TEAM: Temporal–Spatial Consistency Guided Expert Activation for MoE Diffusion Language Model Acceleration

Anonymous Authors[1]

Abstract

Diffusion large language models (dLLMs) enable parallel decoding, and recent MoE dLLMs with autoregressive initialization further improve model capacity and accuracy. However, we identify a mismatch between MoE routing and diffusion decoding: a large number of experts are activated at each denoising step, while only a small subset of tokens is ultimately accepted, resulting in substantial inference overhead and limiting their deployment in latency-sensitive applications. In this work, we propose **TEAM**, a plug-and-play framework that accelerates MoE dLLMs by enabling more accepted tokens with fewer activated experts. TEAM is motivated by the observation that expert routing decisions exhibit strong temporal consistency across denoising levels as well as spatial consistency across token positions. Leveraging these properties, TEAM employs three complementary expert activation and decoding strategies, conservatively selecting necessary experts for decoded and masked tokens and simultaneously performing aggressive speculative exploration across multiple candidates. Experimental results demonstrate that TEAM achieves up to 2.2× speedup over vanilla MoE dLLM, with negligible performance degradation.

1. Introduction

Diffusion large language models (dLLMs) (Nie et al., 2025; Ye et al., 2025; Khanna et al., 2025) address limitations of autoregressive (AR) generation by adopting bidirectional attention, which enables parallelized decoding and positions dLLMs as a compelling alternative to conventional AR models. Recent advances (Wang et al., 2025; Wu et al., 2025; Fu et al., 2025; Liu et al., 2025a) with AR initialization further strengthen this paradigm by incorporating strong autoregressive training priors while remaining compatible with KV cache, achieving both higher accuracy and better inference efficiency than AR models of comparable scale.

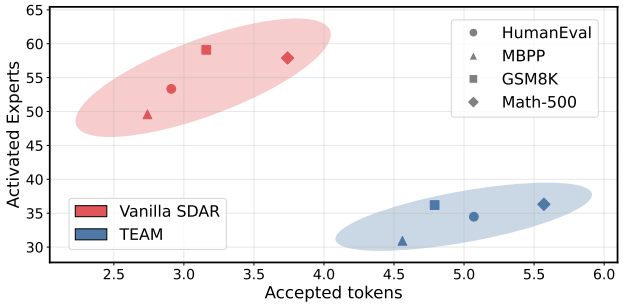

Figure 1. Activated experts vs. accepted tokens per forward pass in SDAR 30B-A3B. TEAM decodes more tokens with fewer experts activated in an iteration.

Mixture-of-Experts (MoE) architectures (Shazeer et al., 2017; Jiang et al., 2024) have become dominant in state-of-the-art AR language models (Yang et al., 2025; Liu et al., 2024; Comanici et al., 2025). Initializing dLLMs from such models, like SDAR (Cheng et al., 2025) and LLaDA 2.0 (Bie et al., 2025), further enhances the effectiveness of diffusion-based decoding, reinforcing the competitiveness of this paradigm compared to the latest AR LLMs (Ma et al., 2025b; Team et al., 2025).

Nevertheless, we observe that naively integrating MoE architectures into dLLMs can substantially degrade inference efficiency. Each diffusion iteration processes all tokens in a block and lets each token select experts independently, but only high-confidence tokens are accepted. Thus, one forward pass activates many distinct experts while yielding few decoded tokens, causing substantial memory and communication overhead. As illustrated in Figure 1, SDAR (Cheng et al., 2025) routes 8 experts per token, yet the number of distinct activated experts per accepted token is much higher in practice. This weakens MoE sparsity and makes decoding latency a practical bottleneck, especially for low-batch or resource-constrained deployment.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

To address this challenge, we propose **TEAM**, which is developed based on our core observation that although both involve multi token decoding, block-wise inference in dLLMs is fundamentally different from multi batch inference in autoregressive models, exhibiting strong temporal and spatial consistency. **Temporally**, accepted tokens remain fixed but continue to trigger expert activations in later iterations. **Spatially**, the routing of masked tokens is highly concentrated, with relatively little variation in expert selection across tokens. Moreover, the acceptance order exhibits spatial locality, suggesting that a substantial portion of masked tokens can be predicted to remain unaccepted in early iterations.

Building on these observations, TEAM applies three token-aware strategies. For **decoded tokens**, we introduce a delayed caching mechanism, activating experts only for recently accepted tokens. For masked tokens, we further partition them into **hot tokens**, which are more likely to be accepted in the near future, and **cold tokens**, which are unlikely to be accepted. Exploiting the concentration of expert routing among masked tokens, TEAM performs speculative exploration on hot tokens to increase the acceptance rate, while rerouting cold tokens to experts that are already activated by decoded or hot tokens. Our main contributions can be summarized as follows:

- We investigate the inefficiency of naively applying MoE architectures to dLLMs. To the best of our knowledge, this is the first study to specifically analyze expert activation characteristics in MoE dLLMs.
- Leveraging the temporal-spatial consistency of block-wise decoding, we propose TEAM with three complementary expert activation and decoding strategies.
- Experiments demonstrate that TEAM achieves up to 2.2× speedup while preserving model performance.

2. Related work

Diffusion Large Language Models (dLLMs). Autoregressive large language models (LLMs) (Achiam et al., 2023; Grattafiori et al., 2024; Yang et al., 2025) have demonstrated remarkable capabilities, yet their inference efficiency is constrained by autoregressive decoding. Diffusion Large Language Models (dLLMs) (Nie et al., 2025; Ye et al., 2025; Li et al., 2025a) mitigate this limitation by enabling parallel decoding via bidirectional attention mechanism. In this paradigm, the entire response is represented as masked tokens, and all positions are decoded in each forward pass. Tokens whose confidence exceeds a predefined threshold are accepted, while the remaining tokens are re-masked and refined in subsequent iterations. However, the global bidirectional attention prevented reuse of KV cache, resulting in limited efficiency gains. Recent block-diffusion models (Wang et al., 2025; Tian et al., 2025; Arriola et al., 2025; Gong et al., 2025) combine AR initialization, intra-block bidirectional attention, and inter-block causal attention. In this way, dLLMs inherit strong autoregressive priors for accuracy while simultaneously improving inference efficiency through parallel decoding and cache reuse.

Mixture-of-Experts (MoE). Mixture-of-Experts (MoE) architectures (Shazeer et al., 2017; Jiang et al., 2024) replace a single feedforward layer with a group of parallel expert networks. By enabling expert specialization and sparse parameter activation, MoE architectures scale more effectively to achieve higher model capacity while improving both training and inference efficiency, and have become a dominant design in recent LLMs (Yang et al., 2025; Liu et al., 2024; Comanici et al., 2025). This paradigm has also been extended to dLLMs (Zhu et al., 2025; Cheng et al., 2025; Bie et al., 2025). However, in dLLMs, all tokens within a block are repeatedly processed and independently routed. As a result, even with a low batch size, a large fraction of parameters may be activated in a single forward pass, thereby limiting the practical deployment of MoE dLLMs.

Acceleration of dLLMs. Existing dLLM acceleration methods reduce redundant computation through approximate KV caching (Wu et al., 2025; Liu et al., 2025b; Ma et al., 2025a), sparsification (Chen et al., 2025; Song et al., 2025; Jiang et al., 2025; Qian et al., 2026), or speculative decoding (Gao et al., 2025; Agrawal et al., 2025; Wei et al., 2025; Wu & Zhang, 2025; Chen et al., 2023; Leviathan et al., 2023). Beyond dense models, the strong empirical performance of MoE dLLMs has recently motivated efforts to accelerate this paradigm, such as dInfer (Ma et al., 2025b). However, dInfer primarily targets general dLLM acceleration and focuses only on expert-parallel execution for cloud-scale MoE deployment. In contrast, we present the first dedicated analysis of expert activation behavior in MoE dLLMs and propose TEAM, which exploits temporal–spatial consistency to tailor distinct expert activation strategies for different tokens, thereby improving decoding efficiency.

3. TEAM Methodology

3.1. Preliminary and Motivation

A dLLM initializes the response Y as $N = B \times L$ $[\texttt{MASK}]$ tokens and partitions it into B blocks of length L, with $Y_i = \left[y_i^0, \cdots, y_i^{L-1}\right]$. Given prompt P, block-wise decoding factorizes the response $\widehat{Y}$ as:

$$p_\vartheta(\widehat{Y} \mid P) = \prod_{i=1}^{B} p_\vartheta\left(\widehat{Y}_i \mid P, Y_{\leq i}\right) \tag{1}$$

Concretely, within each block, the model iteratively samples from the $[\texttt{MASK}]$ tokens. For the i-th block, a single

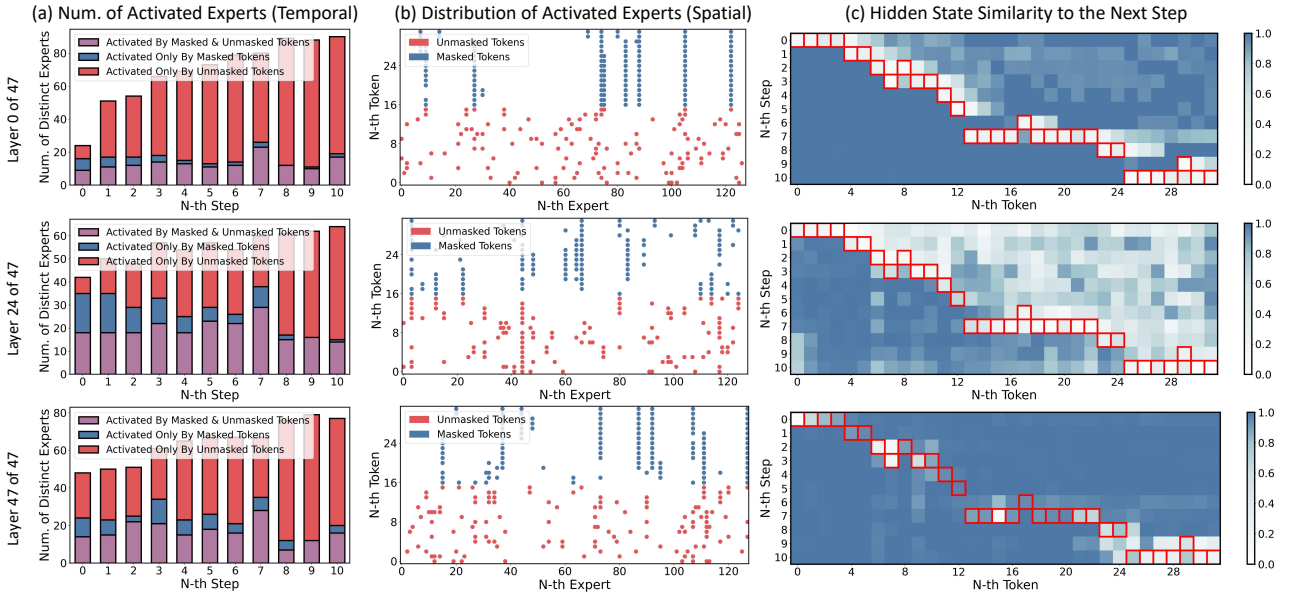

Figure 2. Temporal-spatial characteristics with SDAR 30B-A3B model on a prompt from GSM8K. Results are shown for layers 0, 24, and 47 (of 47). (a) Number of activated experts across decoding iterations. (b) Distribution of experts activated by decoded and masked tokens at step 6 (of 11). (c) Token acceptance positions at each iteration, together with hidden state similarity relative to the subsequent iteration.

forward pass produces token predictions and corresponding confidence scores, given by:

$$\begin{aligned} \widehat{y_i^k} &= \underset{v \in V}{\operatorname{argmax}} p_\vartheta \left(y_i^k = v \mid P, Y_{\leq i} \right) \text{ and} \\ c_k &= p_\vartheta \left(y_i^k = \widehat{y_i^k} \mid P, Y_{\leq i} \right), k \in [0, 1, \cdots, \mathrm{L}] \end{aligned} \tag{2}$$

where V is the vocabulary. Tokens with confidence above τ are accepted and others are re-masked for later iterations:

$$y_i^k = \begin{cases} \widehat{y_i^k}, & \text{if } c_k > \tau \\ [\mathrm{MASK}], & \text{otherwise} \end{cases}, k \in [0, 1, \cdots, \mathrm{L}] \tag{3}$$

Once all positions are unmasked, decoding moves to the next block until the end-of-sequence [EOS] token is generated.

Block-wise dLLMs gain efficiency by accepting multiple tokens per iteration. However, when this decoding paradigm is combined with MoE architectures, the parallel tokens collectively activate a large fraction of the experts, negating the benefits of sparse parameter activation. We analyze the decoding trajectory of a single block along with its associated expert activation patterns, as illustrated in Figure 2, from which we derive several key observations.

Temporal Consistency. Block-wise decoding in dLLMs requires repeatedly processing the same block across successive denoising iterations, during which tokens are gradually accepted and propagated to subsequent steps. Although these accepted tokens no longer change and merely provide context for decoding the remaining masked tokens, they still incur full computation at every iteration and independently trigger expert activations in MoE layers. Figure 2(a) reports the number of experts activated across iterations at three representative layers (first, middle, and last). This repeated computation on already decoded tokens leads to substantial additional expert activations, which grow as decoding proceeds and dominate later iterations.

Spatial Consistency. In contrast to the token-specific activations of decoded tokens, variability among masked tokens primarily stems from positional encodings in their input embeddings. Consequently, spatially adjacent masked tokens exhibit highly consistent expert routing patterns across layers. As shown in Figure 2(b), while decoded tokens activate a diverse set of experts with an approximately uniform distribution across candidates, masked tokens tend to concentrate their routing decisions on a small subset of experts. This observation suggests that a specific group of experts dominates the decoding of nearly all masked tokens, whereas experts outside this subset contribute only marginally or are invoked by very few tokens.

Temporal-Spatial Locality. We further analyze the step-wise similarity of hidden states produced by each layer and mark the positions that are unmasked (highlighted by red boxes), as illustrated in Figure 2(c). We find that the hidden state of a token changes most when it is accepted and in the

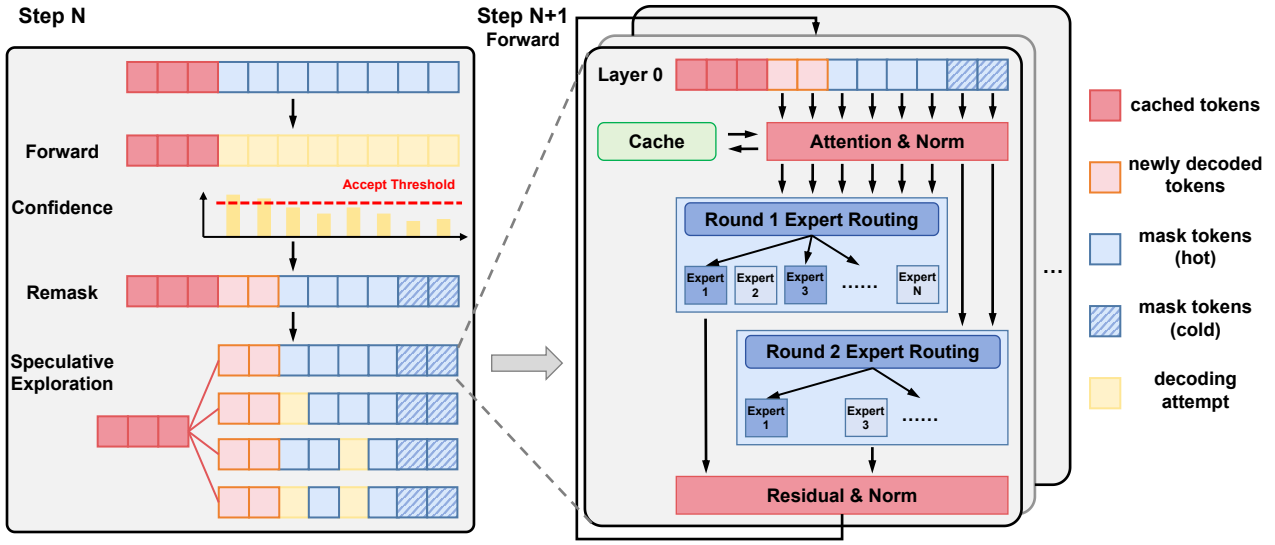

Figure 3. Overview of our proposed **TEAM**. We apply differentiated expert activation and decoding strategies to tokens within each block. For **decoded tokens**, redundant computation is reduced through one-step delayed caching. For **mask tokens (hot)**, we adopt aggressive multi-branch speculative exploration to exploit idle compute resources and increase the token acceptance rate. For **mask tokens (cold)**, a double-round routing mechanism is introduced to constrain unnecessary expert activations.

immediately following iteration, consistent with observations in prior work (Ma et al., 2025a; Song et al., 2025; Li et al., 2025b). Once a token has been accepted and processed through one additional forward pass, its representation can be regarded as approximately stable. Moreover, token acceptance also follows a near-autoregressive, spatially clustered order, making nearby masked tokens more likely to be accepted soon while distant low-confidence tokens remain cold. This behavior is expected, given that dLLMs are initialized from autoregressive models and that natural language generation inherently follows a causal structure.

Building on these insights, we propose TEAM, which implements three complementary expert activation and decoding strategies, as shown in Figure 3.

3.2. Delayed Caching for Decoded Tokens (DCD)

As discussed above, once decoded tokens are incorporated as input and processed by one additional forward pass after acceptance, their hidden representations become approximately stable. This motivates caching them to avoid redundant computation across iterations. At each iteration, DCD computes only the masked tokens and tokens newly accepted in the previous step, while reusing the KV pairs of earlier decoded tokens. After each forward pass, KV pairs of newly accepted tokens are inserted into the cache for subsequent iterations.

A related strategy, dKV-Cache (Ma et al., 2025a), targets global bidirectional attention and periodically recomputes all tokens every N iterations to mitigate KV drift. This refresh mechanism is less effective for block-diffusion dLLMs with native KV-cache support, where decoding is confined to a single block rather than the full sequence. Since such block-level parallelism is often memory-bound on modern GPUs, fine-grained refresh within a block brings limited additional benefit.

However, as observed earlier, under MoE architectures, decoded tokens activate a large set of experts that are largely distinct from those activated by masked tokens, substantially increasing parameter activation density and memory access. This property makes caching decoded tokens particularly beneficial in MoE-based dLLMs. Moreover, by leveraging autoregressive priors and the near-autoregressive acceptance order during decoding, our delayed caching mechanism eliminates the need for periodic global cache refresh.

3.3. Speculative Exploration for Hot Tokens (SEH)

Beyond redundant computation on decoded tokens, expert activation for masked tokens is also inefficient: each expert handles few tokens, leaving compute underutilized, and many distant low-confidence tokens are likely to be remasked. SEH therefore aims to improve decoding for tokens likely to be accepted soon, referred to as **hot tokens**, while reducing overhead for unlikely tokens, referred to as **cold tokens**. We identify two characteristics that make masked tokens more likely to be accepted in the next iteration: (1) their decoding attempt y_i^k at the current iteration yields a relatively high confidence score c_k, even if it does not yet exceed the acceptance threshold; and (2) they are spatially

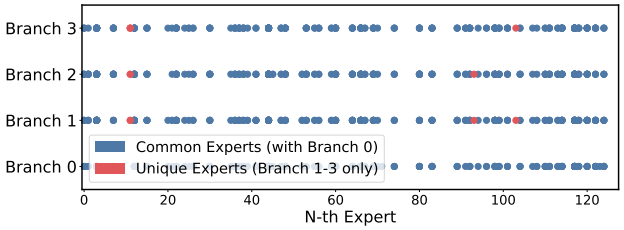

Figure 4. Expert activation with speculative exploration in SDAR for a response from GSM8K, measured at layer 24 (of 47).

closer to decoded tokens and receives stronger contextual guidance. Formally, hot tokens are defined as:

$$y_i^{k-hot} = \left\{ y_i^k \mid (c_k > \tau_h) \text{ or } (\forall j, |k-j| < L_h) \right\} \quad (4)$$

where τ_h denotes the confidence threshold to identify hot tokens, j indexes the positions of decoded tokens, and L_h specifies the maximum allowable distance from them.

For hot tokens, we perform speculative exploration by additionally accepting top-confidence candidates to construct multiple branches. These branches are decoded and verified in parallel, increasing the acceptance rate per iteration and reducing the number of decoding steps. Under bidirectional attention, modifying any single token affects the whole block, implying that dense models may incur prohibitive extra computation. In MoE models, however, inference is dominated by feedforward experts and naturally distributed across them. As illustrated in Figure 4, similar speculative branches activate largely overlapping experts while increasing per-expert arithmetic intensity, making SEH effective.

3.4. Limited Activation for Cold Tokens (LAC)

Cold tokens are masked tokens that are far from decoded positions and have low confidence in previous iterations. They are unlikely to be accepted in the following iterations, so activating experts uniquely routed to them is often unnecessary. Since their predictions are likely to be re-masked, dedicated cold-token expert activations are largely wasteful.

Leveraging spatial consistency, masked tokens tend to route to a largely shared expert subset. Based on this property, we apply a limited activation strategy for cold tokens, as summarized in Algorithm 1. LAC first routes newly accepted tokens that have not yet been cached together with hot tokens, whose accurate expert activation is important for decoding quality. The union of their selected experts forms a necessary expert set. Cold tokens are then routed in a second round restricted to this set, preventing token-specific extra expert activations while retaining the possibility that cold tokens may still be unexpectedly accepted.

Algorithm 1 Limited Activation for cold tokens

Input: Decoded tokens D, Mask tokens M, Experts E_0
Output: Activated Experts E_A, Routing Weights W
// 1. Classfication of tokens
Find newly accepted tokens $D_a \subseteq D$
Find hot tokens $H \subseteq M$ via **Eq. 4**
Define cold tokens $C \leftarrow M \setminus H$
// 2. First-round Routing
Necessary activation $W_1 \leftarrow Router(D_a, H, E_0)$
Necessary experts $E_A \leftarrow$ top-k (W_1)
// 3. Second-round Routing
Activation for cold tokens $W_2 \leftarrow Router(C, E_A)$
Routing Weights $W \leftarrow Concat(W_1, W_2)$
Return E_A, W

4. Experiments

4.1. Experimental Setup

Our experiments are primarily conducted on SDAR 30B-A3B (Cheng et al., 2025), a representative MoE-based diffusion language model following the block diffusion paradigm. Since an official HuggingFace-format evaluation pipeline for LLaDA 2.0 (Bie et al., 2025) is unavailable, we do not use it as the primary platform. We evaluate TEAM on diverse benchmarks, including HumanEval (Chen et al., 2021), MBPP (Austin et al., 2021), GSM8K (Cobbe et al., 2021), and Math-500 (Lightman et al., 2023). All experiments are conducted on an NVIDIA A100 80GB GPU.

Unless otherwise specified, we follow the official SDAR implementation. The acceptance threshold is set to $\tau = 0.95$, and the block size is fixed to 32. For hot-token classification, TEAM uses confidence threshold $\tau_h = 0.7$ or distance threshold $L_h = 3$ from decoded tokens. The number of speculative branches is set to 4.

4.2. Main Results

Table 1 summarizes TEAM's improvements over vanilla SDAR in expert activation and decoding efficiency. Without TEAM, each layer activates over 50 experts per forward pass on average, close to half of the total 128 experts. Yet each forward pass accepts only about three tokens, leading to more than twice the nominal routing cost (8 experts per token) for decoding a single token, and in the worst case up to 18 activated experts per decoded token. This confirms that although parallel decoding in dLLMs and sparse parameter activation in MoE architectures are both individually designed to be inference efficient, their naive combination becomes counterproductive. This inherent incompatibility significantly degrades the overall decoding speed.

In contrast, TEAM achieves decoding of more tokens with substantially fewer expert activations through its carefully

Table 1. **Performance of TEAM on SDAR.** APF denotes the number of Activated experts Per Forward pass, TPF denotes accepted Tokens Per Forward pass, and APT denotes the equivalent number of Activated experts Per decoded Token.

Benchmark	Method	Score↑	APF↓	TPF↑	APT↓	Speedup
HumanEval	Vanilla	79.27	53.34	2.91	18.33	1×
(0-shot)	TEAM	79.88 (+0.61)	34.48 (35%↓)	5.07 (1.74×)	6.80 (63%↓)	2.20×
MBPP	Vanilla	65.76	49.59	2.74	18.10	1×
(0-shot)	TEAM	65.76 (+0.00)	30.92 (38%↓)	4.56 (1.66×)	6.78 (63%↓)	2.08×
GSM8K	Vanilla	90.60	59.11	3.16	18.71	1×
(0-shot)	TEAM	90.30 (–0.30)	36.20 (39%↓)	4.79 (1.52×)	7.56 (60%↓)	1.83×
Math-500	Vanilla	76.00	57.90	3.74	15.48	1×
(0-shot)	TEAM	75.40 (–0.60)	36.31 (37%↓)	5.57 (1.49×)	6.52 (58%↓)	1.64×
Average	Vanilla	77.91	54.99	3.14	17.66	1×
	TEAM	77.84 (–0.07)	34.48 (37%↓)	5.00 (1.59×)	6.92 (61%↓)	1.94×

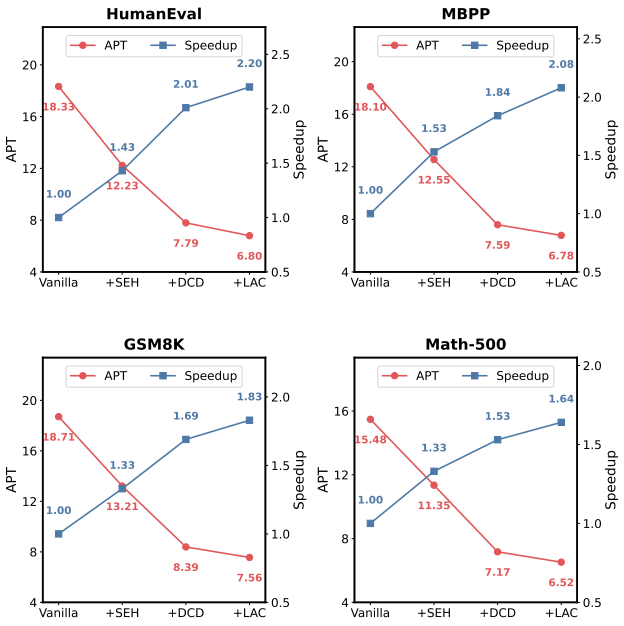

Figure 5. Ablation study on the Activated experts Per decoded Token (APT) and speedup compared to the vanilla model.

designed expert activation and decoding strategies, thereby simultaneously realizing the benefits of parameter sparsity and decoding parallelism. By introducing delayed caching for decoded tokens and limited expert activation for cold tokens, TEAM reduces the number of activated experts by 35–39% across all benchmarks, even after accounting for the additional experts introduced by multi-branch exploration. Moreover, speculative exploration for hot tokens further enhances decoding parallelism, increasing the number of accepted tokens per iteration by 1.49–1.74×. Benefiting from both higher sparsity and increased parallelism, TEAM requires on average only 6.92 activated experts to decode a single token, which is even lower than the nominal routing cost of 8 experts per token. As a result, TEAM achieves an average speedup of 1.94×, with a peak speedup of up to 2.2× on the HumanEval benchmark.

4.3. Ablation Study and Analysis

To assess the contribution of TEAM to accelerating inference in MoE dLLM, we progressively integrate its core techniques into the vanilla model. As illustrated in Figure 5, we evaluate both the average number of activated experts required to decode a single token and the corresponding speedup throughout this process.

The ablation study shows that Speculative Exploration for Hot Tokens (SEH) substantially reduces activated experts per decoded token by increasing accepted tokens per iteration, while adding only marginal expert activations due to branch similarity. Delayed Caching for Decoded Tokens (DCD) further eliminates a large fraction of expert activations triggered by decoded tokens that are irrelevant to the decoding process itself and only provide contextual guidance. Finally, Limited Activation for Cold Tokens (LAC) strictly confines expert activation to the subset responsible for newly decoded tokens and hot tokens. This design further reduces the number of activated experts per decoded token and yields additional speedup, resulting in the highest overall decoding efficiency among all benchmarks.

5. Conclusion

We propose **TEAM**, a plug-and-play framework that accelerates MoE dLLMs by exploiting temporal-spatial consistency in expert routing. By applying three complementary strategies tailored to decoded, hot masked, and cold masked tokens, TEAM enables more tokens to be decoded with fewer activated experts and achieves up to 2.2× speedup over vanilla inference, demonstrating an efficient and practical integration of dLLM and MoE architectures.

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Agrawal, S., Garrepalli, R., Goel, R., Lee, M., Lott, C., and Porikli, F. Spiffy: Multiplying diffusion llm acceleration via lossless speculative decoding. *arXiv preprint arXiv:2509.18085*, 2025.

Arriola, M., Gokaslan, A., Chiu, J. T., Yang, Z., Qi, Z., Han, J., Sahoo, S. S., and Kuleshov, V. Block diffusion: Interpolating between autoregressive and diffusion language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

Austin, J., Odena, A., Nye, M., Bosma, M., Michalewski, H., Dohan, D., Jiang, E., Cai, C., Terry, M., Le, Q., et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Bie, T., Cao, M., Chen, K., Du, L., Gong, M., Gong, Z., Gu, Y., Hu, J., Huang, Z., Lan, Z., Li, C., Li, C., Li, J., Li, Z., Liu, H., Liu, L., Lu, G., Lu, X., Ma, Y., Tan, J., Wei, L., Wen, J.-R., Xing, Y., Zhang, X., Zhao, J., Zheng, D., Zhou, J., Zhou, J., Zhou, Z., Zhu, L., and Zhuang, Y. Llada2.0: Scaling up diffusion language models to 100b, 2025. URL https://arxiv.org/abs/2512.15745.

Chen, C., Borgeaud, S., Irving, G., Lespiau, J.-B., Sifre, L., and Jumper, J. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*, 2023.

Chen, M., Tworek, J., Jun, H., Yuan, Q., de Oliveira Pinto, H. P., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., Ray, A., Puri, R., Krueger, G., Petrov, M., Khlaaf, H., Sastry, G., Mishkin, P., Chan, B., Gray, S., Ryder, N., Pavlov, M., Power, A., Kaiser, L., Bavarian, M., Winter, C., Tillet, P., Such, F. P., Cummings, D., Plappert, M., Chantzis, F., Barnes, E., Herbert-Voss, A., Guss, W. H., Nichol, A., Paino, A., Tezak, N., Tang, J., Babuschkin, I., Balaji, S., Jain, S., Saunders, W., Hesse, C., Carr, A. N., Leike, J., Achiam, J., Misra, V., Morikawa, E., Radford, A., Knight, M., Brundage, M., Murati, M., Mayer, K., Welinder, P., McGrew, B., Amodei, D., McCandlish, S., Sutskever, I., and Zaremba, W. Evaluating large language models trained on code. 2021.

Chen, X., Huang, S., Guo, C., Wei, C., He, Y., Zhang, J., Li, H., Chen, Y., et al. Dpad: Efficient diffusion language models with suffix dropout. *arXiv preprint arXiv:2508.14148*, 2025.

Cheng, S., Bian, Y., Liu, D., Zhang, L., Yao, Q., Tian, Z., Wang, W., Guo, Q., Chen, K., Qi, B., et al. Sdar: A synergistic diffusion-autoregression paradigm for scalable sequence generation. *arXiv preprint arXiv:2510.06303*, 2025.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Comanici, G., Bieber, E., Schaekermann, M., Pasupat, I., Sachdeva, N., Dhillon, I., Blistein, M., Ram, O., Zhang, D., Rosen, E., et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Fu, Y., Whalen, L., Ye, Z., Dong, X., Diao, S., Liu, J., Wu, C., Zhang, H., Xie, E., Han, S., et al. Efficient-dlm: From autoregressive to diffusion language models, and beyond in speed. *arXiv preprint arXiv:2512.14067*, 2025.

Gao, Y., Ji, Z., Wang, Y., Qi, B., Xu, H., and Zhang, L. Self speculative decoding for diffusion large language models. *arXiv preprint arXiv:2510.04147*, 2025.

Gong, S., Agarwal, S., Zhang, Y., Ye, J., Zheng, L., Li, M., An, C., Zhao, P., Bi, W., Han, J., Peng, H., and Kong, L. Scaling diffusion language models via adaptation from autoregressive models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=j1tSLYKwg8.

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., Casas, D. d. l., Hanna, E. B., Bressand, F., et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Jiang, Y., Cai, Y., Luo, X., Fu, J., Wang, J., Liu, C., and Yang, X. d^2 cache: Accelerating diffusion-based llms via dual adaptive caching. *arXiv preprint arXiv:2509.23094*, 2025.

Khanna, S., Kharbanda, S., Li, S., Varma, H., Wang, E., Birnbaum, S., Luo, Z., Miraoui, Y., Palrecha, A., Ermon, S., et al. Mercury: Ultra-fast language models based on diffusion. *arXiv preprint arXiv:2506.17298*, 2025.

Leviathan, Y., Kalman, M., and Matias, Y. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pp. 19274–19286. PMLR, 2023.

Li, J.-N., Guan, J., Wu, W., and Li, C. Refusion: A diffusion large language model with parallel autoregressive decoding. *arXiv preprint arXiv:2512.13586*, 2025a.

Li, S., Gu, J., Liu, K., Lin, Z., Wei, Z., Grover, A., and Kuen, J. Sparse-lavida: Sparse multimodal discrete diffusion language models. *arXiv preprint arXiv:2512.14008*, 2025b.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

Liu, A., Feng, B., Xue, B., Wang, B., Wu, B., Lu, C., Zhao, C., Deng, C., Zhang, C., Ruan, C., et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Liu, A., He, M., Zeng, S., Zhang, S., Zhang, L., Wu, C., Jia, W., Liu, Y., Zhou, X., and Zhou, J. Wedlm: Reconciling diffusion language models with standard causal attention for fast inference. *arXiv preprint arXiv:2512.22737*, 2025a.

Liu, Z., Yang, Y., Zhang, Y., Chen, J., Zou, C., Wei, Q., Wang, S., and Zhang, L. dllm-cache: Accelerating diffusion large language models with adaptive caching. *arXiv preprint arXiv:2506.06295*, 2025b.

Ma, X., Yu, R., Fang, G., and Wang, X. dkv-cache: The cache for diffusion language models. *arXiv preprint arXiv:2505.15781*, 2025a.

Ma, Y., Du, L., Wei, L., Chen, K., Xu, Q., Wang, K., Feng, G., Lu, G., Liu, L., Qi, X., et al. dinfer: An efficient inference framework for diffusion language models. *arXiv preprint arXiv:2510.08666*, 2025b.

Nie, S., Zhu, F., You, Z., Zhang, X., Ou, J., Hu, J., Zhou, J., Lin, Y., Wen, J.-R., and Li, C. Large language diffusion models. *arXiv preprint arXiv:2502.09992*, 2025.

Qian, Y.-Y., Su, J., Hu, L., Zhang, P., Deng, Z., Zhao, P., and Zhang, H. d3llm: Ultra-fast diffusion llm using pseudo-trajectory distillation. *arXiv preprint arXiv:2601.07568*, 2026.

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Song, Y., Liu, X., Li, R., Liu, Z., Huang, Z., Guo, Q., He, Z., and Qiu, X. Sparse-dllm: Accelerating diffusion llms with dynamic cache eviction. *arXiv preprint arXiv:2508.02558*, 2025.

Team, L., Li, A., Liu, B., Hu, B., Li, B., Zeng, B., Ye, B., Tang, C., Tian, C., Huang, C., et al. Every activation boosted: Scaling general reasoner to 1 trillion open language foundation. *arXiv preprint arXiv:2510.22115*, 2025.

Tian, Y., Liang, Y., Sun, J., Zhang, S., Yang, G., Shu, Y., Fang, S., Guo, T., Han, K., Xu, C., et al. From next-token to next-block: A principled adaptation path for diffusion llms. *arXiv preprint arXiv:2512.06776*, 2025.

Wang, X., Xu, C., Jin, Y., Jin, J., Zhang, H., and Deng, Z. Diffusion llms can do faster-than-ar inference via discrete diffusion forcing. *arXiv preprint arXiv:2508.09192*, 2025.

Wei, L., Chen, W., Tang, P., Guo, X., Ye, L., Wang, R., and Li, M. Orchestrating dual-boundaries: An arithmetic intensity inspired acceleration framework for diffusion language models. *arXiv preprint arXiv:2511.21759*, 2025.

Wu, C., Zhang, H., Xue, S., Diao, S., Fu, Y., Liu, Z., Molchanov, P., Luo, P., Han, S., and Xie, E. Fast-dllm v2: Efficient block-diffusion llm. *arXiv preprint arXiv:2509.26328*, 2025.

Wu, S. and Zhang, J. Free draft-and-verification: Toward lossless parallel decoding for diffusion large language models. *arXiv preprint arXiv:2510.00294*, 2025.

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., Zheng, C., Liu, D., Zhou, F., Huang, F., Hu, F., Ge, H., Wei, H., Lin, H., Tang, J., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Zhou, J., Lin, J., Dang, K., Bao, K., Yang, K., Yu, L., Deng, L., Li, M., Xue, M., Li, M., Zhang, P., Wang, P., Zhu, Q., Men, R., Gao, R., Liu, S., Luo, S., Li, T., Tang, T., Yin, W., Ren, X., Wang, X., Zhang, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Zhang, Y., Wan, Y., Liu, Y., Wang, Z., Cui, Z., Zhang, Z., Zhou, Z., and Qiu, Z. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Ye, J., Xie, Z., Zheng, L., Gao, J., Wu, Z., Jiang, X., Li, Z., and Kong, L. Dream 7b: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025.

Zhu, F., You, Z., Xing, Y., Huang, Z., Liu, L., Zhuang, Y., Lu, G., Wang, K., Wang, X., Wei, L., et al. Llada-moe: A sparse moe diffusion language model. *arXiv preprint arXiv:2509.24389*, 2025.