Diffusion Language Models Can Approximate Optimal Infilling Lengths Implicitly

Hengchang Liu[1,2,3]* **Zhao Yang**[1,2,3]* **Bing Su**[1,2,3]†

[1]Gaoling School of Artificial Intelligence
Renmin University of China, Beijing, China

[2]Beijing Key Laboratory of Research on Large Models
and Intelligent Governance

[3]Engineering Research Center of Next-Generation
Intelligent Search and Recommendation, MOE

{liuhengchang,yangyz1230,bingsu}@ruc.edu.cn

Abstract

Diffusion language models (DLMs) provide a bidirectional generation framework naturally suited for infilling, yet their performance is constrained by the pre-specified infilling length. In this paper, we show that DLM first-step confidence can provide a useful signal for infilling length. We identify two key statistical phenomena in the first-step denoising confidence: a local *Oracle Peak* that emerges near the ground-truth length and a systematic *Length Bias* that often obscures this signal. By leveraging this signal and calibrating the bias, our training-free method **CAL** (**C**alibrated **A**daptive **L**ength) selects an infilling length through a search before formal decoding. On LLaDA-Base Single-Line infilling, CAL improves average Pass@1 from 44.8% to 65.4% across four initial lengths; gains reach 47.7pp over Fixed and 40.5pp over the chat-based adaptive baseline at $L_{init} = 4$. Text-infilling gains reach 8.5pp in BLEU-2 and 9.9pp in ROUGE-L. These results establish calibrated first-step confidence as an effective and practical signal for training-free adaptive length control in DLM infilling. Code is available at `https://github.com/NiuHechang/Calibrated_Adaptive_Length`.

1 Introduction

Recently, Diffusion Language Models (DLMs) (Gong et al., 2024; Nie et al., 2025; Ye et al., 2025) have demonstrated significant potential in natural language processing. Unlike traditional Autoregressive (AR) models that rely on token-by-token causal generation, diffusion models generate sequences via iterative denoising over the entire sequence (Ho et al., 2020; Austin et al., 2021). While mainstream DLMs predominantly focus on chat tasks akin to AR models, this usage underutilizes their unique potential. With their natural bidirectional context modeling, diffusion models are inherently suited for infilling tasks (Lewis et al., 2020; Donahue et al., 2020) that require satisfying constraints from both the prefix and suffix. This capability is particularly critical for applications such as code completion (Bavarian et al., 2022; Fried et al., 2022), text infilling (Zhu et al., 2019; Lewis et al., 2020), and controllable generation (Zhou et al., 2023; Xiong et al., 2025), where the generated segment must syntactically and logically bridge the surrounding context.

*Equal contribution.
†Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

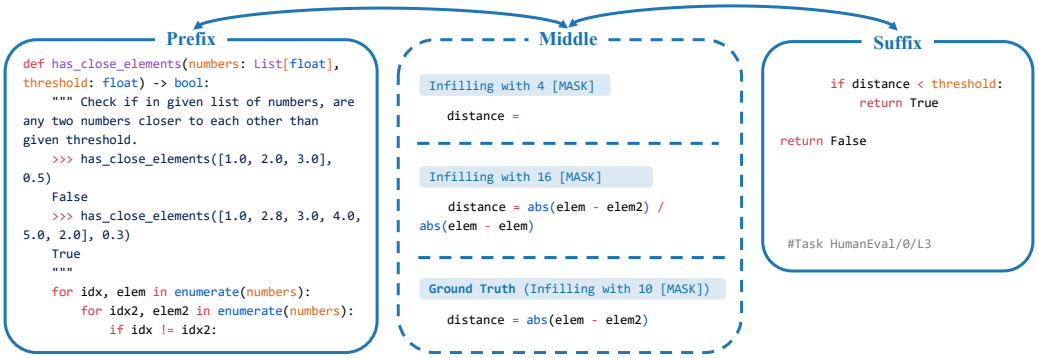


Figure 1: **Infilling sensitivity to a fixed mask length.** Evaluated on HumanEval-Infilling (Bavarian et al., 2022) using LLaDA-8B-Instruct. An underestimated length ($L = 4$) truncates the completion, whereas an overestimated length ($L = 16$) introduces redundant tokens and may yield invalid logic (e.g., division by zero) to fill the remaining masked positions. Among the lengths shown in this example, the correct solution is recovered at the ground-truth length ($L = 10$).

Despite this theoretical promise, the actual performance of DLMs in infilling tasks often falls short of expectations (see Section 3). We find that this performance gap primarily stems from the lack of adaptive length adjustment in DLMs, as quantified in Table 1. In chat-based tasks, DLMs can generate a specific `<eos>` token to terminate generation (Nie et al., 2025; Gong et al., 2024; Ye et al., 2025). In contrast, for infilling tasks, these models lack an intrinsic stopping mechanism and thus rely on a fixed, pre-specified mask length (Gong et al., 2025; Xie et al., 2025). Without a dynamic mechanism, the generation quality of DLMs is highly sensitive to the pre-specified mask length. As shown in Figure 1, an underestimated infilling length truncates the completion, whereas an overestimated length causes the model to generate redundant or incorrect tokens to fill the remaining masked positions. This length constraint significantly limits DLM performance on infilling tasks.

To overcome this limitation without retraining, we investigate whether DLMs implicitly encode infilling length information within their generation process. We analyze the model's inherent confidence during first-step denoising from a fully masked state. By evaluating the average confidence across different infilling lengths, we observe two key statistical phenomena (see Figure 2): (1) *Oracle Peak*: under the same context, the average first-step confidence exhibits a local maximum when the mask length is close to the ground-truth length; (2) *Length Bias*: the average confidence decreases rapidly and then gradually stabilizes as the infilling length increases.

Intuitively, the Oracle Peak suggests that first-step confidence encodes a notion of semantic completeness: when the infilling length matches the ground-truth span, the model assigns higher certainty to its initial token predictions (see Figure 2a). In contrast, Length Bias reflects a systematic decline of average confidence as the number of masked positions increases in a fully masked state (see Figure 2b): longer masked regions weaken contextual constraints and introduce more low-certainty positions, which lowers the average token-level confidence. By calibrating the first-step confidence with the estimated bias, the length-dependent trend is largely removed, making the Oracle Peak more salient. Overall, these observations suggest that calibrated first-step confidence can serve as a practical signal for infilling length discovery, enabling adaptive length control without additional training.

Based on these findings, we propose **CAL** (**C**alibrated **A**daptive **L**ength), a training-free method for infilling length discovery. CAL introduces a length probing stage prior to formal decoding (see Figure 3 and Algorithm 1). Starting from an initial length, we evaluate the calibrated first-step confidence across a set of candidate lengths and perform a bidirectional hill-climbing search to select a suitable infilling length. Since each probe requires only a single forward pass to compute the first-step logits, the additional inference overhead is modest.

Empirical evaluations on both code and text infilling benchmarks (see Table 2 and Table 3) demonstrate that CAL consistently outperforms baselines across multiple DLMs. On LLaDA-Base Single-Line infilling at $L_{init} = 4$, gains reach 47.7pp over Fixed and 40.5pp over the chat-based adaptive baseline. These are configuration-specific gains, not typical gains; averaged over four initial lengths, Pass@1 rises from 44.8% to 65.4%. Text-infilling gains reach 8.5pp in BLEU-2 and 9.9pp in ROUGE-

L. Our method reduces sensitivity to the pre-specified mask length, enabling more robust generation under bidirectional constraints. Our main contributions are: (1) revealing and modeling the *Oracle Peak* and *Length Bias* phenomena in first-step denoising for infilling; (2) proposing a training-free strategy for adaptive length probing via calibrated confidence; and (3) demonstrating the effectiveness of our method through extensive experiments.

2 Preliminaries

Diffusion Language Models. Diffusion language models (Austin et al., 2021; Hoogeboom et al., 2021) generate sequences via iterative denoising with bidirectional context. Let $\mathbf{x}^0 = [x_1^0, \ldots, x_N^0]$ denote a clean sequence of length N. A forward noising process progressively corrupts $\mathbf{x}^0$ into $\mathbf{x}^t$ over T steps (e.g., by random masking (Sahoo et al., 2024; Shi et al., 2024)), yielding a heavily corrupted state $\mathbf{x}^T$. A DLM defines a reverse generative process parameterized by θ that reconstructs $\mathbf{x}^0$ from $\mathbf{x}^T$:

$$p_\theta(\mathbf{x}^{0:T}) = p(\mathbf{x}^T) \prod_{t=1}^{T} p_\theta(\mathbf{x}^{t-1} \mid \mathbf{x}^t). \tag{1}$$

In practice, many modern DLMs (e.g., LLaDA (Nie et al., 2025)) are trained to predict the clean tokens directly from a noisy state, i.e., to model $p_\theta(\mathbf{x}^0 \mid \mathbf{x}^t)$. The resulting token distributions provide the confidence signal used throughout our analysis and length discovery method.

Infilling Task Formulation. The infilling task (Bavarian et al., 2022; Fried et al., 2022; Zhu et al., 2019) aims to generate a semantically consistent middle segment M of length L, given a prefix context P and a suffix context S. Formally, the complete sequence is $\mathbf{x} = [P; M; S]$. While this formulation applies broadly, a critical constraint in diffusion-based infilling is that the length L must be pre-specified to initialize the masked input state $\mathbf{x}^T$. Specifically, the model constructs an initial noisy input $\mathbf{x}^T = [P; \texttt{[MASK]}^L; S]$ and denoises the masked region conditioned on the bidirectional context. Since a suitable length is typically unknown a priori, a mismatched L can degrade generation quality, as the model cannot dynamically adjust the span size.

Following Wu et al. (2026), we use *oracle length* L^* to denote the tokenized ground-truth length. It is a reference, not necessarily the unique optimal length or a performance upper bound: both code and open-ended text can admit valid completions of different lengths. CAL does not use L^* at inference; reference-length error is only a diagnostic.

First-Step Denoising Confidence. To address the unknown length challenge without auxiliary training, we leverage the intrinsic uncertainty estimates provided by the diffusion model. We focus specifically on the first-step denoising, which corresponds to the model's initial prediction $p_\theta(\mathbf{x}^0 \mid \mathbf{x}^T)$ given the fully masked input state $\mathbf{x}^T = [P; \texttt{[MASK]}^L; S]$. This prediction represents the model's "first guess" of the complete sequence based on the bidirectional context. Crucially, we isolate the probabilities corresponding to the masked segment M. Let $p_j(v)$ denote the probability assigned to token v at the j-th position of the generated sequence. We define the average first-step confidence $\Phi(L)$ over the masked indices $\mathcal{I}_{mask}$ as:

$$\Phi(L) = \frac{1}{L} \sum_{j \in \mathcal{I}_{mask}} \max_{v \in \mathcal{V}} p_j(v), \tag{2}$$

where $\mathcal{I}_{mask}$ represents the set of indices corresponding to the middle segment M, and $\mathcal{V}$ is the vocabulary. This metric serves as a computationally efficient proxy for semantic compatibility, allowing us to probe for the optimal length before committing to the full denoising trajectory.

3 Analysis: Length-Confidence Dynamics

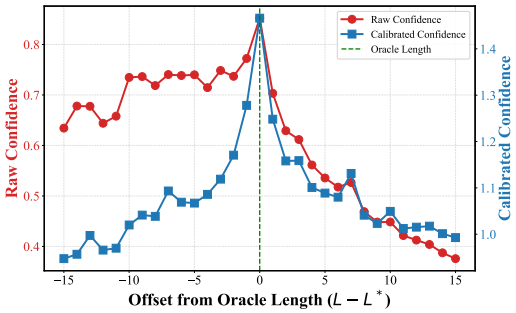
Sensitivity of Diffusion Infilling to Mask Length. Existing DLMs (Nie et al., 2025; Ye et al., 2025) typically follow a training pipeline of pre-training followed by instruction fine-tuning (SFT). During pre-training, the model learns to reconstruct text under random masks, while during SFT, it is optimized for instruction-response pairs. While effective for standard chat generation, this paradigm lacks explicit supervisory signals for `<eos>` or `<padding>` tokens within a masked span. Consequently, these models inherently lack the ability to automatically adjust infilling lengths.

Table 1: **Oracle length significantly improves DLM infilling performance.** Comparison of Pass@1 scores on HumanEval-Infilling (Bavarian et al., 2022) (Single-Line) between fixed-length settings and the Oracle length setting. The Oracle Length is determined by encoding the ground-truth answer with each model's tokenizer. "Avg. Len" indicates the average Oracle length across all test cases. This motivating analysis includes earlier full-set runs; its values are not directly comparable to the calibration-excluded main evaluation.

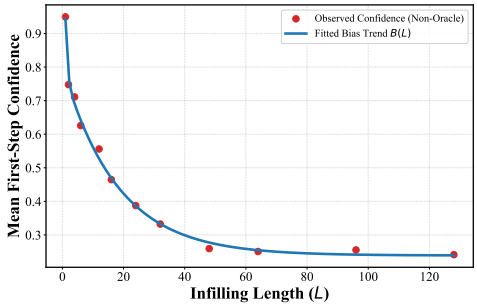
Model	Fixed Length (Pass@1)				Avg.	Oracle	
	4	8	16	32		Pass@1	Avg. Len
LLaDA-Base	22.4	51.0	55.8	48.4	44.4	**87.6**	9.5
DiffuCoder-Base	24.1	51.5	59.8	47.6	45.8	**87.4**	8.6
DreamCoder-Base	24.0	55.6	63.2	54.3	49.3	**88.1**	8.6

As illustrated in Figure 1, generation quality highly depends on the preset [MASK] length: an underestimated length forces semantic truncation, while an overestimated length causes the model to hallucinate redundant content to fill the remaining slots. Experiments confirm that providing the Oracle length (the token count of the ground-truth answer) significantly boosts performance (see Table 1), suggesting that length discovery is the primary bottleneck for unleashing the potential of diffusion models.

Oracle Peak: The Inherent Signal of Semantic Completeness. Although diffusion models lack an explicit stopping mechanism, we discover that a strong signal of logical completeness is encoded within their probability distributions. Inspired by confidence-guided adaptive length methods (Li et al., 2025), we systematically analyze the average first-step denoising confidence $\Phi(L)$ as a function of the offset $(L - L^*)$, where L is the infilling length and L^* is the Oracle length. We observe that, under fixed contextual constraints, the confidence curve exhibits a significant and stable local maximum near the Oracle length (see Figure 2a), which we term the *Oracle Peak*. This phenomenon suggests that diffusion models can discriminate semantic completeness at the very onset of denoising: when the allocated mask span matches the ground-truth length, the model's prediction certainty reaches a maximum. This provides a solid empirical basis for extracting length information without auxiliary training.



(a) **Oracle Peak Phenomenon.** The red line represents raw confidence, while the blue line shows confidence after bias calibration.



(b) **Length Bias Fitting.** Red dots represent empirical $\Phi(L)$ samples; the blue curve shows the fitted function $B(L)$.

Figure 2: **Statistical Analysis of First-Step Denoising Confidence $\Phi(L)$.** (a) A distinct confidence peak emerges when aligned with the ground-truth length. (b) Without length information, confidence exhibits systematic decay, necessitating calibration. Statistics are derived from 100 random HumanEval-Infilling (Bavarian et al., 2022) tasks using LLaDA-8B-Base.

Quantifying Systematic Length Bias. While observing the Oracle Peak, we also identify a systematic phenomenon that interferes with the average first-step denoising confidence $\Phi(L)$, which we term *Length Bias*. Specifically, $\Phi(L)$ exhibits an overall trend of rapid initial decline followed by gradual stabilization as L increases, often causing the Oracle Peak to be obscured by background noise. This bias is consistent with declining prediction certainty as the masked region grows while the prefix and suffix remain fixed. As shown by the red line in Figure 2a, the raw $\Phi(L)$, plotted against $(L - L^*)$, values on the left side of the Oracle length are significantly higher than those on

the right, displaying clear asymmetry. This semantics-agnostic bias constitutes the primary obstacle to cross-length comparison and needs to be eliminated through calibration.

Calibrating Length Bias via Fitting. To address the multi-scale decay of Length Bias, we fit a double-exponential function $B(L) = ae^{-bL} + ce^{-dL} + \beta$ using 100 samples from the HumanEval-Infilling benchmark with LLaDA-8B-Base (see Figure 2b). To isolate structural bias, we exclude probe lengths near the ground-truth length during fitting, and exclude these 100 infilling tasks from the main code evaluation. We define the calibrated confidence $\Phi_c(L)$ as the ratio of raw confidence to the fitted bias:

$$\Phi_c(L) = \frac{\Phi(L)}{B(L)}. \tag{3}$$

We interpret $B(L)$ as a length-dependent baseline that captures the structural component of the confidence decay. Under this interpretation, $\Phi_c(L)$ functions as a signal-to-noise-like ratio, separating semantic signals from length-dependent statistical bias. In the aggregate curve of Figure 2a, calibration makes the near-reference peak more prominent; this is an empirical observation, not a guarantee for every input. The double-exponential form is selected empirically, and Appendix B offers heuristic intuition rather than a derivation. Specific fitting details are provided in Appendix C. We reuse this code-fitted $B(L)$ across the main code and text experiments; in-domain, alternative-score, and multimodal fits are identified separately.

4 Method

Building on the statistical findings in Section 3, we propose CAL (Calibrated Adaptive Length), a training-free framework for infilling length discovery. As illustrated in Figure 3, CAL introduces a length probing stage prior to the formal DLM decoding. This stage formulates length selection as a discrete optimization problem over a calibrated confidence landscape, aiming to approximate the Oracle length to enhance infilling performance.

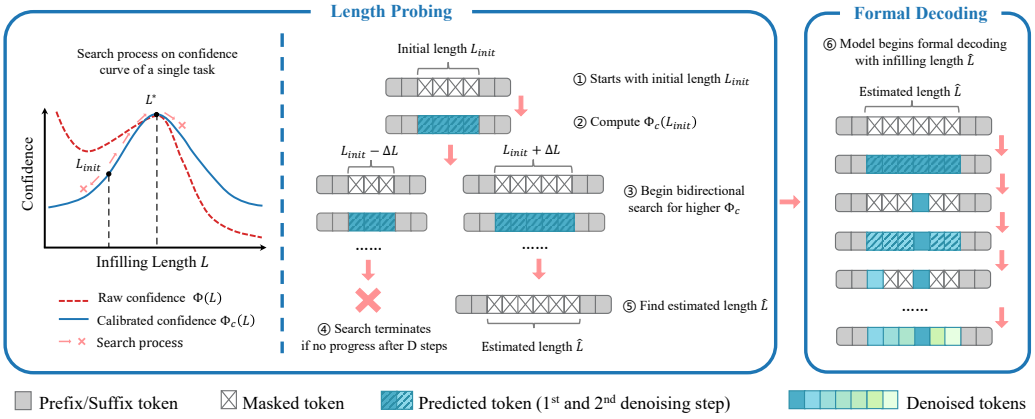


Figure 3: **Calibrated Adaptive Length (CAL).** Two stages: (1) **Length Probing** uses bidirectional hill-climbing to find $\hat{L}$ that maximizes calibrated first-step confidence $\Phi_c(L)$, approximating the oracle length L^*; (2) **Formal Decoding** performs standard iterative denoising initialized at $\hat{L}$.

Optimization Objective. As demonstrated in Section 3, providing the Oracle length substantially improves infilling performance, and confidence often peaks near this reference in our analysis. We use this signal to select an estimated length $\hat{L}$, without assuming that the reference is uniquely optimal. Directly maximizing the raw confidence $\Phi(L)$ is suboptimal because the Oracle Peak is typically obscured by systematic Length Bias. We therefore formulate length discovery as maximizing the calibrated confidence $\Phi_c(L)$:

$$\hat{L} = \arg\max_{L \in \mathcal{R}} \Phi_c(L), \tag{4}$$

where $\mathcal{R}$ denotes the search space of candidate lengths.

Bidirectional Search Algorithm. Exhaustively computing $\Phi_c(L)$ for every candidate length is computationally inefficient. We instead use a bidirectional hill-climbing heuristic motivated by the

observed peaks. As detailed in Algorithm 1, the algorithm starts from L_{init} and searches increasing then decreasing lengths, sharing the best score found across both directions. Each direction stops after D consecutive probes fail to exceed that best score. This approximates the objective without guaranteeing a global optimum. Formal decoding then uses the selected length $\hat{L}$. Each probe requires one forward pass; Appendix D reports search and decoding time. A detailed case study is provided in Appendix I.

5 Experiment

5.1 Experimental Setup

We evaluate the effectiveness of CAL on both code and text infilling benchmarks.

Code Infilling. We utilize the HumanEval-Infilling benchmark (Bavarian et al., 2022), a standard dataset for Python code completion, evaluating on both Single-Line and Multi-Line tasks. Crucially, the 100 samples used for Length Bias fitting in Section 3 were strictly excluded from the test set. We apply our method to four representative discrete diffusion language models: the general-purpose LLaDA-8B (Base and Instruct) (Nie et al., 2025), and the code-specialized DiffuCoder-7B-Base (Gong et al., 2025) and DreamCoder-Base-7B (Xie et al., 2025). Additionally, we adapt DAEDAL (Li et al., 2025), a confidence-based training-free method originally designed for chat tasks, as a baseline on LLaDA-8B-Base. All models employ greedy sampling for formal decoding, and we report Pass@1 as the primary metric.

Text Infilling. To assess generalization across diverse domains, we conduct experiments on three datasets: ROCStories (short five-sentence stories) (Mostafazadeh et al., 2016), CS Abstracts (academic writing) (Donahue et al., 2020), and Yelp Reviews (informal user-generated content) (Zhang et al., 2015). For each sample, we randomly mask a contiguous span of 2 to 8 words. Experiments are conducted using LLaDA-8B-Base. We employ greedy sampling and report BLEU-2 and ROUGE-L as our main metrics. Extended metrics on ROCStories are provided in Appendix F. For detailed hyperparameters and implementation specifics, please refer to Appendix G.

Decoding Budget. Unless stated otherwise, the LLaDA code sampler uses one denoising step per selected canvas token. Text infilling instead fixes the configured decoding steps to L_{init}. Search forwards are additional and are reported separately below.

5.2 Code Infilling Results

Table 2 reports code infilling results. Following Appendix C, we adopt a double-exponential fit for Length Bias. On Single-Line Infilling with LLaDA-Base, we compare CAL with linear, single-exponential (1exp), and single-exponential with offset (1exp+off) variants. The double-exponential fit achieves the highest average Pass@1, consistent with Appendix C.

Consistent Improvement over Baselines. Our adaptive length discovery method consistently achieves superior performance across all models and initial length configurations compared to fixed-length baselines. For instance, on the Single-Line task, our approach improves the average Pass@1 of LLaDA-Base from 44.8% to 65.4%. As indicated by the *Stps.* column, achieving this substantial performance gain requires only 11 to 18 additional first-step forward passes on average. This overhead is acceptable given the substantial capability improvements.

Outperforming Local-Confidence Methods. Notably, our method significantly outperforms DAEDAL, a confidence-based adaptive strategy originally designed for DLM chat tasks. While effective for open-ended generation, DAEDAL's strategy of locally inserting masks based on token-level confidence proves suboptimal for infilling. As shown in Table 2, it occasionally underperforms even the fixed-length baseline, particularly when the initial length is overestimated (e.g., $L = 16$ or 32). This performance gap highlights two key advantages of our method. First, our *bidirectional search* allows for both increasing and decreasing the span length, whereas DAEDAL is primarily limited to extending it. Second, relying on *global average confidence* proves more effective than local heuristics. Since DLMs model the joint probability of all tokens holistically, the average confidence over the entire span serves as a more reliable indicator of semantic completeness.

Table 2: **Code Infilling results.** We compare the Pass@1 scores of various models under four fixed-length settings ($L = 4, 8, 16, 32$) against our proposed adaptive length discovery algorithm (+ ours). Stps. denotes the average number of search steps incurred during the length discovery phase.

Model	Length = 4		Length = 8		Length = 16		Length = 32		Avg.	
	Pass@1	Stps.	Pass@1	Stps.	Pass@1	Stps.	Pass@1	Stps.	Pass@1	Stps.
Single-Line Infilling										
LLaDA-Base	22.7	–	51.0	–	56.6	–	49.0	–	44.8	–
+ daedal	29.9	–	52.8	–	53.4	–	43.1	–	44.8	–
+ cal-linear	59.5	10.5	68.1	12.0	63.5	15.2	55.7	18.4	61.7	14.0
+ cal-1exp	66.1	11.1	70.5	12.0	64.1	15.1	51.3	18.0	63	14.1
+ cal-1exp+off	**71.1**	11.9	72.8	12.5	63.8	14.8	51.8	15.4	64.8	13.7
+ cal	70.4	11.6	**73.5**	12.3	**65.2**	14.8	**52.6**	15.7	**65.4**	13.6
LLaDA-Instruct	23.0	–	54.9	–	64.0	–	57.0	–	49.7	–
+ cal	**74.6**	12.2	**76.9**	13.0	**69.4**	14.3	**58.7**	15.3	**69.9**	13.7
DiffuCoder	24.5	–	51.5	–	60.1	–	47.7	–	46.0	–
+ cal	**73.1**	11.1	**74.8**	12.4	**68.4**	14.6	**55.8**	18.2	**68.0**	14.1
DreamCoder	24.7	–	55.4	–	63.1	–	53.9	–	49.3	–
+ cal	**75.1**	11.1	**76.2**	12.4	**70.5**	14.6	**59.0**	18.1	**70.2**	14.1
Multi-Line Infilling										
LLaDA-Base	4.6	–	10.8	–	18.7	–	27.6	–	15.4	–
+ daedal	6.0	–	12.3	–	20.4	–	26.5	–	16.3	–
+ cal	**17.1**	12.0	**21.2**	13.3	**27.1**	14.7	**31.7**	15.3	**24.3**	13.8
LLaDA-Instruct	4.7	–	11.7	–	21.4	–	30.4	–	17.0	–
+ cal	**18.8**	12.0	**22.8**	13.1	**30.2**	13.8	**34.0**	14.8	**26.5**	13.4
DiffuCoder	5.0	–	11.7	–	21.6	–	30.8	–	17.3	–
+ cal	**19.4**	12.1	**24.5**	13.4	**32.0**	14.3	**38.0**	16.6	**28.5**	14.1
DreamCoder	5.1	–	12.4	–	22.9	–	33.1	–	18.4	–
+ cal	**19.9**	12.1	**24.4**	13.3	**31.9**	14.5	**37.3**	16.9	**28.4**	14.2

Performance Trends and Robustness. We observe that the performance gain diminishes as the initial length increases. We attribute this to two factors: (1) *Signal Decay.* The HumanEval Single-Line split contains short completions (average Oracle length ≈ 9.5, see Table 1). Consequently, a large initial length (e.g., $L = 32$) implies a significant deviation from the ground truth, causing the Oracle Peak signal to decay and become obscured by background noise (as evidenced by confidence fluctuations when $|L - L^*| > 10$ in Figure 2a). (2) *Increased Prediction Difficulty.* As the masked span expands, the first-step reconstruction task becomes inherently more challenging, reducing the model's capacity to produce a sharp confidence peak. Yet, our method consistently outperforms fixed-length baselines even in these difficult scenarios, proving its robustness. Stratified analysis by Oracle length and direct length estimation error are reported in Appendix E.

Stronger Fixed-Length Baselines. The Multi-Line reference length averages about 46.8 tokens, beyond the original fixed grid. Extending this grid to 48 and 64 tokens, CAL still improves Pass@1 by 1.54–3.20 points on LLaDA-Base and Instruct, although the advantage narrows (Appendix E.3).

Long-Span Infilling. Extending the stratified analysis in Appendix E, we evaluate 108 spans with reference lengths of 160–240 tokens. Fixed-200 achieves 20.37%, CAL-200 40.74%, and reference-length decoding 76.85%. CAL adds about 7% forwards and time. These results support long-span infilling near an appropriate initialization, not arbitrary expansion from a short canvas (Appendix E.4).

Search versus Additional Decoding. At $L_{init} = 32$, we compare spending about 16 additional forwards on either more decoding or CAL search. CAL exceeds Fixed-full by 3.00/2.83 Single-/Multi-Line Pass@1 points at similar average forward counts and wall time. This approximately matches the average budget, not compute for each example; full settings and results are in Appendix D.1.

5.3 Text Infilling Results

Table 3 extends our evaluation to general text domains, including short stories, scientific abstracts, and reviews. For additional experimental results including more metrics, please refer to Appendix F.

Generalization Across Domains. Our method improves average BLEU-2 and ROUGE-L over fixed-length baselines on all three datasets. For instance, on the Stories dataset, we achieve a relative improvement of 30.0% in BLEU-2 (from 15.0 to 19.5) compared to the average baseline performance. These results indicate that the discovered length signal is not limited to code syntax but also provides a useful signal in the tested natural language domains.

Domain-Specific Performance Variations. We observe that the performance gains in text infilling are more modest compared to code infilling. We attribute this discrepancy to the inherent differences between the two domains. Code is governed by strict syntax and rigid logical constraints (e.g., variable definitions), which significantly narrow the search space for the ground truth. This constraint allows the model to produce a sharp and distinct Oracle Peak. In contrast, natural language is inherently ambiguous and open-ended; the rich semantic space allows for multiple valid completions of varying lengths. This ambiguity leads to a more diffuse probability distribution, resulting in a less prominent Oracle Peak signal compared to the code domain.

Table 3: **Text Infilling results.** Comparisons of BLEU-2 and ROUGE-L scores on three text datasets using LLaDA-8B-Base. The results are averaged over all test samples. Fixed-length baselines are reported for preset lengths of 2, 4, and 8 tokens.

Dataset	Model	Length = 2		Length = 4		Length = 8		Avg.	
		B-2	R-L	B-2	R-L	B-2	R-L	B-2	R-L
Stories	LLaDA-Base	9.1	25.8	18.0	37.9	17.8	37.6	15.0	33.8
	+ CAL	**17.6**	**35.7**	19.8	38.4	**21.2**	**40.7**	**19.5**	**38.3**
Abstract	LLaDA-Base	8.7	26.8	17.8	38.7	19.3	39.5	15.3	35.0
	+ CAL	**17.1**	**36.5**	**19.6**	**39.6**	**22.1**	**42.4**	**19.6**	**39.5**
Yelp	LLaDA-Base	5.9	20.1	11.4	**30.5**	12.5	30.3	9.9	27.0
	+ CAL	**10.4**	**27.8**	**11.9**	30.2	**13.6**	**32.5**	**12.0**	**30.2**

5.4 Ablation Study

Necessity of Length Bias Calibration. To validate the critical role of bias calibration, we perform comprehensive comparisons across different initial lengths, models, and data domains (see Table 4). Specifically, we utilize the HumanEval-Infilling Single-Line task for code infilling and the ROCStories dataset for text infilling. The results indicate that while the uncalibrated CAL method (w/o LB) generally outperforms the fixed-length baseline, applying Length Bias calibration yields significantly more pronounced improvements. A comparison of alternative functional forms for $B(L)$ using AIC/BIC criteria is provided in Appendix C.

Impact at short lengths. Calibration is particularly vital for short sequences. As shown in panels (a) and (c) of Table 4, the performance gain is substantially larger when the initial length is small. This trend holds for both code and text tasks. The reason lies in the nature of Length Bias: as shown in Figure 2b, the bias curve decays most steeply at shorter lengths. Consequently, without calibration, raw confidence scores are artificially high for extremely short sequences. This often misleads the search into selecting truncated solutions. Our calibration effectively corrects this skew, allowing the model to recover the true Oracle signal.

Effective Across Models and Domains. We fit the calibration function $B(L)$ using only LLaDA-Base on a subset of HumanEval-Infilling data. Yet, this same function demonstrates excellent performance on other models (DiffuCoder, DreamCoder) and text domains. This success across models and domains suggests that Length Bias is a broadly shared statistical regularity across the DLMs we tested, rather than a model-specific artifact. The same curve transfers without re-fitting across the four tested checkpoints, which use two tokenizer families, and the three text domains. Appendix F.2 fits a separate text-domain curve from scratch for comparison; it does not update the code-fitted curve.

Calibration Set Size. On Single-Line, LLaDA-Base, $L_{init} = 8$, three calibration examples give 73.53 ± 0.11 Pass@1 across three subset seeds, versus 73.31 with 100 examples (Appendix C.3). This stability is specific to the tested setting.

Table 4: **Ablation Study of Length Bias (LB) Calibration.** We demonstrate the necessity and robustness of LB calibration across three dimensions: (a) varying initial lengths on code infilling using LLaDA-Base; (b) different model on code infilling ($L_{init} = 8$), where LLa., Dif., and Dre. correspond to LLaDA-Instruct, DiffuCoder-Base, and DreamCoder-Base, respectively; and (c) varying initial lengths on text infilling.

METHOD	(A) LENGTH (CODE)			(B) MODELS (CODE)			(C) LENGTH (TEXT)		
	PASS@1			PASS@1			BLEU-2 / ROUGE-L		
	$L = 4$	$L = 8$	$L = 16$	LLA.	DIF.	DRE.	$L = 2$	$L = 4$	$L = 8$
FIXED LENGTH	22.7	51.0	56.6	54.9	51.5	55.4	9.1 / 25.8	18.0 / 37.9	17.8 / 37.6
CAL *w/o LB*	35.8	59.6	60.8	66.5	69.1	67.1	14.5 / 32.1	16.9 / 35.2	19.5 / 38.4
CAL *w/ LB*	**70.4**	**73.5**	**65.2**	**76.9**	**74.8**	**76.2**	**17.6 / 35.7**	**19.8 / 38.4**	**21.2 / 40.7**
LB Improvement	*+34.6*	*+13.9*	*+4.4*	*+10.4*	*+5.7*	*+9.1*	*+3.1 / +3.6*	*+2.9 / +3.2*	*+1.7 / +2.3*

Confidence Choice. Replacing Top-1 confidence with the probability margin or normalized entropy certainty gives average Pass@1 of 65.06 and 66.59, versus 65.43 for Top-1 across four initial lengths (Appendix H.3). Entropy is not uniformly better, so Top-1 remains the default.

5.5 Image-Conditioned Infilling

We test LaViDa (Li et al., 2026) and LLaDA-V (You et al., 2026) on MS COCO captions (Lin et al., 2014), conditioning on the image and caption prefix/suffix. We remove 2–8 consecutive words, retaining both sides, fit each model's bias on 50 examples, and evaluate 500 image-disjoint examples at $L_{init} \in \{2, 4, 8\}$. CAL improves average BLEU-2/ROUGE-L from 17.3/37.9 to 20.4/41.5 on LaViDa and from 18.9/39.9 to 22.0/43.2 on LLaDA-V. This supports average quality gains after model-specific calibration, not zero-shot transfer of the code bias or a guaranteed reference-length peak. Settings and per-length results appear in Appendix F.4.

6 Related Work

Diffusion Language Models. Discrete diffusion models have recently emerged as a compelling alternative to auto-regressive models, enabling probabilistic modeling of token sequences through iterative denoising. Foundational works (Austin et al., 2021; Hoogeboom et al., 2021) introduced discrete corruption processes, which were further refined by improved training objectives (Zheng et al., 2023; Lou et al., 2023). Within this domain, Masked Diffusion Models (MDMs) have become a dominant paradigm, utilizing masking kernels to achieve scalable training (Sahoo et al., 2024; Shi et al., 2024). Recent efforts have pushed these models to the billion-parameter frontier: LLaDA (Nie et al., 2025) pioneered large-scale training from scratch, while its successor LLaDA-1.5 (Zhu et al., 2025) has integrated reinforcement learning for alignment, and LLaDA-V (You et al., 2026) has extended to multimodal understanding. Parallel research, such as DiffuLLaMA (Gong et al., 2024) and Dream (Ye et al., 2025), explored adapting pre-trained AR models into diffusion frameworks. Furthermore, specialized models like DiffuCoder (Gong et al., 2025) and DreamCoder (Xie et al., 2025) have been developed specifically for code generation. While leveraging the bidirectional nature of diffusion, most prior work has focused primarily on standard instruction-following capabilities.

Generative Infilling. Early approaches to generative infilling predominantly relied on encoder-decoder architectures, such as BART (Lewis et al., 2020), T5 (Raffel et al., 2020), and Insertion Transformers (Stern et al., 2019), which treat infilling as a span corruption and reconstruction task. With the dominance of decoder-only models, research shifted toward adapting AR architectures for infilling. Techniques like Fill-In-the-Middle (FIM) (Bavarian et al., 2022; Fried et al., 2022) and blank-infilling objectives (Donahue et al., 2020; Du et al., 2022) were introduced to enable bidirectional context awareness within a causal generation framework. More recently, the rise of diffusion language models (Nie et al., 2025; Gong et al., 2024; Ye et al., 2025) has offered a new paradigm. Thanks to their non-causal nature, these models naturally model the joint distribution of the prefix, middle, and suffix. While standard DLMs typically require fixed lengths, specialized variants such as DDOT (Zhang et al., 2025), FlexMDM (Kim et al., 2025b), and others (Wu et al., 2026; Xiong et al., 2025) have been proposed to enable variable-length generation capabilities.

Adaptive Length Control in DLMs. Standard diffusion language models typically handle length variations in instruction-following tasks by initializing an oversized canvas and filling unused positions with <eos> or padding tokens (Nie et al., 2025; Gong et al., 2024; Ye et al., 2025). While recent works like Rainbow Padding (Kim et al., 2025a) have optimized this padding mechanism, these strategies are primarily tailored for chat-style interactions rather than infilling. To enable flexible-length generation for infilling, several training-based approaches have been proposed. For instance, Zhang et al. (2025) achieved flexible-length infilling via optimal transport, while Havasi et al. (2025) and Kim et al. (2025b) abandoned the fixed-length mask-decode paradigm in favor of operations like insertion and deletion. Similarly, Wu et al. (2026) introduced special tokens [expand] and [delete] to dynamically adjust sequence length. These methods involve model training, whereas CAL selects the canvas length without updating model parameters.

DreamOn reports about 5 hours on 8 H800 GPUs for code fine-tuning (roughly 40 GPU-hours), using about 110K Python pairs for 10 epochs; its text experiment separately fine-tunes on ROCStories. This is an amortizable training cost. CAL instead incurs per-sample search costs and reuses its code-fitted curve across three text domains. We compare these qualitatively, not through a cross-hardware break-even estimate.

In contrast, training-free approaches such as DAEDAL (Li et al., 2025) attempted to dynamically adjust length at inference time in chat tasks by monitoring local token confidence. Similarly, Xiong et al. (2025) prompted DLMs to generate <null> tokens as placeholders to regulate length in controllable JSON generation. However, these methods rely on explicit termination signals or rigid structural constraints, which are often absent in general infilling scenarios. Our work bridges this gap by proposing a training-free framework that leverages the global first-step confidence of the entire masked region. By calibrating the systematic Length Bias, we select a suitable infilling length without relying on additional training.

7 Conclusion

In this paper, we address a critical limitation of Diffusion Language Models (DLMs) in infilling tasks: their sensitivity to pre-specified mask lengths. Our analysis reveals that DLMs naturally encode a length signal within their first-step denoising confidence. We characterize this through two key phenomena: the *Oracle Peak*, which signals semantic completeness, and *Length Bias*, a systematic confidence decay that must be calibrated. Leveraging these insights, we propose CAL (Calibrated Adaptive Length), a training-free framework for adaptive length discovery. By calibrating the bias and employing an efficient bidirectional search, CAL selects a suitable infilling length prior to formal decoding. Extensive experiments on code and text benchmarks demonstrate that our approach consistently outperforms fixed-length baselines across diverse models and domains.

Limitations. CAL has three principal limitations. First, search adds approximately 1 second per sample in the original A40 measurements; this observed cost is not a constant bound across span lengths or deployment settings. Second, static calibration transfers across the tested code/text settings, but arbitrary OOD transfer and online fitting remain untested. Online fitting would require a specified feedback or unsupervised estimation protocol. Third, search depends on initialization and range: CAL can refine a canvas near 200 tokens, but the original $L_{init} = 32, L_{max} = 128$ setting cannot bridge that gap (Appendix E.4). For multiple spans, a possible extension searches a length vector by initializing each span separately and then updating one span at a time given the other canvases. This extension remains untested.

Acknowledgments and Disclosure of Funding

We thank Zeyan Li for helpful discussions and feedback on this work. This work was supported in part by the National Natural Science Foundation of China No. 62376277, Beijing Natural Science Foundation No. Z260013, Public Computing Cloud, Renmin University of China, and the Fund for Building World-Class Universities (Disciplines) of Renmin University of China.

References

Austin, J., Johnson, D. D., Ho, J., Tarlow, D., and Van Den Berg, R. Structured denoising diffusion models in discrete state-spaces. *Advances in neural information processing systems*, 34:17981–17993, 2021.

Bavarian, M., Jun, H., Tezak, N., Schulman, J., McLeavey, C., Tworek, J., and Chen, M. Efficient training of language models to fill in the middle. *arXiv preprint arXiv:2207.14255*, 2022.

Donahue, C., Lee, M., and Liang, P. Enabling language models to fill in the blanks. *arXiv preprint arXiv:2005.05339*, 2020.

Du, Z., Qian, Y., Liu, X., Ding, M., Qiu, J., Yang, Z., and Tang, J. Glm: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 320–335, 2022.

Fried, D., Aghajanyan, A., Lin, J., Wang, S., Wallace, E., Shi, F., Zhong, R., Yih, W.-t., Zettlemoyer, L., and Lewis, M. Incoder: A generative model for code infilling and synthesis. *arXiv preprint arXiv:2204.05999*, 2022.

Gong, S., Agarwal, S., Zhang, Y., Ye, J., Zheng, L., Li, M., An, C., Zhao, P., Bi, W., Han, J., et al. Scaling diffusion language models via adaptation from autoregressive models. *arXiv preprint arXiv:2410.17891*, 2024.

Gong, S., Zhang, R., Zheng, H., Gu, J., Jaitly, N., Kong, L., and Zhang, Y. Diffucoder: Understanding and improving masked diffusion models for code generation. *arXiv preprint arXiv:2506.20639*, 2025.

Havasi, M., Karrer, B., Gat, I., and Chen, R. T. Edit flows: Flow matching with edit operations. *arXiv preprint arXiv:2506.09018*, 2025.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Hoogeboom, E., Nielsen, D., Jaini, P., Forré, P., and Welling, M. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in neural information processing systems*, 34:12454–12465, 2021.

Kim, B., Jeon, D., Kim, D., Jeung, W., and No, A. Rainbow padding: Mitigating early termination in instruction-tuned diffusion llms. *arXiv preprint arXiv:2510.03680*, 2025a.

Kim, J., Cheuk-Kit, L., Domingo-Enrich, C., Du, Y., Kakade, S., Ngotiaoco, T., Chen, S., and Albergo, M. Any-order flexible length masked diffusion. *arXiv preprint arXiv:2509.01025*, 2025b.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 7871–7880, 2020.

Li, J., Dong, X., Zang, Y., Cao, Y., Wang, J., and Lin, D. Beyond fixed: Training-free variable-length denoising for diffusion large language models. *arXiv preprint arXiv:2508.00819*, 2025.

Li, S., Kallidromitis, K., Bansal, H., Gokul, A., Kato, Y., Kozuka, K., Kuen, J., Lin, Z., Chang, K.-W., and Grover, A. LaViDa: A large diffusion language model for multimodal understanding. *Advances in Neural Information Processing Systems*, 38:105101–105134, 2026.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision*, pp. 740–755. Springer, 2014.

Lou, A., Meng, C., and Ermon, S. Discrete diffusion modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*, 2023.

Mostafazadeh, N., Chambers, N., He, X., Parikh, D., Batra, D., Vanderwende, L., Kohli, P., and Allen, J. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 839–849, 2016.

Nie, S., Zhu, F., You, Z., Zhang, X., Ou, J., Hu, J., ZHOU, J., Lin, Y., Wen, J.-R., and Li, C. Large language diffusion models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=KnqiC0znVF.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Sahoo, S., Arriola, M., Schiff, Y., Gokaslan, A., Marroquin, E., Chiu, J., Rush, A., and Kuleshov, V. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184, 2024.

Shi, J., Han, K., Wang, Z., Doucet, A., and Titsias, M. Simplified and generalized masked diffusion for discrete data. *Advances in neural information processing systems*, 37:103131–103167, 2024.

Stern, M., Chan, W., Kiros, J., and Uszkoreit, J. Insertion transformer: Flexible sequence generation via insertion operations. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5976–5985. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/stern19a.html.

Wu, Z., Zheng, L., Xie, Z., Ye, J., Gao, J., Gong, S., Feng, Y., Li, Z., Bi, W., Zhou, G., and Kong, L. DreamOn: Diffusion language models for code infilling beyond fixed-size canvas. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=EQTPmqukiU.

Xie, Z., Ye, J., Zheng, L., Gao, J., Dong, J., Wu, Z., Zhao, X., Gong, S., Jiang, X., Li, Z., et al. Dream-coder 7b: An open diffusion language model for code. *arXiv preprint arXiv:2509.01142*, 2025.

Xiong, Z., Cai, Y., Li, Z., and Wang, Y. Unveiling the potential of diffusion large language model in controllable generation. *arXiv preprint arXiv:2507.04504*, 2025.

Ye, J., Xie, Z., Zheng, L., Gao, J., Wu, Z., Jiang, X., Li, Z., and Kong, L. Dream 7b: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025.

You, Z., Nie, S., Zhang, X., Zhou, J., Lu, Z., Wen, J.-R., and Li, C. LLaDA-V: Large language diffusion models with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10093–10105, 2026.

Zhang, A., Sivakumar, A., Tang, C.-W., and Thomas, C. Flexible-length text infilling for discrete diffusion models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 31332–31347, 2025.

Zhang, X., Zhao, J., and LeCun, Y. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

Zheng, L., Yuan, J., Yu, L., and Kong, L. A reparameterized discrete diffusion model for text generation. *arXiv preprint arXiv:2302.05737*, 2023.

Zhou, W., Jiang, Y. E., Wilcox, E., Cotterell, R., and Sachan, M. Controlled text generation with natural language instructions. In *International Conference on Machine Learning*, pp. 42602–42613. PMLR, 2023.

Zhu, F., Wang, R., Nie, S., Zhang, X., Wu, C., Hu, J., Zhou, J., Chen, J., Lin, Y., Wen, J.-R., et al. Llada 1.5: Variance-reduced preference optimization for large language diffusion models. *arXiv preprint arXiv:2505.19223*, 2025.

Zhu, W., Hu, Z., and Xing, E. Text infilling. *arXiv preprint arXiv:1901.00158*, 2019.

A Method Algorithm

Algorithm 1 Adaptive Length Discovery

Input: Context P, S, Model $\mathcal{M}$, Bias $B(L)$
Hyperparams: Step ΔL, Tolerance D, Init L_{init}, Max L_{max}
$\hat{L} \leftarrow L_{init}, \quad \hat{\Phi}_c \leftarrow \Phi(L_{init})/B(L_{init})$
for $dir \in \{+1, -1\}$ **do**
  $L \leftarrow L_{init} + dir \cdot \Delta L, \quad cnt \leftarrow 0$
  while $L \in [1, L_{max}] \wedge cnt < D$ **do**
    $\Phi_c \leftarrow \Phi(L)/B(L)$
    if $\Phi_c > \hat{\Phi}_c$ **then**
      $\hat{L}, \hat{\Phi}_c, cnt \leftarrow L, \Phi_c, 0$ {Update best length}
    else
      $cnt \leftarrow cnt + 1$
    end if
    $L \leftarrow L + dir \cdot \Delta L$
  end while
end for
return Denoise$(\mathcal{M}, P, S, \hat{L})$

B Intuition For the Length Bias Form

We outline a two-scale view that motivates the qualitative structure of the empirical fit $B(L) = a\,e^{-bL} + c\,e^{-dL} + \beta$ used in the main paper. We rename the offset parameter to β in this section to avoid notational collision with the natural exponential. We emphasize at the outset that this discussion is heuristic, not a derivation: the specific exponential form is selected empirically based on goodness-of-fit (Appendix C.1), and the role of the discussion below is only to convey why a two-scale plus offset ansatz is natural.

Setup. At the first denoising step, the model receives a fully masked input $[P;\, [\texttt{MASK}]^L;\, S]$. The quantity we model is the average first-step confidence $\Phi(L)$, computed over the L masked positions. Throughout this section we focus on the regime $|L - L^*| > 4$, since $B(L)$ is fit only on those samples (Appendix C); semantic peaks near the oracle length are excluded from the fit by construction.

Two scales of contextual constraint. We posit that contextual information reaches a masked position through (at least) two qualitatively different mechanisms: **Local structural constraints** (syntax, adjacent-token dependencies). These operate on a short range ξ_{syn}; positions near the two span boundaries are strongly constrained, while interior positions are not. Boundary effects therefore dominate only for short spans and weaken quickly as L grows. **Global semantic constraints** (topic, intent, type signature). These operate on a longer range $\xi_{\text{sem}} \gg \xi_{\text{syn}}$ and reach interior positions more uniformly. Their per-position effective strength nonetheless weakens with L, since a fixed bidirectional context must be "shared" across an increasing number of masked positions.

We do not derive a closed form for either contribution. The qualitative claim we rely on is that, holding (P, S) fixed, the per-position contribution from each mechanism is monotone-decreasing in L on the relevant range, with the local mechanism decaying much faster than the global one.

Why an exponential ansatz? A sum of two exponentials $a\,e^{-bL} + c\,e^{-dL}$ is a flexible parametric family that (i) admits two distinct decay rates, matching the two-scale picture above; (ii) is monotone-decreasing on $L \geq 0$; and (iii) interpolates smoothly between a sharp early drop and a long flat tail, matching the empirical curve. A direct comparison against alternative forms (single exponential, linear, single exponential without offset) is reported in Appendix C.1: the double-exponential dominates on every fit-quality criterion (R^2, AIC, BIC) and on downstream Pass@1. We therefore adopt it as an empirical ansatz consistent with the two-scale view, not as a derived form.

Prior baseline. As L grows, contextual constraints diminish, but the model retains pre-training token-frequency priors. High-frequency tokens (e.g., code keywords `return`, `if`) are predicted with systematically elevated confidence regardless of context, contributing a non-vanishing asymptote $\lim_{L\to\infty} \Phi(L) = \beta > 0$. This motivates the additive offset in $B(L)$.

Combining the three components.

$$B(L) \;=\; \underbrace{a\, e^{-bL}}_{\text{fast decay}} + \underbrace{c\, e^{-dL}}_{\text{slow decay}} + \underbrace{\beta}_{\text{prior offset}}, \qquad b \gg d. \tag{5}$$

Why ratio rather than difference? A natural question is why we calibrate via the ratio $\Phi_c(L) = \Phi(L)/B(L)$ rather than the difference $\Phi(L) - B(L)$. Both quantities exhibit peaks near the oracle length on our data, but we adopt the ratio for two practical reasons. (i) $B(L)$ varies by more than an order of magnitude on the range of interest; treating the calibrated score as a signal-to-noise ratio renders it dimensionless and stable across that range. (ii) The relative excess captured by the ratio is more numerically discriminative than the absolute residual captured by the difference, particularly when $B(L)$ becomes small at large L. We therefore adopt the ratio as a practical, scale-invariant measure rather than as a theoretically privileged choice.

Cross-domain check. The two-scale ansatz makes two qualitative predictions: (i) in natural language, where rigid syntactic structure is weaker than in code, the fast-decay amplitude a should be smaller; (ii) in vocabulary-diverse domains, where no single token dominates the prior, the offset β should be smaller. We fit $B(L)$ in-domain on 1,000 held-out ROCStories samples (Appendix C) and report the comparison in Table 5. The text-domain fits drive a and β to near zero, qualitatively consistent with both predictions, while the slow-decay rate d is comparable across domains (0.06 versus 0.07).

We caveat this observation in two respects. First, the 95% confidence interval for a in the code-domain fit is wide (Table 7), reflecting the well-known identifiability limitations of multi-exponential models; the cross-domain reduction in a should therefore be read as qualitatively consistent with our intuition rather than as a statistically tight test. Second, the more robust empirical observation is the stability of c and d across domains, together with the fact that Φ_c yields well-localized Oracle Peaks in both code and text experiments (Tables 2, 3).

Table 5: **Cross-domain comparison of fitted $B(L)$ parameters.** Code parameters are fitted on 100 HumanEval-Infilling samples; text parameters on 1,000 held-out ROCStories samples. The reduction of a and β in the text domain, together with the stability of d across domains, is qualitatively consistent with the two-scale picture; see the discussion above for caveats regarding the wide confidence interval of a.

PARAM.	CODE	TEXT	INTERPRETATION
a	1.00	≈ 0	FAST-DECAY AMPLITUDE; REDUCED WHEN SYNTAX IS FLEXIBLE
b	1.77	—	FAST-DECAY RATE ($\xi_{\text{syn}} \approx 0.6$ TOK.)
c	0.56	0.48	SLOW-DECAY AMPLITUDE; COMPARABLE ACROSS DOMAINS
d	0.06	0.07	SLOW-DECAY RATE ($\xi_{\text{sem}} \approx 14$–$17$ TOK.)
β	0.24	≈ 0	PRIOR OFFSET; HIGH FOR CODE (CONCENTRATED KEYWORDS)

C Length Bias Fitting Details

Data Sampling and Oracle Exclusion Strategy. We randomly sampled 100 single-line infilling tasks with varying ground-truth lengths from the HumanEval-Infilling benchmark (Bavarian et al., 2022). These infill task IDs were excluded from the main test split; different spans from the same underlying function may still occur across splits. For each task i, we fixed the prefix P_i and suffix S_i, probing the first-step denoising confidence across a wide range of lengths $\mathcal{L}_{test} =$

$\{1, 2, 4, 6, 12, 16, 24, 32, 48, 64, 96, 128\}$. To decouple the systematic Length Bias from semantic signals, we implemented a strict *Oracle exclusion strategy*: any probe length L falling within the neighborhood of the Oracle length L_i^* (specifically, $L \in [L_i^* - 4, L_i^* + 4]$) was discarded. This is intended to separate the background confidence trend from reference-aligned peaks.

Double-Exponential Fitting. Based on the characteristic trend of *rapid initial decay followed by stabilization* (visualized in Figure 2b), we adopted a double-exponential model $B(L) = ae^{-bL} + ce^{-dL} + \beta$. The first exponential term accounts for the sharp dilution of local constraints in short sequences, while the second term models the long-tail asymptotic behavior. We solved for the parameters using non-linear least squares, weighting each length point L by $w_L = 1/\sqrt{N_L}$ to account for sample size variations.

Fitted Parameters and Robustness. For the LLaDA-8B-Base model (Nie et al., 2025), the parameters used in the main experiments were: $a = 1.00, b = 1.77$ (fast decay), $c = 0.56, d = 0.06$ (slow decay), and $\beta = 0.24$ (baseline). Although fitting on different subsets may yield minor parameter fluctuations, our ablation studies (see Table 4) demonstrate that the derived bias function is robust in the tested settings. It generalizes across the tested models (DiffuCoder (Gong et al., 2025), DreamCoder (Xie et al., 2025)) and text domains (Mostafazadeh et al., 2016; Donahue et al., 2020) without re-fitting. The score alternatives and multimodal experiments below use separately fitted curves. We provide the core fitting implementation with the released code.

C.1 Functional Form Comparison

To verify that the double-exponential is the appropriate choice, we compare four functional forms for $B(L)$ both in terms of fit quality and downstream Pass@1 on HumanEval-Infilling (Single-Line) using LLaDA-8B-Base. Fit quality is assessed via R^2, the Akaike Information Criterion (AIC), and the Bayesian Information Criterion (BIC); both AIC and BIC penalize additional parameters, with $\Delta > 10$ constituting strong evidence against an alternative model.

Table 6: **Fit quality comparison of alternative $B(L)$ forms.** ΔAIC and ΔBIC are reported relative to the double-exponential (lower is better; 0 is best). A $\Delta > 10$ indicates strong evidence against the alternative.

FORM	FITTED CURVE	R^2	ΔAIC	ΔBIC
DOUBLE-EXP	$1.000\,e^{-1.765L} + 0.564\,e^{-0.056L} + 0.239$	**0.9955**	0	0
SINGLE-EXP	$0.668\,e^{-0.070L} + 0.248$	0.9684	+19.5	+18.5
SINGLE-EXP (NO OFFSET)	$0.770\,e^{-0.020L}$	0.8005	+39.6	+38.2
LINEAR	$0.612 - 0.004L$	0.5816	+48.5	+47.1

The double-exponential achieves the best fit ($R^2 = 0.9955$) and the highest average Pass@1 (65.4%), and is unambiguously preferred by both AIC and BIC (ΔAIC > 10 over all alternatives). Removing the offset β sharply degrades R^2 (0.97 $\rightarrow$ 0.80), confirming that the pre-training prior baseline is a necessary component. The second exponential term provides a consistent further improvement over single-exp (Pass@1: 64.8% $\rightarrow$ 65.4%), reflecting the two-scale structure motivated in Appendix B.

C.2 Parameter Confidence Intervals and Cross-Domain Comparison

Table 7 reports 95% confidence intervals for each double-exponential parameter, obtained via non-linear least squares on the 100 HumanEval calibration samples. The wide CIs for a and b reflect a known identifiability property of multi-exponential models (the two parameters are strongly anti-correlated), not instability of the fitted curve; the downstream-relevant parameters c, d, and β are all tightly estimated.

The cross-domain parameter comparison (Table 5) is consistent with the heuristic picture in Appendix B: in-domain fitting for text yields an approximately single-exponential curve $B_{\text{text}}(L) \approx 0.48 \cdot e^{-0.07L}$, and the slow-decay rate d is comparable across domains (0.056 vs. 0.070). As discussed in Appendix B, the wide CI for a means this should be read as qualitatively supportive rather than as a tight statistical test.

Table 7: **Fitted parameters and 95% confidence intervals** for the double-exponential $B(L)$ on HumanEval-Infilling (LLaDA-8B-Base). Wide CIs for a, b reflect model identifiability, not curve instability.

PARAMETER	ESTIMATE	95% CI
a	1.000	$\pm$1.766
b	1.766	$\pm$1.976
c	0.564	$\pm$0.063
d	0.056	$\pm$0.010
β	0.239	$\pm$0.019

C.3 Calibration Set Size

We fit $B(L)$ on subsets of the 100 calibration examples and evaluate on the same 933 Single-Line test examples using LLaDA-Base, $L_{init} = 8$, $D = 4$, and $\Delta L = 1$. For $n < 100$, we use three nested-subset seeds (40, 41, 42). Curve RMSE is measured against the $n = 100$ fit over integer lengths 1–64. Table 8 reports mean and standard deviation across calibration subsets, not decoding seeds.

Table 8: **Calibration-size sensitivity.** Pass@1 is in percent. The $n = 100$ curve is fitted once.

Calibration n	Subsets	Curve RMSE	Pass@1
3	3	0.0386 ± 0.0079	73.53 ± 0.11
5	3	0.0268 ± 0.0084	73.35 ± 0.27
10	3	0.0307 ± 0.0060	73.42 ± 0.29
25	3	0.0221 ± 0.0064	73.17 ± 0.62
50	3	0.0150 ± 0.0054	73.42 ± 0.00
100	1	0	73.31

Pass@1 remains stable down to $n = 3$ in this setting, although curve RMSE increases. This does not establish that three examples suffice in other domains. These runs use full-precision fitted parameters; the main Top-1 result (73.53 at $L_{init} = 8$) uses the rounded default parameters listed above.

D Computational Overhead Analysis

Table 9 reports per-sample wall-clock time (NVIDIA A40 GPU, LLaDA-8B-Base) decomposed into the length search phase (*Search*) and the formal denoising phase (*Decode*).

Table 9: **Wall-clock latency per sample (seconds)** on an NVIDIA A40 GPU using LLaDA-8B-Base. *Search* is the length probing overhead; *Decode* is the standard denoising time; *Total* is their sum, excluding model loading, tokenization, and evaluation.

TASK	L_{init}	METHOD	SEARCH (S)	DECODE (S)	TOTAL (S)	PASS@1
SINGLE	4	FIXED	0.00	0.28	0.28	22.7%
		CAL	0.82	0.53	1.35	70.4%
	8	FIXED	0.00	0.56	0.56	51.0%
		CAL	0.86	0.61	1.47	73.5%
MULTI	8	FIXED	0.00	0.57	0.57	10.8%
		CAL	0.96	0.75	1.72	21.2%
	32	FIXED	0.00	2.44	2.44	27.6%
		CAL	1.15	2.43	3.58	31.7%

The search overhead ranges from 0.82 to 1.15s in these configurations. The tolerance D controls stopping after consecutive probes fail to improve the best score; it does not bound the total number of

probes. For Multi-Line at $L_{init} = 32$, search accounts for 32% of search-plus-decode time. CAL adds ≈1s for a +47.7 pp gain in the Single-Line $L_{init} = 4$ example (22.7% → 70.4%). These results describe a quality–latency trade-off in the measured settings, not constant overhead for arbitrary span or context lengths.

D.1 Search versus Additional Decoding

The latency comparison above reports CAL's additional cost. Here we test whether spending a similar budget on more decoding provides the same benefit. On LLaDA-Base at $L_{init} = 32$, Fixed-fast uses 16 decoding steps, Fixed-full uses 32, and CAL uses 16 decoding steps plus length search. We use this setting because the LLaDA sampler releases no additional tokens when decoding steps exceed the canvas length. CAL uses $D = 4$ and $\Delta L = 1$, with maximum lengths of 64 for Single-Line and 128 for Multi-Line.

Table 10: **Search versus additional decoding** on LLaDA-Base, $L_{init} = 32$. Forwards and time are per-example means. Time measures search plus decoding, excluding loading and evaluation.

Task	Method	Pass@1	Search	Decode	Total	Time (s)
Single	Fixed-fast	44.59	0.00	16	16.00	1.191
	Fixed-full	48.98	0.00	32	32.00	2.382
	CAL-search	**51.98**	15.72	16	31.72	2.347
Multi	Fixed-fast	24.90	0.00	16	16.00	1.225
	Fixed-full	27.63	0.00	32	32.00	2.438
	CAL-search	**30.46**	15.33	16	31.33	2.384

Table 10 shows that CAL exceeds Fixed-full by 3.00/2.83 Single-/Multi-Line Pass@1 points at similar average forward counts and wall time. This is an approximate average-budget comparison, not a per-example compute guarantee.

E Extended Code Infilling Analysis

E.1 Length Estimation Accuracy

Table 11 reports the average absolute length estimation error $|\hat{L} - L^*|$ alongside Pass@1, comparing fixed-length baselines and CAL on HumanEval-Infilling (LLaDA-8B-Base).

Table 11: **Length estimation error and Pass@1** on HumanEval-Infilling (LLaDA-8B-Base). LenErr $= |\hat{L} - L^*|$; percentage in parentheses is the relative reduction over Fixed.

TASK	L_{init}	FIXED LENERR	CAL LENERR	FIXED PASS@1	CAL PASS@1
SINGLE	4	5.51	3.57 (−35%)	22.7%	70.4%
	8	3.91	3.58 (−8%)	51.0%	73.5%
	16	7.99	7.93 (−1%)	56.6%	65.2%
	32	22.60	21.09 (−7%)	49.0%	52.6%
MULTI	4	42.81	39.69 (−7%)	4.6%	17.1%
	8	39.20	37.31 (−5%)	10.8%	21.2%
	16	33.69	31.91 (−5%)	18.7%	27.1%
	32	28.61	27.35 (−4%)	27.6%	31.7%

For Single-Line tasks, CAL reduces length error by up to 35% (at $L_{init} = 4$), directly explaining the large Pass@1 gain. For Multi-Line tasks, Pass@1 improves substantially even when residual length errors remain large (e.g., Multi $L_{init} = 4$: 4.6% → 17.1%), because code solutions are not unique: CAL steers the model toward valid alternative completions at the corrected length, improving Pass@1 beyond what length error alone would predict.

E.2 Performance by Oracle Length

To assess the effect of ground-truth span length on CAL's effectiveness, we stratify Pass@1 by the Oracle length L^*. Tables 12 and 13 report results for Single-Line ($L_{init} = 8$, Oracle avg. ≈9.5 tokens) and Multi-Line ($L_{init} = 32$, Oracle avg. ≈46.8 tokens).

Table 12: **Single-Line Pass@1 stratified by Oracle length** ($L_{init} = 8$, LLaDA-8B-Base). N is the number of test samples in each bin.

ORACLE LEN	N	FIXED PASS@1	CAL PASS@1	Δ
1–5	213	59.2%	83.6%	+24.4
6–10	442	71.9%	85.3%	+13.4
11–15	161	16.1%	67.7%	+51.6
16–44	117	5.1%	18.8%	+13.7

Table 13: **Multi-Line Pass@1 stratified by Oracle length** ($L_{init} = 32$, LLaDA-8B-Base). N is the number of test samples in each bin.

ORACLE LEN	N	FIXED PASS@1	CAL PASS@1	Δ
1–15	1227	50.5%	55.3%	+4.8
16–30	1247	48.8%	55.0%	+6.2
31–50	1233	25.8%	32.4%	+6.6
51–80	1092	2.6%	3.8%	+1.2
81–128	654	0.8%	0.9%	+0.2

For Single-Line, CAL improves all Oracle length ranges, with the largest gain at $L^* = 11$–15 (+51.6 pp), where fixed-length baselines suffer most from span mismatch. For Multi-Line, gains narrow beyond $L^* = 50$ under $L_{init} = 32$. This does not establish an absolute length limit or isolate the backbone's generation capacity: the initialization and search range also constrain CAL. The longer-span experiment below tests this distinction. Table 13 displays only the bins up to 128 tokens.

E.3 Stronger Multi-Line Fixed-Length Baselines

We add $L_{init} = 48, 64$ for LLaDA-Base and LLaDA-Instruct, using the original code-fitted bias and one decoding step per selected token. These lengths are near and above the mean Multi-Line reference length. CAL retains gains of 1.54–3.20 Pass@1 points over these stronger baselines.

Table 14: **Stronger Multi-Line fixed-length baselines.** Gains use unrounded scores.

Model	L_{init}	Fixed	CAL	Gain
LLaDA-Base	48	28.50	31.71	+3.20
	64	29.96	31.50	+1.54
LLaDA-Instruct	48	32.51	34.87	+2.36
	64	31.88	34.58	+2.69

E.4 Long-Span Infilling

We evaluate LLaDA-Base on 108 Multi-Line spans with ground-truth lengths of 160–240 tokens. These spans come from six HumanEval functions, so this is a focused diagnostic subset. Fixed and CAL use $L_{init} = 200$; CAL uses $L_{max} = 256$, $D = 4$, $\Delta L = 1$, and the original code-fitted bias without re-fitting. Formal decoding uses one step per selected token.

CAL improves Pass@1 from 20.37% to 40.74% with about 7% more forwards and time. It changes 95/108 canvases and selects lengths 184–217; the mean absolute change across all examples is 4.06 tokens. In contrast, the original $L_{init} = 32, L_{max} = 128$ run selects 36.04 tokens on average and at most 91 on these examples. CAL can refine a long canvas near 200, but this local search does not bridge a large initialization gap. This experiment is not compute-matched.

Table 15: **Long-span results.** Pass@1 is in percent; length, forward counts, and search-plus-decode time are sample averages. GT LenErr is $|\hat{L} - L_{\text{gt}}|$. Reference length uses the ground-truth length at inference and is not a guaranteed upper bound.

Method	Pass@1	$\hat{L}$	GT LenErr	Search FWD	Decode FWD	Total FWD	Time (s)
Reference length	76.85	189.59	0.00	0.00	189.59	189.59	23.685
Fixed-200	20.37	200.00	22.37	0.00	200.00	200.00	24.936
CAL-200	40.74	200.32	22.42	13.93	200.32	214.25	26.693

F Extended Text Infilling Evaluation

F.1 Additional Metrics on ROCStories

Table 16 extends the ROCStories evaluation with BLEU-4, METEOR, and BERTScore (RoBERTa-large), supplementing the BLEU-2 and ROUGE-L reported in the main paper (Table 3). CAL results here use in-domain bias calibration fitted on 1,000 held-out ROCStories samples excluded from evaluation (see Section F.2).

Table 16: **Extended text infilling metrics on ROCStories** (LLaDA-8B-Base). CAL uses in-domain $B(L)$. BERTScore computed with RoBERTa-large.

METHOD	L_{init}	BLEU-2	BLEU-4	ROUGE-L	METEOR	BERTSCORE
FIXED	2	9.1	3.7	25.8	12.7	80.5
	4	18.0	9.8	37.9	25.0	88.9
	8	17.8	9.3	37.6	31.2	88.8
	Avg	15.0	7.6	33.8	23.0	86.1
CAL (IN-DOMAIN)	2	17.4	9.0	35.4	22.9	84.1
	4	20.2	10.7	39.1	26.7	86.3
	8	20.1	10.9	39.3	31.1	88.9
	Avg	**19.2**	**10.2**	**37.9**	**26.9**	**86.4**
Δ AVG		+4.3	+2.6	+4.2	+3.9	+0.3

CAL consistently improves all n-gram and METEOR metrics. BERTScore improvement is modest (+0.3 on average), which reflects its narrow dynamic range across conditions (80–89): BERTScore saturates in high-overlap short-span infilling settings and is therefore less discriminative than n-gram metrics in this regime.

F.2 In-Domain Bias Calibration

We fit a separate in-domain $B(L)$ from scratch on 1,000 held-out ROCStories samples, rather than updating the code-fitted curve. The fitted curve is approximately $B_{\text{text}}(L) \approx 0.48 \cdot e^{-0.07L}$ $(a \to 0)$, qualitatively consistent with the heuristic in Appendix B. Table 17 compares calibration strategies over the remaining 51,665 examples, averaged over $L_{init} \in \{2, 4, 8\}$. The original main-text ROCStories table uses 52,665 examples; the two evaluation sets should not be treated as identical.

Table 17: **Calibration strategy comparison on ROCStories** (LLaDA-8B-Base, averaged over $L_{init} \in \{2, 4, 8\}$). Oracle uses the ground-truth L^* directly as a reference-length baseline, not a guaranteed upper bound.

CALIBRATION	BLEU-2	BLEU-4	ROUGE-L	METEOR	BERTSCORE
FIXED (NO CAL)	14.9	7.6	33.8	23.0	86.1
CODE-FITTED $B(L)$	19.5	10.2	38.2	26.2	86.4
IN-DOMAIN $B(L)$	19.2	10.2	37.9	26.9	86.4
ORACLE	29.1	15.9	46.6	35.9	91.0

Both CAL variants substantially outperform Fixed and yield similar average results. This supports using the code-fitted curve directly on the tested text domain without domain-specific updates. It does not establish arbitrary out-of-distribution or online adaptation.

F.3 LLM-as-Judge Evaluation

To complement n-gram and embedding-based metrics, we conducted an LLM-as-judge evaluation using Gemini-3.1-Pro on ROCStories. For each of the $N = 452$ stories with valid judge outputs across all three L_{init} values (sampled from 1,100 per L_{init}, presentation order randomized), we presented the judge with the prefix, suffix, and three candidate completions (Fixed, code-fitted CAL, in-domain CAL) in random order, asking it to rank them by coherence and naturalness. Win rate is the fraction of comparisons where a method is ranked first; ties are permitted, so rows may sum to more than 100%.

Table 18: **LLM-as-judge win rates on ROCStories** ($N = 452$ stories with valid outputs at all three initial lengths, Gemini-3.1-Pro judge). Win rate is the fraction ranked first; ties permitted.

L_{init}	FIXED WIN%	CODE-FITTED WIN%	IN-DOMAIN WIN%
2	29.2%	56.0%	**57.5%**
4	39.6%	47.1%	**48.9%**
8	38.1%	42.7%	**42.9%**
AVG	35.6%	48.6%	**49.8%**

Both CAL variants substantially outperform Fixed across all settings (avg. ≈49% vs. 36%), with the largest margin at $L_{init} = 2$ (+28 pp), where fixed 2-token spans are systematically too short for coherent completions. The results confirm that CAL's quality improvements extend beyond n-gram overlap to human-interpretable coherence and naturalness.

F.4 Image-Conditioned Caption Infilling

We evaluate LaViDa (Li et al., 2026) and LLaDA-V (You et al., 2026) on MS COCO captions (Lin et al., 2014). Given an image and caption prefix/suffix, the model fills a contiguous span of 2–8 randomly removed whitespace-separated words, retaining at least one word on each side. Each model uses its own bias fitted on the same 50 calibration examples, excluding probes within ± 1 token of the reference length. We freeze each curve and evaluate on the same 500 examples, whose images are disjoint from calibration. CAL searches token lengths 1–16 with $\Delta L = 1$, $D = 2$, and $L_{init} \in \{2, 4, 8\}$.

Fixed and CAL use the same configured decoding budget, $S = L_{init}$; effective decoding can finish earlier, and CAL adds search forwards. This is not a compute-matched comparison. We report mean generated-span BLEU-2 and ROUGE-L using the text-infilling metrics.

Table 19: **Image-conditioned caption infilling.** Scores are multiplied by 100; Avg. averages the three initial lengths. Each model uses a separately fitted multimodal bias.

Model	Method	$L_{init} = 2$		$L_{init} = 4$		$L_{init} = 8$		Avg.	
		B-2	R-L	B-2	R-L	B-2	R-L	B-2	R-L
LaViDa	Fixed	12.8	32.2	19.8	42.2	19.4	39.4	17.3	37.9
	CAL	19.2	40.8	22.4	44.4	19.7	39.4	20.4	41.5
LLaDA-V	Fixed	12.8	33.6	23.0	45.4	20.8	40.8	18.9	39.9
	CAL	19.4	41.6	23.7	45.6	22.8	42.3	22.0	43.2

CAL improves average quality on both models, with the largest gains at $L_{init} = 2$. Larger fixed canvases may already provide sufficient space, while alternative valid captions need not score higher against one reference. An earlier LaViDa experiment, masking a middle token span, did not show a robust reference-aligned Oracle Peak. The revised quality results therefore support CAL in this tested image-conditioned setting, not a universal peak or zero-shot transfer of the code-fitted bias.

G Experimental Setup Details

Hyperparameters and Resources. For our adaptive length discovery method (CAL), we set the search step size $\Delta L = 1$ for all tasks to ensure fine-grained length probing. The search tolerance is set to $D = 4$ for code infilling tasks and $D = 2$ for text infilling tasks. The original code experiments use $L_{init} \in \{4, 8, 16, 32\}$, and the text experiments use $L_{init} \in \{2, 4, 8\}$. These code and text experiments were conducted on a node equipped with 8 NVIDIA A40 GPUs. Additional initialization and calibration settings are reported with the corresponding experiments above.

Input Format. We adopted the standard infilling paradigm where the input sequence is constructed as [Prefix] [MASK] [Suffix]. During the denoising process, the tokens in the prefix and suffix regions remain clamped (i.e., unmasked and fixed), while the model performs length adjustment and decoding exclusively on the masked span.

Baseline DAEDAL Implementation. We adapted DAEDAL (Li et al., 2025), a training-free dynamic length method originally for chat tasks, to serve as our infilling baseline. The original DAEDAL algorithm consists of two stages: (1) an initial length extension based on `<eos>` confidence, and (2) a runtime expansion based on low-confidence tokens during decoding. For the infilling adaptation, we omitted the first stage because DLMs in infilling mode typically do not generate explicit `<eos>` tokens, rendering the stop-signal-based extension ineffective. We focused on the second stage, where the span length is dynamically increased if tokens exhibit low confidence. Based on tuning results (see Table 20), we adjusted the confidence thresholds to better suit the infilling distribution: the low-confidence threshold was raised to 0.3 (compared to 0.1 in the original paper), as we found 0.1 to be too conservative for infilling scenarios. The high-confidence threshold was maintained at the original value of 0.9. The expansion span was set to 2 tokens to facilitate fine-grained length adjustment. The maximum generation length was constrained to 64 for Single-Line tasks and 128 for Multi-Line tasks.

H Parameter Ablation

H.1 DAEDAL Threshold Sensitivity

We first examine the impact of the low-confidence threshold in the DAEDAL baseline. As shown in Table 20, the performance of DAEDAL is highly sensitive to this threshold. A lower threshold (e.g., 0.1 or 0.2) makes the model conservative in expanding the length, failing to correct underestimated lengths (e.g., $L = 4$). Conversely, a higher threshold (e.g., 0.5) encourages aggressive expansion, which benefits short initial lengths but degrades performance when the initial length is already sufficient ($L = 16$ or 32) by introducing unnecessary redundancy. We selected a threshold of 0.4 as the optimal trade-off, achieving the highest average Pass@1 (44.8%) across all settings, which matches the fixed-length baseline on average but provides better robustness for shorter initializations.

Table 20: **Impact of DAEDAL low-confidence threshold on code infilling performance.** Evaluations on HumanEval-Infilling (SingleLine) by LLaDA-8B-Base.

Threshold	Initial Length (Pass@1)				Avg.
	$L = 4$	$L = 8$	$L = 16$	$L = 32$	
Fixed Length	22.7	51.0	**56.6**	**49.0**	**44.8**
0.1	22.7	49.8	54.8	46.0	43.3
0.2	23.6	50.0	54.8	43.5	43.0
0.3	26.4	51.2	55.6	43.5	44.2
0.4	29.9	52.8	53.4	43.1	**44.8**
0.5	**32.9**	**54.2**	50.0	38.8	44.0

H.2 CAL Search Parameters

We further investigate the robustness of our CAL method with respect to its two key search parameters: the tolerance D (stopping criterion) and the step ΔL (search granularity).

Impact on Code Infilling. Table 21 presents the results on the HumanEval-Infilling benchmark. We observe a clear trade-off between search accuracy and computational cost. Increasing the tolerance D from 2 to 4 leads to a performance gain (from 71.4% to 74.4%), as it prevents the search from terminating prematurely at local optima. However, this comes at the cost of increased search steps. Beyond $D = 4$, the performance plateaus while the cost continues to rise. Regarding step size, a finer granularity ($\Delta L = 1$) outperforms a coarser one ($\Delta L = 2$), confirming that precise length estimation is critical for code generation. Consequently, we adopt $D = 4$ and $\Delta L = 1$ as the default configuration for code tasks.

Impact on Text Infilling. For text infilling tasks, as shown in Table 22, the method exhibits greater robustness to parameter variations. Even with a minimal tolerance of $D = 1$, CAL significantly outperforms the fixed-length baseline. Increasing D to 2 or 3 yields marginal improvements in BLEU-2 and ROUGE-L scores. This suggests that the confidence landscape in natural language is smoother than in code, making the Oracle Peak easier to locate. To balance performance and efficiency, we select $D = 2$ and $\Delta L = 1$ for all text infilling experiments.

Table 21: **Impact of CAL search parameters on code infilling.** LLaDA-8B-Base with $L_{init} = 8$. The CAL rows retain the original 1,033-instance Single-Line ablation, including calibration instances; the Fixed row is the 933-instance test result. Stps. denotes average search steps. The new ablations in Tables 8 and 23 use the 933-instance test split.

TOLERANCE D	STEP ΔL	PASS@1	STPS.
FIXED LENGTH ($L = 8$)		51.0	–
2	1	71.4	7.3
2	2	65.9	6.4
3	1	73.1	9.8
4	1	**74.4**	12.2
5	1	**74.4**	14.4

Table 22: **Impact of CAL parameters on text infilling performance.** Evaluations are conducted on ROCStories dataset by LLaDA-8B-Base.

TOLERANCE D	STEP ΔL	LENGTH = 2		LENGTH = 4		LENGTH = 8		AVG.	
		B-2	R-L	B-2	R-L	B-2	R-L	B-2	R-L
FIXED LENGTH		9.1	25.8	18.0	37.9	17.8	37.6	15.0	33.8
1	1	15.0	32.6	20.2	39.5	21.0	40.9	18.7	37.7
1	2	18.0	36.4	19.5	38.3	20.7	40.2	19.4	**38.3**
2	1	17.6	35.7	19.8	38.4	21.2	40.7	19.5	**38.3**
3	1	18.8	37.1	19.5	37.9	20.8	39.9	**19.7**	**38.3**

H.3 Alternative Search Confidence Scores

We compare Top-1 confidence with the top-1/top-2 probability margin $p_j^{(1)} - p_j^{(2)}$ and entropy certainty $1 - H(p_j)/\log|\mathcal{V}|$, where $H(p_j) = -\sum_v p_j(v) \log p_j(v)$. Each score is averaged over masked positions. The alternatives use separate bias curves fitted on the same 100 calibration examples; Top-1 retains the main experiment's rounded default curve. Evaluation uses LLaDA-Base on the 933 Single-Line test examples with $D = 4$ and $\Delta L = 1$. Only length probing changes; formal decoding keeps the original remasking rule.

The three fitted curves have $R^2 = 0.9955/0.9910/0.9987$. Margin performs similarly to Top-1. Entropy improves mean Pass@1 by 1.15 points using unrounded scores, but is worse at $L_{init} = 32$. We retain Top-1 as the default.

Table 23: **Alternative confidence scores for length search.** Pass@1 (%) on HumanEval Single-Line.

L_{init}	Top-1	Margin	Entropy
4	70.42	69.88	**73.74**
8	73.53	73.31	**75.67**
16	65.17	63.99	**65.81**
32	52.63	**53.05**	51.13
Mean	65.43	65.06	**66.59**

I Case Studies

To provide a concrete understanding of how CAL operates, we present a detailed case study on the task HumanEval/0/L3 (Bavarian et al., 2022). As shown in Table 24, this task requires the model to infill the core logic for calculating the distance between two numbers within a nested loop. The ground-truth middle segment is `distance = abs(elem - elem2)\n`, which consists of exactly 10 tokens using the LLaDA tokenizer.

Search Process Analysis. Table 25 demonstrates the robustness of our bidirectional search. Whether starting from an underestimated length ($L = 4$, Table a) or an overestimated one ($L = 16$, Table b), the calibrated confidence $\Phi_c(L)$ effectively suppresses length bias and guides the algorithm to find the Oracle length ($L = 10$).

Visualization of the Oracle Peak. Figure 4 plots the confidence landscape for this task. It clearly shows that while the raw confidence (red) decays due to structural noise (blue dashed), the calibrated score (blue solid) reveals a prominent Oracle Peak at $L = 10$. Although local fluctuations exist, the calibrated confidence metric $\Phi_c(L)$ reaches its maximum at the Oracle length.

Table 24: **Illustration of an Infilling Task Structure.** This example (HumanEval/0/L3) demonstrates the decomposition of a code generation task into a prefix (P), a ground-truth middle (M^*), and a suffix (S). The model's goal is to recover the core semantic logic within the masked span.

Prefix (P)

```
from typing import List

def has_close_elements(numbers, threshold):
  """ Check if in given list of numbers, are any
  two numbers closer to each other than given
  threshold.
  >>> has_close_elements([1.0, 2.0, 3.0], 0.5)
  False
  >>> has_close_elements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0],
0.3)
  True
  """
  for idx, elem in enumerate(numbers):
    for idx2, elem2 in enumerate(numbers):
      if idx != idx2:
```

Middle (M^*)

```
        distance = abs(elem - elem2)
```

Suffix (S)

```
      if distance < threshold:
        return True

  return False
```

Table 25: **Case Studies of Bidirectional Length Probing.** We illustrate the search process for Task HumanEval/0/L3 on LLaDA-8B-Instruct (matching the example in Figure 1) starting from different initial lengths. $\Phi(L)$ denotes the raw confidence, $B(L)$ the fitted bias, and $\Phi_c(L)$ the calibrated confidence. In both scenarios, the algorithm accurately identifies the Oracle length $L^* = 10$ (highlighted), where the calibrated confidence $\Phi_c(L)$ is maximized and the model recovers the ground-truth logic.

(a) Starting from an underestimated length ($L_{init} = 4$).

Probe L	$\Phi(L)$	$B(L)$	$\Phi_c(L)$	**First-Step Denoising Prediction**
$L = 4$	0.956	0.681	1.403	`distance =\n`
$L = 5$	0.941	0.655	1.437	`distance = elem\n`
$L = 6$	0.926	0.631	1.468	`distance = elem2\n`
$L = 7$	0.820	0.608	1.348	`distance = abs(elem2\n`
$L = 8$	0.979	0.587	1.669	`distance = elem - elem2\n`
$L = 9$	0.790	0.566	1.395	`distance = abs(elem elem2)\n`
$L = 10$	0.997	0.547	**1.821**	`distance = abs(elem - elem2)\n`
$L = 11$	0.886	0.529	1.674	`distance = abs(elem - elem2)\n\n`
$L = 12$	0.604	0.513	1.179	`distance = abs(elem abs -2 elemabs)\n`
$L = 13$	0.583	0.497	1.173	`distance = abs(elem - abs(elem - elem2)\n`
$L = 14$	0.472	0.482	0.979	`distance = abs  distance abs abs(elem - elem2)\n`
$L = 3$	0.887	0.713	1.244	`distance\n`
$L = 2$	0.926	0.766	1.209	`\n`
$L = 1$	1.000	0.938	1.066	`\n`

(b) Starting from an overestimated length ($L_{init} = 16$).

Probe L	$\Phi(L)$	$B(L)$	$\Phi_c(L)$	**First-Step Denoising Prediction**
$L = 16$	0.524	0.454	1.153	`distance = abs(elem - elem2) abs(elem - elem2)\n`
$L = 17$	0.520	0.442	1.177	`distance = abs(elem - elem2)\n abs(elem - elem2)\n`
$L = 18$	0.414	0.430	0.963	`distance = abs(elem - elem2) distance abs abs(elem - elem2)\n`
$L = 19$	0.425	0.419	1.014	`distance = abs(elem - elem2\n distance = abs(elem - elem2)\n`
$L = 20$	0.389	0.409	0.951	`distance = abs(elem - elem2\n\n distance = abs(elem - elem2)\n`
$L = 21$	0.351	0.399	0.879	`distance = abs(elem - elem2\n\n distance =(elem abs(elem - elem2)\n`
$L = 15$	0.427	0.468	0.913	`distance = abs(elem - -2) abs - elem2)\n`
$L = 14$	0.472	0.482	0.979	`distance = abs  distance abs abs(elem - elem2)\n`
$L = 13$	0.583	0.497	1.173	`distance = abs(elem - abs(elem - elem2)\n`
$L = 12$	0.604	0.513	1.179	`distance = abs(elem abs -2 elemabs)\n`
$L = 11$	0.886	0.529	1.674	`distance = abs(elem - elem2)\n\n`
$L = 10$	0.997	0.547	**1.821**	`distance = abs(elem - elem2)\n`
$L = 9$	0.790	0.566	1.395	`distance = abs(elem elem2)\n`
$L = 8$	0.979	0.587	1.669	`distance = elem - elem2\n`
$L = 7$	0.820	0.608	1.348	`distance = abs(elem2\n`
$L = 6$	0.926	0.631	1.468	`distance = elem2\n`

J Broader Impacts

CAL improves the infilling quality of Diffusion Language Models in applications such as code completion and text generation. On the positive side, it enables better performance from existing DLMs without any additional training, lowering the deployment barrier for practitioners and reducing the need for manual length tuning. On the negative side, improvements to generative models can in principle facilitate the production of misleading or low-quality content at scale. However, CAL is a training-free inference algorithm that introduces no new generative capabilities beyond those of the underlying DLM; any associated societal risks are therefore commensurate with those already present in the base models themselves and not amplified by our method. This work relies solely on publicly available benchmarks and pre-trained models, conducts no human subjects research, and does not release new training data.

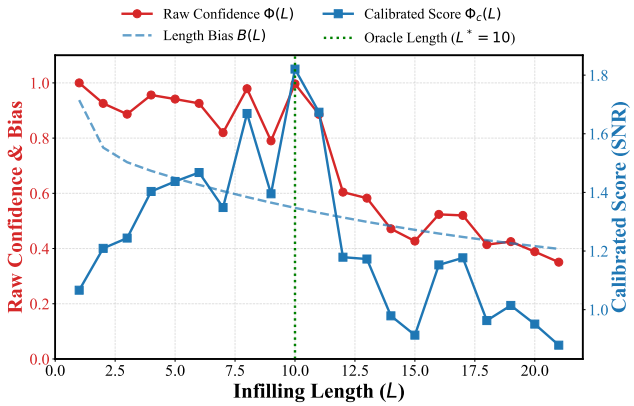


Figure 4: **Dynamics of the confidence landscape during bidirectional search.** We compare the raw confidence $\Phi(L)$ (red) and the calibrated score $\Phi_c(L)$ (blue solid). While the raw signal is biased towards shorter lengths, our calibration against the systematic bias $B(L)$ (blue dashed) reveals a prominent Oracle Peak at the ground-truth length $L^* = 10$ (green dotted line).

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: All contributions (Oracle Peak/Length Bias phenomena, two-scale heuristic motivation for the calibration form, CAL method) and scope limitations are clearly stated in the abstract and introduction.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 7 contains a dedicated Limitations paragraph covering latency overhead, static bias function, and the single-span restriction.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.

- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not contain formal theoretical results. Appendix B provides a heuristic motivation for the empirical double-exponential form of B(L), explicitly stated as intuition rather than derivation; the form is selected empirically based on goodness-of-fit (Appendix C).

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Algorithm 1, all hyperparameters, and implementation details are provided; the bias fitting script is included in supplementary material.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.

- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The release provides core CAL inference and calibration code, selected evaluation scripts, and data preparation instructions. It does not include every exploratory script or raw output; the benchmarks are obtained from their original sources.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: All hyperparameters, hardware (8× NVIDIA A40), data splits, and baseline implementation details are in Appendix (Experimental Setup Details).

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Appendix C.3 reports standard deviations over three seeded calibration subsets for each size below 100. These quantify subset sensitivity, not decoding randomness. Pass@1 measures functional test success; deterministic decoding does not remove finite-sample uncertainty.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Hardware for the original experiments is specified in the appendix. Search and decoding costs are reported in Tables 9, 10, and 15; the release documents the multimodal model environments.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.

- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This work uses only publicly available models and benchmarks, conducts no human subjects research, and raises no privacy or fairness concerns.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: A dedicated discussion is provided in Appendix J.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: No new model weights or raw image collection are released. The release contains CAL code and evaluation preparation scripts using existing datasets.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All models and benchmarks are properly cited and used for research purposes consistent with their public terms.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Core CAL code and the image-conditioned caption-infilling protocol are documented with their parameters and data preparation procedure. No new model weights are introduced.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: No human annotators are involved; the LLM-as-judge evaluation uses an automated model evaluator.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: No human subjects research is conducted.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs appear only as an automated evaluator in Appendix F.3 and are not part of the proposed method CAL.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.