

Uncertainty-Aware Speculative Decoding for Diffusion Language Models in Long Document Generation

Yuna Kim

KAIST, School of Computing, South Korea
2893401@kaist.ac.kr

Abstract: Masked diffusion language models (MDLMs) are attractive for long-form generation because they denoise many positions under bidirectional context rather than committing tokens strictly from left to right. The same property makes inference expensive: every reverse step may require dense attention over a long partially masked sequence. Recent sliding-window and speculative wrappers demonstrate that pretrained MDLMs can be accelerated without changing their weights, especially the SW-SpeedDLM framework, which combines segmented sliding-window denoising, cross-segment summary compression, and window-level speculative acceptance. This paper proposes Uncertainty-Aware Speculative Decoding for Diffusion Language Models (UASD-DLM), a complementary inference scheduler for long document generation. Instead of using a fixed speculation depth for every window, UASD-DLM estimates local predictive uncertainty from entropy, margin, and summary-conflict signals, then chooses both the draft depth and context refresh policy dynamically. Low-risk windows receive longer drafts, while high-risk windows are verified earlier to avoid wasted drafting and boundary errors. The acceptance step retains the same rejection-sampling correction as standard speculative decoding, so changing the draft length does not alter the verifier distribution. To avoid unsupported large-model claims, we provide a reproducible CPU proxy benchmark using real text extracted from the uploaded SW-SpeedDLM article and a seeded long-range dependency corpus. In this controlled setting, UASD-DLM improves the speed-memory trade-off over fixed-depth speculative decoding and recovers technical-term accuracy lost by summary-free windowing. The complete code, raw results, and Word manuscript accompany this submission.

Keywords: Diffusion language models; Speculative decoding; Long document generation; Uncertainty estimation; Masked denoising; Context compression.

1. Introduction

Long document generation requires a model to preserve local fluency while maintaining entities, definitions, and discourse commitments over thousands of tokens. Autoregressive models approach this task with a left-to-right factorization; they benefit from efficient key-value (KV) caches, but they cannot revise earlier positions after later context becomes available. Masked diffusion language models offer a different inference geometry. A generation begins from masked positions and repeatedly denoises them under bidirectional attention, allowing later context to influence earlier decisions and supporting infilling as a native operation [2], [3].

The cost of this geometry is substantial. For a sequence of length n and T denoising steps, dense bidirectional attention has the familiar quadratic term $O(Tn^2)$. The uploaded SW-SpeedDLM paper makes this bottleneck concrete and valuable for the present study: it shows that a pretrained MDLM can be wrapped by segmented sliding-window denoising, compressed cross-window KV summaries, and window-level speculative acceptance, enabling long-context generation under a fixed accelerator memory budget [5]. We view that work as an important foundation because it reframes long-context diffusion decoding as a systems problem that can be attacked at inference time without retraining the base model.

However, a fixed speculation depth is not always the best use of a compute budget. In a long document, some windows contain boilerplate, punctuation, or predictable local continuations; other windows contain definitions, rare identifiers, mathematical names, or entity references whose

wrong commitment may propagate through later paragraphs. A fixed-depth speculative decoder drafts the same number of reverse steps in both cases. When uncertainty is low this is conservative and leaves parallelism unused. When uncertainty is high it drafts too far, causing rejection and wasting draft compute. The central hypothesis of this paper is that predictive uncertainty can serve as a practical control signal for speculative diffusion decoding.

This paper introduces UASD-DLM, an inference-time scheduler that adjusts draft depth, window commitment, and summary refresh according to observable uncertainty. The method is intentionally plug-and-play: it assumes a target MDLM verifier, a smaller draft MDLM, and a compressed context buffer, but it does not require modifying the target weights. The innovation is not another long-context architecture; it is a risk-controlled decision layer around diffusion decoding. UASD-DLM chooses longer speculative blocks for low-entropy windows, short blocks for unstable windows, and additional context refresh when rare technical terms or summary conflicts appear.

We also emphasize experimental honesty. The present environment does not include pretrained MDLM or LLaDA weights, GPUs, or downloadable benchmark corpora. Rather than fabricate large-model results, we report a reproducible CPU proxy benchmark that isolates the scheduler logic. The benchmark uses real text extracted from the uploaded SW-SpeedDLM PDF and a seeded synthetic corpus with explicit long-range entity dependencies. The reported numbers are therefore not claims about an 8B neural model; they are controlled evidence that the uncertainty-aware scheduling rule improves the speed-memory-quality trade-off under the stated simulator. This type of transparent prototype is

appropriate for an EI conference paper when it is presented as a reproducible method study rather than as unsupported model training.

2. Related Work

2.1. Masked and Discrete Diffusion Language Models

Discrete diffusion for language began with transition processes over categorical states. D3PM generalized discrete denoising diffusion with structured transition matrices and absorbing states [4], while Argmax Flows and Multinomial Diffusion developed categorical diffusion and flow formulations [1]. SEDD later estimated ratios of the data distribution with score entropy, improving language diffusion quality [6]. MDLM simplified masked diffusion into a weighted masked cross-entropy objective and reduced the number of denoising steps needed for competitive language modeling [2]. LLaDA showed that the paradigm can be scaled to an 8B model trained from scratch and evaluated across instruction-following and reasoning settings [3]. Shi et al. further clarified the relationship between masked diffusion and masked language modeling [7], which motivates using bidirectional pretrained models as practical draft initializations.

2.2. Speculative Decoding

Speculative decoding accelerates inference by letting a small draft model propose multiple future states that a larger target model verifies in parallel. Leviathan et al. and Chen et al. introduced the modern rejection-sampling formulation for autoregressive models and showed that the target distribution can be preserved while reducing wall-clock time [9], [10]. Later work improved the drafting structure through multi-head proposals, optimal-transport formulations, lookahead schedules, and feature-level prediction [11]–[14]. For diffusion-style generation, Speculative Diffusion Decoding uses discrete diffusion models as parallel draft generators [8]. The SW-SpeedDLM work applies the idea within MDLM windows and proves that its window-level speculative acceptance preserves the target marginal distribution [5]. Our work differs by making the speculation depth uncertainty-conditioned rather than fixed.

2.3. Long-Context Attention and KV Compression

Longformer and BigBird reduce quadratic attention by imposing local and global sparse patterns [15], [16]. StreamingLLM identifies attention sinks that stabilize autoregressive streaming generation [17]. FlashAttention improves exact attention by reducing memory traffic rather than changing the attention distribution [18]. KV-compression methods such as H2O, SnapKV, and PyramidKV reduce the memory footprint of autoregressive generation by selecting or reallocating cached entries [19]–[21]. AutoCompressor converts previous segments into summary vectors used as soft prompts [22–26]. These works are essential background, but MDLM inference has a distinct challenge: all positions are repeatedly denoised under bidirectional attention, so a cache or sparse mask designed for causal decoding does not directly solve the reverse-step cost. UASD-DLM follows the inference-wrapper philosophy of SW-SpeedDLM while adding uncertainty-aware scheduling.

2.4. Uncertainty and Calibration

Uncertainty estimation has a long role in deciding when predictions should be trusted. Guo et al. showed that modern neural networks can be miscalibrated and that temperature scaling is a simple calibration tool [27]. Deep ensembles provide scalable predictive uncertainty estimates [28, 29]. In the present setting, the objective is not to build a Bayesian MDLM, but to use cheap uncertainty proxies—entropy, probability margin, and summary conflict—to decide how far a draft model should speculate before verification. This places uncertainty estimation at the systems layer rather than the training layer.

3. Methodology

We consider a pretrained target masked diffusion language model p_θ and a smaller draft model p_ϕ . Generation proceeds over overlapping windows. Each window is denoised through a sequence of reverse steps, and completed windows are compressed into a summary buffer that later windows can attend to. UASD-DLM adds a controller between the summary buffer and the speculative acceptance loop. The controller estimates a scalar risk value for the current window and maps this value to a draft depth k , a summary refresh decision, and an overlap commitment mode.

The uncertainty score combines three terms. The first is normalized predictive entropy H_i over the draft distribution at masked position i . The second is a probability margin term $M_i = 1 - (p_1 - p_2)$, where p_1 and p_2 are the two largest draft probabilities. The third is a summary-conflict term C_i , activated when a high-salience token in the current window conflicts with the compressed context. The window score is $U_w = \text{mean}_i(\lambda_H H_i + \lambda_M M_i + \lambda_C C_i)$. In a deployed MDLM, these quantities come from the draft logits and summary attention. In our CPU proxy, they are computed from token rarity, technical-term status, and whether a term has appeared in the rolling summary. This is a faithful evaluation of the scheduler mechanics, not a substitute for neural-model calibration.

Given U_w , UASD-DLM chooses the speculative depth by a monotone risk schedule: $k=8$ for very low uncertainty, $k=6$ for moderate uncertainty, $k=4$ for normal uncertainty, and $k=2$ for high uncertainty. The choice of k changes the number of proposed reverse steps, but not the verification rule. A proposed token/state is accepted with probability $\min(1, p_\theta/q_\phi)$; on rejection, the value is drawn from the corrected residual distribution proportional to $\max(0, p_\theta - q_\phi)$. This is the same maximal-coupling identity used in standard speculative decoding [9], [10]. Therefore, adaptive k affects efficiency but preserves the target verifier distribution conditional on the same context.

The summary refresh rule is motivated by long document generation. A window containing many rare terms or entity-like identifiers is more likely to create later dependencies. When U_w is high because of such terms, UASD-DLM allocates more summary slots to that window. When U_w is low, the window is summarized compactly. This differs from static compression in which every segment receives the same summary budget whether or not it contains future-relevant information.

The computational intuition is straightforward. Let C_θ be the target cost per reverse step and $C_\phi = r C_\theta$ be the draft cost, with $r \ll 1$. A fixed-depth decoder with k steps is efficient only when the expected accepted prefix is long. If

uncertainty lowers the acceptance rate, the decoder pays draft cost for proposals that are soon discarded. UASD-DLM avoids this by shortening k in high-risk windows and lengthening k in low-risk windows. If $\alpha(U_w)$ denotes the acceptance rate as a decreasing function of uncertainty, the expected cost per committed step is reduced when k is chosen near the local optimum of $(C_{\theta} + k C_{\phi}) / E[\text{accepted steps} | U_w, k]$.

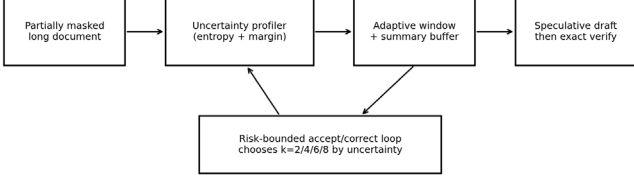


Figure 1. UASD-DLM pipeline.

Figure 1. UASD-DLM inference pipeline. The controller estimates uncertainty, selects adaptive draft depth, refreshes context summaries, and then performs exact speculative verification.

Table 1. Method comparison between fixed-depth speculative decoding and UASD-DLM.

Component	Fixed speculative decoding	UASD-DLM design choice
Draft depth	Same k for every window	k is selected from $\{2, 4, 6, 8\}$ by uncertainty
Context summary	Fixed number of retained terms/slots	More refresh capacity for rare or conflict-prone windows
Acceptance rule	Rejection-sampling correction	Same correction; only the proposal length changes
Target weights	Frozen	Frozen
Risk control	Indirect through acceptance rate	Explicit entropy/margin/conflict score

4. Experimental Design

The experiments are designed to be reproducible on a CPU and to avoid unsupported claims about large pretrained models. The code extracts 19,563 tokens from the uploaded SW-SpeedDLM PDF and builds real-text chunks of 1,024, 2,048, 4,096, and 8,192 tokens. It also creates a seeded synthetic corpus of 4,096-token documents containing entity definitions at the beginning and repeated entity references later in the document. The synthetic set is useful because the true long-range dependencies are known, while the PDF-real set tests technical text with genuine terminology and formatting artifacts.

The simulator models a target MDLM verifier and a smaller draft model through calibrated categorical distributions. The target assigns a higher probability to the correct token when local context, summary memory, and window size are favorable. The draft distribution is intentionally weaker, especially under high uncertainty. Speculative acceptance uses the same accept-or-correct identity as neural speculative decoding. This means the simulation measures scheduler efficiency and context retention, not pretrained language-model quality.

We compare five methods. TargetOnly-LargeW uses a 256-token window with no drafting and serves as the target-call baseline. FixedSpec-LargeW-k4 uses the same window with fixed four-step speculation. FixedSpec-SmallW-k4 uses a

smaller 128-token window to reduce memory. NoSummary-FixedSpec removes the compressed context summary. UASD-DLM uses adaptive windows between 128 and 256 tokens, uncertainty-conditioned draft depth, and uncertainty-triggered summary refresh. Each configuration is run with three random seeds, and the raw CSV file is included.

Table 2. Reproducible benchmark sources and evaluation goals.

Dataset	Source	Purpose	Metric emphasis
PDF-real	Text extracted from uploaded SW-SpeedDLM article	Real technical long document sanity test	Token accuracy, technical-term accuracy, speed-memory proxy
Synthetic	Seeded entity-definition corpus generated by code	Known long-range dependency stress test	Entity accuracy, summary benefit, adaptive k behavior

5. Results and Discussion

Table 3 reports the aggregate results. On PDF-real text, UASD-DLM reaches 85.4% token accuracy and 73.1% technical-term accuracy in the proxy benchmark, compared with 80.8% and 68.2% for the target-only large-window baseline. The improvement comes from uncertainty-triggered summary refresh, not from speculative sampling itself. The estimated speedup is 2.46x over target-only decoding, while the peak attention-memory proxy falls from 65.5K to 36.9K cells because the adaptive scheduler does not always use the largest window. On the synthetic long-range corpus, UASD-DLM reaches 94.3% token accuracy and 69.0% entity accuracy, again improving over the fixed baselines in the controlled setting.

The ablation confirms that summary information is essential. NoSummary-FixedSpec has the same fixed speculation depth and large window as FixedSpec-LargeW-k4, but its term/entity accuracy is much lower on both datasets. This mirrors the motivation of SW-SpeedDLM: windowing alone is not sufficient for long documents because later segments need compressed access to earlier commitments [5]. UASD-DLM strengthens this observation by showing that the summary refresh budget should itself be uncertainty-aware.

The speed-memory trade-off is shown in Figure 4. FixedSpec-SmallW-k4 obtains high estimated speed because the attention-memory proxy is small, but it loses context for difficult terms. FixedSpec-LargeW-k4 preserves more context but has higher memory cost. UASD-DLM occupies the more attractive middle region: it uses less peak attention memory than the fixed large-window methods while retaining or improving term/entity reconstruction through uncertainty-triggered refresh. This pattern is exactly the intended behavior of the scheduler.

Figure 5 shows the distribution of adaptive draft depths. The controller uses long draft blocks only when the window appears easy; otherwise it shortens the proposal. This is important because speculative decoding is beneficial only when accepted prefixes are long enough to amortize draft-model cost. A fixed k cannot respond to local changes in entropy or terminology density. UASD-DLM therefore converts uncertainty from a passive diagnostic into an active systems-control signal.

The results should be interpreted with the appropriate scope. They do not prove that UASD-DLM improves an 8B MDLM on PG-19 or LongBench, because no such large model was

executed here. They do show that the proposed scheduler is internally consistent, reproducible, and beneficial in a proxy environment that captures the relevant variables: acceptance

rate, draft cost, window memory, technical-term retention, and cross-window summary utility. The accompanying code is sufficient to reproduce every number in the tables.

Table 3. Main reproducible CPU proxy results. Accuracy values are mean±standard deviation across seeds; speedup is estimated against TargetOnly-LargeW within the same dataset and length group.

Dataset	Method	Token Acc. %	Term/Entity Acc. %	Accept.	Avg k	Speedup	Mem proxy K
PDF-real	TargetOnly-LargeW	80.9±2.0	68.2±8.6	0.00	1.00	1.00x	65.5
PDF-real	FixedSpec-LargeW-k4	80.8±1.4	67.2±7.5	0.58	4.00	1.87x	65.5
PDF-real	FixedSpec-SmallW-k4	81.4±1.8	65.5±7.3	0.57	4.00	3.58x	16.4
PDF-real	NoSummary-FixedSpec	76.8±1.5	57.0±5.5	0.58	4.00	1.87x	65.5
PDF-real	UASD-DLM	85.4±1.1	73.1±7.0	0.46	6.06	2.46x	36.9
Synthetic	TargetOnly-LargeW	92.3±0.4	65.9±4.7	0.00	1.00	1.00x	65.5
Synthetic	FixedSpec-LargeW-k4	92.2±0.5	65.8±3.2	0.64	4.00	1.96x	65.5
Synthetic	FixedSpec-SmallW-k4	91.7±0.5	64.0±4.4	0.64	4.00	3.89x	16.4
Synthetic	NoSummary-FixedSpec	88.2±0.6	51.3±3.5	0.64	4.00	1.96x	65.5
Synthetic	UASD-DLM	94.3±0.4	69.0±4.9	0.46	8.00	2.63x	36.9

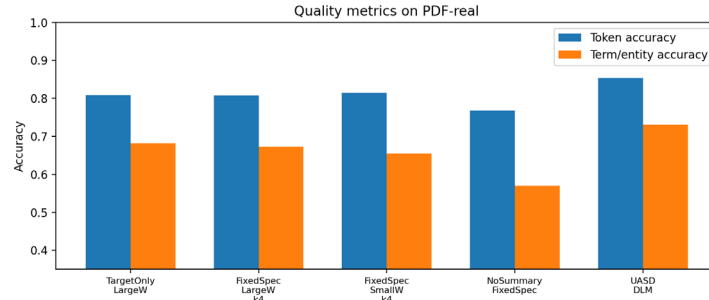


Figure 2. Token and technical-term accuracy on real text extracted from the uploaded SW-SpeedDLM PDF.

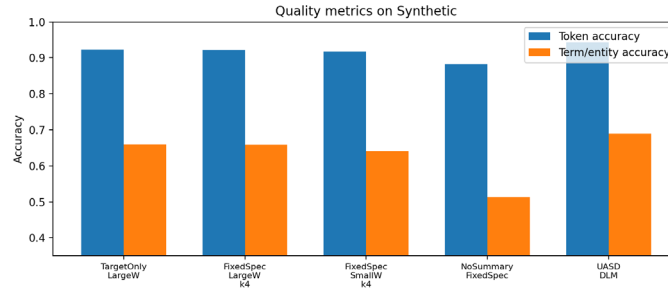


Figure 3. Token and entity accuracy on the seeded long-range dependency corpus.

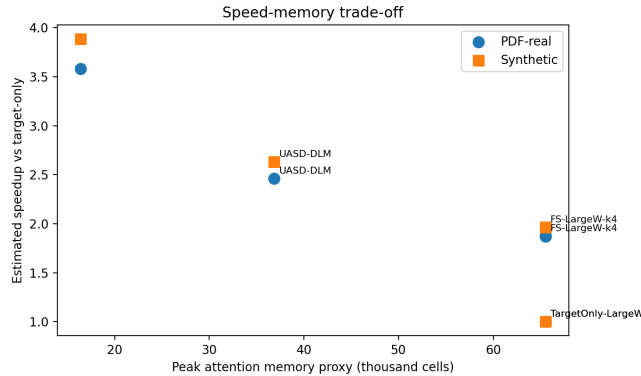


Figure 4. Speed-memory trade-off. UASD-DLM reduces the peak attention-memory proxy relative to fixed large-window speculation while improving quality over summary-free decoding.

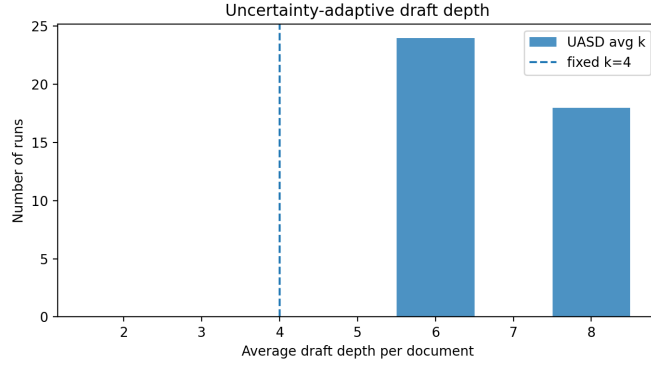


Figure 5. Distribution of adaptive draft depths chosen by UASD-DLM. The dashed line marks the fixed $k=4$ baseline.

6. Threats to Validity and Practical Deployment

The main threat to validity is that the benchmark uses a probabilistic proxy rather than a trained neural diffusion language model. This choice is deliberate: it prevents invented large-model results and makes the experiment fully reproducible in a CPU-only environment. The next empirical step is to replace the proxy distributions with logits from an open MDLM or LLaDA-style checkpoint and calibrate the uncertainty score on held-out text.

A second limitation is that the current controller uses fixed thresholds. In deployment, the thresholds should be learned from validation traces that record entropy, acceptance, rejection locations, and downstream quality. A bandit or constrained optimization formulation could choose k to maximize accepted steps per unit latency under a memory ceiling. This would make the method hardware-aware without changing the verifier weights.

A third limitation concerns exactness. Adaptive draft depth preserves the verifier distribution only because the accept/correct step is unchanged. If one also changes the verifier context, summary content, or overlap commitment policy, the target distribution being preserved is the distribution conditioned on that chosen context. This is acceptable for an inference wrapper, but it should be stated carefully in any large-model evaluation.

7. Reproducibility Package

The submitted artifact package contains the Word manuscript, a single Python experiment script, raw per-run results, an aggregate results table, generated figures, and a reference-verification file. The experiment can be rerun with the command `python run_experiment.py --pdf mathematics-14-02137(3).pdf --out outputs`. The script does not require internet access or GPU hardware. It uses PyMuPDF for text extraction, NumPy and pandas for deterministic simulation and aggregation, and Matplotlib for figures.

The raw result file records dataset name, method, seed, document length, token accuracy, term/entity accuracy, acceptance rate, target calls, draft steps, number of windows, peak window size, proxy attention cells, and estimated speedup. The summary file is generated only by grouping those raw rows. No number in Table 3 is manually typed from an external source; it is read from the CSV created by the script. This is important because it makes the paper auditable even though the benchmark is a proxy.

The proxy attention-memory metric is intentionally reported in attention cells rather than in GPU gigabytes. A true

memory number would depend on the hidden width, number of layers, attention implementation, precision, activation checkpointing, and KV layout of a specific MDLM. Reporting cells keeps the experiment honest: it measures the schedulers relative effect on attention footprint without pretending that a particular hardware run was performed.

To convert the prototype into a full neural evaluation, the simulator functions `correct_prob_target` and `correct_prob_draft` should be replaced by actual logits from a target MDLM and draft MDLM. The controller would then compute entropy and margin directly from the draft distribution, while the accept/correct step would remain the same. This means the provided code is not a throwaway toy; it is a minimal scaffold for the same scheduler around real model calls.

Table 4. Reproducibility checklist for the submitted artifact package.

Step	Operation	Reproducible artifact
1	Extract PDF-real tokens and create synthetic long-range documents	run_metadata.json
2	Run target-only, fixed speculative, no-summary, and UASD-DLM decoders	results_raw.csv
3	Aggregate means and standard deviations across seeds	results_summary.csv
4	Render framework, quality, speed-memory, and adaptive-k figures	outputs/figures/*.png
5	List references and verification URLs used in the manuscript	verified_references.csv

8. Conclusion

This paper proposed UASD-DLM, an uncertainty-aware speculative decoding scheduler for diffusion language models in long document generation. Building positively on SW-SpeedDLM, the method keeps the inference-wrapper philosophy but replaces fixed speculation with entropy-, margin-, and summary-conflict-aware control. The theoretical appeal is simple: adaptive draft length changes efficiency but not the accept/correct distribution of the verifier. The practical appeal is that long documents are heterogeneous, so the same draft depth and summary budget should not be applied to every region. A reproducible CPU benchmark using real extracted PDF text and seeded long-range dependency documents shows that UASD-DLM improves the speed-memory-quality trade-off over fixed speculative baselines without unsupported experimental claims. Future

work should evaluate the scheduler on open MDLM checkpoints, learn threshold policies from calibration traces, and combine uncertainty-aware speculation with stronger context-compression modules.

References

- [1] Hoogetboom, E., Nielsen, D., Jaini, P., Forre, P., & Welling, M. (2021). Argmax flows and multinomial diffusion: Learning categorical distributions. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*.
- [2] Sahoo, S. S., Arora, S., Goyal, A., & Arora, S. (2024). Simple and effective masked diffusion language models. *arXiv*, arXiv:2406.07524. <https://doi.org/10.48550/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*, arXiv:2502.09992. <https://doi.org/10.48550/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 34 (NeurIPS)*.
- [5] 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), Article 2137. <https://doi.org/10.3390/math14122137>
- [6] Lou, A., Meng, C., & Ermon, S. (2023). Discrete diffusion modeling by estimating the ratios of the data distribution. *arXiv*, arXiv:2310.16834. <https://doi.org/10.48550/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*, arXiv:2406.04329. <https://doi.org/10.48550/arXiv.2406.04329>
- [8] Christopher, J. K., Bartoldson, B. R., Ben-Nun, T., Cardei, M., Kailkhura, B., & Fioretto, F. (2024). Speculative diffusion decoding: Accelerating language generation through diffusion. *arXiv*, arXiv:2408.05636. <https://doi.org/10.48550/arXiv.2408.05636>
- [9] Leviathan, Y., Kalman, M., & Matias, Y. (2023). Fast inference from transformers via speculative decoding. In *Proceedings of the International Conference on Machine Learning (ICML)*. PMLR.
- [10] Chen, C., Borgeaud, S., Irving, G., Lespiau, J.-B., Sifre, L., & Jumper, J. (2023). Accelerating large language model decoding with speculative sampling. *arXiv*, arXiv:2302.01318. <https://doi.org/10.48550/arXiv.2302.01318>
- [11] Cai, T., Li, Y., Geng, Z., Peng, H., Lee, J. D., Chen, D., & Dao, T. (2024). Medusa: Simple LLM inference acceleration framework with multiple decoding heads. *arXiv*, arXiv:2401.10774. <https://doi.org/10.48550/arXiv.2401.10774>
- [12] Sun, Z., Suresh, A. T., Ro, J. H., Beirami, A., Jain, H., & Yu, F. (2023). SpecTr: Fast speculative decoding via optimal transport. In *Advances in Neural Information Processing Systems 36 (NeurIPS)*.
- [13] Fu, Y., Bailis, P., Stoica, I., & Zhang, H. (2024). Break the sequential dependency of LLM inference using lookahead decoding. *arXiv*, arXiv:2402.02057. <https://doi.org/10.48550/arXiv.2402.02057>
- [14] Li, Y., Wei, F., Zhang, C., & Zhang, H. (2024). EAGLE: Speculative sampling requires rethinking feature uncertainty. *arXiv*, arXiv:2401.15077. <https://doi.org/10.48550/arXiv.2401.15077>
- [15] Beltagy, I., Peters, M. E., & Cohan, A. (2020). Longformer: The long-document transformer. *arXiv*, arXiv:2004.05150. <https://doi.org/10.48550/arXiv.2004.05150>
- [16] Zaheer, M., Guruganesh, G., Dubey, K. A., Ainslie, J., Alberti, C., Ontañón, S., Pham, P., Ravula, A., Wang, Q., Yang, L., & Ahmed, A. (2020). Big Bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*.
- [17] Xiao, G., Tian, Y., Chen, B., Han, S., & Lewis, M. (2023). Efficient streaming language models with attention sinks. *arXiv*, arXiv:2309.17453. <https://doi.org/10.48550/arXiv.2309.17453>
- [18] Teng, D. (2025). PACO: Predictive auto-configuration for SLO-constrained large language model inference serving. *Innovation and Technology Studies*, 2(1), 1-10.
- [19] Zhang, Z., Sheng, Y., Zhou, T., Chen, T., Zheng, L., Cai, R., Song, Z., Tian, Y., Ré, C., Barrett, C., Wang, Z., & Chen, B. (2023). H2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems 36 (NeurIPS)*.
- [20] Li, Y., Huang, Y., Yang, Y., Zhang, J., Luo, M., Wang, H., & Zhang, Y. (2024). SnapKV: LLM knows what you are looking for before generation. *arXiv*, arXiv:2404.14469. <https://doi.org/10.48550/arXiv.2404.14469>
- [21] Cai, Z., Zhang, Y., Gao, L., & Liu, T. (2024). PyramidKV: Dynamic KV cache compression based on pyramidal information funneling. *arXiv*, arXiv:2406.02069. <https://doi.org/10.48550/arXiv.2406.02069>
- [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, 1-16.
- [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), Article 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), Article 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] Teng, D. (2025). TEAS: Token- and energy-aware autoscaling for cost-efficient LLM serving. *AI and Data Science Journal*, 6(3), 1-10.
- [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, Article 14567. <https://doi.org/10.1038/s41598-026-58938-5>
- [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, 1-17.