In-Context Learning in Diffusion Language Models

Julianna Piskorz[1], **Cristina Pinneri**[2], **Alvaro Correia**[2], **Motasem Alfarra**[2], **Christos Louizos**[2]
[1]University of Cambridge [2]Qualcomm AI Research*

Abstract

While the masked diffusion loss of Diffusion Language Models (DLMs) offers opportunities for faster inference, its impact on inductive biases and In-Context Learning (ICL) capabilities of language models remains largely under-explored. We evaluate DLMs on a range of ICL benchmarks and find that they match the performance of Autoregressive Language Models (ARLMs) despite being trained with a non-autoregressive loss. We further explore whether the alternative loss function allows DLMs to overcome some of the local processing tendencies typical of ARLMs. We conduct controlled experiments targeting two biases: recency and left-directedness. Our results indicate that DLMs exhibit both, though less strongly than ARLMs. These findings reveal that **DLMs inherit ARLM-like locality biases** despite their distinct training objective.

1 Introduction

Diffusion Language Models (DLMs) have recently emerged as an alternative to Autoregressive Language Models (ARLMs) owing to their potential for parallel decoding and faster inference [23, 18, 36]. Unlike ARLMs, which are trained to predict the next token given a left-to-right context, DLMs are trained to iteratively unmask tokens across the entire sequence. The effect of this difference in training objective on models' ability to learn from context and their inductive biases remains largely under-explored.

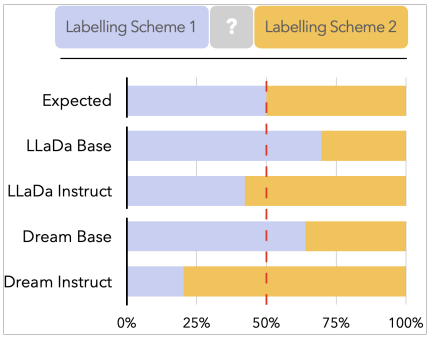

Figure 1: **Directional bias in DLMs.** Models exhibit left-directed bias, though weaker than ARLM baselines. More in Appendix B.

ARLMs are known for their strong In-Context Learning (ICL) capabilities [6, 7, 44]. However, these models also exhibit undesirable locality biases–such as recency bias [38]–that hinder effective conditioning on long contexts [19, 38, 17, 27]. These biases are often linked to the autoregressive (AR) loss function, which enforces a strictly sequential prediction order [4]. This inspires a key question: if we move away from the AR objective, can we retain ICL while reducing the locality biases?

DLMs present a compelling opportunity to examine this question. Their masked diffusion objective is equivalent to any-order ARLMs [34], suggesting that they may leverage context more globally. Yet, it remains unclear whether ICL–previously studied mainly under the AR objective [11, 32]–emerges under this alternative loss. We therefore ask: (1) Do DLMs exhibit ICL, indicating that the AR objective is not strictly necessary for this capability? (2) Does the diffusion objective reduce the locality biases, enabling more effective context processing?

We address these questions by evaluating open-source DLMs (LLaDA [23] and Dream [13]) on carefully-designed ICL experiments. Our findings reveal that DLMs can perform ICL at the same level as similarly-sized ARLMs. While DLMs still exhibit AR-like local biases, such as recency

*Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: What Can('t) Transformers Do?.

and left-directedness (Figure 1), these effects are weaker than in ARLMs. This suggests that ICL is not exclusive to ARLMs, yet the masked diffusion objective alone does not fully eliminate local processing tendencies, offering new insights into how training objectives shape inductive biases.

2 Summary of Experimental Results

Below we summarise the findings of our experiments. We compare the performance of LLaDA-8B with Llama3-8B [2] (an ARLM with a similar architecture) and Dream-7B against Qwen2.5-7B [46, 39] (an ARLM model used to initialize Dream-7B). We also note that while LLaDA was trained from scratch using the masked diffusion loss [31], Dream was initialised with weights of an ARLM, thus constituting an interesting interpolation point between ARLMs and and LLaDA. For details of the experiments, see Appendix B.

Can DLMs perform ICL? We evaluate DLMs on standard ICL benchmarks, including AG News [47], SST-2 [35], and a subset of GLUE classification tasks [42], without using any system prompts in order to isolate ICL ability. Results in Figure 2 show that DLMs match ARLMs on these tasks. However, as these benchmarks involve well-established priors (e.g. sentiment or entailment classification), performance saturates with as few as four in-context examples across all models. To further probe ICL abilities we evaluate performance on a suite of more abstract tasks proposed by Todd et al. [40]. As shown in Figure 3 DLMs also match the performance of ARLMs on these tasks, suggesting a well-rounded ICL ability, despite training with a non-AR loss.

Given that the masked diffusion objective can also induce ICL, we also ask whether it helps overcome some of the locality biases observed in ARLMs limiting effective context utilisation. Specifically, we examine if DLMs inherit two characteristic ARLM-like biases: (i) recency bias (favouring the most recent examples), and (ii) left-directed bias (prioritizing examples positioned to the left of the mask).

Do DLMs suffer from a recency bias? To study recency bias, we adopt the label-flip experiment from [17], in which the labels in a binary classification task are flipped halfway through the in-context examples (e.g., 'Yes' $\rightarrow$ 'No'). Contrary to expectations, Figure 4 shows that DLMs are highly sensitive to ordering: they adapt to the flipped distribution as fast as the ARLMs and exhibit significant performance gaps between default-then-flipped and flipped-then-default sequences.

To further probe recency bias, we evaluate DLMs on a multidimensional classification task. We vary the position of the most relevant examples (closest in L2 distance) to be left-most, right-most, or randomly located within the prompt. Results in Figure 5 show that performance of DLMs (particularly Dream) drops significantly when relevant examples are far from the masked question, confirming a recency bias – though weaker than in ARLMs.

Do DLMs have a left-directed bias? To further probe AR tendencies, we test whether DLMs exhibit a left-directed bias. We design a variant of the label-flip experiment, modifying a sentiment classification task so that the first chunk of in-context examples uses one labelling scheme (e.g., 'Yes'/'No') and the second chunk uses an alternative, non-contradictory scheme (e.g., 'A'/'B'). The masked question is placed between the two segments, and we observe which labelling the model adopts–the one on the left or on the right. Results in Figures 6 and 7 show that, despite dataset-specific variation (likely due to prior label preferences), on average both base DLMs favour the left labelling over the right, indicating a left-directed bias.

3 Discussion

Instruction fine-tuning alleviates AR bias. Across our experiments, instruction-tuned models consistently exhibit weaker recency and left-directed biases compared to their base counterparts, reinforcing findings from previous works [19]. We hypothesise that this stems from exposure to more structured, multi-step tasks during instruction tuning, which likely encourages more global context utilisation. Combined with our finding that DLMs display AR-like biases despite a non-AR objective, this suggests that such biases may arise primarily from the properties of training data rather than the loss function alone, aligning with observations in prior work [4].

Masked diffusion fine-tuning does not erase AR bias. Our results further indicate that retraining an ARLM with a masked diffusion loss (as in Dream) does not fully eliminate its AR bias, whereas training from scratch under this objective (as in LLaDA) yields models that are comparatively less biased (cf. Figure 5). This suggests that the inductive biases established during initial AR training persist through fine-tuning, and that achieving more global context comprehension may require training from scratch rather than repurposing ARLMs.

References

[1] K. Ahn, X. Cheng, H. Daneshmand, and S. Sra. Transformers learn to implement preconditioned gradient descent for in-context learning. *Advances in Neural Information Processing Systems*, 36:45614–45650, 2023.

[2] AI@Meta. Llama 3 model card. 2024. URL https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

[3] E. Akyürek, D. Schuurmans, J. Andreas, T. Ma, and D. Zhou. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*, 2022.

[4] S. An, Z. Ma, Z. Lin, N. Zheng, and J.-G. Lou. Make your LLM fully utilize the context. *Neural Information Processing Systems*, abs/2404.16811:62160–62188, 25 Apr. 2024. URL http://dx.doi.org/10.48550/arXiv.2404.16811.

[5] J. Austin, D. D. Johnson, J. Ho, D. Tarlow, and R. van den Berg. Structured denoising diffusion models in discrete state-spaces. *arXiv [cs.LG]*, 7 July 2021. URL http://arxiv.org/abs/2107.03006.

[6] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020. URL https://papers.nips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.

[7] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, P. Schuh, K. Shi, S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes, Y. Tay, N. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski, X. Garcia, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Dohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta, M. Diaz, O. Firat, M. Catasta, J. Wei, K. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, and N. Fiedel. Palm: scaling language modeling with pathways. *J. Mach. Learn. Res.*, 24(1), Jan. 2023. ISSN 1532-4435.

[8] H. Firooz, M. Sanjabi, W. Jiang, and X. Zhai. Lost-in-distance: Impact of contextual proximity on LLM performance in graph tasks. *arXiv [cs.AI]*, 2 Oct. 2024. URL http://arxiv.org/abs/2410.01985.

[9] K. Frans, D. Hafner, S. Levine, and P. Abbeel. One step diffusion via shortcut models. *arXiv [cs.LG]*, 23 June 2025. URL http://arxiv.org/abs/2410.12557.

[10] T. Gaintseva, C. Ma, Z. Liu, M. Benning, G. Slabaugh, J. Deng, and I. Elezi. CASteer: Steering diffusion models for controllable generation. *arXiv [cs.GR]*, 11 Mar. 2025. URL http://arxiv.org/abs/2503.09630.

[11] Z. Gong, X. Hu, H. Tang, and Y. Liu. Towards auto-regressive next-token prediction: In-context learning emerges from generalization. In *The Thirteenth International Conference on Learning Representations*, 4 Oct. 2024. URL https://openreview.net/forum?id=gK1rl98VRp.

[12] M. Hahn and N. Goyal. A theory of emergent in-context learning as implicit structure induction. *arXiv preprint arXiv:2303.07971*, 2023.

[13] HKU NLP Group. Dream 7B. https://hkunlp.github.io/blog/2025/dream/.

[14] D. Israel, G. Van den Broeck, and A. Grover. Accelerating diffusion LLMs via adaptive parallel decoding. *arXiv [cs.CL]*, 31 May 2025. URL http://arxiv.org/abs/2506.00413.

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

[16] J. Kim, K. Shah, V. Kontonis, S. Kakade, and S. Chen. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. *arXiv [cs.LG]*, 10 Feb. 2025. URL http://arxiv.org/abs/2502.06768.

[17] J. Kossen, Y. Gal, and T. Rainforth. In-context learning learns label relationships but is not conventional learning. *arXiv [cs.CL]*, 23 July 2023. URL http://arxiv.org/abs/2307.12375.

[18] I. Labs, S. Khanna, S. Kharbanda, S. Li, H. Varma, E. Wang, S. Birnbaum, Z. Luo, Y. Miraoui, A. Palrecha, S. Ermon, A. Grover, and V. Kuleshov. Mercury: Ultra-fast language models based on diffusion. *arXiv [cs.CL]*, 17 June 2025. URL http://arxiv.org/abs/2506.17298.

[19] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang. Lost in the middle: How language models use long contexts. *arXiv [cs.CL]*, 6 July 2023. URL http://arxiv.org/abs/2307.03172.

[20] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM computing surveys*, 55(9):1–35, 2023.

[21] X. Liu, Z. Liu, Z. Huang, Q. Guo, Z. He, and X. Qiu. LongLLaDA: Unlocking long context capabilities in diffusion LLMs. *arXiv [cs.CL]*, 22 June 2025. URL http://arxiv.org/abs/2506.14429.

[22] A. Lou, C. Meng, and S. Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. *arXiv [stat.ML]*, 25 Oct. 2023. URL http://arxiv.org/abs/2310.16834.

[23] S. Nie, F. Zhu, Z. You, X. Zhang, J. Ou, J. Hu, J. Zhou, Y. Lin, J.-R. Wen, and C. Li. Large language diffusion models. *arXiv [cs.CL]*, 14 Feb. 2025. URL http://arxiv.org/abs/2502.09992.

[24] B. Pang and L. Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the ACL*, 2005.

[25] C. Pani, Z. Ou, and Y. Li. Test-time alignment of discrete diffusion models with sequential monte carlo. *arXiv [cs.LG]*, 28 May 2025. URL http://arxiv.org/abs/2505.22524.

[26] Y.-H. Park, C.-H. Lai, S. Hayakawa, Y. Takida, and Y. Mitsufuji. *Jump Your Steps*: Optimizing sampling schedule of discrete diffusion models. *arXiv [cs.LG]*, 10 Oct. 2024. URL http://arxiv.org/abs/2410.07761.

[27] G. Qin, Y. Feng, and B. Van Durme. The nlp task effectiveness of long-range transformers. *arXiv preprint arXiv:2202.07856*, 2022.

[28] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. 2019. URL https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

[29] J. Rector-Brooks, M. Hasan, Z. Peng, Z. Quinn, C. Liu, S. Mittal, N. Dziri, M. Bronstein, Y. Bengio, P. Chatterjee, A. Tong, and A. J. Bose. Steering masked discrete diffusion models via discrete denoising posterior prediction. *arXiv [cs.LG]*, 10 Oct. 2024. URL http://arxiv.org/abs/2410.08134.

[30] P. Reizinger, S. Ujváry, A. Mészáros, A. Kerekes, W. Brendel, and F. Huszár. Position: understanding llms requires more than statistical generalization. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

[31] S. Sahoo, M. Arriola, Y. Schiff, A. Gokaslan, E. Marroquin, J. T. Chiu, A. Rush, and V. Kuleshov. Simple and effective masked diffusion language models. *Neural Information Processing Systems*, abs/2406.07524:130136–130184, 11 June 2024. URL http://dx.doi.org/10.48550/arXiv.2406.07524.

[32] M. E. Sander, R. Giryes, T. Suzuki, M. Blondel, and G. Peyré. How do transformers perform in-context autoregressive learning? *arXiv [stat.ML]*, 8 Feb. 2024. URL http://arxiv.org/abs/2402.05787.

[33] G. Shansan, Z. Ruixiang, Z. Huangjie, G. Jiatao, J. Navdeep, K. Lingpeng, and Z. Yizhe. DiffuCoder: Understanding and improving masked diffusion models for code generation. *arXiv [cs.CL]*, 25 June 2025. URL http://arxiv.org/abs/2506.20639.

[34] X. Shuchen, X. Tianyu, H. Tianyang, F. Zijin, S. Jiacheng, K. Kenji, L. Zhenguo, and M. Zhi-Ming. Any-order GPT as masked diffusion model: Decoupling formulation and architecture. *arXiv [cs.LG]*, 24 June 2025. URL http://arxiv.org/abs/2506.19935.

[35] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In D. Yarowsky, T. Baldwin, A. Korhonen, K. Livescu, and S. Bethard, editors, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA, Oct. 2013. Association for Computational Linguistics. URL https://aclanthology.org/D13-1170/.

[36] Y. Song, Z. Zhang, C. Luo, P. Gao, F. Xia, H. Luo, Z. Li, Y. Yang, H. Yu, X. Qu, Y. Fu, J. Su, G. Zhang, W. Huang, M. Wang, L. Yan, X. Jia, J. Liu, W.-Y. Ma, Y.-Q. Zhang, Y. Wu, and H. Zhou. Seed diffusion: A large-scale diffusion language model with high-speed inference. *arXiv [cs.CL]*, 4 Aug. 2025. URL http://arxiv.org/abs/2508.02193.

[37] Y. Song, Z. Zhang, C. Luo, P. Gao, F. Xia, H. Luo, Z. Li, Y. Yang, H. Yu, X. Qu, et al. Seed diffusion: A large-scale diffusion language model with high-speed inference. *arXiv preprint arXiv:2508.02193*, 2025.

[38] S. Sun, K. Krishna, A. Mattarella-Micke, and M. Iyyer. Do long-range language models actually use long-range context? *arXiv [cs.CL]*, 19 Sept. 2021. URL http://arxiv.org/abs/2109.09115.

[39] Q. Team. Qwen2.5: A party of foundation models, September 2024. URL https://qwenlm.github.io/blog/qwen2.5/.

[40] E. Todd, M. L. Li, A. S. Sharma, A. Mueller, B. C. Wallace, and D. Bau. Function vectors in large language models. *arXiv [cs.CL]*, 23 Oct. 2023. URL http://arxiv.org/abs/2310.15213.

[41] J. Von Oswald, E. Niklasson, E. Randazzo, J. Sacramento, A. Mordvintsev, A. Zhmoginov, and M. Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pages 35151–35174. PMLR, 2023.

[42] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In T. Linzen, G. Chrupała, and A. Alishahi, editors, *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium, Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5446. URL https://aclanthology.org/W18-5446/.

[43] C. Wu, H. Zhang, S. Xue, Z. Liu, S. Diao, L. Zhu, P. Luo, S. Han, and E. Xie. Fast-dLLM: Training-free acceleration of diffusion LLM by enabling KV cache and parallel decoding. *arXiv [cs.CL]*, 28 May 2025. URL http://arxiv.org/abs/2505.22618.

[44] Y. Wu, Z. Sun, S. Li, S. Welleck, and Y. Yang. Inference scaling laws: An empirical analysis of compute-optimal inference for problem-solving with language models. *arXiv preprint arXiv:2408.00724*, 2024.

[45] S. M. Xie, A. Raghunathan, P. Liang, and T. Ma. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*, 2021.

[46] A. Yang, B. Yang, B. Hui, B. Zheng, B. Yu, C. Zhou, C. Li, C. Li, D. Liu, F. Huang, et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

[47] X. Zhang, J. Zhao, and Y. LeCun. Character-level convolutional networks for text classification. In *Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 649–657, Cambridge, MA, USA, 2015. MIT Press.

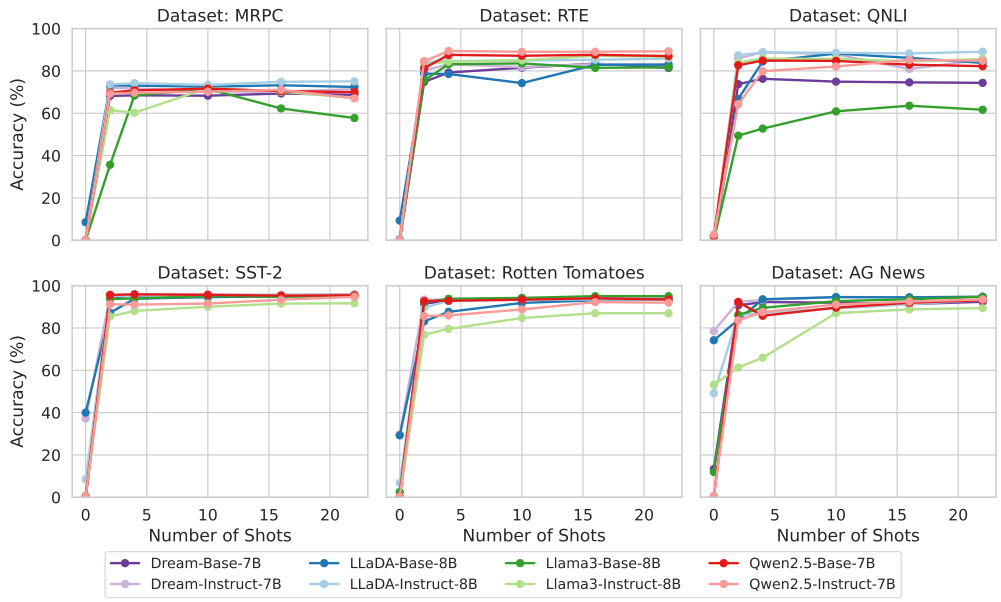

Figure 2: **DLMs match the performance of ARLMs on standard text classification in-context learning tasks.** For experimental setup and details of the datasets used, see Appendix B.

A Related Works

Diffusion Language Models. Diffusion Language Models (DLMs) have recently emerged as a promising alternative to the dominant autoregressive paradigm for text generation [23, 13, 37, 15]. Unlike GPT-style models that generate text sequentially, DLMs employ an iterative denoising process that starts from a noisy representation and progressively reconstructs coherent text, enabling parallel token generation and bidirectional context modelling. While early research explored both continuous and discrete diffusion formulations for text, the discrete masked diffusion objectives [31, 22, 5] have recently dominated the landscape, allowing DLMs to effectively scale to larger model sizes and achieve competitive perplexity on standard benchmarks [13, 23]. DLMs have attracted a lot of attention for their potential to speed up inference [16, 9, 37, 14, 43, 26] as well as improve controllability [29, 10, 25]. However, to the best of our knowledge, a comprehensive evaluation of the influence of the masked diffusion training objective on the models' capabilities and inductive biases is still missing.

In-Context Learning. In-Context Learning (ICL) refers to the ability of large language models (LLMs) to perform new tasks by conditioning on information provided in the input context, without updating their parameters [6]. This capability enables LLMs to solve problems not explicitly encountered during training, substantially enhancing their generalization abilities [7, 28, 44]. A growing body of research has sought to explain this phenomenon, proposing that language models may implicitly implement general-purpose learning algorithms such as gradient descent or Bayesian inference [45, 1, 41, 3, 12]. These studies share the view that understanding the emergence and mechanisms of ICL can provide valuable insights into the internal representations and reasoning processes of LLMs. However, most investigations to date have focused on models trained with the prevalent autoregressive objective. Extending this analysis to models trained with a masked diffusion objective offers an opportunity to test whether ICL is inherently tied to a specific loss function or instead reflects deeper inductive biases shaped by architecture and training dynamics [30]. Throughout this paper, we study ICL abilities of DLMs by focusing on few-shot ICL specifically, which involves augmenting the input with *examples* indicating the input-output relationships [20].

Locality Biases in Language Models. Although powerful, ICL exhibits behaviours that differ from conventional learning algorithms, revealing important inductive biases in language models. One such bias is sensitivity to the ordering of in-context examples. Kossen et al. [17] showed that

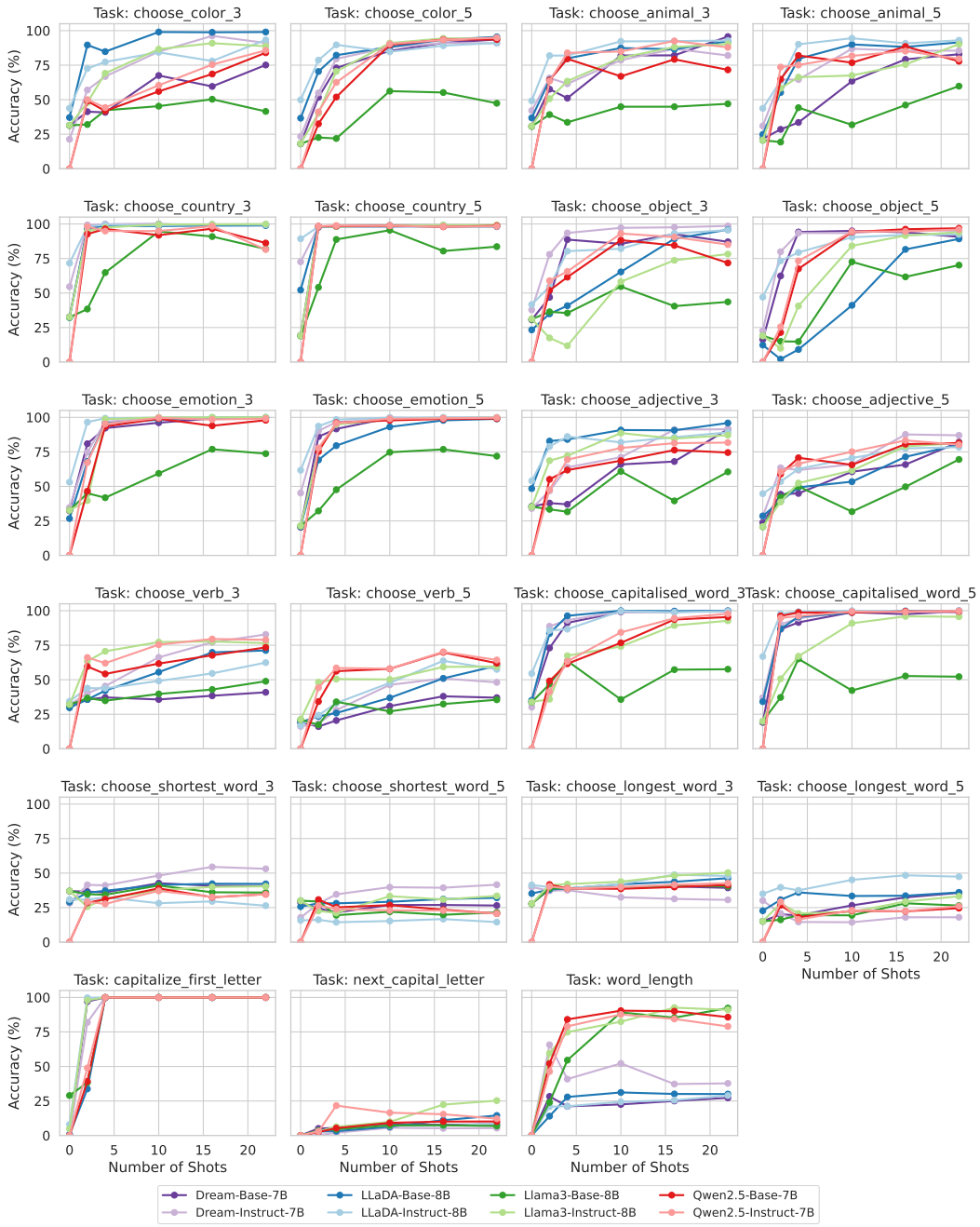

Figure 3: **Even on more abstract tasks, DLMs match in-context learning capabilities of ARLMs.** The only task where DLMs significantly underperform is the `word_length` task, which requires counting the number of letters in the given word. Details of the different tasks can be found in Appendix B.

ARLMs tend to prioritize examples near the end of the prompt—a phenomenon known as recency bias [38]. Variants of the needle-in-the-haystack experiment further confirm that ARLM performance depends strongly on the placement of relevant information, with accuracy peaking when information appears at the beginning (primacy bias) or end (recency bias) of the input context [19, 8, 4, 27]. Whether DLMs exhibit similar locality biases remains an open question. Recent work has addressed related but distinct aspects: Liu et al. [21] evaluated LLaDA on the needle-in-the-haystack task to study generalization to unseen context lengths, but the simplicity of the task they considered

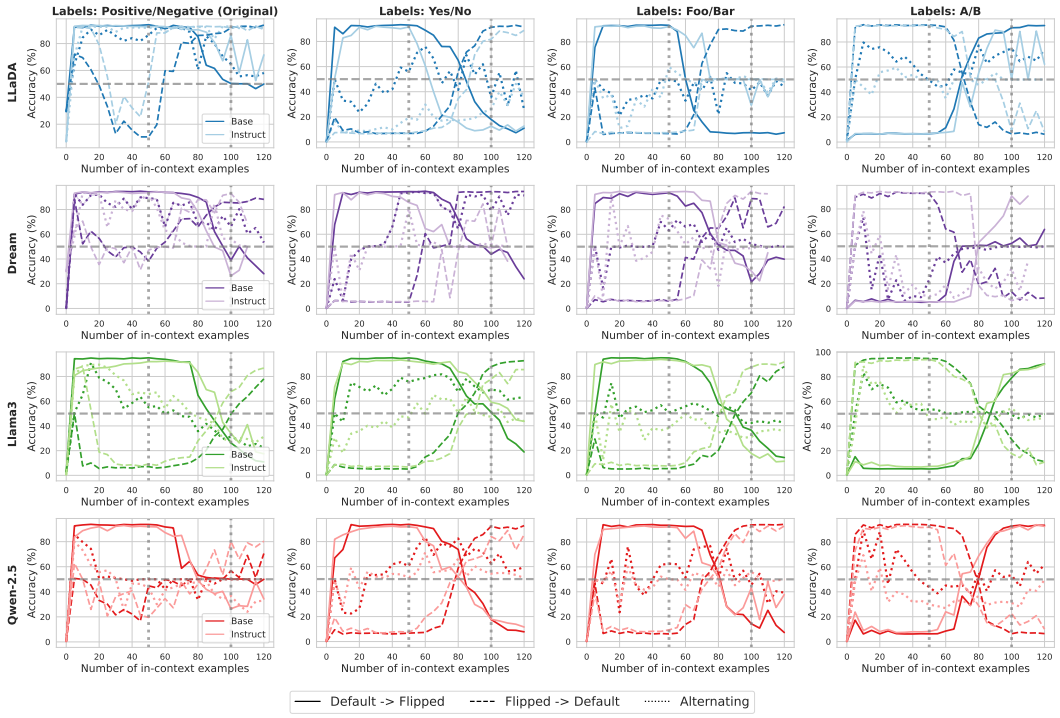

Figure 4: **DLMs exhibit a strong recency bias.** We perform the experiment on the Rotten Tomatoes dataset, choosing different labelling schemes for the two classes, to mitigate the strong prior associated with the original ('Positive'/'Negative') labels. We flip the labels to the opposite labelling distribution after 50 in-context examples (and thus after 100 examples the models have seen equal number of points from the default and flipped distribution). We measure the accuracy with respect to the default labelling distribution. The dotted line marks the performance in the setting where we alternate between the default and flipped labels with each example. The performance, consistent across different labelling schemes, shows that DLMs (particularly LLaDA) are as fast as ARLMs to converge to the flipped distribution. They are also order-sensitive, as evidenced by the large gap in performance between the default-then-flipped and flipped-then-default settings. The results also offer additional evidence that DLMs can learn label distribution in-context.

limits conclusions about recency effects. Similarly, Shansan et al. [33] examined the "AR-ness" of DLMs, defined as a preference for left-to-right decoding. In contrast, our work investigates whether DLMs display AR-like tendencies in *processing* in-context information, rather than in their decoding strategy.

B Experimental Details

B.1 Models

Throughout our experiments we use the following open-source model families, all accessed via the Huggingface API:

- **LLaDA [23]:** An 8B diffusion language model pre-trained from scratch using the masked diffusion loss [31].
- **Dream [13]:** A 7B diffusion language model, whose weights are initialised from those of an autoregressive Qwen-2.5-7B.
- **Qwen-2.5-7B [46, 39]:** A fully AR model.
- **Llama3-8B [2]:** A fully AR model, with the architecture similar to that of LLaDA [23].

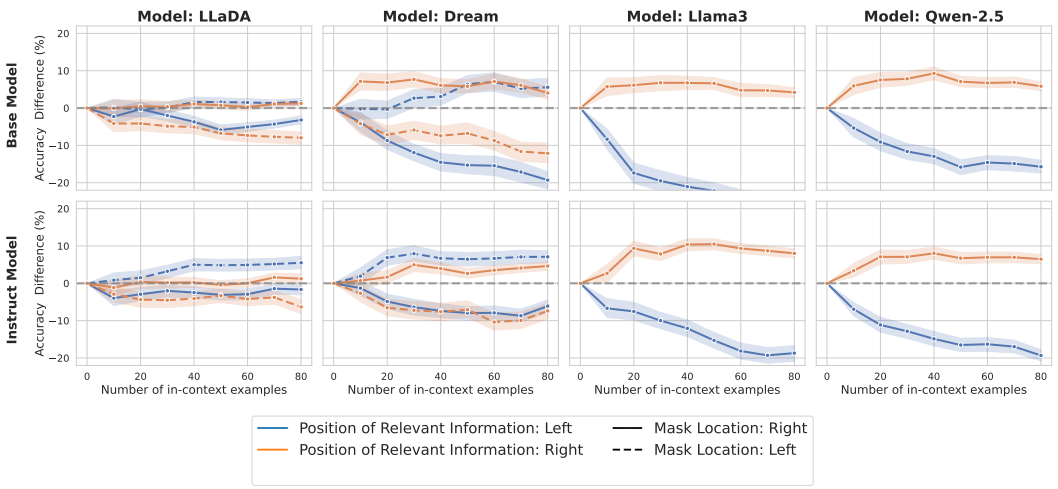

Figure 5: **Across all models, the performance degrades when the relevant information is distant from the masked question.** We report the accuracy difference when placing relevant information on the left versus randomly (blue line), and on the right versus randomly (orange line). For DLMs, we additionally vary the position of the masked question–placing it at either the left or right end of the in-context examples (solid vs. dashed lines). If the models displayed global comprehension abilities, the ordering of the in-context examples and the placement of the mask should not affect performance. To the contrary, across all models performance consistently drops when the relevant information is far from the masked question (blue solid and orange dashed lines), with the effect being most pronounced in ARLMs. Notably, Dream exhibits a stronger recency bias when the masked question is positioned on the right than on the left, suggesting an underlying AR bias. *Shaded regions indicate 95% confidence intervals computed across the 4 dataset types, and 5 seeds.*

To ensure computational efficiency, all models are quantized to 4-bit precision using the Quanto library. For all models, we use greedy decoding strategy (no sampling). We design all of our experiments in a way such that the correct answer consists of only a single token across all the different models and tokenisers. This is to ensure that our experiments can isolate the context-processing abilities of the different models, without being confounded by the effect of tokenisation and/or decoding schemes. This is particularly relevant for DLMs, for which the number of masks added to the prompt can constitute a strong prior about the answer.

B.2 Formatting of the In-Context Learning Examples

Throughout each experiment, we pre-select a certain group of examples from the specified train set to serve as the in-context learning examples for all the test examples (that is, each test example sees exactly the same in-context learning examples, put in the same order). Each example is formatted according to dataset-specific templates that clearly delineate the input text and expected output format. For example, for entailment classification tasks we use:

```
f"Sentence 1: '{sentence1}'\nSentence 2: '{sentence2}'\nLabel:[{label}]
\n\n"
```

While for text classification tasks we use:

```
f"Text: '{sentence}'\nLabel:[{label}]\n\n"
```

We always embed the final answer within the square brackets to avoid issues around tokenisation of spaces. For instruct models, each in-context example is formatted as a pair of messages: a user message containing the question and an assistant message containing the answer. The test question is added as the final user message, with the answer prefix included in the assistant's response.

Autoregressive models. For ARLMs, we add the full test question and the beginning of the answer (e.g., `"Label:["`) to the final formatted prompt and ask the model to continue the generation. We always decode only one new token.

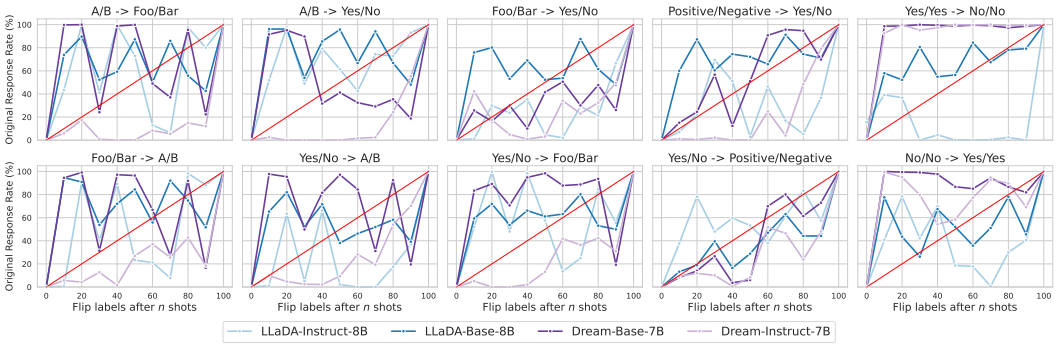

Figure 6: **The base DLMs prefer to answer according to the left labelling scheme.** We design a variant of the label-flip experiment, modifying a sentiment classification task so that the first chunk of in-context examples uses one labelling scheme (e.g., 'Yes'/'No') and the second chunk uses an alternative, non-contradictory scheme (e.g., 'A'/'B'). The masked question is placed between the two segments, and we observe which labelling the model adopts–the one on the left or the right. We keep the total number of examples fixed while varying the switch point (shown on the x-axis). To control for the effect of the prior preference of different labelling schemes, for each pair of labels we consider both orderings (upper and lower row). The y-axis represents how often the model decides to respond using the original (left) labelling. The red diagonal line represents the expected behaviour: the model using the original labelling $x\%$ of the time when $x\%$ of all the in-context examples use this labelling.

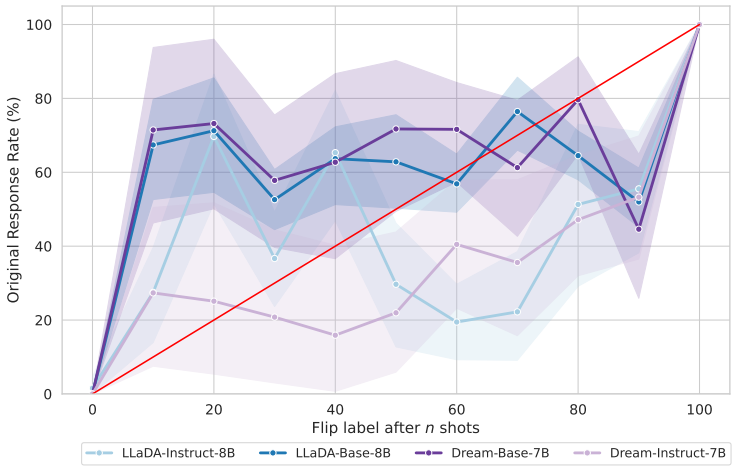

Figure 7: **On average, both LLaDA and Dream prefer to respond with the left-side labels.** By averaging the results across all the labelling schemes considered in Figure 6, we note that both LLaDA-Base and Dream-Base prefer to answer using the original (left) labelling scheme in the majority of cases. *Shaded regions indicate 95% confidence intervals computed across the datasets in Figure 6.*

Diffusion models. For diffusion models, to allow to robustly compare performance between different locations of the masked question within the provided in-context examples, we structure the answer of the masked question as `"Label:[<|mask|>]."` and add this to the prompt, where `<|mask|>` is the textual representation of the mask token, specific to each DLM. We add exactly one copy of the mask token in between the square brackets.

Extracting answers. To evaluate the models' accuracy, we perform string matching on the greedily decoded answer (that is, we perform evaluation on the decoded answer, rather than on the generated tokens).

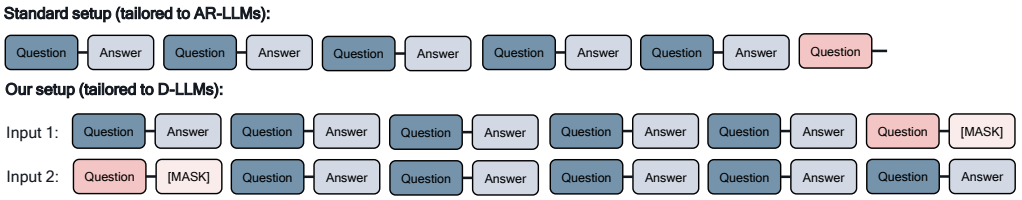

Figure 8: Visualisation of the setting used to evaluate ICL in DLMs. We format the whole in-context learning prompt, placing the mask in place of the correct answer. This allows us to compare the performance of the model when the mask is placed at different positions within the sequence.

Changing the location of the masked question. In Figure 5, to evaluate the sensitivity of the DLMs to the positioning of the mask, we design experiments in which the masked question is placed at different positions within the in-context examples (at the beginning (left) or end (right)). This setup is visualised in Figure 8.

B.3 Details of Individual Experiments

B.3.1 Standard ICL Evaluation Benchmarks (Figure 2)

Datasets. We use the following datasets: AG News [47], SST-2 [35], Rotten Tomatoes [24], as well as MRPC, RTE and QNLI from GLUE [42]. For AG News, we restrict the dataset to three categories only (excluding 'Science and Technology') such that each of the correct labels can be expressed with a single token only. For RTE and WNLI datasets, we use the original validation set for getting the in-context examples and use the train set as the evaluation set, to maximise the number of examples in the evaluation set. For datasets where the examples are ordered by label, we shuffle the datasets upon loading to ensure that there is an even distribution between the different classes within the in-context examples provided to the models. For RTE, QNLI and AG News datasets we also filter the examples in the train and test sets such that the length of the text does not exceed 500 characters.

B.3.2 ICL Evaluation on Abstract Tasks

Datasets. We construct abstract pattern recognition tasks, inspired by the setup proposed by Todd et al. [40]. We compile a list of words from different categories (animal, colour, country, etc.) and construct a number of tasks in which the task is to choose the word from a specified category from amongst several distractor words. In the name of the task, the final suffix (`_3` or `_5`) indicates how many words the model has to choose from. To ensure that each answer only consists of a single token, we phrase the task as a multiple-choice question, for example (for the `choose_color_3` task):

```
Options: (A) dog, (B) volleyball, (C) yellow

Answer:[C]
```

We additionally include three tasks which also require only a single-token answer but do not require multiple-choice answers: `capitalize_first_letter` (output the first letter of the given word, capitalised), `next_capital_letter` (output the letter which is next in the alphabet after the first letter of the given word, capitalised), `word_length` (output the number of letters in the given word, where we limit the words to those which contain less than 10 letters).

For each dataset, we randomly generate 100 train examples and 1000 test examples.

B.4 Label Flip Experiment (Figure 4)

Dataset. We perform the experiment on the Rotten Tomatoes dataset [24]. To mitigate the strong prior associated with the original ('Positive'/'Negative') labels (which is difficult to overwrite using in-context learning [17]), we use different alternative labelling schemes ('Yes'/'No'; 'Foo'/'Bar'; 'A'/'B').

Setup. We adopt the experimental setup from Kossen et al. [17]. The aim of the experiment is to observe to what extent different language models are sensitive to the ordering of the in-context learning examples provided to them in the prompt. To study this, in this experiment we flip the labels on the Rotten Tomatoes sentiment classification task after 50 in-context examples. That is, after 50 in-context examples, we flip the label distribution by changing each label to the opposite label (e.g., 'Positive' -> 'Negative' and vice versa). We then measure how quickly the model detects and adjusts to this change. The underlying assumption is that if the model treated all the examples provided in-context equally (not caring about the order in which they are presented, as a conventional learning algorithm would), after the model has seen equal number of points from both the default and the flipped label distributions (i.e., after seeing 100 in-context examples), it should not matter which of them was presented first (and thus we would expect roughly the same performance from the model if we used default-then-flipped and flipped-then-default ordering). As a baseline, we also consider alternating between the flipped and default label distributions every other example.

Performance evaluation. We measure and plot the accuracy of the model with respect to the default labelling scheme (where the 'default' corresponds to the original 'Positive'/'Negative' labelling if these labels are used, or is chosen arbitrarily in case of the other labels). We then measure the performance of the model with a different number of shots provided in-context. If the number of shots is smaller than the flipping point (50 in-context examples), that means that we are using only the examples with the first labelling.

B.5 Multidimensional Classification Experiment (Figure 5)

Dataset. To evaluate recency bias, we constructed several synthetic binary classification datasets with varying complexity. Each dataset was designed to present different learning challenges, from nonlinear decision boundaries to complex manifold structures. For reproducibility, we generated each dataset type with 5 different random seeds. We utilized four distinct dataset types in our experiments:

1. **Nonlinear dataset:** This dataset features nonlinear decision boundaries created through polynomial feature transformations. We first generated base features as random integers between 1 and 100. We then augmented these with squared terms and interaction terms between features, creating a nonlinear feature space. The final binary labels were determined by applying a logistic function to a weighted sum of these features (with randomly generated coefficients), followed by thresholding at 0.5.
2. **Swiss-roll dataset:** We employed scikit-learn's `make_swiss_roll` function to generate data points along a 3D swiss roll manifold. The continuous position along the roll (colour parameter) was converted to binary labels by thresholding at the median value, creating two interleaved classes that cannot be separated by a linear boundary. The 3D coordinates were then scaled to integers between 1 and 100 to maintain consistency with our other datasets.
3. **Moons dataset:** Using scikit-learn's `make_moons` function, we created two interleaving half-moon shapes in 2D space. This dataset presents a clear nonlinear boundary challenge. The resulting coordinates were scaled to integers between 1 and 100.
4. **Circles dataset:** We generated concentric circles using scikit-learn's `make_circles` function, creating another challenging nonlinear classification problem. As with the other datasets, the coordinates were scaled to integers between 1 and 100.

To ensure class balance, we generate equal numbers of positive and negative examples for both training (100 examples) and test splits (1000 examples) of each dataset. Additional dimensions beyond those generated by the base algorithms were filled with random integers, such that each dataset has is three-dimensional. Each input vector is stored as a space-separated string of integers. We use the class labels 'Above' and 'Below'.

Setup. To study the recency bias (i.e., whether or not the models have a tendency to pay more attention towards examples which are closer to the generation point), we employ the following ordering schemes to the selected in-context examples:

- **Random ordering:** The in-context examples are ordered randomly.

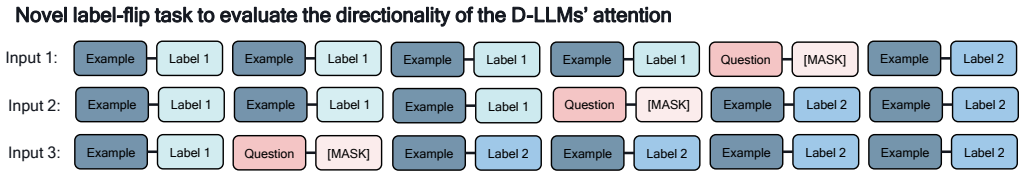

Figure 9: Visualisation of the setting used in Figure 6 & Figure 7. We create two alternative labelling schemes (Label 1 and Label 2) and create prompts by using Label 1 for the first chunk of examples and Label 2 for the second chunk. We then place the masked question in between the two chunks, thus asking the model to choose which labelling scheme it prefers to use. We record the proportion of times when the model responded using Label 1.

- **Ordered by distance to the test point, in decreasing order:** When formatting each prompt, we compute the L2 distance of each in-context example to the test example. We then order the in-context examples in decreasing order, such that examples on the far left of the prompt are furthest away from the test point, and examples on the far right are closest in distance to the test point. This corresponds to the 'Position of relevant information: Right' setting.
- **Ordered by distance to the test point, in decreasing order:** We again compute the L2 distance of each in-context example to the test example, but now order the points in increasing order, such that examples on the far left of the prompt are closest to the test point, and examples on the far right are furthest away in distance to the test point. This corresponds to the 'Position of relevant information: Left' setting.

We note that in all settings, the selected in-context examples are fixed, we just change their order within the prompt. Under this setting, a conventional supervised learning algorithm should be agnostic to the ordering of the provided information. To evaluate whether the recency bias in DLMs is location-agnostic, we run the experiments with the masked example placed both on the left-end of the prompt and on the right-end of the prompt.

B.6 Label Flip Experiment with Two Labelling Schemes (fig. 6, fig. 7 & fig. 1)

Datasets. We base this experiment on the Rotten Tomatoes dataset (as in fig. 4). This time, we fix the number of in-context learning examples to be 100, and we modify the switch point. Before the switch point, we use one type of labelling (e.g., 'A'/'B') and after the switch point we transition into an alternative (yet not contradictory) labelling scheme (e.g., 'Foo'/'Bar'). We put the masked question exactly in between the two labelling schemes, such that to answer the question, the model needs to choose whether to use the labelling scheme to the left of the mask or to the right of the mask. This setting is visualised in Figure 9. For any pair of labelling schemes, we consider both possible orderings of the labelling schemes (i.e., both labelling schemes of each pair get to be on the left), in order to control for the effect of the prior preferences of the model for the different ordering schemes. The final labelling scheme pair ('Yes'/'Yes' and 'No'/'No') is meant to overwrite any underlying label distribution, such that the correct answer is always 'Yes' or always 'No', regardless of the true answer. This is to check the performance in the case when the task is simply label-independent.

Evaluation. For each dataset, we then measure how often the model decides to respond according to the original (left) labelling scheme. Rather than computing task accuracy, we focus on label preference, abstracting away from the model's ability to solve the classification task itself. This allows us to isolate and examine the model's inductive biases in label selection, independent of its task-solving competence.

B.7 Metric in Fig. 1: Directional Deviation Ratio

We base fig. 1 on the results of the experiment presented in fig. 6 and fig. 7. Let $\{(e_i, o_i)\}_{i=1}^{K}$ be the expected and observed left-response rates at each flip position (or bin) i, with optional weights $w_i \geq 0$ (default $w_i = 1$):

- Left-side excess mass: $LM = \sum_i w_i \max\{0,\, o_i - e_i\}$,

- Right-side excess mass: $RM = \sum_i w_i \max\{0,\, e_i - o_i\}$.

Then we define the Directional Deviation Ratio as:

$$\text{DDR} = \begin{cases} \dfrac{LM}{LM + RM}, & \text{if } LM + RM > 0, \\ 0.5, & \text{if } LM + RM = 0. \end{cases}$$

Interpretation:

- DDR $= 1.0 \longrightarrow$ maximally left-directed (e.g., $o_i = 1$ for all i).
- DDR $= 0.0 \longrightarrow$ maximally right-directed (e.g., $o_i = 0$ for all i).
- DDR $= 0.5 \longrightarrow$ unbiased relative to the flip protocol (perfectly tracks $o_i = e_i$).

Left and right deviations are first separated, then normalized. This ensures that a model that strongly favours left in some regions and right in others does not appear "neutral" simply because deviations cancel.