Any-Order in Theory, Left-to-Right in Practice: The Autoregressive Underpinnings of dLLMs

Dayuan Zhao
Yuanpei College, Peking University
2200017742@stu.pku.edu.cn

Yiduo Guo
Tencent AI Lab
guoyd1999@gmail.com

Abstract

While autoregressive models (ARMs) dominate natural language processing, their strictly left-to-right decoding inherently limits global context utilization. Diffusion language models (DLMs) have recently emerged as a promising paradigm shift, theoretically offering bidirectional attention, parallel decoding, and the flexibility of any-order generation. However, the nature of this generative flexibility remains heavily under-explored. In this work, we systematically investigate the interplay between any-order flexibility, task performance, and inference efficiency. By quantifying deviations from autoregressive (AR) generation order using Normalized Edit Distance (NED) across reasoning, code, and commonsense tasks, we find that **current DLMs rely heavily on near left-to-right generation pathways to achieve high performance.** To weaken this coupling, we demonstrate that the generation block size acts as a key regulator of decoding order. Consequently, we propose a novel mitigation strategy: training the model with a large block size to grant it a vastly expanded exploration space, while performing inference with a comparatively smaller block size to minimize computational overhead. Ultimately, our methodology increases deviation from left-to-right generation while preserving strong task accuracy and inference efficiency.

1 Introduction

Natural language processing has long been dominated by autoregressive models (ARMs) (OpenAI et al., 2024; Grattafiori et al., 2024; Yang et al., 2025), whose strict left-to-right generation limits flexibility and global context utilization (Berglund et al., 2024). Recently, diffusion language models (DLMs) (Sahoo et al., 2024; Lou et al., 2024) have emerged as a powerful alternative, adapting the gradual denoising processes of vision diffusion models (Rombach et al., 2022; Brooks et al., 2024) to text generation. By naturally incorporating bidirectional attention throughout the generative process, DLMs capture global contextual dependencies and enable any-order generation (Ye et al., 2025). Furthermore, because they iteratively refine the entire sequence simultaneously, DLMs natively support parallel decoding. When combined with acceleration strategies like KV caching, they can ultimately outpace the generation speeds of their autoregressive counterparts (Wu et al., 2025).

Despite theoretical claims of any-order flexibility (Ye et al., 2025; Cheng et al., 2025), rigorous empirical validation remains lacking, prompting our investigation into how effectively current DLMs use this flexibility. Across reasoning, code, and commonsense tasks, we find that the evaluated DLMs achieve their strongest and most stable performance with near-left-to-right decoding sequences. Specifically, our analysis reveals a strong positive correlation between AR similarity and model accuracy, showing that the evaluated DLMs favor left-to-right decoding and suffer sharp performance drops as their trajectories deviate from strict AR patterns.

To weaken this coupling between decoding order and performance, we investigate the role of the generation block size. While small block sizes constrain the decoding space and encourage AR-like behavior, scaling up the block size reduces AR similarity and permits more varied generation orders. However, our study reveals a trade-off between block size, task performance, and computational efficiency: simply increasing the block size during inference heavily degrades both accuracy and throughput. To navigate this, we propose a novel asymmetric block-scaling strategy: **explicitly training the DLM with a large block size to expand its contextual exploration space, while executing inference with a comparatively smaller block size to bypass extraneous computational overhead.** This strategy successfully

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 50571–50580
October 24-29, 2026 ©2026 Association for Computational Linguistics

increases deviation from left-to-right generation while preserving strong task performance and inference efficiency.

2 Related Work

Diffusion Language Models Diffusion language models (DLMs), specifically masked diffusion models (MDMs), have emerged as a compelling alternative to the prevalent autoregressive models (ARMs). By utilizing bidirectional attention, these models avoid the causal constraint of ARMs and support new features such as any-order generation (Liu et al., 2025; Li et al., 2025). With foundational works such as SEDD (Lou et al., 2024) and MDLM (Sahoo et al., 2024) verifying the efficacy of this approach on a smaller scale, recent advancements such as LLaDA (Nie et al., 2025) and Dream (Ye et al., 2025) have successfully scaled these architectures to match the sizes of mainstream ARMs with rivaling performance. Recent research has also revitalized the semi-autoregressive paradigm introduced by SSD-LM (Han et al., 2023). By confining diffusion to local windows while maintaining global AR dependencies, modern approaches like BD3-LM (Arriola et al., 2025) and SDAR (Cheng et al., 2025) reintroduce KV caching efficiency for MDMs. The progress has notably extended to the commercial landscape, exemplified by models such as Mercury (Labs et al., 2025) and Gemini Diffusion (Google DeepMind, 2025). However, despite these architectural shifts, methods to reduce AR-like generation while preserving performance and efficiency remain largely unexplored—a gap this work addresses.

Mechanisms of Diffusion Language Models Recent investigations into DLMs have provided a comprehensive understanding of their underlying mechanisms. Several works have observed that conditional independence in parallel decoding leads to globally inconsistent sequences (Jin et al., 2025) and hinder rigorous chain-of-thought reasoning (Chen et al., 2025). Furthermore, assumptions like uniform corruption and block diffusion have been linked to structural challenges, including frequency collapse, early saturation (Jin et al., 2025), and elevated hallucination rates compared to ARMs (Guo and Tan, 2026). Despite these challenges, DLMs are also claimed to have their own theoretical and empirical advantages. The non-sequential nature of MDMs enables adaptive generation speeds and continuous sequence refinement (Zhong et al., 2026), achieving near-optimal token error rates for fluency-focused tasks (Feng et al., 2025). Beyond inference, DLMs consistently outperform AR models in data-constrained settings (Prabhudesai et al., 2025), underscoring their potential as a highly capable alternative at scale (von Rütte et al., 2026). Recent studies also analyze decoding order: Huang et al. (2026) identify biases induced by uncertainty-based decoding, while Zhong et al. (2026) compare the natural generation orders of correct and incorrect LLaDA2.0 outputs. Because these observed orders are jointly determined by the model and its inference strategy, they do not isolate the model's order preference. We instead impose controlled orders and measure the resulting accuracy–order relationship across four full and block DLM families, separating our work from previous works concerning generation orders.

3 The Illusion of Any-Order Flexibility: The Empirical Reliance on Autoregressive Decoding

To investigate whether DLMs genuinely achieve the theoretical promise of any-order generation—effectively decoupling high performance from traditional left-to-right decoding—we examine the relationship between a generation trajectory's autoregressive (AR) similarity and downstream accuracy. Specifically, we quantify the deviation from strict AR decoding using the **Normalized Edit Distance (NED)**, derived from the standard Levenshtein distance. Formally, the metric is defined as:

$$\mathrm{NED}(a, b) = \frac{\mathrm{Lev}(a, b)}{\max(|a|, |b|)} \tag{1}$$

where $|a|$ and $|b|$ represent the lengths of sequences a and b, respectively. The Levenshtein distance, denoted as $\mathrm{Lev}(a, b)$, measures the minimum number of single-element insertions, deletions, or substitutions required to transform sequence a into sequence b. In our context, a represents the sequence of token indices ordered by their generation steps, while $b = (0, 1, \ldots, |a| - 1)$ represents the strict AR order. Consequently, NED provides a normalized measure of how significantly the actual generation trajectory deviates from a left-to-right pattern. Although we discuss AR similarity conceptually, NED has the opposite direction: a higher NED in-

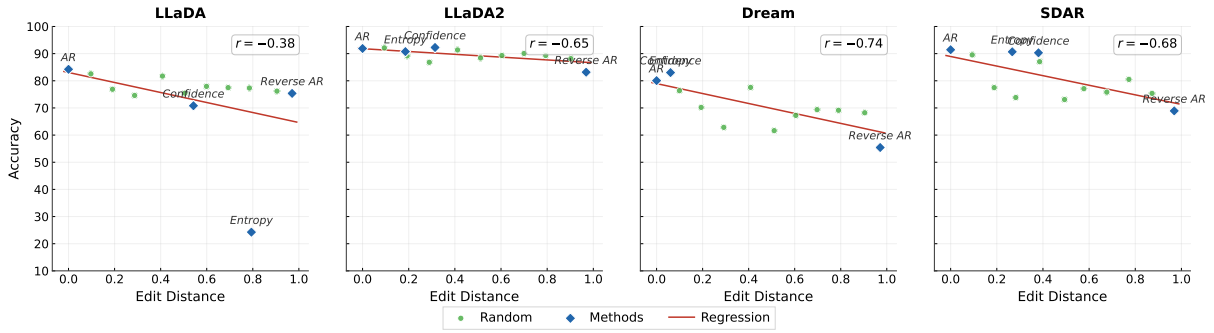


Figure 1: **Performance vs. Normalized Edit Distance (NED) on GSM8K.** We evaluate LLaDA, Dream, SDAR, and LLaDA2.0, comparing specific decoding methods (blue) against sampled generation orders (green).

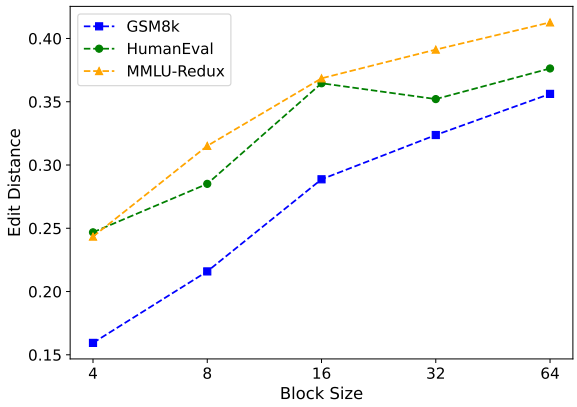


Figure 2: **Average normalized edit distances (NEDs) for SDAR with different block sizes.** SDAR trained with block sizes from 4 to 64 is evaluated on GSM8K, HumanEval, and MMLU-Redux.

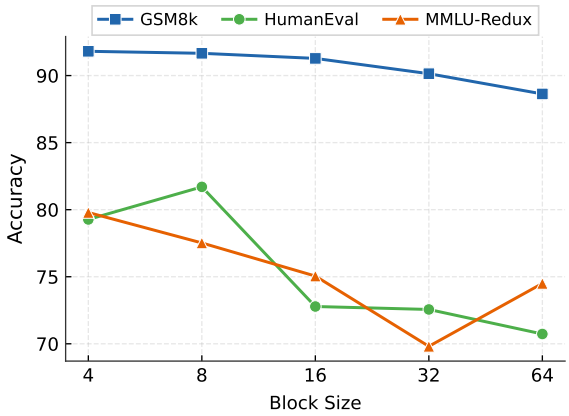


Figure 3: **Impact of Block Size on SDAR Performance.** We compare the performance of SDAR across different block sizes on GSM8K, HumanEval, and MMLU-Redux. Configurations are consistent with Figure 5.

dicates lower similarity to AR order. We justify this metric choice in Appendix B.

Setting: Our evaluation spans a broad spectrum of models, tasks, and decoding strategies. We benchmark LLaDA (Nie et al., 2025), Dream (Ye et al., 2025), SDAR (Cheng et al., 2025), and LLaDA2.0 (Bie et al., 2025) across a diverse suite of tasks, including GSM8K (Cobbe et al., 2021), HumanEval (Chen et al., 2021), and MMLU-Redux (Gema et al., 2025). We assess standard DLM techniques, such as confidence-based and entropy-based sampling, alongside explicit positional constraints like forced AR and reverse AR generation. Finally, to provide a granular analysis of the decoding landscape, we introduce synthetically sampled any-order patterns calibrated to target NED values ranging from 0.1 to 0.9 (see Appendix A for the exact generation orders).

As detailed in Figure 1 and 4, the tested models exhibit a stark contrast at the extremes of decoding strategies: **confidence-based sampling achieves the highest performance but exhibits high AR similarity, whereas forced reverse AR generation results in severely degraded accuracy**. Our synthetically sampled orders show the same empirical relationship. The Pearson correlation coefficients, most of which range approximately from -0.65 to -0.9, indicate that **AR similarity and model performance are strongly positively correlated across the evaluated models and settings**.

4 Block Size as a Regulator of Decoding Order

Hypothesis We hypothesize that the generation block size is a key regulator of the model's decoding order. Specifically, expanding the block size broadens the contextual canvas available at each diffusion step, thereby facilitating non-autoregressive generation. To test this, we conduct experiments using the SDAR (Cheng et al., 2025) framework, analyzing its natively trained variants across vary-

Table 1: Comparison of NED across different datasets. Results are reported as mean ± standard deviation over three repeated evaluation runs; this protocol also applies to the subsequent result tables. A value of ±0.00 indicates variation below the displayed precision, with small residual differences arising from numerical GPU non-determinism.

Model	GSM8K	MATH-500	MMLU-Redux	HumanEval	MBPP	GPQA-Diamond	Avg
SDAR-b64	0.36 ± 0.00	0.15 ± 0.00	0.25 ± 0.00	0.21 ± 0.00	0.35 ± 0.00	0.38 ± 0.00	0.28
SDAR-b64* (Train 64, Infer 64)	0.35 ± 0.00	0.11 ± 0.00	0.24 ± 0.00	0.22 ± 0.00	0.35 ± 0.00	0.35 ± 0.00	0.27
SDAR (Train 128, Infer 64)	0.34 ± 0.00	0.30 ± 0.00	0.45 ± 0.00	0.31 ± 0.00	0.48 ± 0.00	0.52 ± 0.00	0.40
SDAR (Train 128, Infer 128)	0.36 ± 0.00	0.32 ± 0.00	0.47 ± 0.00	0.34 ± 0.00	0.47 ± 0.00	0.54 ± 0.00	0.42
SDAR (Train 256, Infer 64)	0.33 ± 0.00	0.28 ± 0.00	0.39 ± 0.00	0.31 ± 0.00	0.49 ± 0.00	0.66 ± 0.00	0.41
SDAR (Train 256, Infer 128)	0.34 ± 0.00	0.30 ± 0.00	0.40 ± 0.00	0.32 ± 0.00	0.50 ± 0.01	0.65 ± 0.00	0.42
LLaDA	0.54 ± 0.00	0.76 ± 0.00	0.64 ± 0.00	0.37 ± 0.00	0.43 ± 0.00	0.62 ± 0.00	**0.56**
Dream	0.06 ± 0.00	0.06 ± 0.00	0.08 ± 0.00	0.02 ± 0.00	0.07 ± 0.00	0.11 ± 0.00	0.07
LLaDA2.0	0.31 ± 0.00	0.27 ± 0.00	0.36 ± 0.00	0.14 ± 0.00	0.23 ± 0.00	0.29 ± 0.00	0.27

ing block sizes. We also consider LLaDA2.0 as a baseline for comparison.

As Figure 2 illustrates, increasing the block size directly correlates with a higher normalized edit distance. This shows that **larger block sizes enable generation trajectories that deviate further from AR order**.

Performance-Efficiency Trade-off in Block Scaling While larger block sizes effectively mitigate autoregressive bias, they may degrade task performance and inference efficiency. To evaluate this trade-off, we test SDAR (Cheng et al., 2025) and LLaDA2.0 (Bie et al., 2025) on established tasks using their default confidence sampling methods, tracking task accuracy, tokens per forward pass (TPF), and tokens per second (throughput). From Figure 3, we find that simply **increasing the block size during inference gradually degrades task performance. Also, efficiency experiences a slight drop as block size increases, even for models natively trained at that scale.** This trade-off is expected: larger block sizes incur computational overhead by forcing the model to generate extraneous padding or stop tokens for shorter tasks (e.g., using a 128-token block for a 40-token task).

Our Method Based on the above observations, we propose our strategy: **training with a large block size while inferring with a comparatively smaller block size**. On the one hand, training with a large block size (e.g., 256) makes the model learn to unmask tokens across a broader context space, encouraging generation orders that deviate further from left-to-right. On the other hand, executing inference with a comparatively smaller block size (e.g., 64 or 128) allows the model to avoid unnecessary computation while preserving the increased deviation enabled by large-block training. In the following section, we verify our strategy's effectiveness and compare it against other diffusion language models.

5 Experiments

Setting We fine-tune our base model, SDAR-8B-Chat with block size 64 (SDAR-b64), using training block sizes of 128 and 256 on a 5B-token instruction dataset (1 epoch, learning rate 5e-5), and evaluate both at inference block sizes 64 and 128. Baselines include LLaDA, LLaDA2.0-mini, Dream, SDAR-b64, and SDAR-b64* (SDAR-b64 fine-tuned on the same 5B-token instruction dataset with block size 64 to serve as an ablation baseline). We omit evaluating the untrained SDAR-b64 at block size 128 due to severe performance degradation. Benchmarks include GSM8K (Cobbe et al., 2021), MATH-500 (Lightman et al., 2023), GPQA (Rein et al., 2023), MMLU-Redux (Gema et al., 2025), HumanEval (Chen et al., 2021), and MBPP (Austin et al., 2021) (dataset and inference details are in Appendix D).

Results Across the full training/inference block-size comparison in Tables 1, 6, and 7, larger training blocks consistently increase deviation from left-to-right generation. At the same inference block size of 64, increasing the training block size from 64 to 256 raises average NED from 0.270 to 0.408 and average accuracy from 67.5% to 70.0%, while maintaining throughput (45.3 vs. 46.6 tokens/s). Changing only the inference block size produces much smaller NED differences: 0.402 to 0.415 for the model trained at 128, and 0.408 to 0.418 for the model trained at 256. The train-256/infer-128 configuration also surpasses SDAR-b64, LLaDA, Dream, and SDAR-b64* in task accuracy across all six benchmarks, trailing only the much larger 16B-

parameter LLaDA2.0, while maintaining slightly higher throughput. Thus, **large-block training weakens the observed coupling between decoding order and performance while preserving strong task performance and efficiency**.

6 Conclusion

In this work, we demonstrate that the current DLMs evaluated here achieve their best performance with near-left-to-right decoding and degrade as their trajectories deviate from that order. We identify block size as a key regulator of decoding order and introduce an asymmetric block-scaling strategy: training on expansive block sizes to broaden exploration while performing inference on smaller ones. Our evaluations confirm that this approach significantly enhances the any-order flexibility of block diffusion language models like SDAR while keeping its performance.

References

Marianne Arriola, Aaron Gokaslan, Justin T. Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. 2025. Block diffusion: Interpolating between autoregressive and diffusion language models. *Preprint*, arXiv:2503.09573.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and 1 others. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Lukas Berglund, Meg Tong, Max Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. 2024. The reversal curse: Llms trained on "a is b" fail to learn "b is a". *Preprint*, arXiv:2309.12288.

Tiwei Bie, Maosong Cao, Kun Chen, Lun Du, Mingliang Gong, Zhuochen Gong, Yanmei Gu, Jiaqi Hu, Zenan Huang, Zhenzhong Lan, Chengxi Li, Chongxuan Li, Jianguo Li, Zehuan Li, Huabin Liu, Lin Liu, Guoshan Lu, Xiaocheng Lu, Yuxin Ma, and 12 others. 2025. Llada2.0: Scaling up diffusion language models to 100b. *Preprint*, arXiv:2512.15745.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, and 1 others. 2024. Video generation models as world simulators. *OpenAI*.

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

Qiguang Chen, Hanjing Li, Libo Qin, Dengyun Peng, Jinhao Liu, Jiangyi Wang, Chengyue Wu, Xie Chen, Yantao Du, and Wanxiang Che. 2025. Beyond surface reasoning: Unveiling the true long chain-of-thought capacity of diffusion large language models. *Preprint*, arXiv:2510.09544.

Shuang Cheng, Yihan Bian, Dawei Liu, Linfeng Zhang, Qian Yao, Zhongbo Tian, Wenhai Wang, Qipeng Guo, Kai Chen, Biqing Qi, and Bowen Zhou. 2025. Sdar: A synergistic diffusion-autoregression paradigm for scalable sequence generation. *Preprint*, arXiv:2510.06303.

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

Guhao Feng, Yihan Geng, Jian Guan, Wei Wu, Liwei Wang, and Di He. 2025. Theoretical benefit and limitation of diffusion language model. *Preprint*, arXiv:2502.09622.

Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, Claire Barale, Robert McHardy, Joshua Harris, Jean Kaddour, Emile van Krieken, and Pasquale Minervini. 2025. Are we done with mmlu? *Preprint*, arXiv:2406.04127.

Google DeepMind. 2025. Gemini diffusion: Google deepmind's experimental research model. `https://deepmind.google/models/gemini-diffusion/`. Accessed: 2025-12-20.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Zhengnan Guo and Fei Tan. 2026. Lost in diffusion: Uncovering hallucination patterns and failure modes in diffusion large language models. *Preprint*, arXiv:2604.10556.

Xiaochuang Han, Sachin Kumar, and Yulia Tsvetkov. 2023. Ssd-lm: Semi-autoregressive simplex-based diffusion language model for text generation and modular control. *Preprint*, arXiv:2210.17432.

Pengcheng Huang, Tianming Liu, Zhenghao Liu, Yukun Yan, Shuo Wang, Tong Xiao, Zulong Chen, and Maosong Sun. 2026. Empirical analysis of decoding biases in masked diffusion models. In *Proceedings*

of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 6853–6876, San Diego, California, United States. Association for Computational Linguistics.

Ziqi Jin, Bin Wang, Xiang Lin, Lidong Bing, and Aixin Sun. 2025. On the role of discreteness in diffusion llms. *Preprint*, arXiv:2512.22630.

Inception Labs, Samar Khanna, Siddhant Kharbanda, Shufan Li, Harshit Varma, Eric Wang, Sawyer Birnbaum, Ziyang Luo, Yanis Miraoui, Akash Palrecha, Stefano Ermon, Aditya Grover, and Volodymyr Kuleshov. 2025. Mercury: Ultra-fast language models based on diffusion. *Preprint*, arXiv:2506.17298.

Tianyi Li, Mingda Chen, Bowei Guo, and Zhiqiang Shen. 2025. A survey on diffusion language models. *Preprint*, arXiv:2508.10875.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Anji Liu, Xuejie Liu, Dayuan Zhao, Mathias Niepert, Yitao Liang, and Guy Van den Broeck. 2025. Tractable transformers for flexible conditional generation. *Preprint*, arXiv:2502.07616.

Aaron Lou, Chenlin Meng, and Stefano Ermon. 2024. Discrete diffusion modeling by estimating the ratios of the data distribution. *Preprint*, arXiv:2310.16834.

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

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Mihir Prabhudesai, Mengning Wu, Amir Zadeh, Katerina Fragkiadaki, and Deepak Pathak. 2025. Diffusion beats autoregressive in data-constrained settings. *Preprint*, arXiv:2507.15857.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Jackson, Richard Yuanzhe Pang, Elton Dirani, Jha Mikhailova, and 1 others. 2023. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. *Preprint*, arXiv:2112.10752.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. 2024. Simple and effective masked diffusion language models. *Preprint*, arXiv:2406.07524.

Dimitri von Rütte, Janis Fluri, Omead Pooladzandi, Bernhard Schölkopf, Thomas Hofmann, and Antonio Orvieto. 2026. Scaling behavior of discrete diffusion language models. *Preprint*, arXiv:2512.10858.

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

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

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

Yangyang Zhong, Yanmei Gu, Zhengqing Zang, Xiaomeng Li, Yuqi Ding, Xibei Jia, Yuting Shen, Zhenzhong Lan, Liwang Zhu, Weiping Liu, Junlin Zhou, Haisheng Liu, Zhong Xin Yu, Pengxin Luo, Donglian Qi, Yunfeng Yan, and Junbo Zhao. 2026. Parallelism and generation order in masked diffusion language models: Limits today, potential tomorrow. *Preprint*, arXiv:2601.15593.

Limitations

Our findings are primarily validated on models up to 8 billion parameters; scalability to massive foundation models remains an open question. Additionally, our evaluation focuses on reasoning and objective tasks, leaving the effects of forced any-order generation on open-ended creative writing unexplored. Our training strategy is evaluated only on SDAR and directly applies to block DLMs; validation on other block-DLM families remains future work.

A Sampled Generation Order

For the experiments in Figures 1 and 4, we employ sampled generation orders with NED values spanning from approximately 0.1 to 0.9. The specific permutations used for these evaluations are detailed in Table 2.

Table 2: **Sampled block generation orders.** This table details the specific permutations used in our evaluations, covering normalized edit distances (NEDs) approximately evenly spaced between 0.1 and 0.9.

NED	Sampled Order
0.0938	0, 1, 9, 3, 4, 5, 2, 7, 8, 6, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31
0.1875	0, 1, 2, 3, 4, 25, 8, 21, 6, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 7, 22, 23, 24, 5, 26, 27, 28, 29, 30, 31
0.2813	0, 1, 21, 7, 4, 5, 6, 11, 2, 9, 29, 3, 12, 13, 14, 15, 16, 17, 18, 19, 20, 8, 10, 23, 24, 25, 26, 27, 28, 22, 30, 31
0.4063	0, 1, 2, 3, 4, 5, 8, 7, 17, 9, 15, 13, 20, 11, 14, 10, 16, 6, 26, 30, 12, 21, 22, 23, 24, 25, 18, 27, 28, 29, 19, 31
0.5000	0, 21, 15, 4, 3, 29, 6, 7, 8, 11, 10, 9, 26, 13, 14, 2, 16, 17, 20, 19, 28, 1, 22, 23, 24, 25, 12, 27, 30, 5, 18, 31
0.5938	15, 14, 2, 3, 0, 5, 10, 7, 16, 4, 6, 30, 12, 13, 21, 9, 19, 17, 27, 8, 20, 1, 22, 18, 26, 25, 24, 11, 28, 29, 23, 31
0.6875	5, 1, 23, 6, 4, 27, 31, 7, 28, 10, 9, 11, 13, 12, 8, 15, 16, 22, 14, 21, 20, 3, 0, 2, 26, 25, 17, 19, 18, 29, 30, 24
0.7813	4, 1, 5, 24, 9, 26, 21, 30, 27, 17, 14, 15, 12, 13, 0, 20, 16, 25, 8, 19, 28, 18, 22, 6, 31, 2, 10, 23, 3, 29, 7, 11
0.9063	22, 2, 1, 23, 19, 17, 10, 9, 29, 15, 13, 8, 31, 6, 21, 18, 16, 27, 11, 5, 14, 20, 7, 28, 24, 3, 12, 0, 25, 4, 30, 26

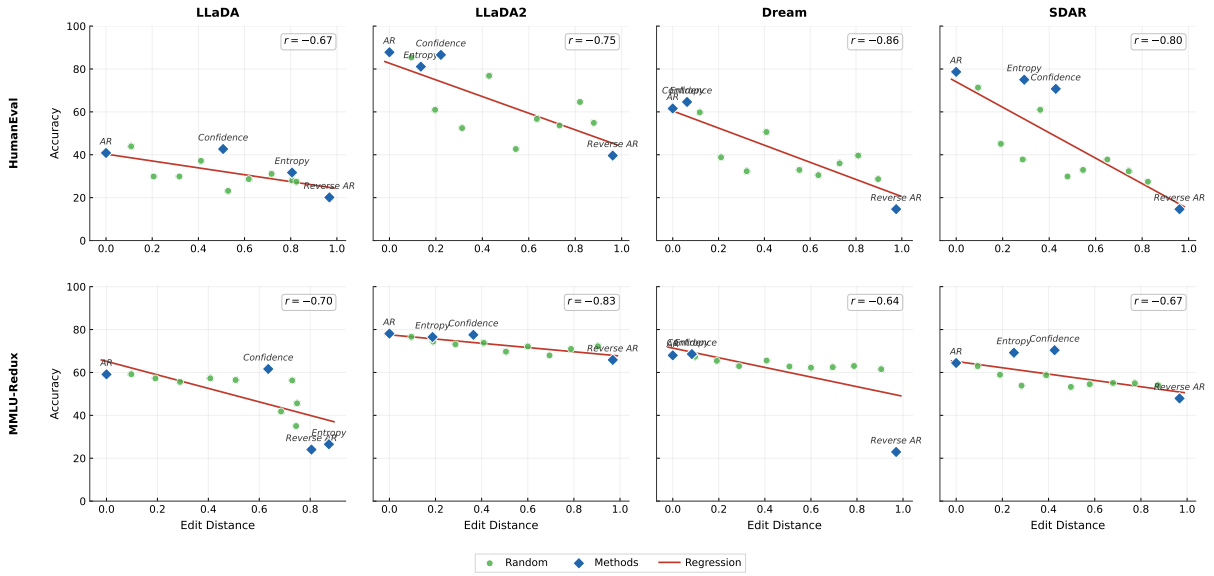


Figure 4: **Performance vs. Normalized Edit Distance (NED) on HumanEval and MMLU-Redux.** Following the same experimental setup as in Figure 1, these results further demonstrate that lower edit distances consistently correlate with improved overall accuracy across different benchmarks.

B Comparison of Generation-Order Metrics

We compare NED with Kendall's τ and Spearman's ρ, two natural rank-based alternatives. For each block size, Table 3 reports the metric span between strict AR order and reverse-AR order within each block, the largest deviation permitted by the block constraint. NED retains a substantially wider usable range, particularly at small block sizes.

Table 3: Usable metric range under different SDAR block-size constraints.

Block size	NED	Kendall's τ	Spearman's ρ
4	0.7538	0.0365	0.0013
8	0.8784	0.0861	0.0054
16	0.9410	0.1952	0.0239
32	0.9720	0.3781	0.0849
64	0.9876	0.7583	0.3352

Table 4 further compares the metrics on actual SDAR generations for GSM8K. While NED distinguishes the block-size settings, Kendall's τ and Spearman's ρ remain close to their AR-similarity ceiling of 1, making them less discriminative in this setting.

Table 4: Generation-order metric values on GSM8K across SDAR block sizes.

Model	Block size	NED	Kendall's τ	Spearman's ρ
SDAR-b4	4	0.1590	0.9965	0.9999
SDAR-b8	8	0.2161	0.9919	0.9996
SDAR-b16	16	0.2889	0.9824	0.9984
SDAR-b32	32	0.3239	0.9749	0.9967
SDAR-b64	64	0.3547	0.9616	0.9917

C Influence of Block Size on SDAR and LLaDA2.0-mini Performance

We demonstrate how block size influences the performance and efficiency of SDAR in Figures 3 and 5 and LLaDA2.0-mini in Figures 6 and 7.

D Benchmark and Model Inference Details

We list the details about the 6 used datasets in Table 5.

Inference For LLaDA and Dream, we follow the inference code of the Dream model to generate the responses (https://github.com/DreamLM/Dream). The number of sampling steps is set equal to the generation length, which is 64 for GPQA and 512 for all other datasets. LLaDA employs a low-confidence remasking strategy, whereas Dream remasks high-entropy tokens. For SDAR and LLaDA2.0, we use block diffusion with dynamic low-confidence remasking for both, with threshold set as 0.95, and a maximum generation length of 2048. Furthermore, their block sizes are configured to 64 for SDAR and 32 for LLaDA2.0. For SDAR, we use its source code https://github.com/JetAstra/SDAR/blob/main/generate.py. For LLaDA2.0-mini, we use its original modeling code https://huggingface.co/inclusionAI/LLaDA2.0-mini/blob/main/modeling_llada2_moe.py#L1233. In the whole paper, the inference batch size is set as 1 for fair comparison.

E Task Performance and Efficiency Results

We list the accuracy results in Table 6 and throughput results in Table 7.

F Computational Budget

Training SDAR with our method takes 20 hours on 100 NVIDIA H20 GPUs. Our training is conducted with a total batch size of 400, sequence length 2048, and the training framework is built with DeepSpeed.

Table 5: **Overview of Evaluated Benchmarks.** A summary of the six datasets utilized in our experiments, detailing their primary domains, evaluated test set sizes, and performance metrics.

Dataset	Domain	Test Size	Metric
GSM8K	Grade School Mathematics	1,319	Accuracy
MATH-500	Advanced Mathematics	500	Accuracy
GPQA Diamond	Graduate-Level Science	198	Accuracy
MMLU-Redux	Multidisciplinary QA	5,700	Accuracy
HumanEval	Python Code Generation	164	Pass@1
MBPP	Basic Python Programming	257	Pass@1

Table 6: Comparison of Accuracy (%) across different datasets.

Model	GSM8K	MATH-500	MMLU-Redux	HumanEval	MBPP	GPQA-Diamond	Avg
SDAR-b64	88.6 ± 0.2	68.1 ± 0.8	73.9 ± 0.4	66.1 ± 2.1	49.8 ± 1.0	34.1 ± 1.1	63.4
SDAR-b64* (Train 64, Infer 64)	89.1 ± 0.1	70.5 ± 0.2	75.1 ± 0.1	70.1 ± 0.1	65.2 ± 1.3	35.2 ± 0.3	67.5
SDAR (Train 128, Infer 64)	90.5 ± 0.0	69.9 ± 0.0	64.0 ± 0.0	70.5 ± 0.0	67.7 ± 0.0	35.2 ± 0.0	66.3
SDAR (Train 128, Infer 128)	89.9 ± 0.0	70.2 ± 0.0	63.6 ± 0.0	69.9 ± 0.0	67.1 ± 0.0	35.4 ± 0.0	66.0
SDAR (Train 256, Infer 64)	91.3 ± 0.0	71.3 ± 0.0	77.0 ± 0.0	74.8 ± 0.0	69.3 ± 0.0	36.4 ± 0.0	**70.0**
SDAR (Train 256, Infer 128)	90.3 ± 0.2	71.6 ± 1.2	75.2 ± 0.4	73.6 ± 0.9	68.9 ± 1.8	35.9 ± 0.5	69.3
LLaDA	71.0 ± 0.0	27.6 ± 0.0	62.1 ± 0.0	42.1 ± 0.0	42.0 ± 0.0	31.8 ± 0.0	46.1
Dream	81.2 ± 0.0	49.2 ± 0.0	67.7 ± 0.0	59.2 ± 0.0	63.4 ± 0.0	34.3 ± 0.0	59.2
LLaDA2.0	92.3 ± 0.0	76.6 ± 0.0	77.6 ± 0.0	88.4 ± 0.0	78.2 ± 0.0	38.9 ± 0.0	75.3

Table 7: Throughput of different models and SDAR configurations across tasks.

Model	GSM8K	MATH-500	MMLU-Redux	HumanEval	MBPP	GPQA-Diamond	Avg
SDAR-b64	57.2 ± 0.5	82.6 ± 0.7	31.5 ± 0.1	44.1 ± 0.5	19.2 ± 0.1	20.7 ± 0.1	42.6
SDAR-b64* (Train 64, Infer 64)	47.7 ± 0.5	60.3 ± 0.2	32.2 ± 0.3	67.5 ± 1.3	48.4 ± 2.7	15.4 ± 0.1	45.3
SDAR (Train 128, Infer 64)	50.2 ± 0.5	62.4 ± 0.7	34.7 ± 0.4	68.3 ± 0.1	40.7 ± 2.2	15.9 ± 0.0	45.4
SDAR (Train 128, Infer 128)	49.9 ± 0.3	63.7 ± 0.2	33.0 ± 0.1	65.9 ± 0.9	40.8 ± 2.6	15.7 ± 0.1	44.8
SDAR (Train 256, Infer 64)	47.6 ± 0.5	59.1 ± 0.3	33.9 ± 0.4	62.9 ± 0.5	51.9 ± 2.3	24.1 ± 0.1	46.6
SDAR (Train 256, Infer 128)	50.8 ± 0.4	62.6 ± 0.6	33.7 ± 0.3	69.0 ± 1.9	44.4 ± 1.5	20.9 ± 4.6	**46.9**
LLaDA	16.2 ± 0.0	15.8 ± 0.0	15.1 ± 0.0	9.8 ± 0.0	11.6 ± 0.0	9.1 ± 0.0	12.9
Dream	14.2 ± 0.0	13.8 ± 0.0	13.1 ± 0.0	12.3 ± 0.0	14.1 ± 0.0	5.5 ± 0.0	12.2
LLaDA2.0	39.9 ± 0.3	47.4 ± 0.2	29.4 ± 0.1	71.2 ± 0.0	38.1 ± 0.0	28.7 ± 0.0	42.5

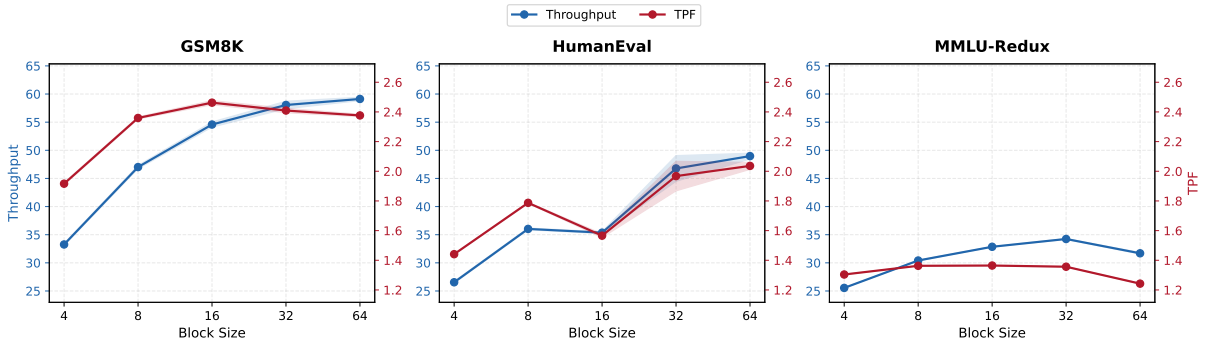


Figure 5: **Impact of Block Size on SDAR Efficiency.** We compare the efficiency of SDAR by measuring throughput (blue) and TPF (red) across different block sizes on GSM8K, HumanEval, and MMLU-Redux. Note that SDAR evaluations utilize models trained on their corresponding block sizes.

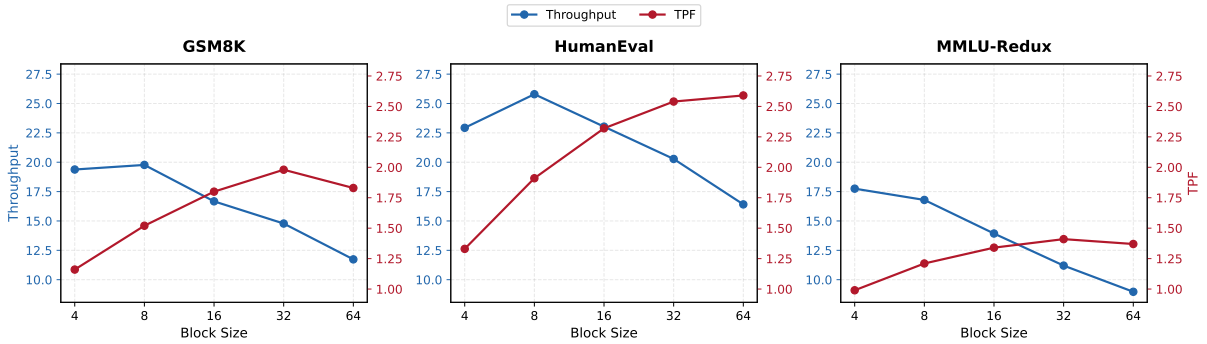


Figure 6: **Impact of Block Size on LLaDA2.0-mini Efficiency.** We compare the efficiency of LLaDA2.0-mini by measuring throughput (blue) and TPF (red) across different block sizes on GSM8K, HumanEval, and MMLU-Redux. Different from Figure 5, we use the same model throughout all evaluations, for LLaDA2.0-mini is trained with block size 32 and doesn't have alternate versions trained with other block sizes.

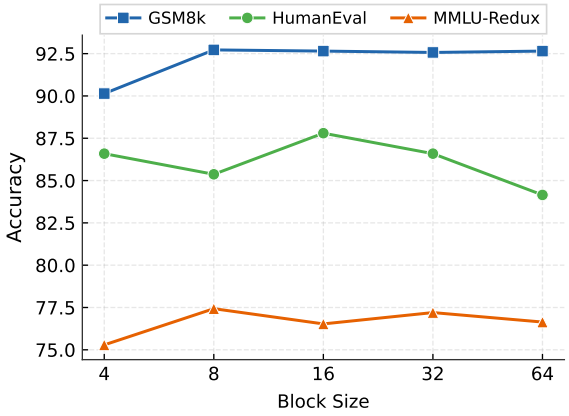


Figure 7: **Impact of Block Size on LLaDA2.0-mini Performance.** We compare the performance of LLaDA2.0-mini across different block sizes on GSM8K, HumanEval, and MMLU-Redux. Other configurations are consistent with Figure 6.