MASKS CAN BE DISTRACTING: ON CONTEXT COMPREHENSION IN DIFFUSION LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Masked Diffusion Language Models (MDLMs) have recently emerged as a promising alternative to Autoregressive Language Models (ARLMs), leveraging a denoising objective that, in principle, should enable more uniform context utilisation. In this work, we examine the context comprehension abilities of MDLMs and uncover two key limitations. First, despite their more global training objective and bidirectional attention mechanism, similarly to ARLMS, **MDLMs exhibit a strong locality bias**: performance is highly sensitive to the position of relevant information within the input, favouring local over distant context. Second, we show that appending a large number of **mask tokens–required for generation–can significantly degrade context comprehension**. Through systematic ablations, we find that these masks act as distractors, reducing the model's ability to process relevant information. To address this, we introduce a **mask-agnostic loss function** that encourages predictions to remain invariant to the number of appended masks. Fine-tuning with this objective substantially mitigates the distracting effect of masks, improving robustness of MDLMs. Overall, our findings reveal critical limitations of the current MDLM training paradigm and provide actionable insights for building diffusion-based language models with stronger context comprehension.

1 INTRODUCTION

Diffusion Language Models (DLMs) have recently emerged as a promising alternative to autoregressive models (ARLMs), offering parallel generation and bidirectional context modelling through iterative denoising (Austin et al., 2021; Lou et al., 2023). Within this family, masked DLMs (MDLMs) (Sahoo et al., 2024) have seen rapid progress, scaling competitively and showing strong perplexity and speed on standard benchmarks (Nie et al., 2025; Song et al., 2025). Yet, despite these advances, it remains unclear how MDLMs use context during inference, and to what extent their distinct training objective alters well-known inductive biases of ARLMs (Liu et al., 2023; Barbero et al., 2024). In this work, we present a systematic study of context comprehension in MDLMs and uncover limitations that have immediate implications for how these models are trained, evaluated and deployed.

Despite their promise, our analysis reveals that MDLMs introduce new challenges that are not yet widely recognised. Although their masked diffusion objective permits global context processing in principle, we find that context use is far from uniform. In practice, MDLMs exhibit strong locality effects, relying heavily on nearby context rather than uniformly integrating information across the input sequence. Moreover, generation-time design choices such as the number and placement of mask tokens can dramatically alter performance. These findings raise fundamental questions about the robustness of MDLMs and the validity of current evaluation practices.

This sensitivity traces back to a core design element of MDLMs: the use of mask tokens both during training (via randomised masking in the denoising objective) and at inference (to delimit the predicted span). While intended as neutral scaffolding, these tokens can disrupt attention and distort context processing. Our analysis uncovers a striking inverse scaling law: as more masks are appended, performance drops predictably and sharply in LLaDA models. This degradation is not just quantitative, it alters how models interpret context, reducing their focus on nearby information and levelling performance across input positions, but at a consistently lower baseline. In short, the very tokens meant to guide generation end up clouding it. Far from passive placeholders, masks can be

distracting, and their influence must be treated as a first-class factor in the design, evaluation, and deployment of MDLMs.

We make the following contributions towards identifying these limitations and proposing practical steps to mitigate them:

- **Locality bias (Section 4)**: We provide the first systematic evidence that MDLMs exhibit a strong *locality bias*, prioritising information near the masked token despite their global denoising objective.
- **Inverse scaling with masks (Section 5)**: We uncover an *inverse scaling law with extra masks*: additional mask tokens can significantly degrade performance of MDLMs trained from scratch, especially in long-context settings and when using few decoding steps.
- **Mask-agnostic fine-tuning (Section 6)**: We propose a *mask-agnostic objective* that enforces prediction invariance to mask count, improving robustness without requiring explicit length control.
- **Evaluation guidelines (Section 7)**: We establish mask configuration as a critical experimental factor in MDLM evaluations and recommend standardised practices for fair benchmarking.

2 Related Works

Diffusion Language Models. Diffusion Language Models (DLMs) have recently emerged as a promising alternative to the dominant autoregressive paradigm for text generation (Nie et al., 2025; HKU NLP Group; Song et al., 2025; Khanna et al., 2025). 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 (Shi et al., 2024; Ou et al., 2025; Sahoo et al., 2024; Lou et al., 2023; Austin et al., 2021) have recently dominated the landscape, allowing DLMs to effectively scale to larger model sizes and achieve competitive perplexity on standard benchmarks (HKU NLP Group; Nie et al., 2025). MDLMs have attracted a lot of attention for their potential to speed up inference (Kim et al., 2025b; Frans et al., 2025; Song et al., 2025; Israel et al., 2025; Wu et al., 2025a; Park et al., 2024; Agrawal et al., 2025) and improve controllability (Rector-Brooks et al., 2024; Gaintseva et al., 2025; Pani et al., 2025). However, to the best of our knowledge, a comprehensive evaluation of the influence of the masked diffusion training objective on the models' context comprehension abilities is still missing.

Context Comprehension in Language Models. Language models do not process information provided in the input uniformly (Sun et al., 2021; Qin et al., 2022; Barbero et al., 2024). Two well-documented position biases are primacy bias–a tendency to favour information appearing early in the input–and recency bias, where information near the end is weighted more heavily. These effects combine to produce the characteristic U-shaped accuracy curve in autoregressive models, often referred to as the lost-in-the-middle phenomenon (Liu et al., 2023). Empirical evidence for these biases comes from variations of the needle-in-the-haystack experiments (Kamradt) and related benchmarks across diverse tasks, including information retrieval (An et al., 2024), multi-document question answering (Liu et al., 2023), graph reasoning (Firooz et al., 2024), and in-context learning (Kossen et al., 2023). Primacy bias has been often attributed to the causal attention mask (Barbero et al., 2024), while recency bias has been linked to the training data distributions and the next-token prediction objective (Sharan et al., 2016; Barbero et al., 2024; An et al., 2024).

Whether MDLMs–trained on similar text corpora but with a fundamentally different objective–exhibit comparable position biases remains an open question. Prior work has explored related but distinct aspects of MDLMs: Liu et al. (2025) evaluated LLaDA on needle-in-the-haystack tasks to study generalization to unseen context lengths, but their setup was too simple to reveal recency effects within the models' context lengths. Similarly, Shansan et al. (2025) examined the "AR-ness" of MDLMs, defined as a preference for left-to-right decoding. In contrast, our work investigates whether MDLMs display AR-like tendencies in processing the context, rather than in their decoding strategy. A concurrent work from Rulli et al. (2025) further studies attention sinks in MDLMs, identifying crucial differences in attention patterns between MDLMs and ARLMs, as well as between MDLMs trained from scratch and those initialised from autoregressive models.

3 Experimental Setup

In this work we focus our analysis on two publicly available MDLMs: LLaDA-8B (Nie et al., 2025) and Dream-7B (Ye et al., 2025). We compare the performance of LLaDA-8B with Llama3-8B (AI@Meta, 2024) (an ARLM with a similar architecture) and Dream-7B against Qwen2.5-7B (Yang et al., 2024; Team, 2024) (an ARLM model used to initialise Dream-7B). We also note that while LLaDA was trained from scratch using the masked diffusion loss (Sahoo et al., 2024), Dream was initialised with weights of an ARLM, thus constituting an interesting interpolation point between ARLMs and and LLaDA. Since we prioritise accuracy over generation diversity, we use greedy decoding for all models we consider.

The positional bias in LMs has typically been studied using information retrieval tasks, such as multi-document question answering tasks (Liu et al., 2023) or needle-in-a-haystack tasks (Kamradt; An et al., 2024). However, the problem with these tasks is that they are relatively easy (An et al., 2024), and as such, in order to illuminate the positional bias of the models, they require significantly larger context lengths than those of the MDLMs we consider in this work (LLaDA and Dream have been trained with the context length of 4096 and 2048 tokens respectively). For instance, Liu et al. (2025) have demonstrated that LLaDA-8B-Base achieves 100% retrieval accuracy on the Needle-In-A-Haystack tests (Kamradt) within the context length that it has been trained on. Thus, as in this work we are *not* interested in studying generalisation to larger context lengths, we depart from traditional information retrieval setups.

Instead, we propose a suite of few-shot learning tasks allowing to evaluate the sensitivity of LMs to information placement while remaining within the context limits of all the studied models. The few-shot learning tasks we consider are inspired by Todd et al. (2023) and take the form of multiple-choice questions, requiring the model to infer an abstract rule from examples. For example, given three words, the model needs to choose the one which is an adjective, which would be formatted as:

```
Options: (A) knit, (B) quirky, (C) persuade\nAnswer:[B].\n\n
```

We purposefully design the tasks so that the correct answer spans just a single token, to enable robust evaluation of context processing abilities through the accuracy metric (avoiding relying on more ambiguous metrics such as generative perplexity (Zheng et al.)), and control for the possibly negative effects of the order used in the decoding process. The resulting tasks require a small number of tokens, thus allowing us to remain within the context length of all the studied models.

To be able to manipulate the placement of relevant information within the context, alongside our selection of 8 relevant *word* tasks (e.g., choose adjective, verb, fruit), we also construct 2 distractor tasks based on *numbers* instead (choose the largest or smallest number). This leads to a suite of 16 tasks which we use for evaluation, each containing 1000 test points. Crucially, the tasks are constructed so that the governing rule can only be learned through exposure to multiple examples. Thus, by randomly mixing the relevant and irrelevant examples, we can test the model's ability to comprehend long contexts. Further, we also present results on other datasets (HotPotQA and a multidimensional classification dataset) accompanied with extra experimental details in Appendix C. *Unless otherwise stated, in our figures the shaded regions mark 95% confidence intervals, computed over the 16 datasets contained in our suite of few-shot learning tasks.*

4 Are Diffusion Language Models Location-Sensitive?

Motivation. Autoregressive language models (ARLMs) dominate the current text generation paradigm, but their next-token prediction objective enforces a strict left-to-right generation order, introducing locality biases–such as recency bias–that hinder effective use of long-range context (Sun et al., 2021; Liu et al., 2023; Kossen et al., 2023; Qin et al., 2022). These biases are widely attributed to the autoregressive loss, which inherently prioritizes recent tokens due to its sequential structure (An et al., 2024; Barbero et al., 2024; Bachmann & Nagarajan, 2024; Sharan et al., 2016). This raises a key question: if we move away from the AR objective, can we reduce these biases? Masked Diffusion Language Models (MDLMs) offer a compelling opportunity to explore this hypothesis. Unlike ARLMs, MDLMs optimize a masked diffusion objective that denoises tokens *across the whole sequence* in parallel, at different noise levels, potentially enabling more global context integration. Moreover, the masked diffusion objective has been shown to be equivalent to any-order autoregressive

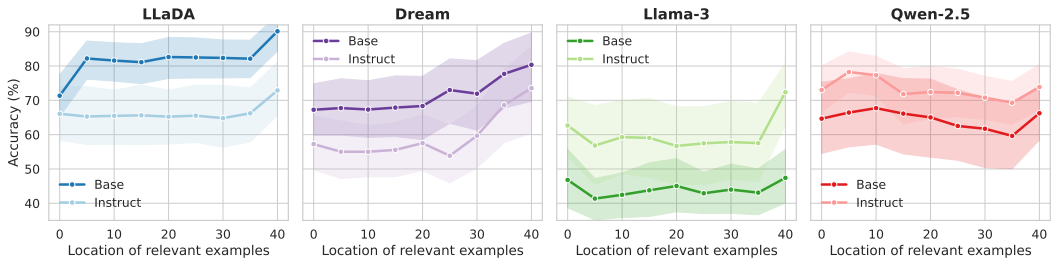

Figure 1: **MDLMs display a recency bias.** The performance of both MDLMs (LLaDA and Dream) and ARLMs is sensitive to the placement of relevant information within the context. For MDLMs, the performance degrades significantly when the relevant information is placed far away from the test question, suggesting a recency bias.

modelling (Shuchen et al., 2025), suggesting that MDLMs might overcome positional constraints inherent to ARLMs. We investigate whether the diffusion objective truly mitigates locality biases and improves long-context comprehension or if such biases persist despite the training paradigm shift.

4.1 Is the Performance Sensitive to the Location of Relevant Information?

Setup. To assess whether model performance depends on the position of relevant information, we systematically vary the location of the *relevant* in-context learning examples within the prompt and measure the resulting accuracy on test questions. Specifically, we use 10 relevant examples (grouped together into one block), and 40 distractor examples. We keep the order of examples within the relevant and distractor groups fixed across all conditions, varying only the position of the relevant block within the overall sequence. We put the test example at the right end of the provided context, in an auto-regressive fashion.

Results. Figure 1 summarises the effect of information placement on model accuracy. Despite being trained with a masked diffusion objective–which does not enforce a strictly sequential prediction order–both MDLMs exhibit strong sensitivity to the position of relevant examples. Performance is highest when relevant information appears immediately before the test question, indicating a significant **recency bias**. Unlike ARLMs, which often display a U-shaped pattern (high accuracy when relevant examples are at the beginning *or* end of the prompt) (Liu et al., 2023; Barbero et al., 2024), MDLMs show a monotonic decline in accuracy as relevant information moves farther away. This suggests that MDLMs lack a strong primacy effect. This observation aligns with expectations, as the primacy effect has been attributed primarily to the causal attention mechanism (Barbero et al., 2024), not present in the MDLMs.

4.2 Does 'Locality' Depend on the Location of the Mask?

Setup. The previous experiment revealed a strong recency bias in MDLMs, but it did not clarify its origin: does the bias arise because models generally prioritise information near the right edge of the context, or because they attend most strongly to the region around the mask token? To disentangle these factors, we repeat the previous experiment while varying the position of the test question (with its answer masked) within the prompt. By moving only the masked question, we can isolate the effect of mask placement on model performance.

Results. Figure 2 shows that across all settings, model performance is highest when relevant information is placed near the masked question. This indicates that the previously observed recency bias is, in fact, a broader locality bias: MDLMs prioritise information close to the prediction target, regardless of its absolute position in the prompt. Interestingly, performance is consistently lowest when the masked question appears at the beginning of the input.

Where does the locality bias come from? Although MDLMs are trained on a more general decoding objective than next-token prediction, the masked diffusion loss is scaled by $1/p$, where p is the probability of masking a token (Nie et al., 2025; Sahoo et al., 2024). Consequently, training places greater weight on cases where only a few tokens are masked–scenarios where nearby context is

usually sufficient for prediction, similar to next-token prediction setting (Sharan et al., 2016). We hypothesise that this encourages MDLMs to rely on nearby context when processing the inputs.

4.3 Further Quantifying the Bias with Gradient Attribution Analysis

Setup. To deepen our understanding of locality bias in MDLMs and ARLMs, we perform gradient attribution analysis (Lopardo et al., 2024), which quantifies how sensitive the model's prediction is to changes in each input token. Specifically, we compute the L2 norm of the gradients of the logit corresponding to the predicted answer token with respect to the input token embeddings. This provides a more mechanistic measure of each token's influence on the output. We use a dataset containing 10 relevant examples and 40 distractors, randomly mixed together to ensure the model must process the entire context to arrive at the answer. While the examples remain fixed across runs, their relative ordering is randomised across 30 seeds. If the models were location-invariant, gradient magnitudes would be roughly uniform across the positions. As the in-context examples do not change for different test questions, for computational efficiency we evaluate the gradients for a sample of 20 test questions for each task only.

Results. Figure 3 shows *normalised* gradient scores across the different in-context examples. Consistent with earlier performance trends (fig. 1), all models exhibit a non-uniform pattern, forming the characteristic U-shape associated with primacy and recency effects. However, MDLMs display more uniform gradients than ARLMs, suggesting more global comprehension abilities. Notably, MDLMs also show less pronounced primacy bias compared to their autoregressive counterparts. Further results of the gradient attribution analysis across the models can be found in Section A.2

> **Takeaways.** Although MDLMs are trained with a masked diffusion objective that denoises tokens across the entire sequence in parallel, they still exhibit a strong **locality bias**: performance depends heavily on the proximity of relevant information to the masked question.

5 The Distracting Effect of Extra Masks

Motivation. The previous section established that MDLMs do not process context uniformly; instead, they rely on heuristics that prioritise information closest to the mask. However, our analysis so far has been restricted to single-token answers, for which we allocated *a single mask token* during decoding. To gain a more comprehensive understanding of MDLMs' context comprehension, we now investigate how their performance changes when additional masks are introduced. As mask tokens are a unique feature of MDLMs, their impact on context processing has not been systematically studied before. Our hypothesis is that increasing the number of masks may reduce the model's local focus, encouraging it to integrate information more globally across the context.

5.1 What is the Effect of Extra Masks on MDLMs' Performance?

Setup. To measure how additional mask tokens affect MDLMs' context comprehension, we append varying numbers of mask tokens to the input prompt. We use 10 relevant and 40 distractor examples, mixing these two groups randomly together, to force the model to process the entire input context. In

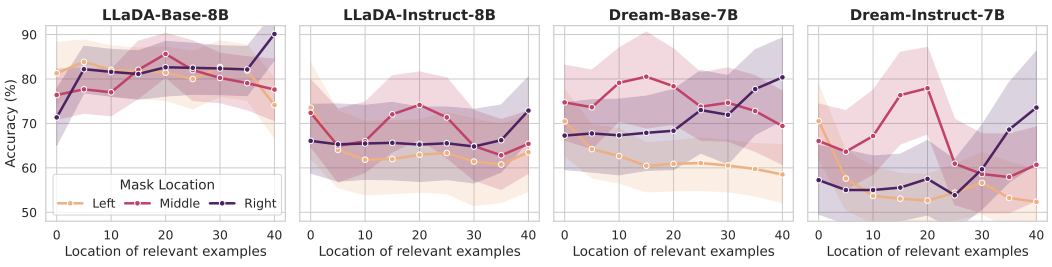

Figure 2: **MDLMs prioritise information placed closest to the mask.** All studied MDLMs perform best when relevant information is near the masked token, regardless of question position.

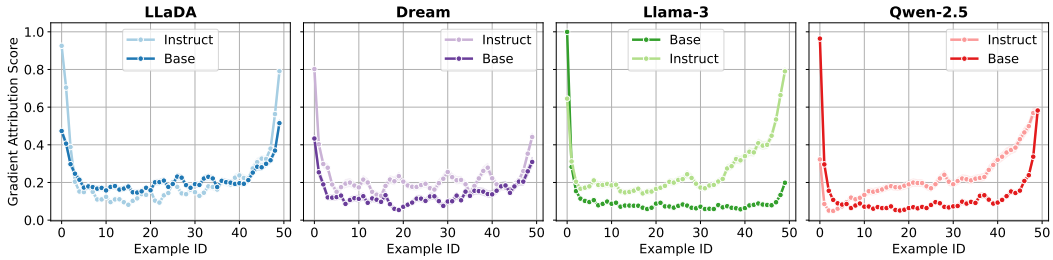

Figure 3: **Gradient attribution analysis further illuminates the locality bias of the models.** Although all models display the characteristic U-shaped behaviour, MDLMs demonstrate more uniform gradients across different positions, indicating reduced locality bias compared to their ARLM counterparts.

our format, the first mask token always corresponds to the answer for the test question. We decode the entire sequence in a single step but evaluate only the prediction for this first mask, ignoring all others. This setup isolates the effect of extra masks on the model's ability to correctly predict the target answer token, without introducing confounding factors from multi-step decoding (we study the effect of decoding the extra masks sequentially on the model's performance in Section 5.4).

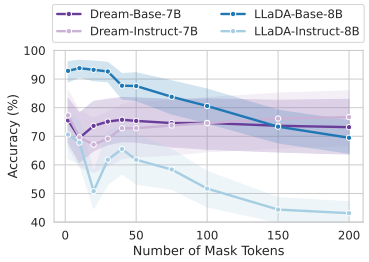

Figure 4: **Performance of LLaDA decreases significantly with added mask tokens,** while Dream is more robust to the extra masks.

Results. Contrary to our initial hypothesis—that additional masks might improve global reasoning—we observe a consistent performance degradation as the number of masks increases (fig. 4). This trend holds for both LLaDA-Base and LLaDA-Instruct. For the Base model, one plausible explanation is a distribution shift: during training, random noising rarely produces long unmasked prefixes followed by large contiguous mask spans. However, this scenario is not unusual for the Instruct model, which nonetheless exhibits a similar decline. Similar detrimental effect of too large numbers of masks was also seen in concurrent works (Li et al., 2025), although to a lesser extent.

Dream models appear more robust to large numbers of masks, but still exhibit a noticeable drop (6 and 8 percentage points for the base and instruct model, respectively) when approximately 20 masks are added, indicating that they are not fully invariant to the extra masks. We hypothesise that this difference may stem from Dream models being initialised from the weights of the autoregressive Qwen-2.5, making mask tokens less integral to their architecture and training dynamics. In the following sections, we study this phenomenon of performance degradation due to extra masks in more detail, analysing several possible hypotheses, aiming to find an explanation for this behaviour.

5.2 Do MDLMs Get Equally Distracted on All Tasks?

Setup. We begin our analysis by investigating whether performance drop caused by extra masks is linked to impaired context comprehension. To that end, we examine how the effect of extra masks changes as the context length required to solve the task increases. Specifically, we vary the number of distractor examples in the prompt while keeping the number of relevant examples fixed, mixing the two groups together randomly. If extra masks indeed disrupt context processing, we expect their negative impact to grow as more distractors are added. We believe that this is because the model must filter relevant from irrelevant information over a longer context, and extra masks may disrupt its attention allocation.

Results. Figure 5 shows that for LLaDA, performance degradation due to extra masks generally increases with the number of distractors, and thus with the effective context length. This suggests that additional masks impair the model's ability to process long contexts, acting as strong 'distractors'.

We provide further evidence for this claim in Appendix A.3. There, we compare the degree of performance degradation caused by extra masks with the gains achieved when increasing the number of in-context examples across different tasks. We find a correlation: tasks that benefit most from

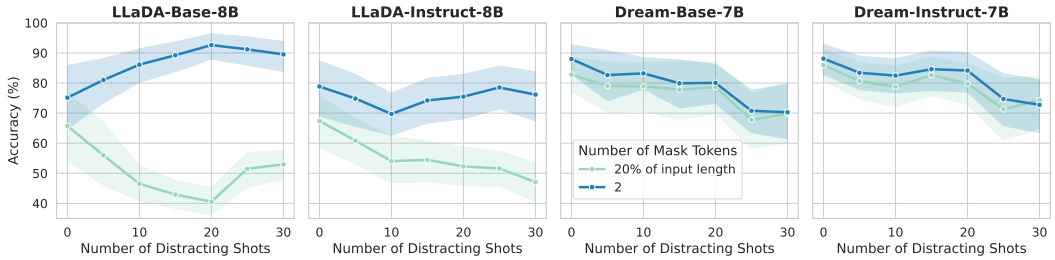

Figure 5: **For LLaDA, performance degradation becomes more significant as the context length increases.** We do not observe a similar effect for Dream, which is robust to the effect of extra masks.

additional context are also the most vulnerable to mask-induced degradation in LLaDA models. This reinforces the conclusion that extra masks specifically inhibit long-context comprehension.

5.3 Is the Degradation Caused Simply by Repeated Tokens?

Setup. In the previous section, we hypothesised that extra masks degrade performance because they act as distractors, drawing attention away from relevant context. To further validate this hypothesis and rule out alternative explanations, we test whether the performance degradation observed earlier is caused by the presence of the mask tokens specifically–rather than by simply appending many identical tokens. To evaluate this, we repeat the experiment from Section 5.1 but replace the extra masks with a relatively neutral token sequence: the string `" . "` repeated multiple times. This ablation allows us to isolate the effect of mask tokens and verify that the observed behaviour is not merely due to an out-of-distribution repetition.

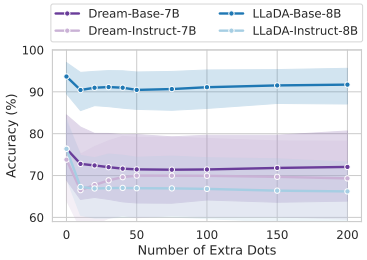

Figure 6: **Extra dots do not degrade performance as strongly as extra masks.**

Results. Figure 6 shows that appending extra dots to the input has only a minor impact on the performance of LLaDA, especially compared to the substantial degradation caused by additional mask tokens (performance decreases by up to 3 and 10 percentage points for the base and instruct models respectively, compared to 23 and 27 percentage points in fig. 4). This confirms that in LLaDA the performance drop is driven by the presence of *mask tokens* specifically, rather than by the mere repetition of identical tokens. For Dream, the effect of the masks and the dots are largely similar.

5.4 Can the Negative Effects be Rectified by Unmasking?

Setup. We examine whether the degradation caused by extra masks can be alleviated at inference time via iterative unmasking, that is, progressively resolving masked positions, rather than using just a single unmasking step as in the previous experiments. We run 40 decoding steps and compare two selection strategies for unmasking: choosing which tokens to unmask at random or according to the highest confidence.

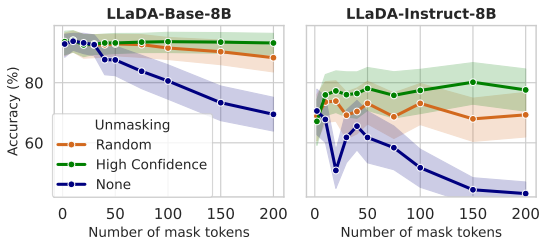

Figure 7: **Unmasking recovers accuracy lost to mask-induced distraction**. Both unmasking strategies improve performance compared to no (i.e. 1-step) unmasking (None, blue).

Results. Figure 7 shows that iterative unmasking (with 40 steps) markedly improves accuracy, recovering the performance lost due to the extra masks when using only a single decoding step. This is especially true for the high-confidence unmasking strategy. This corroborates our findings in Section 5.3: extra masks seem to act as strong distractors and removing them during the decoding process, by replacing them even potentially imperfect generations, can restore focus on relevant context. While effective, this approach adds latency as it requires multiple decoding passes to reduce the effect of extra masks, which might not be desirable for specific hardware-constrained applications.

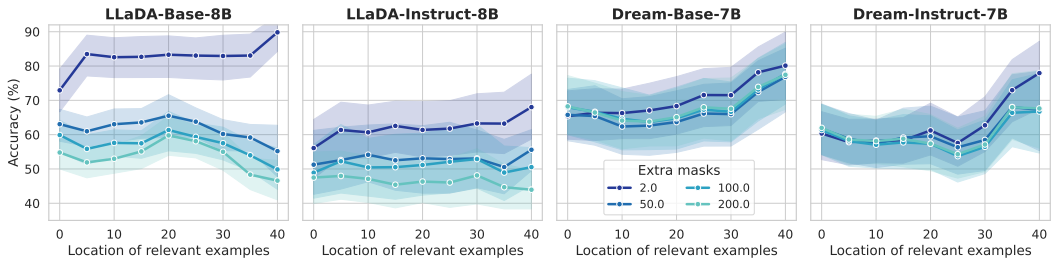

Figure 8: **Extra masks diminish the locality bias.** We measure performance sensitivity to the location of relevant information as extra masks are added. With more masks, accuracy becomes less dependent on the location of the relevant examples, mainly because it declines across all positions.

5.5 Do Extra Masks Affect the Locality of the MDLMs?

Setup. Finally, we revisit the question that motivated us to explore the effect of masks: can additional masks alter the locality bias observed in MDLMs (Section 4)? To test this, we repeat the experiment from Figure 1 but append varying numbers of extra masks to the input. This allow us to assess how extra masks influence the model's ability to use information at different positions within the prompt.

Results. Figure 8 shows a surprising pattern: while extra masks degrade performance across all positions, the drop is more severe when relevant information is *closest* to the test question, creating an 'inverse recency effect' for LLaDA models. This suggests that masks appended near the question strongly disrupt processing of local information. As hypothesised, the performance becomes more uniform across positions, but this uniformity mainly reflects consistently poor results.

Takeaways. The context comprehension abilities of MDLMs can be severely impaired by the presence of **extra mask tokens** in the input. This degradation worsens as context length increases, suggesting that masks act as strong **distractors** which can interfere with long-context processing. Controlled experiments confirm that this effect is specific to mask tokens (rather than a byproduct of repeated tokens only), and can be largely rectified by unmasking. We further find that the extra masks most significantly interfere with the processing of the local information.

6 Reducing the Distracting Effect Through Mask-Agnostic SFT

Motivation. In many practical settings, estimating the expected token length of a valid answer a priori may be challenging. Therefore, we see robustness to variations in the number of mask tokens as a desirable property for MLDMs. To support this, we propose a supervised fine-tuning scheme that promotes prediction invariance with respect to the number of masks appended at the end of the context, encouraging the model to focus on the core task.

This approach assumes that mask count should not serve as an informative prior for generation. While in some applications–such as summarisation–the number of masks can intentionally signal the desired output length, our design targets scenarios where the correct answer length is unknown or where one wants to resort to a default generation length. This avoids burdening users with specifying the output length, and promotes robustness in settings where length constraints are ambiguous or undesirable.

Connection to variable-length generation. The sensitivity to extra masks reflects a broader length prior: MDLMs assume a fixed canvas, so changing the initial mask budget perturbs early denoising steps. A complementary line of work removes the fixed-canvas constraint by enabling *insertions and deletions* during generation (Kim et al., 2025a; Wu et al., 2025b; Havasi et al., 2025). Unlike these approaches, which modify the forward process or introduce special tokens, our method enforces invariance to mask count via fine-tuning, with no architectural changes. Notably, the initial number of masks remains impactful even in these variable-length frameworks, as it influences early decoding steps and the conditioning signal, underscoring the importance of robustness to mask configurations.

6.1 Formulation of the Mask-Agnostic Loss Function

To encourage invariance to the number of extra masks, we propose a **mask-agnostic (MA) loss**. Consider prompt–answer pairs, where $\boldsymbol{q} = (q^1, \ldots, q^{n_q})$ is the tokenised prompt and $\boldsymbol{a} = (a^1, \ldots, a^{n_a})$ is the tokenised answer. We construct a noised version of the answer, $\tilde{\boldsymbol{a}} = (\mathbf{1} - \boldsymbol{u}) \circ \boldsymbol{a} + \boldsymbol{u} \circ \boldsymbol{m}$, where $\mathbf{1} \in \mathbb{R}^{n_a}$ is the vector of 1s, $\boldsymbol{m} \in \mathbb{R}^{n_a}$ is the vector of mask tokens, and $\boldsymbol{u} = (u^1, \ldots, u^{n_a})$ is a vector of samples from a Bernoulli distribution $(u^1, \ldots, u^{n_a} \overset{\text{iid}}{\sim} Ber(p))$ for masking probability p. Here, $\circ$ denotes element-wise vector multiplication and $\oplus$ denotes vector concatenation.

Let $\boldsymbol{q} \oplus \tilde{\boldsymbol{a}}$ denote the concatenation of the prompt and the noised answer tokens. To compute our MA loss, we construct two alternative versions of this input, with different numbers of mask tokens appended. That is, we select $l_1, l_2 \in \mathbb{Z}$ randomly without replacement from the range $[0, N - (n_a + n_q)]$, where N is some pre-defined maximum context length. We then construct two inputs: $\boldsymbol{x}_1 = \boldsymbol{q} \oplus \tilde{\boldsymbol{a}} \oplus (m) * l_1 = (x_1^1, \ldots, x_1^{n_q+n_a+l_1})$ and $\boldsymbol{x}_2 = \boldsymbol{q} \oplus \tilde{\boldsymbol{a}} \oplus (m) * l_2 = (x_2^1, \ldots, x_2^{n_q+n_a+l_2})$. The corresponding labels (not noised) are: $\boldsymbol{x} = \boldsymbol{q} \oplus \boldsymbol{a} = (x^1, \ldots, x^{n_q+n_a})$. Further, let $\mathcal{A}$ denote the set of indices of the elements of $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ which correspond to the answer-part of the input. With this notation in hand we can define our loss as follows:

$$\mathcal{L}_{CE} = -\frac{1}{2pn_m} \sum_{i=1,2} \sum_{j \in \mathcal{A}} \mathbb{1}\{x_i^j = m\} \log p_\theta(x^j | \boldsymbol{x_i}), \tag{1}$$

$$\mathcal{L}_{TV} = \frac{p}{n_m} \sum_{j \in \mathcal{A}} \mathbb{1}\{x_1^j = m\} TV\left(p_\theta(x^j | \boldsymbol{x_1}), p_\theta(x^j | \boldsymbol{x_2})\right), \tag{2}$$

where p_θ is the MDLM distribution and $n_m = \sum_{j \in \mathcal{A}} \mathbb{1}\{x_i^j = m\}$ is the number of masked tokens. Our final MA-loss is then constructed as $\mathcal{L}_{MA} = \alpha \mathcal{L}_{CE} + \beta \mathcal{L}_{TV}$ for scaling parameters α and β.

The first term (**CE loss**) is a cross-entropy loss on the generated answer, ensuring that the model's predictions match the ground-truth answers regardless of how many additional masks are appended. We scale this term by $1/p$, following the standard masked diffusion objective (Sahoo et al., 2024; Nie et al., 2025). The second term (**TV loss**) is a total variational distance that encourages the probability distributions of the answer tokens to remain consistent across different masking configurations. We scale this term by p to ensure that the distributions are aligned even when there are scarcely any unmasked tokens in the answer. As we explain in Section 4.2, in this case the generations are less constrained by the neighbouring tokens, and thus similarity under different masking conditions is crucial to ensure robustness. We further divide both terms by n_m to ensure that loss is calculated on a per-token basis (to account for the possible large variations in the answer lengths, and hence in the number of masked tokens per input). We provide a pseudo-code for the loss in Appendix B.

6.2 Experimental Validation

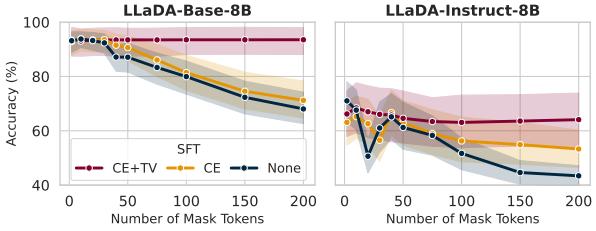

Figure 9: **MA loss rectifies the effect of extra masks.**

Training Details To empirically verify the effectiveness of our MA loss function, we fine-tune LLaDA-Base and LLaDA-Instruct models using LoRA adapters. We also train an ablated model with the CE loss only, setting $\beta = 0.0$. We train on a subset of the OpenOrca dataset (Lian et al., 2023), for 930 gradient descent steps. OpenOrca is an instruction-tuning dataset and not specifically designed for our ICL evaluation setup. Thus, its diversity ensures that fine-tuning induces more global changes in the model, rather than overfitting to the ICL-specific task structure. Training details are in Appendix B.

Does the MA Loss Rectify Performance Degradation Caused by Extra Masks? In Figure 9 we show that fine-tuning both the base and the instruct model with our MA loss allows to improve the performance of the models, making them more robust to variations in the number of masks appended to the input. The CE loss on its own does not have a similar effect, emphasising the importance of regularising generation with the TV loss directly. In Figure 26 we also show the effect of MA loss on the logits of the model, showing that our SFT procedure reduces the entropy of the model and makes it significantly smoother as a function of masks, thus increasing robustness. Unlike iterative unmasking

(Section 5.4), which recovers accuracy through multiple decoding passes, our approach achieves similar robustness in a **single decoding step**. This makes it particularly attractive for low-latency applications and for distillation pipelines, where minimising generation steps is critical for efficiency and model compression. We show that our MA loss improves robustness also in few decoding steps (2-6) in Appendix A.4.

Does the MA Loss Affect the Locality of the Model? We further show in Figure 10 that the MA loss has the added benefit of reducing the locality bias of the LLaDA-Base model. We hypothesise that by teaching the model to effectively ignore the extra masks, we encourage it to pay more attention to the provided context, thus improving context comprehension. This further emphasises the inherent link between the extra masks and context comprehension abilities. We provide the results for the fine-tuned LLaDA-Instruct mod in the Appendix.

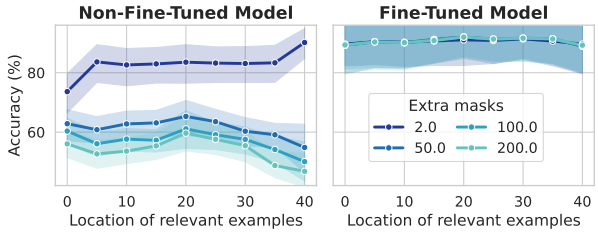

Figure 10: **MA loss (CE+TV) reduces the locality bias of LLaDA-Base.**

Takeaways. We introduced the **mask-agnostic loss**, a fine-tuning objective that enforces invariance to the number of appended masks during generation. This approach consistently improves robustness with a single decoding step, particularly for the Base model, and further mitigates locality bias, highlighting its effectiveness in enhancing long-context comprehension.

7 Discussion, Conclusions and Future Work

MDLM Evaluation Guidelines. Our findings motivate two key recommendations for evaluating MDLMs. First, benchmark reports should explicitly state the number of mask tokens used during evaluation. Our work suggests that this detail is critical for reproducibility, yet we observe that it is often omitted in prior work (Ye et al., 2025; Song et al., 2025). Second, we advocate incorporating mask-sensitivity analysis as a standard component of MDLM evaluation pipelines, conducted specifically on tasks requiring long-context comprehension. Popularising such analysis could reveal how novel decoding strategies and post-training methods influence robustness to variations in mask count, fostering a deeper understanding of model behaviour under realistic usage conditions.

Limitations. In this work, we have focused on the analysis of open-source masked diffusion language models–LLaDA and Dream. However, we note that important details of the pre-training protocol for these models (such as the exact datasets used) have not been publicly released. Their differing training data and pipelines likely contribute to some of the behaviours we uncovered, and a clearer picture could emerge with access to these details. Understanding how specific data distributions, masking schedules, and architectural choices interact with the diffusion objective would help disentangle model-specific quirks from general properties of MDLMs. This represents a limitation of our study, as a more complete understanding of context processing in diffusion models would benefit from controlled comparisons across models with fully transparent training setups.

Moreover, the studied MDLMs have limited context windows (2048 tokens for Dream and 4096 for LLaDA), which complicates direct comparison with autoregressive models capable of handling substantially longer sequences. In particular, concurrent work by Veseli et al. (2025) shows that positional biases in autoregressive models shift as inputs approach the context limit. Whether a similar effect occurs in MDLMs remains unclear; we plan to investigate how context window length influences locality biases in future work.

Future Work. Our findings open several avenues for future work. While we document strong locality biases in MDLMs, the mechanisms behind these biases remain unclear. A deeper analysis of the masked diffusion objective, particularly its weighting schemes and noise schedules, could help explain why models favour nearby context and suggest ways to adjust training dynamics for fewer positional biases. Another direction is to examine uniform diffusion models (Lou et al., 2023), which avoid explicit masks and instead apply noise more evenly across the input. If such models prove more robust to context placement and less sensitive to decoding configurations, this could clarify whether the issues we identify are intrinsic to the diffusion paradigm or specific to masked variants.

REFERENCES

Sudhanshu Agrawal, Risheek Garrepalli, Raghavv Goel, Mingu Lee, Christopher Lott, and Fatih Porikli. Spiffy: Multiplying diffusion llm acceleration via lossless speculative decoding, 2025. URL https://arxiv.org/abs/2509.18085.

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

Shengnan An, Zexiong Ma, Zeqi Lin, Nanning Zheng, and Jian-Guang Lou. Make your LLM fully utilize the context. *Neural Information Processing Systems*, abs/2404.16811:62160–62188, 25 April 2024. URL http://dx.doi.org/10.48550/arXiv.2404.16811.

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

Gregor Bachmann and Vaishnavh Nagarajan. The pitfalls of next-token prediction. *arXiv [cs.CL]*, 11 March 2024. URL http://arxiv.org/abs/2403.06963.

Federico Barbero, Andrea Banino, Steven Kapturowski, Dharshan Kumaran, João G M Araújo, Alex Vitvitskyi, Razvan Pascanu, and Petar Veličković. Transformers need glasses! information over-squashing in language tasks. *arXiv [cs.CL]*, 6 June 2024. URL http://arxiv.org/abs/2406.04267.

Hamed Firooz, Maziar Sanjabi, Wenlong Jiang, and Xiaoling Zhai. Lost-in-distance: Impact of contextual proximity on LLM performance in graph tasks. *arXiv [cs.AI]*, 2 October 2024. URL http://arxiv.org/abs/2410.01985.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. *arXiv [cs.LG]*, 23 June 2025. URL http://arxiv.org/abs/2410.12557.

Tatiana Gaintseva, Chengcheng Ma, Ziquan Liu, Martin Benning, Gregory Slabaugh, Jiankang Deng, and Ismail Elezi. CASteer: Steering diffusion models for controllable generation. *arXiv [cs.GR]*, 11 March 2025. URL http://arxiv.org/abs/2503.09630.

Marton Havasi, Brian Karrer, Itai Gat, and Ricky T Q Chen. Edit flows: Flow matching with edit operations. *arXiv [cs.LG]*, 10 June 2025. URL http://arxiv.org/abs/2506.09018.

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

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

Greg Kamradt. LLMTest_NeedleInAHaystack: Doing simple retrieval from LLM models at various context lengths to measure accuracy. URL https://github.com/gkamradt/LLMTest_NeedleInAHaystack.

Samar Khanna, Siddhant Kharbanda, Shufan Li, Harshit Varma, Eric Wang, Sawyer Birnbaum, Ziyang Luo, Yanis Miraoui, Akash Palrecha, Stefano Ermon, et al. Mercury: Ultra-fast language models based on diffusion. *arXiv preprint arXiv:2506.17298*, 2025.

Jaeyeon Kim, Lee Cheuk-Kit, Carles Domingo-Enrich, Yilun Du, Sham Kakade, Timothy Ngotiaoco, Sitan Chen, and Michael Albergo. Any-order flexible length masked diffusion. *arXiv preprint arXiv:2509.01025*, 2025a.

Jaeyeon Kim, Kulin Shah, Vasilis Kontonis, Sham Kakade, and Sitan Chen. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. *arXiv [cs.LG]*, 10 February 2025b. URL http://arxiv.org/abs/2502.06768.

Jannik Kossen, Yarin Gal, and Tom 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.

Jinsong Li, Xiaoyi Dong, Yuhang Zang, Yuhang Cao, Jiaqi Wang, and Dahua Lin. Beyond fixed: Training-free variable-length denoising for diffusion large language models. *arXiv [cs.CL]*, 18 August 2025. URL http://arxiv.org/abs/2508.00819.

Wing Lian, Bleys Goodson, Eugene Pentland, Austin Cook, Chanvichet Vong, and "Teknium". Openorca: An open dataset of gpt augmented flan reasoning traces. https://https://huggingface.co/datasets/Open-Orca/OpenOrca, 2023.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *arXiv [cs.CL]*, 6 July 2023. URL http://arxiv.org/abs/2307.03172.

Xiaoran Liu, Zhigeng Liu, Zengfeng Huang, Qipeng Guo, Ziwei He, and Xipeng Qiu. LongLLaDA: Unlocking long context capabilities in diffusion LLMs. *arXiv [cs.CL]*, 22 June 2025. URL http://arxiv.org/abs/2506.14429.

Gianluigi Lopardo, Frederic Precioso, and Damien Garreau. Attention meets post-hoc interpretability: A mathematical perspective, 2024. URL https://arxiv.org/abs/2402.03485.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. *arXiv [stat.ML]*, 25 October 2023. URL http://arxiv.org/abs/2310.16834.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. *arXiv [cs.CL]*, 14 February 2025. URL http://arxiv.org/abs/2502.09992.

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=sMyXP8Tanm.

Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the ACL*, 2005.

Chinmay Pani, Zijing Ou, and Yingzhen 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.

Yong-Hyun Park, Chieh-Hsin Lai, Satoshi Hayakawa, Yuhta Takida, and Yuki Mitsufuji. *Jump Your Steps*: Optimizing sampling schedule of discrete diffusion models. *arXiv [cs.LG]*, 10 October 2024. URL http://arxiv.org/abs/2410.07761.

Guanghui Qin, Yukun Feng, and Benjamin Van Durme. The nlp task effectiveness of long-range transformers. *arXiv preprint arXiv:2202.07856*, 2022.

Jarrid Rector-Brooks, Mohsin Hasan, Zhangzhi Peng, Zachary Quinn, Chenghao Liu, Sarthak Mittal, Nouha Dziri, Michael Bronstein, Yoshua Bengio, Pranam Chatterjee, Alexander Tong, and Avishek Joey Bose. Steering masked discrete diffusion models via discrete denoising posterior prediction. *arXiv [cs.LG]*, 10 October 2024. URL http://arxiv.org/abs/2410.08134.

Maximo Eduardo Rulli, Simone Petruzzi, Edoardo Michielon, Fabrizio Silvestri, Simone Scardapane, and Alessio Devoto. Attention sinks in diffusion language models, 2025. URL https://arxiv.org/abs/2510.15731.

S Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr 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.

Gong Shansan, Zhang Ruixiang, Zheng Huangjie, Gu Jiatao, Jaitly Navdeep, Kong Lingpeng, and Zhang Yizhe. DiffuCoder: Understanding and improving masked diffusion models for code generation. *arXiv [cs.CL]*, 25 June 2025. URL http://arxiv.org/abs/2506.20639.

Vatsal Sharan, Sham Kakade, Percy Liang, and Gregory Valiant. Prediction with a short memory. *arXiv [cs.LG]*, 7 December 2016. URL http://arxiv.org/abs/1612.02526.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 103131–103167. Curran Associates, Inc., 2024. doi: 10.52202/079017-3277. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/bad233b9849f019aead5e5cc60cef70f-Paper-Conference.pdf.

Xue Shuchen, Xie Tianyu, Hu Tianyang, Feng Zijin, Sun Jiacheng, Kawaguchi Kenji, Li Zhenguo, and Ma 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.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard (eds.), *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://aclanthology.org/D13-1170/.

Yuxuan Song, Zheng Zhang, Cheng Luo, Pengyang Gao, Fan Xia, Hao Luo, Zheng Li, Yuehang Yang, Hongli Yu, Xingwei Qu, et al. Seed diffusion: A large-scale diffusion language model with high-speed inference. *arXiv preprint arXiv:2508.02193*, 2025.

Simeng Sun, Kalpesh Krishna, Andrew Mattarella-Micke, and Mohit Iyyer. Do long-range language models actually use long-range context? *arXiv [cs.CL]*, 19 September 2021. URL http://arxiv.org/abs/2109.09115.

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

Eric Todd, Millicent L Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. Function vectors in large language models. *arXiv [cs.CL]*, 23 October 2023. URL http://arxiv.org/abs/2310.15213.

Blerta Veseli, Julian Chibane, Mariya Toneva, and Alexander Koller. Positional biases shift as inputs approach context window limits, 2025. URL https://arxiv.org/abs/2508.07479.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In Tal Linzen, Grzegorz Chrupała, and Afra Alishahi (eds.), *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353–355, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5446. URL https://aclanthology.org/W18-5446/.

Wen Wang, Bozhen Fang, Chenchen Jing, Yongliang Shen, Yangyi Shen, Qiuyu Wang, Hao Ouyang, Hao Chen, and Chunhua Shen. Time is a feature: Exploiting temporal dynamics in diffusion language models. *arXiv [cs.CL]*, 12 August 2025. URL http://arxiv.org/abs/2508.09138.

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dLLM: Training-free acceleration of diffusion LLM by enabling KV cache and parallel decoding. *arXiv [cs.CL]*, 28 May 2025a. URL http://arxiv.org/abs/2505.22618.

Zirui Wu, Lin Zheng, Zhihui Xie, Jiacheng Ye, Jiahui Gao, Yansong Feng, Zhenguo Li, Victoria W., Guorui Zhou, and Lingpeng Kong. Dreamon: Diffusion language models for code infilling beyond fixed-size canvas. https://hkunlp.github.io/blog/2025/dreamon/, 2025b. HKU NLP Group Blog Post.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models, 2025. URL https://arxiv.org/abs/2508.15487.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pp. 649–657, Cambridge, MA, USA, 2015. MIT Press.

Kaiwen Zheng, Yongxin Chen, Hanzi Mao, Ming-Yu Liu, Jun Zhu, and Qinsheng Zhang. Masked diffusion models are secretly time-agnostic masked models and exploit inaccurate categorical sampling. In *The Thirteenth International Conference on Learning Representations*.

Fengqi Zhu, Zebin You, Yipeng Xing, Zenan Huang, Lin Liu, Yihong Zhuang, Guoshan Lu, Kangyu Wang, Xudong Wang, Lanning Wei, Hongrui Guo, Jiaqi Hu, Wentao Ye, Tieyuan Chen, Chenchen Li, Chengfu Tang, Haibo Feng, Jun Hu, Jun Zhou, Xiaolu Zhang, Zhenzhong Lan, Junbo Zhao, Da Zheng, Chongxuan Li, Jianguo Li, and Ji-Rong Wen. Llada-moe: A sparse moe diffusion language model, 2025. URL https://arxiv.org/abs/2509.24389.

Appendix Contents

A Additional Experimental Results

A.1 Results on LLaDA-MoE

Motivation. As the area of MDLMs is still in early stages of development, the number of open-source MDLMs available for evaluation is still heavily limited. In our work, we follow the example of existing works and conduct all the evaluations on the LLaDA and Dream models (Shansan et al., 2025; Israel et al., 2025; Li et al., 2025; Wang et al., 2025). We believe that the identified limitations of these models, particularly given their prevalence, can guide training and deployment of future MDLMs and hence significantly contribute to the field. To improve the generalisability of our results, we have rerun the experiments in sections section 4 and section 5 of the paper also on LLaDA-MoE (Zhu et al., 2025)–a mixture of experts MDLM, providing significant training details in the provided model report. Importantly, LLaDA-MoE has been fine-tuned to context lengths of 8k, thus increasing the context length compared to LLaDA and Dream.

Results. Figures 11-18 show the results of our analysis conducted on LLaDA-MoE. LLaDA-MoE largely displays patterns similar to that of LLaDA, although some results merit further discussion. In Figures 11 and 12 we note that LLaDA-MoE-Base does not display a significant recency bias (its performance is mostly agnostic to the location of relevant examples). We hypothesise that this might be because LLaDA-MoE was fine-tuned to handle context lengths of 8k, which is significantly more than the length of the tasks considered in our evaluation. Nevertheless, the gradient attribution analysis (Figure 13) still demonstrates patterns consistent with the recency bias present in other MDLMs, suggesting that this issue might still affect performance in tasks with longer input. The fine-tuning experiment with the MA loss (fig. 18) clearly demonstrates that the MA loss can be effective in reducing the negative effect of extra masks.

Remark. We note that the performance of LLaDA-MoE presented in Figure 11 does not align exactly with the performance for the case when we use 2.0 masks in Figure 17. We note that this discrepancy stems from the fact that in Figure 11 we use only a single mask, followed by the end of sentence, rather than two separate masks (see Section C.4 for details). This discrepancy indicates that

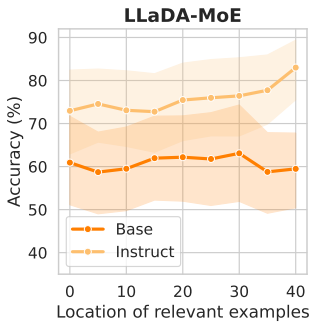

Figure 11: **Recency bias in LLaDA-MoE (re: Fig 1).** LLaDA-Moe-Instruct displays a strong recency bias, as seen also in other MDLMs, while the performance of LLaDA-MoE-Base is more agnostic to the location of relevant examples within the context.

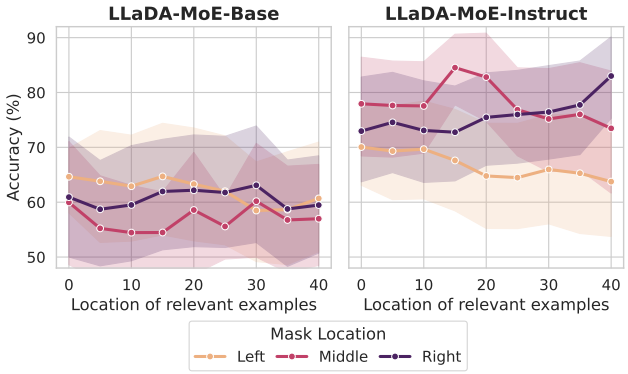

Figure 12: **Locality bias in LLaDA-MoE (re: Fig 2).** LLaDA-MoE-Instruct displays a strong locality bias, as seen also in other MDLMs (the performance is best when the relevant examples are located close to the masked question). The performance of LLaDA-MoE-Base is more uniform across the locations of relevant examples, although at a significantly lower level overall.

LLaDA-MoE is highly sensitive to the number of masks, and small variations can significantly affect performance, further reiterating the importance of our findings.

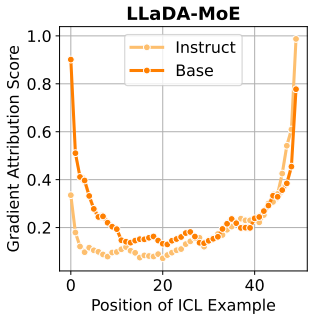

Figure 13: **Gradient attribution analysis reveals recency bias in LLaDA-MoE (re: Fig 3).** Both LLaDA-MoE-Base and LLaDA-MoE-Instruct display a strong recency and primacy bias, based on the gradient attribution analysis.

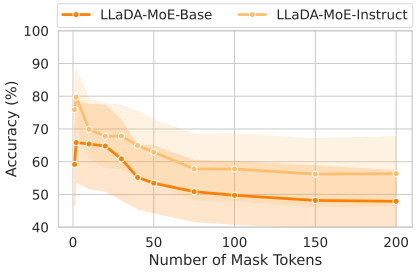

Figure 14: **Performance of LLaDA-MoE decreases significantly with added masks (re: Fig. 4).**

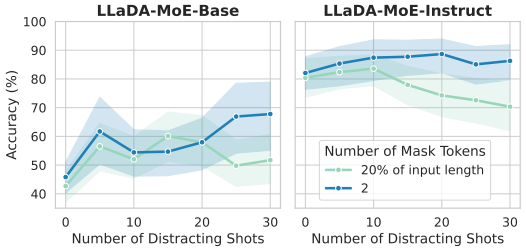

Figure 15: **For LLaDA-MoE, the performance degradation becomes more significant as the context length increases (re: Fig 5).** This effect is particularly visible in LLaDA-MoE-Instruct.

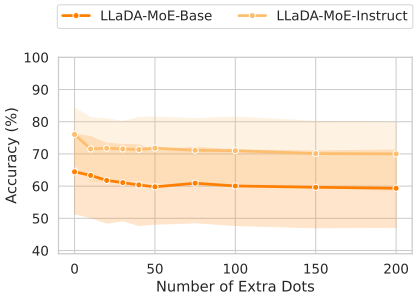

Figure 16: **For LLaDA-MoE, extra dots do not degrade performance as strongly as extra masks (re: Fig 6).**

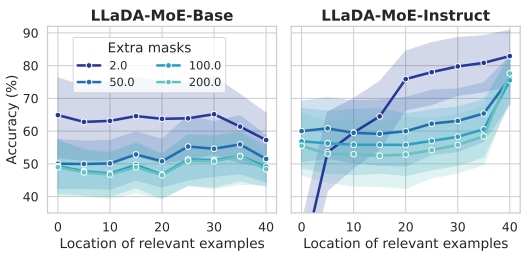

Figure 17: **Extra masks alter the locality bias in LLaDA-MoE (re: Fig. 8)**. For both the Base and the Instruct model, the performance becomes significantly worse as we add extra masks, across all locations. For LLaDA-MoE-Instruct in particular, the performance is more uniform across most locations with the extra masks.

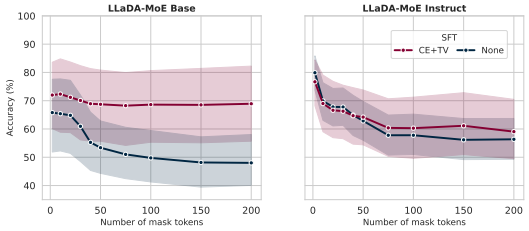

Figure 18: **MA loss rectifies the effect of extra masks.** Particularly in the LLaDA-MoE-Base model, fine-tuning with the MA loss allows to induce the robustness to extra masks, leading to improved performance. We use random unmasking strategy.

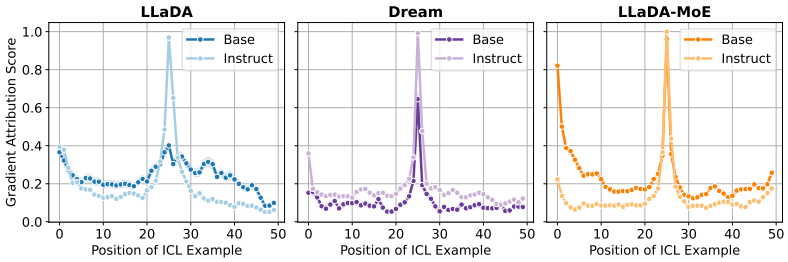

Figure 19: **Gradient attribution analysis confirms locality bias in MDLMs.**Normalised gradient attribution results for when the target question is placed in the **middle** of the input context.

A.2 Gradient Attribution Analysis of MDLMs

A.2.1 Measuring the Locality Bias in MDLMs

Motivation. To further study the locality bias of MDLMs, we conduct further gradient attribution analysis. We use the setup described in section Section 4.3, however this time we move the masked question to different locations within the input context (similarly to what we did in Section 4.2).

Results. The results of our analysis, presented in Figures 20 and 19, show a clear locality bias of the studied MDLMs: the normalised gradients have consistently larger values at positions closer to the mask of interest (i.e. for positions 20-30 when the masked question is located in the centre of the input, and for positions 0-10 when the masked question is located on the left end of the input). This provides additional evidence for our results presented in Section 4, indicating that MDLMs display a strong locality bias.

A.2.2 Gradient Attributed Towards the Extra Masks

Motivation. To assess how strongly MDLMs prioritise the extra mask tokens over any other tokens in the input, we analyse gradient-based attributions. Using the configuration from Section 4.2, we

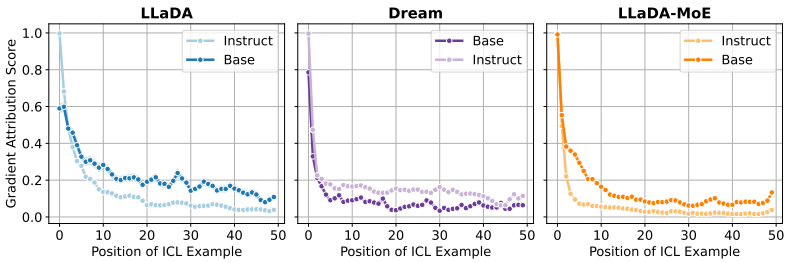

Figure 20: **Gradient attribution analysis confirms locality bias in MDLMs.** Normalised gradient attribution results for when the target question is placed on the **left** end of the input context.

Model Name	Masks	Non-Masks (Last 50)	Non-Masks
Dream-Base-7b	0.282 ± 0.040	0.012 ± 0.007	0.005 ± 0.003
Dream-Instruct-7b	0.144 ± 0.031	0.030 ± 0.005	0.018 ± 0.002
LLaDA-Base-8b	0.234 ± 0.021	0.005 ± 0.002	0.005 ± 0.002
LLaDA-Instruct-8b	0.220 ± 0.031	0.057 ± 0.014	0.017 ± 0.003
LLaDA-MoE-Base	0.237 ± 0.034	0.094 ± 0.016	0.029 ± 0.003
LLaDA-Moe-Instruct	0.188 ± 0.032	0.150 ± 0.024	0.028 ± 0.004

Table 1: **MDLMs are particularly sensitive to mask tokens.** We show the average normalised gradients attributed to the mask tokens, compared to all the other tokens in the input sequence.

append 50 mask tokens to the input and measure the normalised gradient of the masked answer token (i.e., the first mask) with respect to all other tokens in the sequence. This quantifies the influence of the added masks on the model's prediction and the model's sensitivity to their presence relative to the surrounding context.

Results. Table 1 reports the average normalised gradients for three token groups: (i) the added mask tokens, (ii) the last 50 non-mask tokens closest to the target mask, and (ii) all non-mask tokens. Across all models, gradient magnitudes attributed to mask tokens are markedly higher than those attributed to either non-mask group. This pattern indicates that MDLMs allocate disproportionate attention to the added masks, consistent with our broader observation that these models are heavily influenced by the mask tokens at the expense of effective context utilisation. We also note that the last 50 non-mask tokens (i.e. the 50 tokens located directly to the left of the mask) have significantly higher gradient scores than non-mask tokens in general, reiterating the recency bias.

A.3 Correlation Between Mask Degradation and Context Significance

Extra Masks Hurt Behaviour On Tasks Requiring Long Context Comprehension. In Section section 5.2, we presented initial evidence that additional masks impair the model's ability to utilise long contexts. Here, We investigate this effect further by analysing the performance of MDLMs on a variety of few-shot learning tasks with single-token answers.

Setup. For each task, we evaluate performance along two axes. First, we measure the gain in accuracy when increasing the number of in-context examples from 5 to 25, which serves as a proxy for the task's dependence on long-context information. Second, we compare performance between two masking configurations–one with a single extra mask and one with 200 extra masks–using the 25-shot setting. This quantifies the degradation in predictive accuracy induced by extra masks.

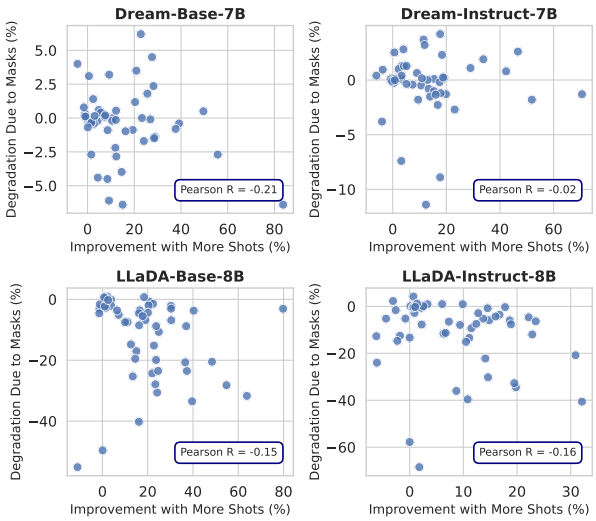

Figure 21: **For LLaDA models, tasks that benefit more from additional ICL shots exhibit stronger performance degradation under extra masks.** Dream shows no such trend, remaining more robust to extra masks.

Results. Figure 21 visualises the relationship between performance gains from additional in-context examples and degradation due to extra masks (both expressed as absolute accuracy differences). For LLaDA-Base and LLaDA-Instruct, most points lie below the $y = 0$ line, indicating substantial degradation—up to 60% on some tasks. The negative Pearson correlations ($R = -0.15$ and $R = -0.16$, respectively) suggest that tasks benefitting most from longer contexts are also those most affected by extra masks. While the correlations are modest, they

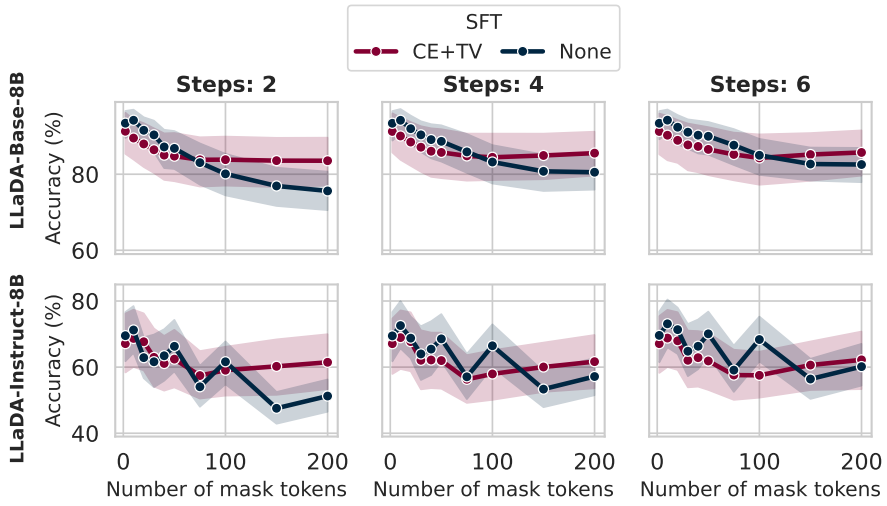

Figure 22: Performance across varying numbers of mask tokens for different decoding steps (2, 4, and 6) using random unmasking strategy, for LLaDA models. The base model (None) shows significant performance degradation as the number of mask tokens increases, while our CE+TV fine-tuned model maintains more stable performance across all configurations.

reinforce the hypothesis that masking disproportionately disrupts long-context processing, though other factors likely also determine the level of degradation.

By contrast, Dream models show minimal and less consistent degradation ($\leq 12\%$), aligning with our earlier observation that MDLMs initialised from autoregressive (AR) weights exhibit increased robustness to masking effects.

Details of the few-shot learning tasks used. Each point on the scatterplots presented in Figure 21 corresponds to a different few-shot learning task. We use the following few-shot learning datasets investigated in the different sections of the paper: (1) The pattern recognition tasks described in Section 3 (16 combinations). (2) All the variants of the multi-dimensional classification dataset described in Section A.8. Additionally, we use the following popular ICL datasets: AG News (Zhang et al., 2015), SST-2 (Socher et al., 2013), Rotten Tomatoes (Pang & Lee, 2005), as well as MRPC, RTE and QNLI from GLUE (Wang et al., 2018). 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 dataset, 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.

A.4 Robustness Analysis: Decoding with Few Steps on the LLaDA Models

Motivation. Our results in Figure 7 demonstrated that when using 40 decoding steps, the performance degradation due to extra masks is alleviated. In Figure 10 we showcased that the same effect can be achieved in just 1 decoding step with the help of our mask-agnostic fine-tuning, achieving much lower latency. Here, we provide intermediate results showing how performance varies when using 2, 4, and 6 decoding steps to further evaluate the benefits of using the mask-agnostic loss.

Results. Figure 22 shows the results obtained using the random unmasking strategy (tokens were unmasked in random order). Across all configurations, we observe that increasing the number of mask tokens leads to performance degradation in both the base model and our fine-tuned model. However,

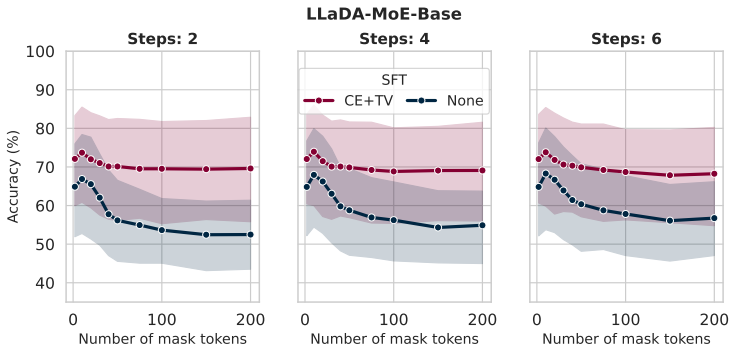

Figure 23: Performance across varying numbers of mask tokens for different decoding steps (2, 4, and 6) using random unmasking strategy, for LLaDA-MoE-Base model. The MA loss helps to rectify the negative effect of extra masks across all numbers of decoding steps considered.

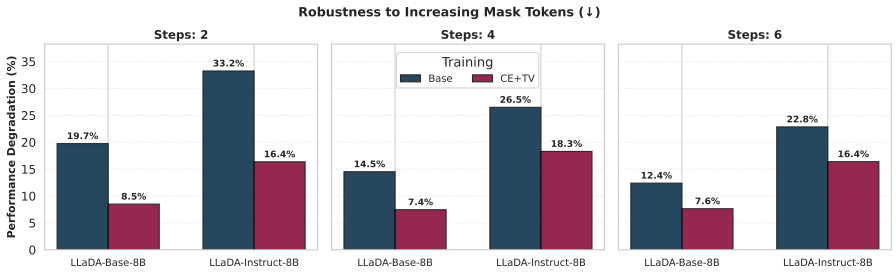

Figure 24: Relative performance degradation (measured as $\frac{\text{max accuracy} - \text{min accuracy}}{\text{max accuracy}} \times 100\%$) across different numbers of decoding steps. Lower values indicate better robustness to increasing mask tokens. Our CE+TV fine-tuning reduces degradation by 38-49% compared to the base model, demonstrating significantly improved robustness with minimal accuracy trade-offs.

the CE+TV fine-tuned model consistently maintains higher performance and exhibits significantly less degradation.

Robustness Analysis. To quantify this improved robustness, we measure the relative performance degradation as the percentage drop from maximum to minimum accuracy: $\frac{\text{max accuracy} - \text{min accuracy}}{\text{max accuracy}} \times 100\%$. As shown in Figure 24, our CE+TV fine-tuning substantially reduces performance degradation across all step configurations:

- **LLaDA-Base-8B**: Degradation reduced from 15.5% to 7.9% (49% reduction)
- **LLaDA-Instruct-8B**: Degradation reduced from 27.5% to 17.0% (38% reduction)

Importantly, this improved robustness comes with minimal accuracy trade-offs. The CE+TV model maintains competitive or superior performance at low mask token counts while being significantly more robust as the number of mask tokens increases. This demonstrates that our mask-agnostic fine-tuning not only enables efficient single-step decoding but also fundamentally improves the model's ability to handle varying numbers of mask tokens, making it more practical for real-world applications where computational constraints may vary.

However, we emphasise that while our method mitigates the degradation due to extra masks, it does not fully eliminate it. The fact that a non-negligible performance drop persists–even after targeted fine-tuning and multiple decoding steps–underscores the severity of the mask distraction phenomenon. It suggests this is not a trivial artifact, but a deep-seated characteristic of current MDLM architectures that cannot be easily ignored and requires continued investigation.

A.5 Additional Results for the Fine-Tuned LLaDA-Instruct

In Figure 25 we provide additional results visualising how the fine-tuning procedure affects the locality of the LLaDA-Instruct model, under different numbers of masks. We observe that the model is more robust to the extra masks, and its performance is more uniform over the different positions of relevant information. However, this also comes at a slight decrease in performance, compared to the non-fine-tuned model.

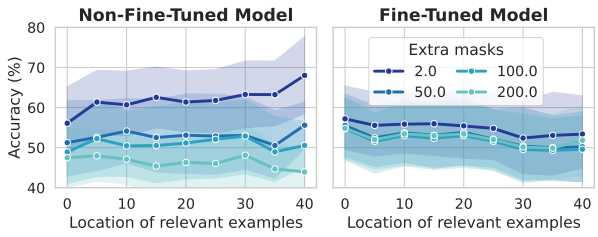

Figure 25: **MA loss (CE+TV) reduces the degrading effect of extra masks in LLaDA-Instruct, and removes the locality of model, however, at the cost of an overall performance decrease.**

Performance of the fine-tuned LLaDA-Instruct Model. Throughout our experiments, we observed that our MA loss yields consistent robustness gains for base (pre-trained) MDLM (LLaDA-Base) but inconsistent boost for instruction-tuned checkpoints (LLaDA-Instruct). Our hypothesis is that instruction tuning biases decoding toward autoregressive-like behaviour (as noted in prior work such as DiffuCoder Shansan et al. (2025)), which interacts unfavourably with vanilla ELBO based objective. The ELBO's factorisation and noise-schedule assumptions can be disrupted by instruction-conditioned guidance, leading to brittle adaptation when mask schedules vary. These results suggest that extending MDLMs to instruction-following regimes requires revisiting both objective design and hyperparameters such as loss weightings between denoising, invariance, and instruction objectives.

A.6 Confidence and Entropy as a Function of Masks

In Figure 26 we plot the effect of fine-tuning the LLaDA models with the MA loss on the confidence (calculated as the probability of the generated token, under the greedy decoding scheme) and the entropy of the model's generations. We observe that training with the MA loss significantly increases the confidence in the generated answer for the Base model, and makes the confidence more smooth as a function of extra masks for both models. Furthermore, MA loss also significantly decreases the entropy for both models, also making it more smooth.

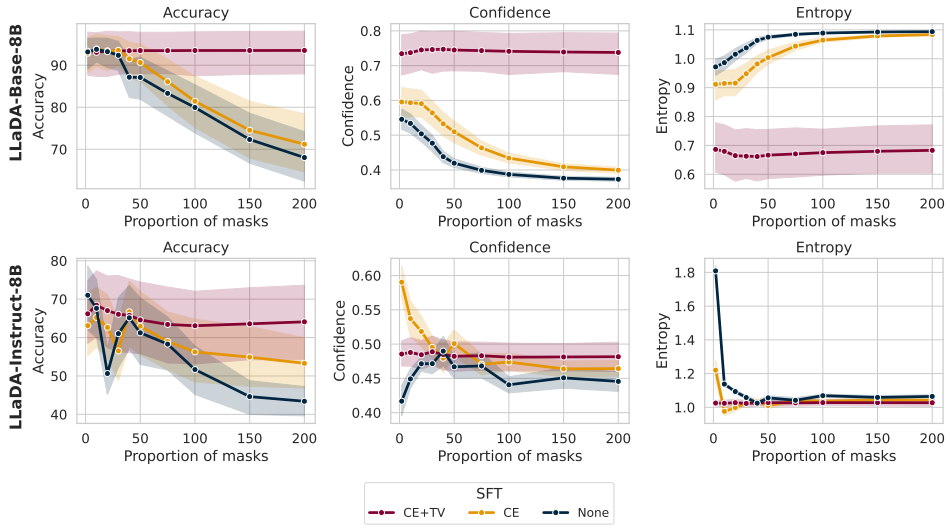

Figure 26: **MA loss (CE + TV) decreases the model's entropy and increases the confidence in the generated token**, while making both a smoother function of the number of extra masks, thus increasing the robustness of the model.

A.7 Experiments on the HotPotQA dataset

Motivation. To further apply whether our results generalise to other in-context learning tasks, beyond the few-shot learning setting, we use a subset of the HotPotQA dataset (Yang et al., 2018). This dataset consists of Wikipedia-based question-answer pairs. The questions require finding and *simultaneously* reasoning over multiple supporting documents (facts), thus ensuring that the dataset requires long-context comprehension.

Dataset. We utilised the 'distractor' configuration of HotPotQA and loaded it via the Hugging Face datasets library.[1] Our preprocessing focused on extracting binary-choice questions by filtering for examples containing "or" in the question text. Using regular expression pattern matching, we parsed such questions to extract the question stem and two possible options (A and B). We applied additional filtering to remove examples with input lengths exceeding 1000 tokens (to fit within the context window of the studied MDLMs) and those that could not be reliably converted to multiple-choice format. This approach allowed us to work with a standardized set of binary-choice questions from HotPotQA with single-token answers that were suitable for our controlled experiments and could be reliably evaluated using the accuracy metric. For each example, we concatenated the provided supporting facts (context) together with the question:

```
f"**Context**:\n'{entry['context']}'.\n\n"
    + f"**Question**: '{entry['question']}'"
    + f" [A] {entry['option_A']}\n"
    + f" [B] {entry['option_B']}\n"
    + "**Answer**:[{entry['answer']}]"
```

We use a system prompt ("Which of the following answers is true? Respond with [A] or [B].") and append one in-context learning example to ensure that the model can correctly format its answer.

As the input lengths in this dataset are more variable, rather than adding a pre-determined number of masks as in previous experiments, we add a number of masks proportional to the number of tokens in the input text.

Results. We evaluated the performance of LLaDA-Base and LLaDA-Instruct, with and without the mask-agnostic fine-tuning. Without the fine-tuning, we observe a high sensitivity of the Base model

[1] https://huggingface.co/datasets/hotpotqa/hotpot_qa

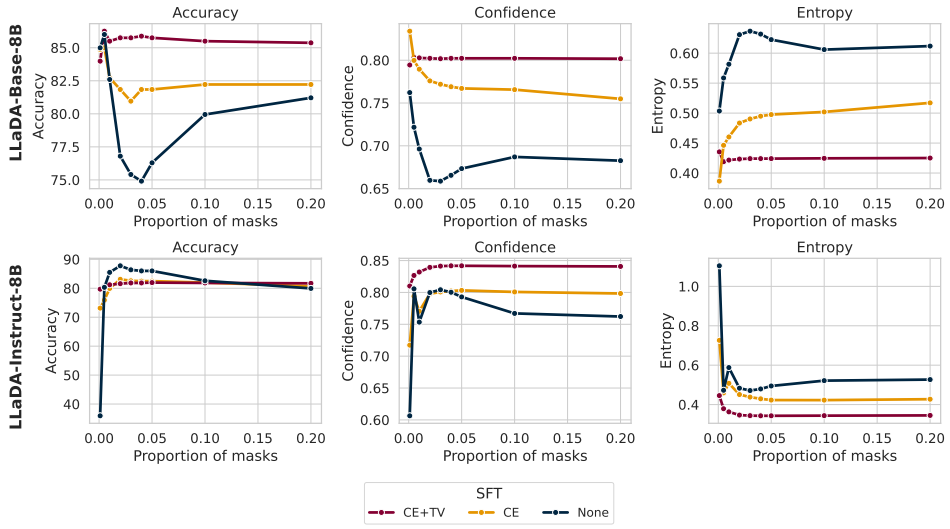

Figure 27: **On the HotPotQA dataset, the MA loss also improves the robustness of the models to the varying number of masks.** We observe improved performance particularly for the LLaDA-Base model.

to the number of extra masks, with performance decreasing sharply when the number of masks is equal to $\approx 5\%$ of the input length (which corresponds to 90-100 tokens). The fine-tuning allows to effectively remove the variability to the extra masks, once again smoothing out the confidence and entropy curves.

For the Instruct model, we note that even before the fine-tuning the model is more robust to the number of extra masks in this setting. However, the MA loss still allows to smooth out the confidence and the entropy of the model. Further, the MA loss makes the model more robust in the case when the number of available tokens is small (1-2) tokens, in which case the original model fails to provide a coherent answer.

A.8 Experiments on the Multi-dimensional Classification Dataset

Motivation. While in the pattern recognition tasks presented in the main paper it is relatively clear which examples carry signal for the test question (number vs word tasks), we also consider the setting where this distinction is more blurry, and the contribution of each example to the answer is more ambiguous. Specifically, we construct a multidimensional classification task, where each point is described using a three-dimensional integer coordinates and a binary label. To make the tasks difficult, we use different non-linear decision boundaries, described below. The task of the model is to predict the label for a new point. To measure the sensitivity to the position of information, we manipulate the order in which the points are presented: ordering them either randomly, or by the L2 distance in the input space to the test point.

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.
- **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. 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.

Results. Firstly, we use the created setup to further evaluate the locality bias of the MDLMs and ARLMs. Results in Figure 28 show that performance of MDLMs (particularly Dream, initialised with the weights of an ARLM) drops significantly when relevant examples are far from the masked question, confirming a locality bias – though weaker than in ARLMs.

Secondly, we evaluate the robustness of the LLaDA-Base and LLaDA-Instruct models to the varying numbers of masks under the different ordering schemes. We focus on the case with 30 in-context examples, with the test question located on the right-end of the in-context examples. Results in Figure 29 show that for the Base model, our MA loss prevents performance degradation, particularly for the random ordering and the ordering by increasing distance (where the most relevant information is far away from the test question). For the Instruct model, our fine-tuning scheme improves robustness to the number of masks, and consistently prevents significant performance degradation with small numbers of masks.

B Details of the Supervised Fine-Tuning Pipeline

Below, we present the pseudo-code for calculating our MA loss, and list the hyperparameters we used during fine-tuning. We use batch-size of three, and we pad the inputs with the end of

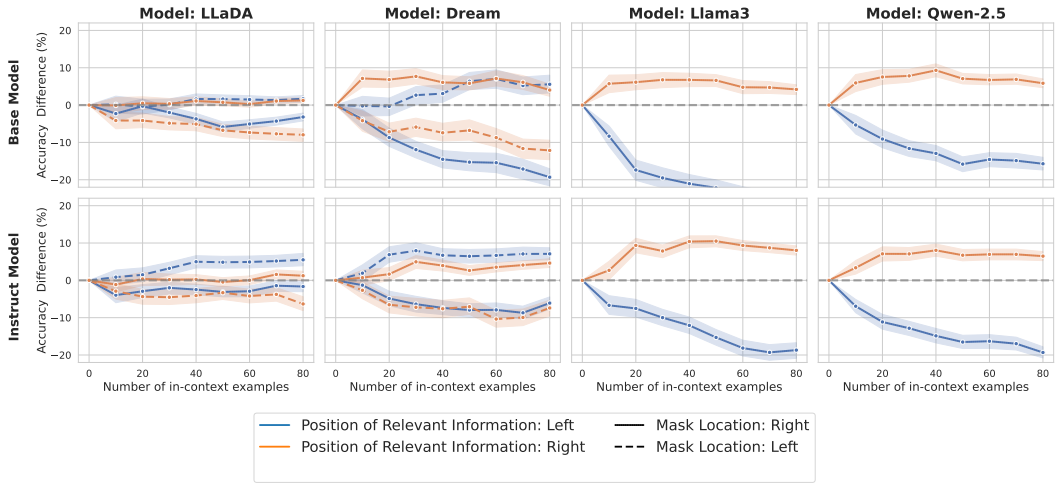

Figure 28: **In the multidimensional classification dataset, across all models, performance degrades when the relevant information is distant from the test question.** We report the accuracy difference when placing relevant information on the left versus randomly (blue line), and on the right versus 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). 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.*

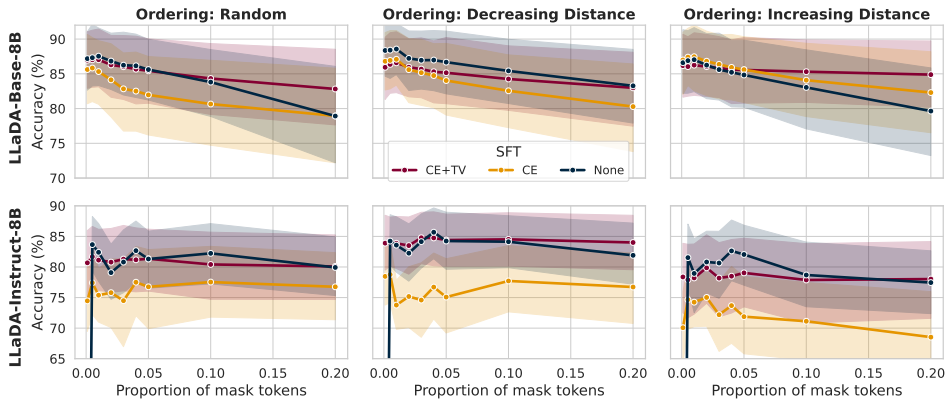

Figure 29: **In the multidimensional-classification dataset, the MA loss prevents performance degradation with the extra masks (particularly for the Base model).** We observe how the performance of the models changes under different ordering schemes of the in-context examples: random ordering, ordering by decreasing distance (most relevant information is located close to the test question) and ordering by increasing distance (most relevant information is located far from the test question). *Shaded regions indicate 95% confidence intervals computed across the 4 dataset types, and 5 seeds.*

sequence tokens to ensure equal lengths of the input. Additionally, to make training more stable, we introduce a curriculum for the lengths of the masks added at the end of the inputs, starting from minimal numbers of extra masks, and reaching 600 masks over 5000 gradient descent steps. As in our language modelling setup p_θ is a categorical distribution, we compute the TV distance $TV\left(p_\theta(x^j|x_1), p_\theta(x^j|x_2)\right)$ as the L1 distance between the probabilities (after softmax) obtained for the two inputs. We conduct the LoRA-based fine-tuning (and the subsequent evaluations of the fine-tuned models) on the non-quantised version of the LLaDA models, to ensure more stable training.

We use the following specific values of the hyperparameters for individual settings, chosen based on the value of the loss functions across the considered settings:

- **Base model, CE loss:** $\beta = 0.0$, $LR = 10^{-6}$
- **Base model, CE + TV loss:** $\beta = 1.0$, $LR = 10^{-5}$
- **Instruct model, CE loss:** $\beta = 0.0$, $LR = 10^{-5}$
- **Instruct model, CE + TV loss:** $\beta = 10.0$, $LR = 10^{-6}$

Algorithm 1 Mask-agnostic training

Require: $\mathcal{P}$: set of input pairs (q, a)
Require: p_l, p_u: lower and upper probabilities of masking
Require: N: maximum length of text allowed for the model
Require: max_masks: Maximal number of masks to be appended to the input
Require: α, β: regularisation coefficients
for (q, a) in $\mathcal{P}$ **do**
  Sample $p \sim U(p_l, p_u)$.
  Create a noised version of the answer $\tilde{a}$ with masking probability p.
  Sample $l_1, l_2 \sim \mathcal{U}\left(0, \min(L - \text{len}(p \oplus a), \text{max\_masks})\right)$.
  $x_1 \leftarrow p \oplus \tilde{a} \oplus (\text{[MASK]} * l_1)$
  $x_2 \leftarrow p \oplus \tilde{a} \oplus (\text{[MASK]} * l_2)$
  Pad x_1, x_2 with EOS tokens such that they have equal length.
  $o_1 \leftarrow \text{MDLM}(x_1)$
  $o_2 \leftarrow \text{MDLM}(x_2)$
  Compute the loss according to Equation (1).
  Backpropagate(Loss).
end for

C Experimental Details

C.1 Models

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

- **LLaDA (Nie et al., 2025):** An 8B diffusion language model pre-trained from scratch using the masked diffusion loss (Sahoo et al., 2024).
- **Dream (HKU NLP Group):** A 7B diffusion language model, whose weights are initialised from those of an autoregressive Qwen-2.5-7B.
- **Qwen-2.5-7B (Yang et al., 2024; Team, 2024):** A fully AR model.
- **Llama3-8B (AI@Meta, 2024):** A fully AR model, with the architecture similar to that of LLaDA (Nie et al., 2025).

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.

Table 2: Fine-tuning hyperparameters

Category	Parameter	Description	Value
General	Max Context Length	Maximum number of tokens processed in a single forward pass	1024
	Lower p	Lower threshold for masking probability	0.2
	Upper p	Upper threshold for probability in sampling	0.8
Loss	α	Weight coefficient for the CE loss	0.1
	β	Weight coefficient for the TV loss	See B
	Max Steps Mask Curriculum	Number of steps for mask curriculum learning	5000
	Max Masks	Maximum number of mask tokens that can appended to the sequence	600
Training	Gradient Accumulation Steps	Number of forward passes before parameter update	43
	Batch size	Size of each batch	3
	Mixed Precision	Numerical precision format used during training	bf16
	Max Gradient Norm	Maximum L2 norm of gradients for clipping	1.0
LoRA	Rank	Dimension of low-rank adaptation matrices	64
	α_l	Scaling factor for LoRA adaptation	128
	Dropout	Probability of dropping neurons during training	0.0
	Learning Rate (LR)	Step size for optimizer updates	See B
	Weight Decay	L2 regularization coefficient	0.0

C.2 Quantisation

In the experiments which *did not* involve SFT (for which we opted to use the full models), to ensure computational efficiency, we quantised all models to 4-bit precision using the Quanto library. In Figure 30 and Figure 31 we compare the performance of the quantised and non-quantised models on a single task from the pattern recognition suite, verifying that the quantisation has no significant effect on the models' locality bias, nor on the performance degradation under extra masks.

C.3 Details of the Few-Shot Learning Dataset

Below, we provide further explanations regarding the generation of the few-shot learning tasks used in the main part of the paper. For the relevant (word) tasks, we generate a list of words spanning different categories. We then create the following 8 relevant tasks, by juxtaposing the words from the target category (e.g. adjective) with words from other categories (e.g. verb):

- choose country (out of countries and names),
- choose country (out of countries and names),
- choose capitalised word (out of capitalised and non-capitalised words),
- choose verb (out of verbs, adjectives, prepositions and objects),

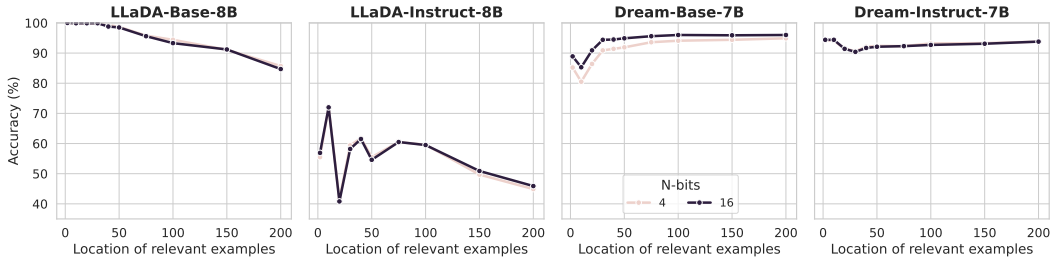

Figure 30: **Quantisation has no significant effect on the performance under varying numbers of mask tokens.**

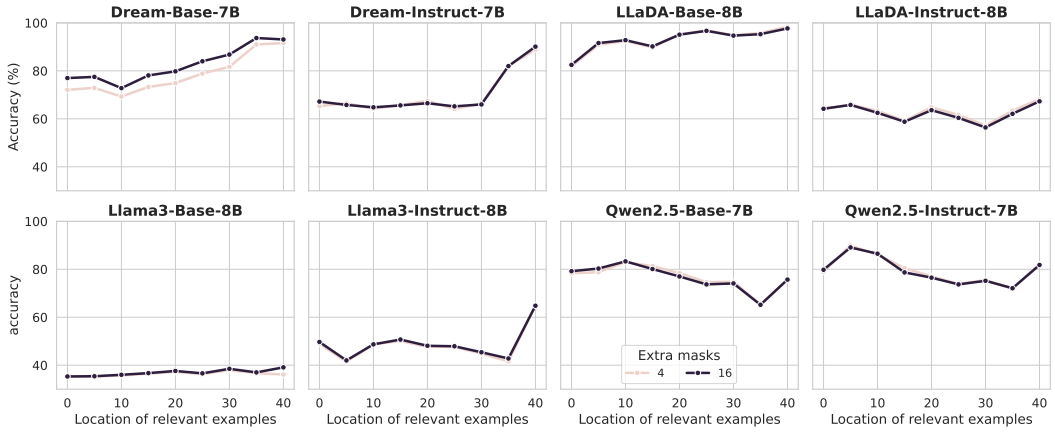

Figure 31: **Quantisation has no significant effect on the the locality of the models.**

- choose adjective (out of adjectives, verbs, prepositions and objects),
- choose animal (out of animals, objects, fruits and sports),
- choose colour (out of colours, animals and objects),
- choose emotion (out of emotions, colours, objects and animals),
- choose object (out of objects, emotions, colours and adjectives).

Additionally, we consider the following distractor (number tasks), where the candidate numbers are integers sampled without replacement from the range 1 to 1000:

- choose smallest number,
- choose largest number.

Each task contains three possible answers (A, B, C) formatted in a way presented in Section 3. To provide further illustration of the dataset considered, below we include an example of the input obtained for a dataset with task “choose verb" and distractor task “choose smallest number", in the settings when the relevant and distractor tasks are mixed (as in Figure 3) or not (as in Figure 1).

```
'Options: (A) 915, (B) 491, (C) 266\nAnswer:[C].\n\nOptions: (A) 610, (B)
    222, (C) 307\nAnswer:[B].\n\nOptions: (A) 576, (B) 510, (C) 31\
   nAnswer:[C].\n\nOptions: (A) 463, (B) 142, (C) 797\nAnswer:[B].\n\
   nOptions: (A) arrive, (B) thoughtful, (C) near\nAnswer:[A].\n\
   nOptions: (A) 941, (B) 371, (C) 341\nAnswer:[C].\n\nOptions: (A) 694,
    (B) 772, (C) 727\nAnswer:[A].\n\nOptions: (A) tall, (B) compete, (C)
    silly\nAnswer:[B].\n\nOptions: (A) 809, (B) 293, (C) 663\nAnswer:[B
   ].\n\nOptions: (A) 755, (B) 63, (C) 166\nAnswer:[B].\n\nOptions: (A)
   450, (B) 398, (C) 750\nAnswer:[B].\n\nOptions: (A) 541, (B) 698, (C)
   124\nAnswer:[C].\n\nOptions: (A) 289, (B) 567, (C) 774\nAnswer:[A].\n
```

```
    \nOptions: (A) reliable, (B) search, (C) zucchini\nAnswer:[B].\n\
    nOptions: (A) 289, (B) 373, (C) 197\nAnswer:[C].\n\nOptions: (A) 402,
     (B) 785, (C) 467\nAnswer:[A].\n\nOptions: (A) 555, (B) 287, (C) 607\
    nAnswer:[B].\n\nOptions: (A) 302, (B) 102, (C) 265\nAnswer:[B].\n\
    nOptions: (A) 790, (B) 409, (C) 904\nAnswer:[B].\n\nOptions: (A)
    deliver, (B) graceful, (C) sensitive\nAnswer:[A].\n\nOptions: (A)
    143, (B) 388, (C) 159\nAnswer:[A].\n\nOptions: (A) 52, (B) 285, (C)
    847\nAnswer:[A].\n\nOptions: (A) 688, (B) 588, (C) 426\nAnswer:[C].\n
    \nOptions: (A) 752, (B) 680, (C) 295\nAnswer:[C].\n\nOptions: (A) 24,
     (B) 868, (C) 400\nAnswer:[A].\n\nOptions: (A) 865, (B) 455, (C) 497\
    nAnswer:[B].\n\nOptions: (A) 214, (B) 506, (C) 469\nAnswer:[A].\n\
    nOptions: (A) 242, (B) 138, (C) 689\nAnswer:[B].\n\nOptions: (A) 159,
     (B) 51, (C) 824\nAnswer:[B].\n\nOptions: (A) 436, (B) 773, (C) 587\
    nAnswer:[A].\n\nOptions: (A) 95, (B) 312, (C) 390\nAnswer:[A].\n\
    nOptions: (A) 30, (B) 982, (C) 727\nAnswer:[A].\n\nOptions: (A) 323,
    (B) 590, (C) 480\nAnswer:[A].\n\nOptions: (A) 640, (B) 621, (C) 525\
    nAnswer:[C].\n\nOptions: (A) 464, (B) 836, (C) 125\nAnswer:[C].\n\
    nOptions: (A) 759, (B) 278, (C) 491\nAnswer:[B].\n\nOptions: (A) 70,
    (B) 435, (C) 386\nAnswer:[A].\n\nOptions: (A) jar, (B) kiss, (C)
    thoughtful\nAnswer:[B].\n\nOptions: (A) 733, (B) 603, (C) 211\nAnswer
    :[C].\n\nOptions: (A) 73, (B) 48, (C) 876\nAnswer:[B].\n\nOptions: (A
    ) passionate, (B) lettuce, (C) master\nAnswer:[C].\n\nOptions: (A)
    169, (B) 784, (C) 919\nAnswer:[A].\n\nOptions: (A) lucky, (B) train,
    (C) igloo\nAnswer:[B].\n\nOptions: (A) for, (B) calculate, (C) cube\
    nAnswer:[B].\n\nOptions: (A) 861, (B) 579, (C) 735\nAnswer:[B].\n\
    nOptions: (A) 844, (B) 207, (C) 774\nAnswer:[B].\n\nOptions: (A) 502,
     (B) 361, (C) 954\nAnswer:[B].\n\nOptions: (A) innocent, (B) relax, (
    C) upbeat\nAnswer:[B].\n\nOptions: (A) underneath, (B) kill, (C)
    spicy\nAnswer:[B].\n\nOptions: (A) 935, (B) 501, (C) 459\nAnswer:[C
    ].\n\nOptions: (A) concerning, (B) hate, (C) painting\nAnswer:['
```

Listing 1: Example of the input for the few shot learning tasks, with the relevant task “choose verb" and the distractor task “choose smallest number", in the case when the examples are mixed.

```
'Options: (A) deliver, (B) graceful, (C) sensitive\nAnswer:[A].\n\
    nOptions: (A) innocent, (B) relax, (C) upbeat\nAnswer:[B].\n\nOptions
    : (A) jar, (B) kiss, (C) thoughtful\nAnswer:[B].\n\nOptions: (A)
    reliable, (B) search, (C) zucchini\nAnswer:[B].\n\nOptions: (A)
    arrive, (B) thoughtful, (C) near\nAnswer:[A].\n\nOptions: (A)
    passionate, (B) lettuce, (C) master\nAnswer:[C].\n\nOptions: (A)
    lucky, (B) train, (C) igloo\nAnswer:[B].\n\nOptions: (A) underneath,
    (B) kill, (C) spicy\nAnswer:[B].\n\nOptions: (A) for, (B) calculate,
    (C) cube\nAnswer:[B].\n\nOptions: (A) tall, (B) compete, (C) silly\
    nAnswer:[B].\n\nOptions: (A) 555, (B) 287, (C) 607\nAnswer:[B].\n\
    nOptions: (A) 463, (B) 142, (C) 797\nAnswer:[B].\n\nOptions: (A) 289,
     (B) 567, (C) 774\nAnswer:[A].\n\nOptions: (A) 464, (B) 836, (C) 125\
    nAnswer:[C].\n\nOptions: (A) 861, (B) 579, (C) 735\nAnswer:[B].\n\
    nOptions: (A) 844, (B) 207, (C) 774\nAnswer:[B].\n\nOptions: (A) 755,
     (B) 63, (C) 166\nAnswer:[B].\n\nOptions: (A) 502, (B) 361, (C) 954\
    nAnswer:[B].\n\nOptions: (A) 52, (B) 285, (C) 847\nAnswer:[A].\n\
    nOptions: (A) 576, (B) 510, (C) 31\nAnswer:[C].\n\nOptions: (A) 242,
    (B) 138, (C) 689\nAnswer:[B].\n\nOptions: (A) 541, (B) 698, (C) 124\
    nAnswer:[C].\n\nOptions: (A) 159, (B) 51, (C) 824\nAnswer:[B].\n\
    nOptions: (A) 610, (B) 222, (C) 307\nAnswer:[B].\n\nOptions: (A) 302,
     (B) 102, (C) 265\nAnswer:[B].\n\nOptions: (A) 915, (B) 491, (C) 266\
    nAnswer:[C].\n\nOptions: (A) 694, (B) 772, (C) 727\nAnswer:[A].\n\
    nOptions: (A) 733, (B) 603, (C) 211\nAnswer:[C].\n\nOptions: (A) 214,
     (B) 506, (C) 469\nAnswer:[A].\n\nOptions: (A) 809, (B) 293, (C) 663\
    nAnswer:[B].\n\nOptions: (A) 865, (B) 455, (C) 497\nAnswer:[B].\n\
    nOptions: (A) 450, (B) 398, (C) 750\nAnswer:[B].\n\nOptions: (A) 323,
     (B) 590, (C) 480\nAnswer:[A].\n\nOptions: (A) 688, (B) 588, (C) 426\
    nAnswer:[C].\n\nOptions: (A) 169, (B) 784, (C) 919\nAnswer:[A].\n\
    nOptions: (A) 790, (B) 409, (C) 904\nAnswer:[B].\n\nOptions: (A) 30,
    (B) 982, (C) 727\nAnswer:[A].\n\nOptions: (A) 73, (B) 48, (C) 876\
    nAnswer:[B].\n\nOptions: (A) 402, (B) 785, (C) 467\nAnswer:[A].\n\
```

```
nOptions: (A) 289, (B) 373, (C) 197\nAnswer:[C].\n\nOptions: (A) 935,
 (B) 501, (C) 459\nAnswer:[C].\n\nOptions: (A) 24, (B) 868, (C) 400\
nAnswer:[A].\n\nOptions: (A) 436, (B) 773, (C) 587\nAnswer:[A].\n\
nOptions: (A) 143, (B) 388, (C) 159\nAnswer:[A].\n\nOptions: (A) 640,
 (B) 621, (C) 525\nAnswer:[C].\n\nOptions: (A) 941, (B) 371, (C) 341\
nAnswer:[C].\n\nOptions: (A) 95, (B) 312, (C) 390\nAnswer:[A].\n\
nOptions: (A) 70, (B) 435, (C) 386\nAnswer:[A].\n\nOptions: (A) 752,
(B) 680, (C) 295\nAnswer:[C].\n\nOptions: (A) 759, (B) 278, (C) 491\
nAnswer:[B].\n\nOptions: (A) concerning, (B) hate, (C) painting\
nAnswer:['
```

Listing 2: Example of the input for the few shot learning tasks, with the relevant task “choose verb" and the distractor task “choose smallest number", in the case when the examples are not mixed, and the relevant examples are at position 0.0.

C.4 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). 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.

Diffusion models.

- In section 4: 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 `"Answer:[<|mask|>]."` and add this to the prompt, where `<|mask|>` is the textual representation of the mask token, specific to each MDLM. We add exactly one copy of the mask token in between the square brackets. We also use this setup in the experiments with extra dots, appending the dots *after* the closing the bracket of the answer.
- In section 5 and 6 (as well as in other experiments with varying number of extra masks), we use a generation style more resembling the setup for ARLMs, where we add the full test question and the beginning of the answer (e.g., `"Answer:["`) to the final formatted prompt, followed by the specified number of masks. As the closing bracket `].` is typically tokenised as a single token, using this setup with exactly two extra masks allows us to mostly recover the performance seen in the previous setup.

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).

Changing the location of the masked question. In Figure 2, 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)).