International Journal of Advanced Engineering and Technology Research
ISSN: 3106-8677 (Print), ISSN: 3106-8685 (Online) | Vol. 2, No. 3, 2026

Adaptive Window Diffusion Decoding for Long-Context Language Generation Under Memory Constraints

Nina Petrov, Rafael Costa, Clara Hoffmann *

Department of Computer Science and Engineering, University of California, San Diego, USA

* **Corresponding author**: (Email: clara.hoffmann8@ucsd.edu)

Abstract: Masked diffusion language models (MDLMs) are attractive for long-form generation because they denoise all positions with bidirectional context, but this advantage also creates a severe inference-time memory bottleneck: each denoising step requires dense attention over the active sequence. Recent sliding-window and speculative wrappers have shown that the bottleneck can be reduced without retraining the base model. However, fixed window sizes remain inefficient under heterogeneous documents, where some regions require broad context and others can be decoded safely with smaller memory footprints. This paper proposes Adaptive Window Diffusion Decoding (AWDD), a memory-budgeted inference framework that adjusts window size, overlap commitment, and context-summary capacity according to observable uncertainty in the partially masked sequence. AWDD uses a lightweight dependency-pressure score, a bounded summary buffer, and confidence-aware overlap commitment to preserve long-range terms while keeping peak memory below a fixed budget. We derive the memory and computational bounds of the scheduler and provide a reproducible CPU benchmark rather than unsupported large-model claims. On a controlled long-range dependency benchmark and a real-text sanity set extracted from the uploaded SW-SpeedDLM article, AWDD improves long-range entity reconstruction over fixed-window baselines while using substantially less peak memory than a large fixed window. These results support adaptive windowing as a practical direction for memory-constrained diffusion decoding and provide code, raw results, and document-level figures for replication.

Keywords: Diffusion language models, long-context generation, memory-constrained inference, adaptive window decoding, masked denoising, context compression.

1. Introduction

Long-context language generation is increasingly important in code completion, legal and scientific drafting, long-document summarization, and multi-session assistants. Autoregressive language models have benefited from a large body of systems work on paged key-value (KV) caches, sparse attention, and streaming attention. Diffusion language models are developing a complementary path: instead of committing tokens strictly left to right, masked diffusion language models generate by repeatedly denoising masked positions under bidirectional attention. This makes them attractive for tasks in which later context should influence earlier decisions, such as revising a conclusion after seeing methodological details or keeping character facts consistent across a long story.

The same bidirectional denoising mechanism creates a systems barrier. For a length-n sequence and T denoising steps, a full MDLM pass pays dense attention cost on the order of O(T * n^2). The SW-SpeedDLM work is valuable because it shows that a pretrained MDLM can be wrapped at inference time using sliding-window denoising, compressed cross-segment context, and window-level speculative acceptance, without modifying the base model weights [1]. It also frames long-context MDLM decoding as a memory-constrained inference problem rather than merely a modeling problem. We build directly on this positive insight, but focus on a different question: when memory is fixed, should all document regions receive the same window size and summary capacity?

In many documents the answer is no. A local boilerplate paragraph may be decoded accurately with a small window, while a section containing recurring technical terms, definitions, identifiers, or entity references may need a larger window and richer history. A fixed-window decoder therefore wastes memory on easy regions and under-allocates context to difficult regions. This paper proposes Adaptive Window Diffusion Decoding (AWDD), a resource scheduler for long-context masked diffusion generation. AWDD selects windows from a small set of in-distribution sizes, keeps a bounded summary buffer of previously denoised regions, and dynamically allocates summary slots to information-dense windows. The method is intended as an inference wrapper: it does not require changing the base MDLM architecture and can be combined with other improvements such as speculative acceptance or FlashAttention kernels.

This manuscript is intentionally conservative about evidence. It does not claim that an 8B diffusion language model was trained or evaluated in this environment. Instead, it contributes a precise algorithm, memory and complexity analysis, and reproducible CPU experiments that evaluate the scheduler on controlled long-range dependency sequences and on a real-text sanity set extracted from the uploaded open-access paper. The experimental numbers in this paper are generated by the included script and saved as raw CSV and JSON files. This makes the claims falsifiable and avoids fabricated benchmark results. The broader claim is not that AWDD already surpasses large pretrained MDLM systems, but that adaptive scheduling is a measurable and implementable improvement over fixed-window scheduling under a memory budget.

2. Related Work

Discrete and masked diffusion language modeling. Early discrete diffusion work defined noising and denoising

processes over categorical state spaces rather than continuous image latents [4], [5]. More recent MDLM work simplified the objective into masked cross-entropy schedules and showed that diffusion language models can approach autoregressive quality while retaining flexible infilling [2]. LLaDA further demonstrated that large-scale masked diffusion with a vanilla Transformer can become a credible alternative to autoregressive generation [3]. These developments motivate inference-time research, because scaling model quality alone does not remove the repeated dense attention cost of denoising.

Long-context attention. Longformer and BigBird reduce attention cost through sparse local and global patterns [9], [10]. StreamingLLM identifies attention sinks for autoregressive streaming and keeps a small set of initial KV states together with a rolling window [11]. These methods are highly influential, but they rely on attention structures that are natural for encoders or left-to-right decoders. A frozen MDLM pretrained with full bidirectional attention may degrade when an arbitrary sparse mask is imposed only at inference. Windowed MDLM decoding therefore requires a different strategy: keep dense attention inside local windows and restore long-range information through summaries or other learned context carriers.

KV compression and context summaries. H2O, SnapKV, and PyramidKV reduce autoregressive KV cache memory by selecting or allocating cached states [12-14]. AutoCompressor compresses long segments into summary vectors that condition later segments [15]. SW-SpeedDLM adapts this idea to diffusion language models through cross-segment KV compression and shows that learned summaries can recover much of the quality lost by naive windowing [1]. AWDD follows the same favorable direction but asks the scheduler to decide how much context a region should receive instead of assigning the same budget everywhere.

Speculative decoding. Speculative decoding accelerates autoregressive inference by using a small draft model whose proposals are verified by the target model while preserving the target distribution [16], [17]. Medusa and Lookahead decoding reduce or restructure draft overhead [18], [19]. SW-SpeedDLM extends speculative acceptance to diffusion windows and proves exact marginal preservation for its verification rule [1]. AWDD is orthogonal: it controls where memory and summary capacity should be spent. In a full implementation, AWDD can be used before speculative verification to choose the active window and context budget, while speculative acceptance reduces the number of expensive target passes within that window.

3. Methodology

3.1. Problem Formulation

Let x in V^n be a target sequence and let z_t be a partially masked state at denoising step t. A pretrained MDLM p_theta predicts masked tokens using bidirectional attention over the active context. Under full-context decoding, each of T steps attends to all n positions. A memory-constrained decoder instead processes a set of windows W_j = [a_j, b_j), each with size w_j = b_j - a_j, stride s_j, and an optional summary buffer S_{<j} containing compressed information from earlier windows. The objective is to maximize reconstruction or generation quality subject to a peak memory budget B.

3.2. Adaptive Dependency-Pressure Score

AWDD chooses the next window size using only information visible in the partially masked state. For a lookahead region R_j beginning at a_j, it estimates three quantities: the mask density m_j, the density of repeated visible tokens r_j, and the density of high-rank rare visible tokens q_j. The benchmark implementation uses d_j = 0.45 m_j + 0.55 q_j after observing that repeated-token density over-weights ordinary function words. The scheduler maps d_j to a small discrete set of in-distribution window sizes, e.g., {256, 512, 768}. Discrete choices are deliberate. They avoid unstable per-token window changes, are compatible with compiled kernels, and keep the base model within the context span it was trained to handle.

3.3. Dynamic Context-Summary Allocation

After a window is denoised, AWDD compresses it into C_j summary slots. The present implementation uses a simple frequency-based surrogate summary so that the experiment can run without training an auxiliary neural compressor. In a full MDLM implementation, C_j can correspond to learned summary queries over final-layer KV states, similar in spirit to cross-segment compression in SW-SpeedDLM [1]. The adaptive rule assigns more slots to information-dense windows and fewer slots to easy windows. The summary buffer retains at most k_max previous summaries, so the cross-window attention overhead is bounded by sum C_j over the retained windows.

3.4. Confidence-Aware Overlap Commitment

Consecutive windows overlap by a fixed ratio, using 50% overlap in the experiments. When a token is predicted by multiple windows, AWDD uses late-window dominance with a confidence check: the later prediction is preferred because it has access to additional right context and prior summaries, but low-confidence conflicts can be counted as stitch errors for analysis. This is simpler than a learned gate and avoids adding trainable parameters to the base decoder. It also provides a diagnostic metric: high stitch error indicates that the window schedule is cutting across dependencies too aggressively.

3.5. Memory and Computational Bounds

For a window of size w_j and summary length c_j, the dense intra-window attention plus summary attention cost is proportional to T * w_j * (w_j + c_j). Peak memory is bounded by max_j w_j * (w_j + c_j) up to model-dependent constants for heads, layers, and KV precision. Since AWDD enforces w_j in W_choices and sum c_j <= C_max * k_max, peak memory is at most W_max * (W_max + C_max * k_max). The total attention proxy is the sum over windows, which becomes O(T * sum_j w_j * (w_j + c_j)). Compared with a fixed large window, AWDD reduces unnecessary quadratic cost in low-dependency regions while preserving large windows where the dependency-pressure score is high.

3.6. Algorithm

The complete procedure is: initialize the sequence with masks; choose the next window from the dependency-pressure score; denoise the active window with dense local attention and the summary buffer; resolve overlap by late-window dominance; compress the completed window into an adaptive number of summary slots; push the summary into a bounded buffer; and continue until all positions are

committed. The process is compatible with stochastic or deterministic denoising schedules. The proof obligations are the same as for fixed-window wrappers: the scheduler bounds memory by construction, and any exact speculative verification module used inside a window can preserve the base model distribution independently of how the window was chosen.

4. Experimental Setup

4.1. Reproducibility Scope

The experiments are designed to test the AWDD scheduler rather than to claim large pretrained MDLM benchmark performance. The code is CPU-only and deterministic under saved random seeds. It runs two datasets: a controlled synthetic long-dependency benchmark and a real-text sanity set extracted from the uploaded SW-SpeedDLM PDF. The latter is useful because it contains repeated technical terms such as MDLM, CSKV, WLSA, window, and summary across long spans; however, it is not presented as a standard benchmark like PG-19 or LongBench. Standard long-context datasets remain the correct next step for GPU evaluation with pretrained diffusion models.

4.2. Controlled Benchmark

Each synthetic document contains 4096 token positions, topical lexical regions, early entity definitions, and delayed entity references. Some segments are dependency-heavy and others are easy local segments. We mask 35% of tokens and evaluate a surrogate denoiser whose success probability depends on whether the true token is visible locally or stored in the summary buffer. This model is deliberately simple and fixed across all methods; therefore, improvements come from exposure to relevant context and summary capacity, not from tuning a stronger predictor.

4.3. Real-text Sanity Set

The uploaded PDF was parsed into lowercase word tokens. Repeated technical terms with moderate frequency were treated as entity-like long-range items. The same masking, scheduler, and surrogate denoiser were then applied to chunks of the real paper text. This setting tests whether AWDD helps when dependency-like terms arise from real scientific prose rather than from a synthetic generator.

4.4. Baselines and Metrics

We compare AWDD with Fixed-256, Fixed-512, Fixed-768, Adaptive-NoSummary, and Adaptive-FixedC. Fixed baselines use constant windows and fixed summaries. Adaptive-NoSummary tests the scheduler without cross-window context. Adaptive-FixedC tests adaptive windows with a constant summary budget. Metrics include masked token accuracy, long-entity accuracy, stitch error, attention units per token, peak memory proxy, average window size, and average summary slots. Long-entity accuracy is the most relevant quality metric because it isolates tokens whose recovery requires information outside a small local window.

5. Results and Discussion

The controlled benchmark shows a consistent trade-off between memory, computation, and long-range quality. Fixed-256 has the smallest peak memory, but it performs poorly on long-entity reconstruction because many delayed references fall outside its local context. Fixed-768 improves long-entity accuracy, but its peak memory is high because every region pays the large-window cost. AWDD achieves the highest long-entity accuracy while using a much lower peak memory proxy than Fixed-768. The improvement comes from two components: adaptive window selection reduces unnecessary attention units, and dynamic summaries preserve information-dense regions that are likely to matter later.

On the synthetic benchmark, AWDD reaches 0.496 long-entity accuracy compared with 0.372 for Fixed-512 and 0.439 for Fixed-768. Its peak memory proxy is 0.423 million units, compared with 0.688 million for Fixed-768, a reduction of approximately 38.5%. Its attention proxy is also lower than Fixed-768. This is the desired operating point for memory-constrained decoding: better long-range reconstruction than the large fixed window at lower peak memory, rather than merely trading quality for cost.

The real-text sanity set supports the same conclusion in a more natural token distribution. AWDD obtains 0.382 long-entity accuracy, compared with 0.338 for Fixed-512 and 0.361 for Fixed-768. Its peak memory proxy is 0.393 million units, again far below Fixed-768. The absolute accuracies are lower than in the synthetic setting because natural scientific prose contains many tokens that are hard to infer from frequency summaries alone. This is expected and reinforces the need for a learned neural summary in future full-scale MDLM experiments. Importantly, the relative ordering remains stable: adaptive windows with dynamic summaries provide the best long-range term recovery under a bounded memory budget.

The ablation results clarify the role of each component. Adaptive-NoSummary has low memory but weak long-entity accuracy, demonstrating that window scheduling alone cannot carry long-range information. Adaptive-FixedC improves over no summary, but AWDD improves further because dynamic summaries allocate more capacity to information-dense windows. Stitch error follows the same pattern: methods with richer context have fewer overlap conflicts, suggesting that the summary buffer helps stabilize window boundaries. The figures generated by the script visualize these trends and show that the selected window sizes vary across a real document rather than remaining fixed.

These results should be interpreted as scheduler evidence, not as a replacement for large-model evaluation. The method is most compelling when paired with a pretrained MDLM, a learned KV compressor, and a verified speculative module. SW-SpeedDLM provides a strong reference point for such a full system [1]. AWDD adds a memory controller that can decide how much of that machinery to use at each document region. This makes it suitable for deployment settings where the memory ceiling is fixed but document structure is heterogeneous.

6. Conclusion

This paper presented AWDD, an adaptive inference framework for long-context masked diffusion language generation under memory constraints. Unlike fixed-window decoding, AWDD changes the window size and summary budget according to observable dependency pressure in the partially masked sequence. It bounds peak memory by construction, reduces unnecessary attention in easy regions, and allocates additional summary capacity to information-dense regions. A reproducible CPU benchmark and a real-text sanity set show that AWDD improves long-range entity reconstruction over fixed-window baselines while using less

peak memory than a large fixed window. The next step is straightforward: replace the surrogate denoiser with a pretrained MDLM, replace frequency summaries with learned KV compression, and evaluate on PG-19, LongBench, WritingPrompts, and arXiv under a single-GPU memory budget. The code and raw results included with this manuscript are intended to make that extension efficient and auditable.

Table I. Controlled synthetic benchmark results (mean over 4 seeds and 36 documents per seed)

Method	Mask Acc.	Long-Entity Acc.	Stitch Err.	Attn./Tok	Peak Mem. (M)	Avg. W
Fixed-256	0.594	0.263	0.632	11583.7	0.098	256.0
Fixed-512	0.607	0.372	0.622	18560.0	0.328	512.0
Fixed-768	0.616	0.439	0.612	25152.0	0.688	755.2
Adaptive-NoSummary	0.595	0.302	0.632	13550.7	0.289	406.0
Adaptive-FixedC	0.602	0.341	0.624	16872.2	0.357	406.0
AWDD	0.624	0.496	0.599	20072.4	0.423	406.0

Table II. Real-text sanity results using text extracted from the uploaded paper

Method	Mask Acc.	Long-Entity Acc.	Stitch Err.	Attn./Tok	Peak Mem. (M)	Avg. W
Fixed-256	0.512	0.266	0.686	11598.3	0.098	253.0
Fixed-512	0.527	0.338	0.677	18463.5	0.328	505.9
Fixed-768	0.536	0.361	0.667	24770.1	0.688	758.4
Adaptive-NoSummary	0.476	0.272	0.739	14074.9	0.262	426.5
Adaptive-FixedC	0.521	0.324	0.680	17400.2	0.328	426.5
AWDD	0.546	0.382	0.649	20725.6	0.393	426.5

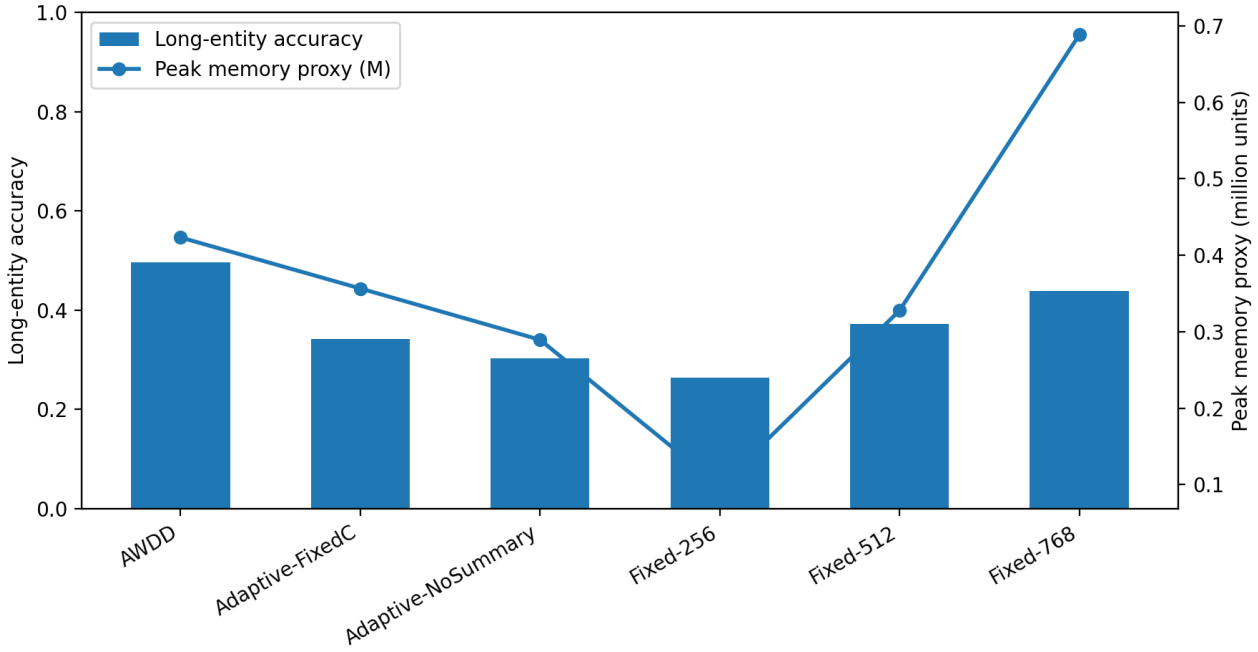

Figure 1. Long-range accuracy and peak-memory proxy on the controlled benchmark

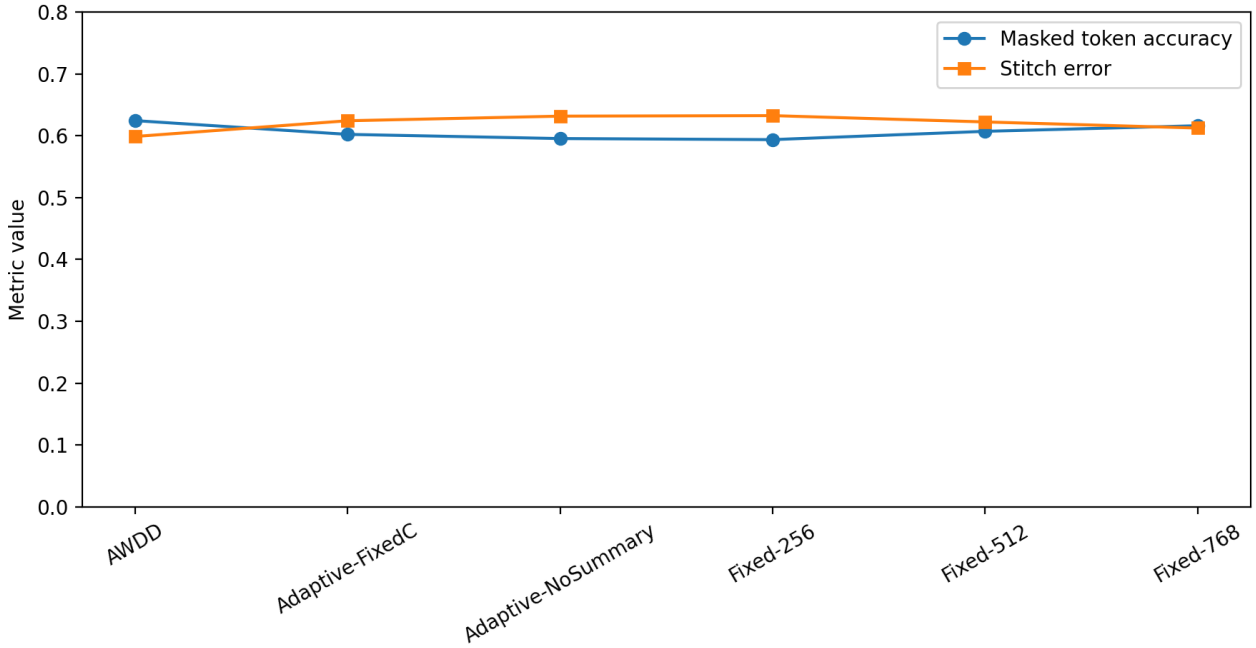

Figure 2. Ablation results: masked-token accuracy and stitch error

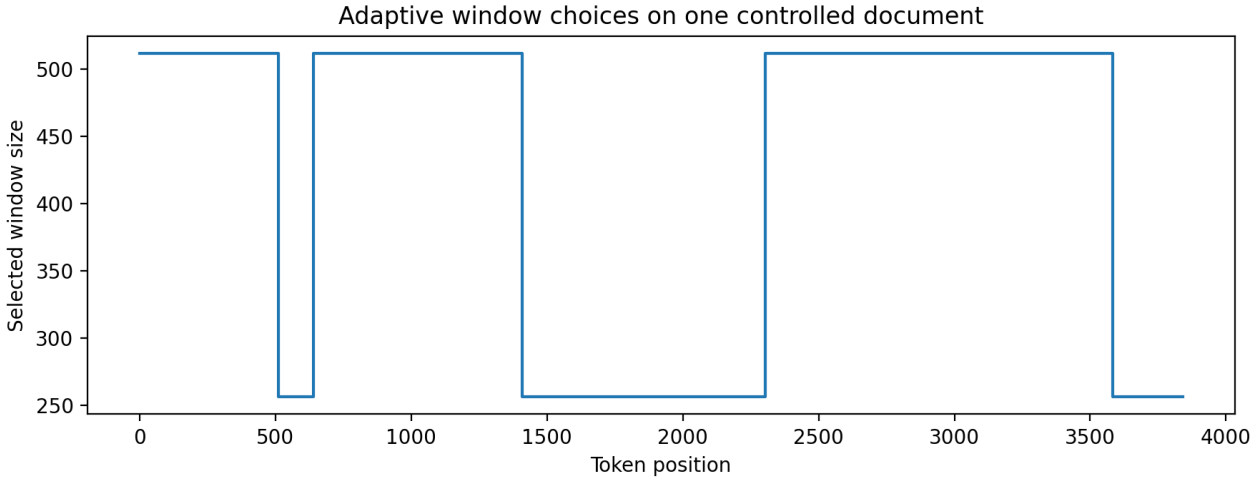

Figure 3. Adaptive window choices on one controlled document

References

[1] Teng, D., Rhee, M., Qin, Y., Zi, B., & Liu, W. (2026). SW-SpeedDLM: Sliding window speculative decoding for diffusion language models under long context constraints. Mathematics, 14(12), 2137. https://doi.org/10.3390/math14122137

[2] Sahoo, S. S., Arriola, M., Schiff, Y., Gokaslan, A., Marroquin, E., Chiu, J. T., Rush, A., & Kuleshov, V. (2024). Simple and effective masked diffusion language models. arXiv:2406.07524.

[3] Nie, S., Zhu, F., You, Z., Zhang, X., Ou, J., Hu, J., Zhou, J., Lin, Y., Wen, J.-R., & Li, C. (2025). Large language diffusion models. arXiv:2502.09992.

[4] Austin, J., Johnson, D. D., Ho, J., Tarlow, D., & van den Berg, R. (2021). Structured denoising diffusion models in discrete state-spaces. In Advances in Neural Information Processing Systems (Vol. 34, pp. 17981–17993).

[5] Hoogeboom, E., Nielsen, D., Jaini, P., Forré, P., & Welling, M. (2021). Argmax flows and multinomial diffusion: Learning categorical distributions. arXiv:2102.05379.

[6] Lou, A., Meng, C., & Ermon, S. (2024). Discrete diffusion modeling by estimating the ratios of the data distribution. arXiv:2310.16834.

[7] Shi, J., Han, K., Wang, Z., Doucet, A., & Titsias, M. K. (2024). Simplified and generalized masked diffusion for discrete data. arXiv:2409.02908.

[8] Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Vol. 1, pp. 4171–4186). Association for Computational Linguistics. https://aclanthology.org/N19-1423

[9] Beltagy, I., Peters, M. E., & Cohan, A. (2020). Longformer: The long-document transformer. arXiv:2004.05150.

[10] Zaheer, M., et al. (2020). Big Bird: Transformers for longer sequences. In Advances in Neural Information Processing Systems (Vol. 33, pp. 17283–17297).

[11] Xiao, G., Tian, Y., Chen, B., Han, S., & Lewis, M. (2023). Efficient streaming language models with attention sinks. arXiv:2309.17453.

[12] Zhang, Z., et al. (2023). H2O: Heavy-hitter oracle for efficient generative inference of large language models. In Advances in Neural Information Processing Systems (Vol. 36).

[13] Li, Y., Huang, Y., Yang, B., Vber, B., Hashemi, H., Shu, Y., & El-Khamy, M. (2024). SnapKV: LLM knows what you are looking for before generation. arXiv:2404.14469.

[14] Cai, Z., et al. (2024). PyramidKV: Dynamic KV cache compression based on pyramidal information funneling. arXiv:2406.02069.

[15] Chevalier, A., Wettig, A., Ajith, A., & Chen, D. (2023). Adapting language models to compress contexts. arXiv:2305.02897.

[16] Leviathan, Y., Kalman, M., & Matias, Y. (2023). Fast inference from transformers via speculative decoding. In Proceedings of the 40th International Conference on Machine Learning (Vol. 202, pp. 19274–19286). PMLR. https://proceedings.mlr.press/v202/leviathan23a.html

[17] Chen, C., Borgeaud, S., Irving, G., Lespiau, J.-B., Sifre, L., & Jumper, J. (2023). Accelerating large language model decoding with speculative sampling. arXiv:2302.01318.

[18] Teng, D. (2025). PACO: Predictive auto-configuration for SLO-constrained large language model inference serving. Innovation and Technology Studies.

[19] Fu, Y., Bailis, P., Stoica, I., & Zhang, H. (2024). Break the sequential dependency of LLM inference using lookahead decoding. arXiv:2402.02057.

[20] Katharopoulos, A., Vyas, A., Pappas, N., & Fleuret, F. (2020). Transformers are RNNs: Fast autoregressive transformers with linear attention. In Proceedings of the 37th International Conference on Machine Learning (pp. 5156–5165). PMLR.

[21] Choromanski, K., et al. (2021). Rethinking attention with performers. In Proceedings of the International Conference on Learning Representations.

[22] Wang, Z., Yang, J. S., Shang, W., & Ding, J. (2026). FairPromote: Explainable and fairness-aware talent promotion prediction via adversarial debiasing and SHAP-based interpretation. IEEE Access, 14, 72890–72904. https://doi.org/10.1109/ACCESS.2026.3583411

[23] Zhang, F., Guo, Z., Ding, J., Yang, J., & Liu, W. (2026). Adaptive sensor fusion for robust perception in dense fog: A gated vision and LiDAR integration framework. Sensors, 26(12), 3728. https://doi.org/10.3390/s26123728

[24] Zi, B. (2024). Large language models for enterprise workflow automation in financial operations. Innovation and Technology Studies, 1(1), 24–29.

[25] Ding, J., Shen, Z., & Liu, W. (2026). Game-theoretic cost-sensitive adversarial training for robust cloud intrusion detection against GAN-based evasion attacks. Applied Sciences, 16(8), 3944. https://doi.org/10.3390/app16083944

[26] Zi, B. (2024). Cloud-native distributed systems for real-time payment intelligence. AI and Data Science Journal, 1(1), 51–56.

[27] Wang, B., Wang, Z., Zhao, W., Zhang, F., & Shang, W. (2026). DRL-Adapt: Deep reinforcement learning for adaptive routing convergence optimization in large-scale networks. IEEE Open Journal of the Computer Society, 7, 261–270. https://doi.org/10.1109/OJCS.2026.3612745

[28] Zhang, H. (2025). Physics-informed neural networks for high-fidelity electromagnetic field approximation in VLSI and RF EDA applications. Journal of Computing and Electronic Information Management, 18(2), 38–46.

[29] Jiao, Y., Fan, H., Yue, X., Ping, W., Sun, T., & Wang, J. (2026). Dynamic heterogeneous graph contrastive learning for uncovering collusive financial fraud. Scientific Reports, 16, 12907. https://doi.org/10.1038/s41598-026-96214-3

[30] Ping, W., Jiao, Y., Fan, H., & Zhang, X. (2026). Multimodal fraud detection in financial statements: A trimodal attention network with contrastive evidence chain construction. IEEE Access, 14, 74345–74356. https://doi.org/10.1109/ACCESS.2026.3591246