing.2) *Architecture-agnostic:* Unlike YOCO-style methods that require cache-once designs or parameter sharing, our method applies to any pretrained model without modification.3) *Complementary to KV-caching:* Layer skipping reduces FLOPs and depth; KV-caching reduces memory and redundant computation across tokens. Both can be combined for compounded benefits.

Skip policy. Based on Figure 4, we define a skip set $\mathcal{S} \subset \{1, \ldots, L\}$ containing layers with consecutive similarity $> \theta$ (we use $\theta = 0.95$ by default). During inference, for each layer $\ell \in \mathcal{S}$, we bypass the transformer block and directly pass $\mathbf{h}_{\ell-1}$ to layer $\ell + 1$. Residual connections ensure gradient flow and representational continuity. Our algorithm is shown in Appendix Section 8 due to space constraint.

Quality vs. Efficiency Hypothesis: Minimal degradation for dLLMs, High similarity indicates redundancy; skipping should preserve task performance. Larger degradation for AR models as AR models may rely more on incremental refinement, making layer skipping more disruptive.

4 Experimental Setup

Models. We evaluate three families to disentangle training objective and initialization: (i) a native diffusion LLM, **LLaDA** (we use the 8B Base / Instruct checkpoints) (Nie et al., 2025); (ii) a native autoregressive (AR) model, **Qwen2.5** (7B Base / Instruct) (Yang et al., 2024); and (iii) an AR-initialized diffusion LLM, **Dream-7B** (Instruct) (Ye et al., 2025). Unless stated otherwise, all models are evaluated with their public inference code and default tokenizers.

Benchmarks. We measure reasoning and code synthesis across standard suites: **GSM8K** (grade-school math; exact-match accuracy) (Cobbe et al., 2021); **HumanEval** (function-level Python synthesis; pass@k using the official harness) (Chen et al., 2021); **MATH-500** (the 500-problem test subset of the MATH benchmark; exact-match accuracy) (Hendrycks et al., 2021).

Prompting and answer extraction. For **GSM8K**, we use a few-shot rationale prompt with an explicit `Final Answer:` line; we strip formatting and compare normalized numbers. For **HumanEval** and **MBPP**, we request a single Python function and evaluate with the official test suites; we report pass@1 and pass@k. The same prompts are used across models to ensure comparability.We follow exact inference setting of Chen & Liu (2026)

Decoding & sampling. For **AR** decoding (Qwen2.5), we use greedy or nucleus sampling (default `top_p`=0.95, `temperature`$\in \{0.2, 0.7, 0.8\}$ depending on task), `max_new_tokens`=2048, and early stopping on task-specific end markers. For **diffusion** decoding (LLaDA, Dream-7B), we follow each repository's default sampler/schedule and report quality–latency tradeoffs across denoising budgets $T \in \{16, 32\}$; other settings (e.g., temperature annealing or remasking) follow the public implementations (Nie et al., 2025; Ye et al., 2025). To compare fairly to AR decoding, we standardize context limits (2,048 tokens total) and stop rules.

Layer-skipping evaluation (ours). To isolate objective-induced redundancy, we introduce a *static, task-agnostic* top-k layer skip policy applied only at inference time, without KV sharing or architectural changes. We evaluate $k \in \{0, 2, 4, 6\}$ on 7–8B models. Metrics include: task score (GSM8K accuracy; HumanEval/MBPP pass@1 and pass@k; MATH-500), *equivalence rate* (fraction of generations identical to full-depth outputs), token-level KL divergence (w.r.t. the full-depth distribution; optional), and end-to-end latency/throughput (prefill+decode or diffusion steps).

Note: Due to varying inference optimizations across model implementations (different kernel implementations), absolute throughput (tokens/sec) is not directly comparable between models. We therefore report: (1) the number of layers skipped for each model, and (2) the resulting task performance relative to full-depth baselines. This isolates the effect of our layer-skip strategy from architecture-specific optimizations.

5 Results

Native dLLMs Enable Aggressive Layer Skipping. Table 1 and Figure 1 show that native diffusion LLMs are markedly more robust to layer skipping than autoregressive models. For LLaDA, skipping 6 layers (18.75% FLOPs reduction) preserves 88.2–102.1% of baseline performance across all evaluated tasks. Even at an 8-layer skip (25% FLOPs reduction), performance retention remains high (62.7–91.8%), placing LLaDA firmly in the favorable efficiency–quality regime (top right of Figure 1).

In contrast, autoregressive models degrade rapidly under similar interventions. Qwen2.5 experiences severe performance collapse: skipping only 2 layers (7.14% FLOPs reduction) reduces retention to 34.9–75.3%, indicating that AR representations are tightly coupled and lack depth-wise redundancy. Notably, Dream-7B—despite diffusion fine-tuning—exhibits similar brittleness. At a 2-layer skip, Dream-7B achieves only 60.5–81.4% retention, closely tracking Qwen2.5 and substantially underperforming LLaDA (100–123.1%). This provides strong evidence that autoregressive pretraining induces a persistent representational structure that diffusion objectives alone do not override, preventing the emergence of the coarse-to-fine hierarchy observed in native dLLMs.

Computational savings: Native dLLMs achieve **2.6× greater FLOPs reduction with 1.4× higher quality retention**. Importantly, these savings are orthogonal to KV-caching: layer skipping reduces depth-wise computation, while KV-caching eliminates token-wise redundancy, enabling multiplicative gains when combined.

Table 1: Performance comparison across models and layer skipping configurations. Values represent retention percentages relative to baseline (0 layers skipped), with absolute accuracy shown in parentheses for baseline rows. Missing entries (−) indicate performance below retention threshold. Higher values (↑) indicate better retention.

Benchmark	Layers Skipped	Llada-Instruct	Dream-Base	Dream-Instruct	Qwen2.5
GSM8K	0 (baseline)	100.0 (0.83)	100.0 (0.77)	100.0 (0.81)	100.0 (0.49)
	2	101.3	76.8	84.6	34.9
	3	96.7	62.6	71.2	15.9
	4	102.5	44.5	25.0	20.6
	5	97.7	–	–	–
	6	91.8	–	–	–
	8	91.8	–	–	–
MATH500	0 (baseline)	100.0 (0.37)	100.0 (0.34)	100.0 (0.43)	100.0 (0.18)
	2	108.5	81.4	78.2	–
	3	89.4	60.5	54.6	–
	4	89.4	58.8	32.7	–
	5	93.6	–	–	–
	6	102.1	–	–	–
	8	70.2	–	–	–
HumanEval	0 (baseline)	100.0 (0.51)	100.0 (0.65)	100.0 (0.64)	100.0 (0.68)
	2	100.0	66.2	70.3	64.7
	3	100.0	63.1	59.4	48.5
	4	92.2	43.1	26.6	41.2
	5	82.3	–	–	–
	6	88.2	–	–	–
	8	62.7	–	–	–
MBPP	0 (baseline)	100.0 (0.52)	100.0 (0.43)	100.0 (0.71)	100.0 (0.81)
	2	123.1	60.5	70.3	75.3
	3	115.4	74.4	54.1	55.6
	4	115.4	53.5	37.8	35.8
	5	111.5	–	–	–
	6	94.2	–	–	–
	8	71.1	–	–	–

5.1 Layer Distribution and Skip Sensitivity

Tables 2a and 2b reveal that *consecutive layer skipping is catastrophic*. For LLaDA at 6-layer skip, allowing consecutive removal drops GSM8K retention from 91.8% to 75.3% and HumanEval from 88.2% to 64.7%. Dream-7B shows extreme sensitivity: 2-layer consecutive skip collapses GSM8K retention from 76.8% to 14.1%.

Our algorithm avoids this by maintaining representational continuity. Analysis and Fig.7 shows skipped layers concentrate in early network depth (first 40–60%), aligning with our observation that early layers establish coarse representations with high redundancy, while later layers perform critical fine-grained refinement.

Table 2: Performance retention for models with layer-skip when **allowed** and **not allowed** to skip consecutive layers. Higher (↑) the better

Layers Skipped	GSM8k		HumanEval	
	Not Allowed	Allowed	Not Allowed	Allowed
4	102.5	95.1	92.2	88.2
6	91.8	75.3	88.2	64.7
8	91.8	45.8	62.7	23.5

(a) **LLaDA** model

Layers Skipped	GSM8k		HumanEval	
	Not Allowed	Allowed	Not Allowed	Allowed
2	76.8	14.1	78.2	67.7
3	62.6	18.2	54.6	53.8
4	44.5	16.2	32.7	23.1

(b) **Dream** model

6 Related Work

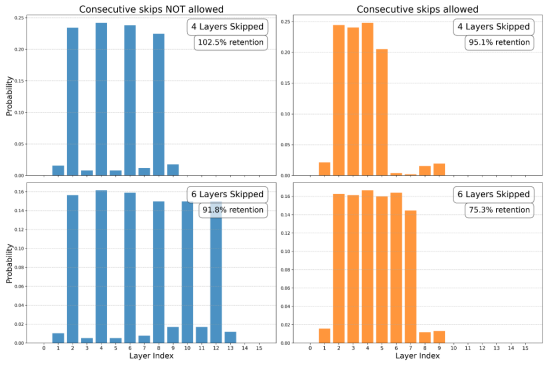


Figure 7: **Which layers are skipped?** Empirical distribution over *layer indices* selected for skipping by Algorithm 1on LLaDA, aggregated across evaluation prompts on GSM8k.The plotted value at layer i is the fraction (probability) of runs/examples in which layer i is chosen to be skipped. **Left:** consecutive (adjacent) layers may be skipped, allowing contiguous skip blocks. **Right:** consecutive skipping is disallowed, restricting selections to non-adjacent layers.

Diffusion Language Models and Representations. Diffusion language models (dLLMs) replace autoregressive decoding with bidirectional denoising objectives, enabling parallel decoding and global context modeling. Foundational work on discrete diffusion Austin et al. (2021) led to recent dLLMs such as SEDD Lou et al. (2023), and LLaDA Nie et al. (2025); Bie et al. (2025), which achieve competitive language modeling performance. Dream-7B Ye et al. (2025) adapts pretrained AR models to diffusion training, while MDLM Sahoo et al. (2024) simplifies diffusion objectives. Despite this progress, how diffusion objectives shape internal representations—especially relative to AR and AR-initialized models—remains insufficiently understood. Additionally, efforts have been made to integrate KV caching mechanisms to reduce redundant computation Ma et al. (2025); Liu et al. (2025). An alternate line of work focuses on step distillation of dLLMs Deschenaux & Gulcehre (2025); Qian et al. (2026) towards accelerating dLLM inference.

Representational Structure and Initialization Effects. Prior work shows that language models organize representations hierarchically across depth, with earlier layers capturing coarse features and deeper layers refining task-specific abstractions Jawahar et al. (2019). While representational dynamics in AR models have been studied, systematic analyses comparing AR and diffusion models are rare. Our work provides direct, layer- and token-level evidence of this initialization bias in AR-adapted dLLMs and contrasts it with the representational redundancy that emerges in native diffusion models.

References

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

Tiwei Bie, Maosong Cao, Kun Chen, Lun Du, Mingliang Gong, Zhuochen Gong, Yanmei Gu, Jiaqi Hu, Zenan Huang, Zhenzhong Lan, et al. Llada2. 0: Scaling up diffusion language models to 100b. *arXiv preprint arXiv:2512.15745*, 2025.

Jian Chen and Zhijian Liu. Dflash: Block diffusion for flash speculative decoding. *arXiv preprint*, 2026. URL https://github.com/z-lab/dflash. Paper coming soon.

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, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374. Introduces HumanEval and Codex.

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, 2021. URL https://arxiv.org/abs/2110.14168. Introduces the GSM8K benchmark.

Justin Deschenaux and Caglar Gulcehre. Beyond autoregression: Fast LLMs via self-distillation through time. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=uZ5K4HeNwd.

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. *NeurIPS Datasets and Benchmarks*, 2021. URL https://arxiv.org/abs/2103.03874. We use the 500-problem test subset commonly referred to as "MATH-500".

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

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.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*, 2023.

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

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, 2025. URL https://arxiv.org/abs/2502.09992.

Yu-Yang Qian, Junda Su, Lanxiang Hu, Peiyuan Zhang, Zhijie Deng, Peng Zhao, and Hao Zhang. d3llm: Ultra-fast diffusion llm using pseudo-trajectory distillation. *arXiv preprint arXiv:2601.07568*, 2026.

Maximo Eduardo Rulli, Simone Petruzzi, Edoardo Michielon, Fabrizio Silvestri, Simone Scardapane, and Alessio Devoto. Attention sinks in diffusion language models. *arXiv preprint arXiv:2510.15731*, 2025.

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

Huangjie Zheng Jiatao Gu Navdeep Jaitly Lingpeng Kong Yizhe Zhang Shansan Gong, Ruixiang Zhang. Diffucoder: Understanding and improving masked diffusion models for code generation. 2025. URL https://arxiv.org/abs/2506.20639.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2024. URL https://arxiv.org/abs/2412.15115.

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

7 APPENDIX

8 LAYER-SKIP ALGORITHM

Algorithm 1 Layer Skipping Algorithm

$\mathcal{S} \leftarrow [], \mathcal{L}_{\text{skip}} \leftarrow \emptyset, \tau \leftarrow$ threshold, $n_{\max} \leftarrow$ max layers to skip ▷ $\mathcal{S}$: similarity list, $\mathbf{H}$: hidden states, p: prompt length
for $i \leftarrow 1$ to $|\mathbf{H}| - 1$ **do**
  $\mathbf{h}_{\text{prev}} \leftarrow \mathbf{H}_{i-1}[:, : p]$
  $\mathbf{h}_{\text{curr}} \leftarrow \mathbf{H}_{i}[:, : p]$
  $s \leftarrow \text{cosine\_similarity}(\mathbf{h}_{\text{prev}}, \mathbf{h}_{\text{curr}}).\text{mean}()$
  $\mathcal{S}.\text{append}(s)$
end for ▷ Sort by similarity (descending)
$\_, \mathbf{idx} \leftarrow \text{sort}(\mathcal{S}, \text{descending} = \text{True})$ ▷ $\mathbf{idx}$: sorted indices ▷ Apply rules to select which layers to skip
for $i \leftarrow 0$ to $|\mathbf{idx}| - 1$ **do**
  $l \leftarrow \mathbf{idx}[i]$ ▷ Check if current layer is not adjacent to any already skipped layer
  if $l \notin \mathcal{L}_{\text{skip}}$ **and** $l - 1 \notin \mathcal{L}_{\text{skip}}$ **and** $l + 1 \notin \mathcal{L}_{\text{skip}}$ **and** $\mathcal{S}[l] \geq \tau$ **then**
    $\mathcal{L}_{\text{skip}}.\text{add}(l)$ ▷ Limit the maximum number of layers to skip
    if $|\mathcal{L}_{\text{skip}}| \geq n_{\max}$ **then**
      break
    end if
  end if
end for ▷ Prepare final list of layers to skip
$\mathcal{L}_{\text{skip}} \leftarrow \text{sorted}(\mathcal{L}_{\text{skip}})$ ▷ Add 1 to indices as we always skip the later layer in each pair
$\mathcal{L}_{\text{skip}} \leftarrow [l + 1 \text{ for } l \text{ in } \mathcal{L}_{\text{skip}}]$
return $\mathcal{L}_{\text{skip}}$

9 ADDITIONAL ANALYSIS AND VISUALIZATIONS

9.1 DETAILED TOKEN-WISE SIMILARITY ANALYSIS

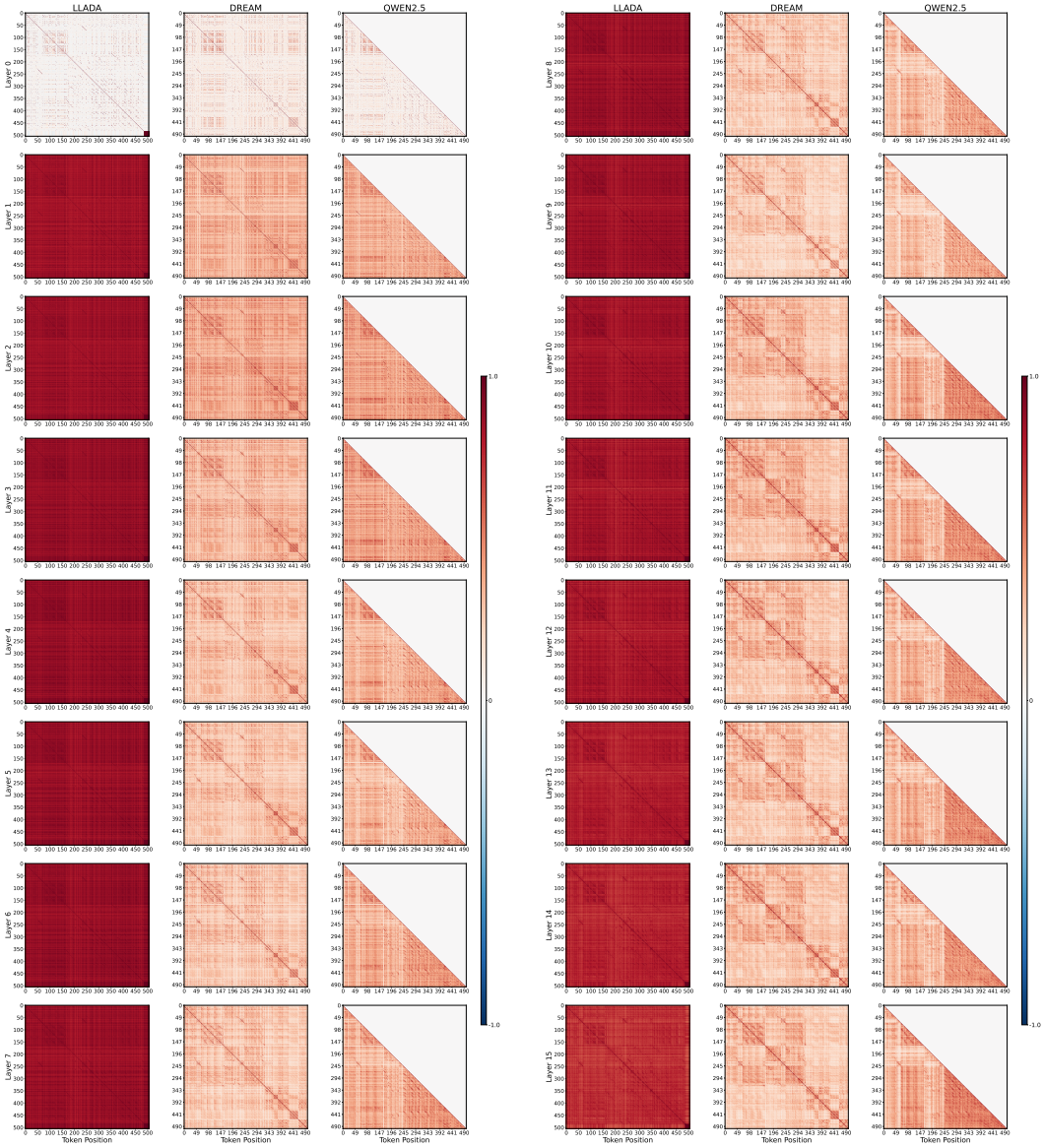
We provide comprehensive token-wise similarity visualizations to complement the layer-wise analysis presented earlier. These reveal how hidden state representations evolve across tokens within individual layers, providing deeper insight into the recency bias and global vs. local representation patterns discussed in the main text.

Figure 10 shows token-wise cosine similarity across all 32 layers of LLaDA. Early layers (0–15) exhibit consistently high similarity (> 0.9) between consecutive tokens, indicating smooth representational transitions with minimal recency bias. This validates our hypothesis that native dLLMs establish stable global context in early layers. Later layers (16–31) show increased variability and lower similarity, reflecting task-specific refinement and decoder-like behavior where representations are actively updated for generation.

In stark contrast, from Fig. 11 reveals that Dream-7B maintains significant recency bias across *all* layers. Consecutive token representations show substantial changes throughout network depth, mirroring the incremental token-by-token refinement characteristic of autoregressive models. This pattern persists despite diffusion training, providing mechanistic evidence that AR initialization creates persistent representational structure. The lack of hierarchical abstraction—with similar update patterns across all depths—explains Dream-7B's brittleness under layer skipping (Table 1), where it behaves more like Qwen2.5 than LLaDA.

9.2 LAYER-WISE TOKEN SIMILARITY BY DEPTH

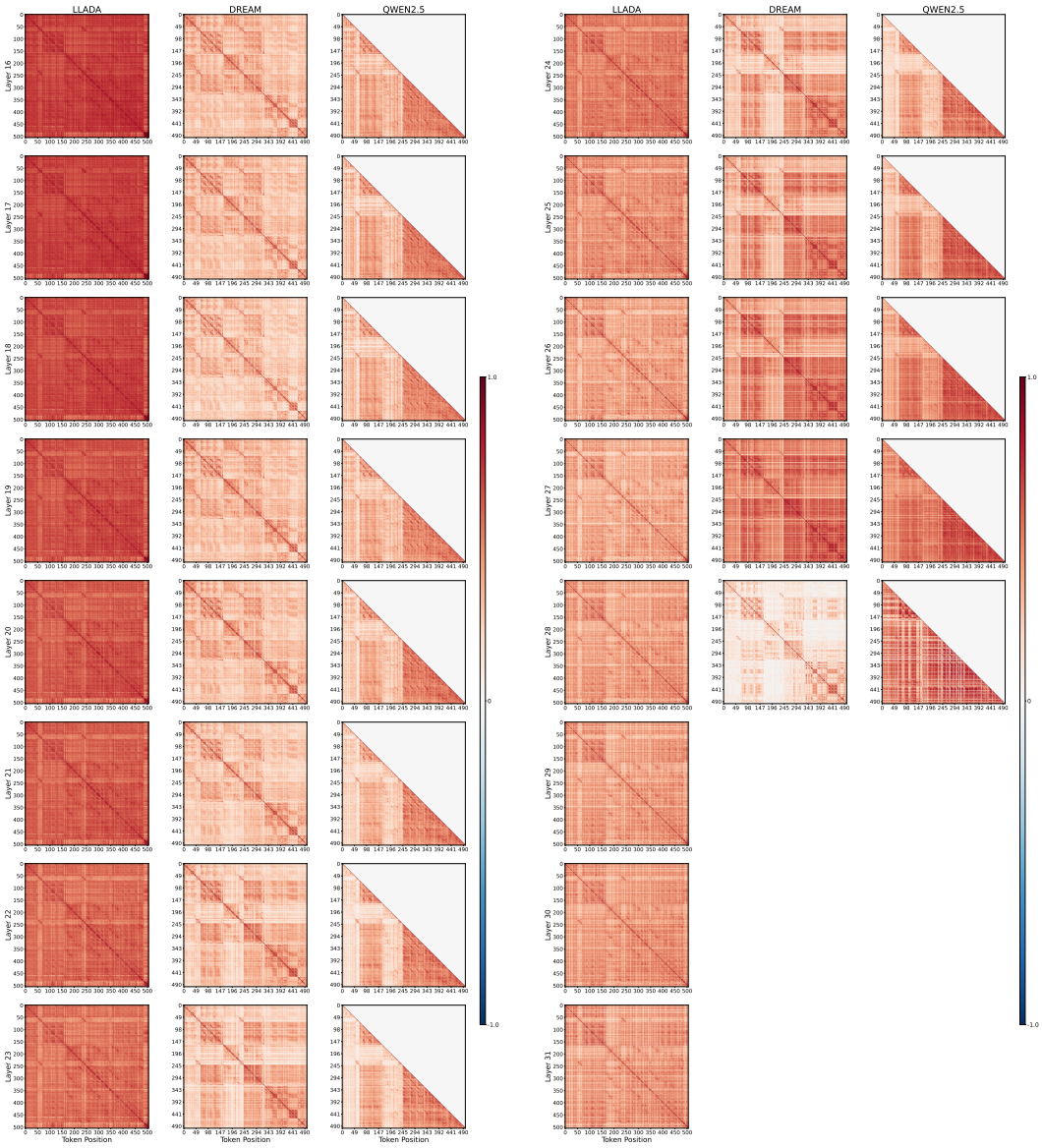
Figures below show token-wise similarity patterns grouped by network depth, revealing the transition from global to local representations:



(a) **Token-wise similarity in early layers (0–7).** LLaDA shows uniformly high similarity across all tokens, indicating stable global representations. Dream-7B and Qwen2.5 exhibit lower similarity with visible recency effects, demonstrating incremental AR-style processing even in early layers.

(b) **Token-wise similarity in early-middle layers (8–15).** LLaDA maintains high similarity, while Dream-7B and Qwen2.5 continue showing strong recency bias. The divergence between native dLLM and AR-initialized models becomes more pronounced.

Figure 8: **Token-wise similarity across early and early-middle layers.** (a) Layers 0–7 and (b) layers 8–15.



(a) **Late-middle layers (16–23).** LLaDA begins transitioning to lower similarity, indicating the onset of task-specific refinement. Dream-7B and Qwen2.5 maintain consistent recency patterns throughout depth.

(b) **Late layers (24–31).** LLaDA shows increased variability and lower similarity, reflecting active decoder-like refinement for generation. Dream-7B and Qwen2.5 continue incremental updates with strong recency bias, lacking the global transition observed in native dLLMs.

Figure 9: **Token-wise similarity in later layers.** (a) Late-middle (16–23) and (b) late (24–31).

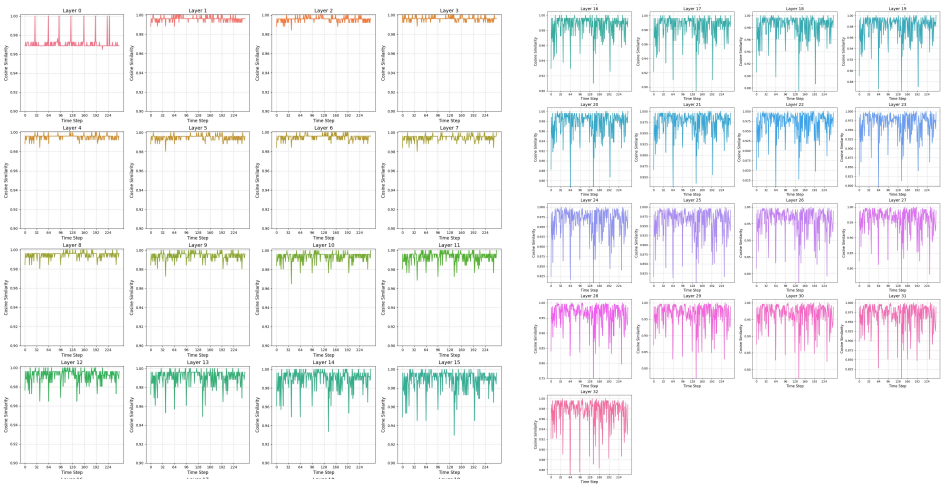


Figure 10: **Token-wise cosine similarity across all layers for LLaDA.** Each subplot shows the cosine similarity between consecutive token representations ($\mathbf{h}_{\ell,i}$ and $\mathbf{h}_{\ell,i+1}$) within a specific layer ℓ. High similarity indicates smooth representational transitions, while low similarity indicates significant representational changes between tokens. LLaDA exhibits consistently high token-wise similarity across early layers, demonstrating minimal recency bias and global representational abstraction. Later layers show increased variability, indicating task-specific refinement and decoder-like behavior. This pattern validates our hypothesis that native dLLMs develop coarse-to-fine abstraction hierarchies, with early layers establishing stable global context and later layers performing iterative refinement.

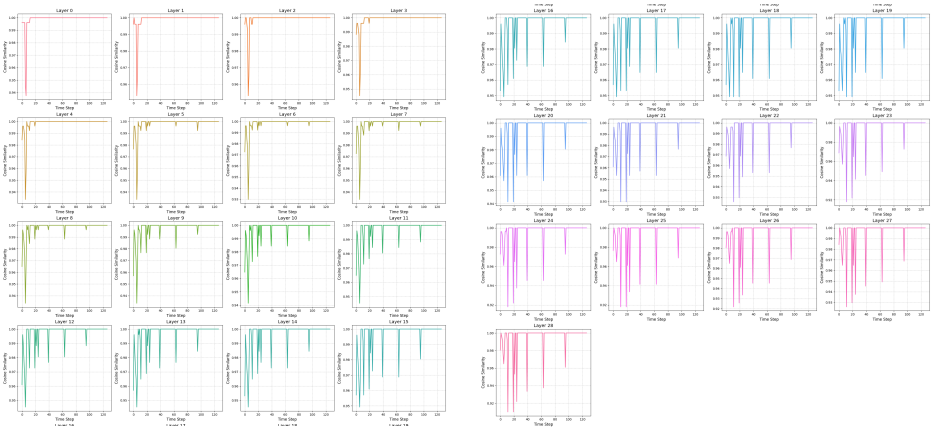


Figure 11: **Token-wise cosine similarity across all layers for Dream-7B.** Each subplot shows the cosine similarity between consecutive token representations within a specific layer. In stark contrast to LLaDA (Figure 10), Dream-7B exhibits significant recency bias across *all* layers, with substantial representational changes for each new token often. The lack of hierarchical abstraction—with similar token-by-token update patterns across all depths—confirms that Dream-7B retains AR-like incremental refinement and retain different representational abstraction compared to native dLLMs. This provides mechanistic evidence for the initialization bias observed in our layer-skip experiments (Table 1), where Dream-7B's brittleness mirrors Qwen2.5 despite diffusion training.