WHEN TO STOP AND WHICH TO RETURN: TRAJECTORY-LEVEL MINIMUM BAYES RISK DECODING FOR DIFFUSION LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Masked diffusion language models can produce high-quality intermediate responses before denoising finishes, yet continued decoding may incur unnecessary computation or overwrite better responses. Existing approaches address these challenges separately. Early-commitment methods reduce computation but cannot recover earlier responses. Temporal voting exploits intermediate responses to improve quality but retains the cost of full decoding. We propose **Trajectory-MBR**, a training-free method that jointly decides when to stop and which current or earlier response to return. It reuses existing denoiser outputs to construct complete response candidates and performs trajectory-score-weighted minimum Bayes risk selection within a sliding window. Once the minimum weighted risk falls to or below a threshold, decoding stops and returns the selected response. We show that accurate trajectory-based utility estimates yield near-optimal selection within the window and provide an offline condition for improvement over full decoding. Across language and multimodal tasks, Trajectory-MBR maintains or improves average response quality with average denoiser-evaluation speedups of up to $3.76\times$, while its token-level extension preserves response quality at a $12.34\times$ speedup on HumanEval.

1 INTRODUCTION

Masked diffusion language models (dLLMs), including LLaDA (Nie et al., 2025), Dream (Ye et al., 2025), and UltraLLaDA (He et al., 2026), generate responses through iterative denoising. At each checkpoint, the denoiser predicts clean-token distributions for all remaining masked positions in parallel (Austin et al., 2021; Sahoo et al., 2024; Shi et al., 2024). These predictions allow complete response candidates to be constructed before denoising finishes. Standard decoding nevertheless typically executes a predetermined number of transitions and returns only the terminal response, leaving useful intermediate predictions unexploited.

Continuing to the terminal state can therefore hurt both efficiency and quality. Li et al. (2025a) find that, on trajectories whose final answer is correct, correct answer tokens on GSM8K and MMLU often become stable top-1 predictions well before decoding finishes. Moreover, Wang et al. (2025b) document temporal oscillation, where later predictions overwrite an intermediate correct answer. These observations reveal two failure modes involving redundant computation on an already stable answer and the loss of a better response as decoding continues. Addressing both requires deciding not only when to stop, but also which response produced so far to return.

Existing methods address these decisions separately. Prophet (Li et al., 2025a), KLASS (Kim et al., 2025), and JoT (Kohut et al., 2026) use confidence or stability signals to commit predictions early. Their outputs remain tied to the predictions they commit to, without retrospective selection among complete checkpoint responses. Temporal Self-Consistency Voting (Wang et al., 2025b) demonstrates the value of aggregating intermediate responses, but collects this evidence over a completed trajectory and therefore retains the cost of full decoding. Together, these approaches motivate an online rule that uses trajectory evidence both to select a response and to determine whether further denoising is necessary.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

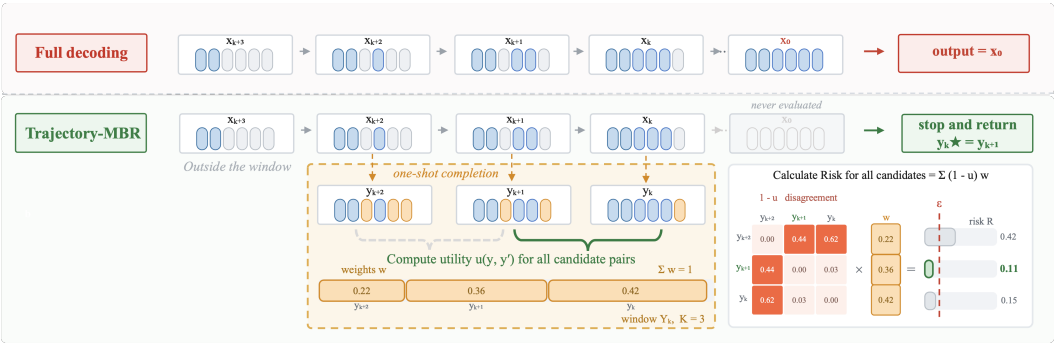

Figure 1: **Overview of Trajectory-MBR.** Full decoding (top) unrolls the sampler to $\mathbf{x}_0$ and returns the terminal state. We instead score the K most recent checkpoint candidates against each other and stop as soon as the weighted risk of the best one falls to or below ϵ (bottom). In this example, the selected candidate is $\mathbf{y}_{k+1}$, and the remaining transitions are never run.

We propose **Trajectory-MBR**[1], a training-free decoding method that decouples the stopping checkpoint from the checkpoint whose response is returned. As illustrated in Figure 1, it constructs complete candidates by retaining revealed tokens and filling masked positions with the denoiser's current top predictions. A sliding window retains recent candidates for selection. Candidate construction and scoring reuse existing denoiser outputs, requiring no additional denoiser evaluations.

To select among these candidates, we use minimum Bayes risk decoding (Freitag et al., 2022; Jinnai et al., 2024). We compare complete responses through task utility and weight each reference candidate by a trajectory score derived from token probabilities, adapting the trajectory-likelihood construction of d^2 (Wang et al., 2026a). The selected candidate minimizes weighted utility loss against the window's references. This minimum risk measures weighted disagreement. Once it falls to or below a threshold, decoding stops and returns the selected current or earlier response. Thus, the same criterion supports both stopping and selection, without gold references or access to future checkpoints.

Agreement among intermediate responses does not by itself establish their quality, motivating an analysis of its relationship to target utility. Under explicit assumptions, we bound trajectory-score approximation error and target-utility regret relative to the best candidate in the current window. The analysis separates score error from reference mismatch and provides an offline sufficient condition for the selected response to outperform full decoding.

To assess the practical impact of this decision rule, we evaluate Trajectory-MBR on 13 LongBench tasks and nine multimodal benchmarks. The method improves average quality while reducing denoiser evaluations. On UltraLLaDA, it raises the LongBench average from 44.90 to 48.37 with a $3.76\times$ average denoiser-evaluation speedup. Extending the trajectory-consensus principle to token-level early commitment yields a $12.34\times$ speedup while preserving quality.

2 Related Work

dLLM Inference Acceleration and Adaptive Decoding. Prior work on dLLM inference acceleration has reduced either the cost of individual denoising steps or the number of denoiser evaluations required for generation. Sparse-dLLM, SparseD, and Focus-dLLM make denoising steps cheaper through cache management or sparse attention computation (Song et al., 2026; Wang et al., 2026b; Long et al., 2026). Fast-dLLM combines approximate KV caching with parallel decoding, reducing redundant computation while also shortening the decoding trajectory (Wu et al., 2026). Adaptive decoding methods such as KLASS, Prophet, and JoT instead use stability, confidence, or convergence signals from intermediate predictions. These signals are used to adapt unmasking or commit predictions early, thereby reducing the number of denoiser evaluations (Kim et al., 2025; Li et al., 2025a; Kohut et al., 2026).

Denoising Trajectories and Trajectory Scoring. Beyond accelerating decoding, the denoising trajectory itself has been studied as a target for refinement and analysis. ReMDM revisits previously

[1]Code is available at `https://anonymous.4open.science/r/Trajectory-MBR`.

unmasked tokens through remasking to correct earlier prediction errors (Wang et al., 2025a), while DDPD and P2 improve subsequent generation by planning which positions require further denoising or by controlling the token unmasking order (Liu et al., 2025a; Peng et al., 2025). Temporal Self-Consistency Voting (Wang et al., 2025b) further shows that correct intermediate answers can be lost during later denoising and combines intermediate answers through temporal voting after completing the trajectory. Trajectory-MBR similarly exploits intermediate responses, but uses trajectory-score-weighted agreement during decoding to jointly decide *when to stop* and *which checkpoint response to return*. Separately, Wang et al. (2026a) estimate sampling-trajectory likelihoods and use them as a training signal for reasoning diffusion language models. Trajectory-MBR treats checkpoint responses as candidate outputs and uses trajectory scores to weight their agreement at inference time.

Minimum Bayes Risk Decoding. Minimum Bayes Risk (MBR) decoding provides a natural framework for this selection problem. Rather than selecting hypotheses solely by model probability, which need not align with output quality (Eikema & Aziz, 2020; Freitag et al., 2022), MBR selects according to expected utility under a reference distribution. This framework has since been extended with neural utility metrics (Freitag et al., 2022) and consensus-based interpretations of generation (Bertsch et al., 2023). Jinnai et al. (2024) further propose Model-Based MBR (MBMBR), which incorporates model sequence probabilities as non-uniform weights over reference hypotheses. Cheng & Vlachos (2023) reduce the cost of MBR decoding by progressively adding reference samples for utility estimation while pruning hypotheses based on confidence. Trajectory-MBR adapts these utility-based and model-weighted principles to correlated checkpoint responses from a single ongoing diffusion trajectory. It uses weighted agreement among the candidates available at the current checkpoint to support both online response selection and adaptive stopping.

3 PRELIMINARIES

Masked diffusion language models. Let $\mathcal{V}$ be the token vocabulary and let $[\texttt{MASK}] \notin \mathcal{V}$ be the absorbing state of the forward corruption process. Given a prompt $\mathbf{c}$ and a response $\mathbf{z} = (z^1, \ldots, z^L) \in \mathcal{V}^L$ of length L, a pretrained masked diffusion language model takes a partly masked response and predicts the original token at every masked position in parallel (Austin et al., 2021; Sahoo et al., 2024; Shi et al., 2024; Ou et al., 2025). We consider a base sampler with T decoding transitions and index its checkpoints by $k \in \{T, \ldots, 0\}$. Let $\mathbf{x}_k \in (\mathcal{V} \cup \{[\texttt{MASK}]\})^L$ be the partly masked response at checkpoint k and let $\mathcal{M}_k \coloneqq \{i \in \{1, \ldots, L\} \mid x_k^i = [\texttt{MASK}]\}$ be its masked positions. For every $i \in \mathcal{M}_k$, let $r_{\theta,k}^i(v \mid \mathbf{x}, \mathbf{c})$ denote the raw denoiser distribution at checkpoint k, including any noise-level input supplied by the base sampler. Let $p_{\theta,k}^i(v \mid \mathbf{x}, \mathbf{c})$ denote the categorical distribution obtained after any deterministic sampler-side transformation of that output, such as temperature scaling, truncation, or guidance, that is applied before token assignment. When no such transformation is used, $p_{\theta,k}^i = r_{\theta,k}^i$. At the actual state $\mathbf{x}_k$, we abbreviate $p_{\theta,k}^i(v) \coloneqq p_{\theta,k}^i(v \mid \mathbf{x}_k, \mathbf{c})$ for $v \in \mathcal{V}$. The deployed token-assignment rule acts on this distribution, and the stochastic ancestral counterpart introduced below samples from it.

Base sampler. Let $\mathcal{A}$ denote the deployed base sampler used at inference time, that is, the decoder that would run without our stopping rule. Starting from $\mathbf{x}_T = [\texttt{MASK}]^L$, it produces $\mathbf{x}_T \to \mathbf{x}_{T-1} \to \cdots \to \mathbf{x}_0$ with $\mathcal{M}_0 = \varnothing$. At transition $k \to k-1$, the sampler selects a subset of the currently masked positions according to its reveal rule, for example by confidence, and assigns their values according to its token-assignment rule, such as top-1 assignment in greedy decoding (Wu et al., 2026). We restrict attention to absorbing samplers in which every response position is revealed exactly once and remains visible thereafter. Accordingly, we write $\mathcal{U}_r \coloneqq \mathcal{M}_r \setminus \mathcal{M}_{r-1}$ for the positions revealed at transition $r \to r-1$, and $\mathcal{U}_1, \ldots, \mathcal{U}_T$ partition $\{1, \ldots, L\}$. Revealing a position is also referred to as unmasking or committing it, and a transition is also called a step.

4 METHODOLOGY

4.1 TRAJECTORY-MBR

As shown in Figure 1, Trajectory-MBR is a training-free stopping and selection rule. Candidate construction and scoring add no denoiser evaluation as long as they reuse the output the base transition already computes at each visited checkpoint, and any evaluation beyond these is counted separately. Appendix A.2 details the reuse condition and the non-denoiser cost of comparing candidates.

Intermediate candidates. At each checkpoint $k \in \{1, \dots, T\}$, we form a full candidate $\mathbf{y}_k$ by keeping every visible token of $\mathbf{x}_k$ and filling every masked position with its top token,

$$y_k^i \coloneqq \begin{cases} x_k^i, & i \notin \mathcal{M}_k, \\ \arg\max_{v \in \mathcal{V}} p_{\theta,k}^i(v), & i \in \mathcal{M}_k. \end{cases} \tag{1}$$

If several tokens attain the maximum in Equation 1, we select the first one under the fixed token-index order of the vocabulary. At the terminal checkpoint, we set $\mathbf{y}_0 \coloneqq \mathbf{x}_0$.

Candidate trajectory score. Adapting the trajectory-likelihood construction of d2 (Wang et al., 2026a), we attach a token probability to every reveal event and use the resulting product as an inference-time score. At checkpoint k, we join the observed prefix $\mathbf{x}_T \to \cdots \to \mathbf{x}_k$ with the one-shot completion $\mathbf{x}_k \to \mathbf{y}_k$ and call and refer to the result as the candidate trajectory of $\mathbf{y}_k$. Its score multiplies the probability of every token revealed along the prefix with the probability of every token used to complete the current state,

$$s_k \coloneqq \prod_{r=k+1}^{T} \prod_{i \in \mathcal{U}_r} p_{\theta,r}^i\big(x_{r-1}^i\big) \prod_{i \in \mathcal{M}_k} p_{\theta,k}^i\big(y_k^i\big), \qquad k = 0, \dots, T. \tag{2}$$

At $k = 0$, the second product is empty because $\mathcal{M}_0 = \varnothing$, so s_0 scores the observed trajectory alone and requires no denoiser evaluation at the terminal state. We use s_k only to compare candidates inside a window and do not require it to be a calibrated sequence probability. Under the deterministic-reveal assumptions of Theorem 1, s_k approximates the candidate likelihood under the stochastic ancestral counterpart $\bar{\mathcal{A}}$ of the deployed sampler, defined in Section 4.2. The approximation error is controlled by how much each candidate token's log probability differs between checkpoint k and the state at which that token would eventually be revealed. It does not require uniform stability at all intervening checkpoints. If the reveal rule is stochastic, the score is instead interpreted conditionally on the realized reveal path unless the reveal-set probabilities are included. Further qualifications for stochastic reveal rules and transformed token distributions are given in Appendix A.1.

Trajectory-score-weighted risk and decision. Let $K \in \{2, \dots, T+1\}$ be the window size. Once K candidates are available, we keep the K most recent entries

$$Y_k \coloneqq (\mathbf{y}_k, \mathbf{y}_{k+1}, \dots, \mathbf{y}_{k+K-1}), \qquad 0 \le k \le T - K + 1, \tag{3}$$

where a larger offset refers to an earlier checkpoint. The same window serves as the decision set and the reference set. MBMBR (Jinnai et al., 2024) removes duplicate references before applying model sequence probabilities, whereas we keep every checkpoint entry. If the same response appears at several checkpoints, each occurrence keeps its own weight, so a repeated response receives more total support. The weights are the trajectory scores normalized within the window,

$$w_{k+j} \coloneqq \frac{s_{k+j}}{\sum_{r=0}^{K-1} s_{k+r}}, \qquad j = 0, \dots, K-1. \tag{4}$$

Let $u(\mathbf{y}, \mathbf{y}') \in [0, 1]$ be the task utility, with $u(\mathbf{y}, \mathbf{y}) = 1$. Any raw utility used in the implementation must be normalized to satisfy these conditions as specified in Appendix A.2. The weighted risk of each entry and the resulting decision are

$$R_{k,\ell} \coloneqq \sum_{j=0}^{K-1} \left(1 - u(\mathbf{y}_{k+\ell}, \mathbf{y}_{k+j})\right) w_{k+j}, \qquad \ell = 0, \dots, K-1, \tag{5}$$

$$\ell_k^\star \in \arg\min_{0 \le \ell < K} R_{k,\ell}, \qquad \mathbf{y}_k^\star \coloneqq \mathbf{y}_{k+\ell_k^\star}. \tag{6}$$

If several entries attain the minimum in Equation 6, we select the most recent one, that is, the entry with the smallest offset ℓ. The selected entry can come from any checkpoint in the window, so it need not be the current completion $\mathbf{y}_k$, and it may disagree with a token already fixed in $\mathbf{x}_k$. Because nearby entries share revealed tokens, the entries of Y_k are not independent samples from the model, and we therefore read Equation 5 as a trajectory-score-weighted agreement measure rather than as a Monte Carlo estimate of Bayes risk. Section 4.2 analyzes the resulting reference-set mismatch separately from the score-approximation error. We update prefix log scores incrementally and reuse cached pairwise utilities as the window advances. After initialization, each window update requires only $O(K)$ new utility evaluations involving the incoming candidate, as detailed in Appendix A.2.

Risk-based stopping. Let $\epsilon \in [0, 1]$ be the risk threshold. Testing starts once $T - k + 1 \geq K$, and decoding stops at the first eligible checkpoint with

$$R_{k,\ell_k^\star} \leq \epsilon, \tag{7}$$

returning $\mathbf{y}_k^\star$, which may have been constructed at an earlier checkpoint than the one at which decoding stops. The stopping test is an agreement-based criterion tuned for the task. By itself, Equation 7 does not certify a small target-utility regret or improvement over full decoding. If no earlier checkpoint passes the test, the decision is made on the terminal window at $k = 0$, so the base-sampler output $\mathbf{x}_0 = \mathbf{y}_0$ always enters the final decision. Because every decision entry also serves as a reference, the minimum risk satisfies $R_{k,\ell_k^\star} \leq 1 - 1/K$, so any $\epsilon \geq 1 - 1/K$ stops at the first complete window, and a single dominant weight can trigger stopping even when the candidates disagree. We therefore select K, ϵ, and the task utility u jointly for each task. Table 8 lists the search grid, and Appendix A.2 gives the implementation details and Algorithm 1.

4.2 Theoretical Analysis of Trajectory-MBR

For the likelihood analysis, let $\bar{\mathcal{A}}$ denote the stochastic ancestral counterpart of the deployed sampler $\mathcal{A}$. It shares the same reveal rule and the denoiser distributions used in Equation 2, but samples each revealed token from the corresponding categorical distribution. If the deployed sampler already satisfies these sampling assumptions, then $\bar{\mathcal{A}} = \mathcal{A}$. The first theorem bounds the discrepancy between the trajectory score and the response likelihood under $\bar{\mathcal{A}}$, extending the trajectory-likelihood analysis of d2 (Wang et al., 2026a). The second gives a deterministic regret bound on a fixed candidate window, adapting the MBR analysis of Ichihara et al. (2025). A third result, Proposition 1 in Appendix A.3.5, gives an offline sufficient condition for improvement over full decoding. None of these results is an online certificate for the stopping test in Equation 7. Proofs are given in Appendix A.

Theorem 1 (Trajectory Score Approximation)**.** *Suppose the reveal rule shared by $\mathcal{A}$ and $\bar{\mathcal{A}}$ is a deterministic function of the current state and reveals every position exactly once. Suppose further that $\bar{\mathcal{A}}$ draws the values revealed at each transition independently from the sampler-side categorical distributions of Equation 2. For every candidate considered below, assume that every score factor and every eventual-reveal factor in the induced response factorization is strictly positive. Let $p_{\theta,\bar{\mathcal{A}}}(\mathbf{y} \mid \mathbf{c})$ denote the response distribution induced by $\bar{\mathcal{A}}$. For each $i \in \mathcal{M}_k$, let $\tau_{k,i}(\mathbf{y}_k) \in \{1, \ldots, k\}$ be the checkpoint at which position i would be revealed if $\bar{\mathcal{A}}$ were continued consistently with $\mathbf{y}_k$, and let $\widetilde{\mathbf{x}}_{k,i}(\mathbf{y}_k)$ be the state at which position i is revealed on that continuation. Define the current-to-reveal discrepancy by*

$$\gamma_{k,i} \coloneqq \left| \log p^i_{\theta,k}\big(y^i_k \mid \mathbf{x}_k, \mathbf{c}\big) - \log p^i_{\theta,\tau_{k,i}(\mathbf{y}_k)}\big(y^i_k \mid \widetilde{\mathbf{x}}_{k,i}(\mathbf{y}_k), \mathbf{c}\big) \right|, \qquad \varepsilon_k^{\mathrm{traj}} \coloneqq \sum_{i \in \mathcal{M}_k} \gamma_{k,i}. \tag{8}$$

Then the trajectory score of Equation 2 satisfies $\big|\log s_k - \log p_{\theta,\bar{\mathcal{A}}}(\mathbf{y}_k \mid \mathbf{c})\big| \leq \varepsilon_k^{\mathrm{traj}}$ at every checkpoint k. If in addition $\max_{0 \leq j < K} \varepsilon_{k+j}^{\mathrm{traj}} \leq \bar{\varepsilon}_k$, the window weights of Equation 4 satisfy

$$e^{-2\bar{\varepsilon}_k} \leq \frac{w_{k+j}}{\overline{p}_{k+j}} \leq e^{2\bar{\varepsilon}_k}, \qquad \overline{p}_{k+j} \coloneqq \frac{p_{\theta,\bar{\mathcal{A}}}(\mathbf{y}_{k+j} \mid \mathbf{c})}{\sum_{r=0}^{K-1} p_{\theta,\bar{\mathcal{A}}}(\mathbf{y}_{k+r} \mid \mathbf{c})}, \qquad j = 0, \ldots, K-1. \tag{9}$$

Thus, when $\bar{\varepsilon}_k$ is small, the window weights approximate the likelihoods under $\bar{\mathcal{A}}$ restricted and normalized to the indexed entries of the current window. The discrepancy $\varepsilon_k^{\mathrm{traj}}$ depends on counterfactual states at which the remaining candidate tokens would be revealed and is an analysis quantity, not an online stopping signal. Although $\varepsilon_0^{\mathrm{traj}} = 0$ because $\mathcal{M}_0 = \varnothing$, no monotonic decrease is guaranteed at intermediate checkpoints. When the deployed sampler is greedy, the likelihood interpretation is with respect to $\bar{\mathcal{A}}$ rather than the deployed sampler's degenerate output distribution.

Theorem 2 (Conditional Decision Regret for Trajectory-MBR)**.** *Let P_{h} be a target reference distribution with expected utility $U_{\mathrm{h}}(\mathbf{y}) \coloneqq \mathbb{E}_{\mathbf{y}' \sim P_{\mathrm{h}}}[u(\mathbf{y}, \mathbf{y}')]$, and let $\mathbf{y}_{k+\ell_k^{\mathrm{h}}}$ be the entry of Y_k that maximizes U_{h}. Write $\widehat{U}_k(\mathbf{y}) \coloneqq \sum_{j=0}^{K-1} u(\mathbf{y}, \mathbf{y}_{k+j})\, w_{k+j}$ for the trajectory-score-weighted utility, which equals $1 - R_{k,\ell}$ at $\mathbf{y} = \mathbf{y}_{k+\ell}$ and is therefore the quantity maximized by Equation 6. We assume that $\widehat{U}_k$ approximates U_{h} on the window, in the sense that $\max_{0 \leq \ell < K} \big|U_{\mathrm{h}}(\mathbf{y}_{k+\ell}) - \widehat{U}_k(\mathbf{y}_{k+\ell})\big| \leq \eta_k$. Then the selected response $\mathbf{y}_k^\star$ of Equation 6 satisfies*

$$U_{\mathrm{h}}(\mathbf{y}_{k+\ell_k^{\mathrm{h}}}) - U_{\mathrm{h}}(\mathbf{y}_k^\star) \leq 2\eta_k. \tag{10}$$

Remark. Under the conditions of Theorem 1, the hypothesis holds with $\eta_k = \delta_k^{\mathrm{win}} + \tanh(\overline{\varepsilon}_k)$, where $U_k^{\mathrm{win}}(\mathbf{y}) := \sum_{j=0}^{K-1} u(\mathbf{y}, \mathbf{y}_{k+j})\,\overline{p}_{k+j}$ and $\delta_k^{\mathrm{win}} := \max_{0\le\ell<K} \left|U_{\mathrm{h}}(\mathbf{y}_{k+\ell}) - U_k^{\mathrm{win}}(\mathbf{y}_{k+\ell})\right|$. This separates the uncontrolled target-to-window mismatch from the trajectory-score approximation error. For the tight choice $\overline{\varepsilon}_k = \max_{0\le j<K} \varepsilon_{k+j}^{\mathrm{traj}}$, the latter vanishes only when every $\varepsilon_{k+j}^{\mathrm{traj}}$ in the window is zero, as shown in Appendix A.3.3. The guarantee is relative to the best entry in the window, whereas Proposition 1 compares with full decoding only offline, since $\mathbf{y}_{\mathrm{full}}$ is unavailable at an early stop.

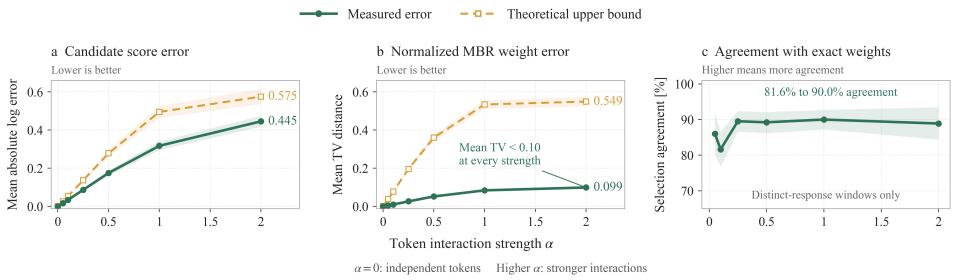

Figure 2: **Mean weight error below 0.10 with 81.6% to 90.0% selection agreement.** Panels a and b compare mean errors with theoretical bounds. Panel c shows same-response agreement under exact likelihood weights on windows containing distinct responses. Shading denotes pointwise 95% context-bootstrap intervals.

Controlled evaluation of trajectory-score approximation. We assess the approximation in Theorem 1 using an exactly computable binary model with varying token interaction strength. Figure 2 shows that mean window-weight TV remains below 0.10, while the theoretical bounds are conservative. Replacing trajectory-score weights with exact likelihood weights preserves the selected response in 81.6% to 90.0% of windows containing distinct responses. Appendix A.4 provides the detailed experimental setup and full results.

Table 1: **Higher LongBench scores with fewer denoiser evaluations.** Across 13 tasks, Trajectory-MBR improves average scores from 44.90 to 48.37 on UltraLLaDA and from 43.03 to 44.56 on Dream-7B-Instruct, with average denoiser-evaluation speedups of 3.76× and 1.94× over full decoding. † denotes results reported by Long et al. (2026), with settings detailed in Appendix B.1.

	Single-Doc. QA		Multi-Doc. QA			Summarization		Few-shot Learning		Synthetic		Code		Avg. Score	Avg. Speedup
Method	Qasper	MF-en	HotpotQA	2WikiMQA	MuSiQue	GovReport	QMSum	TREC	TriviaQA	LSHT	PRe	LCC	RB-P		
UltraLLaDA															
Full decoding†	19.14	25.87	16.27	18.00	12.08	32.83	22.48	**80.00**	91.58	41.00	96.75	68.23	**59.50**	44.90	1.00×
Fast-dLLM†	18.34	29.90	17.03	17.11	13.36	30.05	22.89	79.50	91.03	42.00	94.75	67.50	58.10	44.74	1.70×
Sparse-dLLM†	18.04	27.26	20.59	17.88	13.67	29.95	**23.57**	76.50	**91.93**	41.50	97.12	67.50	57.72	44.86	1.00×
SparseD†	19.09	25.87	15.45	18.04	11.92	32.64	22.50	79.50	90.70	41.50	96.79	68.10	59.02	44.70	1.00×
Focus-dLLM†	17.02	29.11	22.47	21.49	20.20	26.75	21.45	77.00	90.78	41.00	95.73	66.72	57.14	45.14	1.00×
Trajectory-MBR	**23.41**	**34.25**	**28.03**	**27.76**	**20.30**	**32.88**	22.91	**80.00**	91.58	**42.50**	**97.33**	**68.30**	**59.50**	**48.37**	3.76×
Dream-7B-Instruct															
Full decoding†	35.58	40.49	41.59	42.10	23.36	**23.51**	19.66	71.50	87.34	15.75	32.50	63.79	62.23	43.03	1.00×
Fast-dLLM†	37.54	43.24	35.56	35.74	17.97	21.14	19.57	70.50	88.25	16.75	46.17	62.21	61.11	42.75	1.83×
Sparse-dLLM†	37.50	43.23	36.83	34.97	17.05	20.60	20.05	70.00	88.38	17.00	**46.50**	62.80	61.20	42.78	1.00×
SparseD†	37.66	40.96	41.39	41.61	**24.17**	**23.51**	19.66	**72.50**	86.85	15.75	36.83	**63.86**	61.98	43.59	1.00×
Focus-dLLM†	37.38	41.96	38.96	38.56	18.05	21.06	19.26	70.00	88.76	17.25	44.25	60.62	60.50	42.82	1.00×
Trajectory-MBR	**38.33**	**43.36**	**41.82**	**42.46**	23.90	21.87	**20.05**	**72.50**	**89.31**	**18.00**	40.50	**63.86**	**63.38**	**44.56**	1.94×

5 Experiments

We evaluate Trajectory-MBR by applying the sequence-level rule to long-context language generation and to diffusion vision-language models, and by extending the principle to token-level early commitment. Dataset configurations, task utility definitions and hyperparameters are given in Appendix B.

5.1 Experimental Settings

Language dataset. We conduct experiments on 13 LongBench tasks (Bai et al., 2024), following the protocol of Long et al. (2026). We use UltraLLaDA (He et al., 2026) and Dream-7B-Instruct (Ye

Table 2: **Higher multimodal quality with fewer denoiser evaluations.** Trajectory-MBR matches or exceeds full decoding on all nine benchmarks for both backbones, with average denoiser-evaluation speedups of 1.97× on LLaDA-V and 1.13× on LaViDa. LLaVA-Bench scores rise from 61.3 to 75.8 and from 39.5 to 52.6, respectively. †† denotes results reported by Hong et al. (2026), except for the re-evaluated columns specified in Appendix B.2.

Method	General			Visual Grounding			Long-form Generation			Avg. Score	Avg. Speedup
	MME (sum)	MMBench	MMMU	RefCOCOg	Ferret	GQA	LLaVA-Bench	DetailCaps	MIA		
LLaDA-V											
Full decoding††	1993	83.1	48.7	70.7	89.6	61.6	61.3	59.8	66.1	281.54	1.00×
Fast-dLLM	1999	83.0	**49.6**	68.2	29.5	61.6	29.5	58.6	53.4	270.27	1.00×
Sparse-dLLM	2003	83.0	49.4	68.6	88.1	62.0	67.3	57.8	73.5	283.63	1.00×
SparseD	1928	80.4	48.8	70.2	**93.1**	**62.2**	71.4	57.9	**74.6**	276.29	1.01×
Trajectory-MBR	**2006**	**83.2**	49.3	**71.0**	91.4	61.6	**75.8**	**60.0**	72.6	**285.66**	1.97×
LaViDa											
Full decoding††	1691	72.3	**43.8**	38.5	55.2	59.6	39.5	8.3	49.4	228.62	1.00×
Fast-dLLM	1698	71.8	42.9	**41.2**	21.4	60.1	26.4	20.3	39.0	224.58	1.00×
Sparse-dLLM	**1711**	71.8	42.9	38.7	49.6	59.6	51.4	26.8	56.0	234.20	1.00×
SparseD	1661	65.6	40.4	33.6	52.4	58.2	49.5	**32.9**	53.7	227.48	1.00×
Trajectory-MBR	1708	**72.5**	**43.8**	38.6	**60.5**	**60.2**	**52.6**	24.1	**56.3**	**235.16**	1.13×

et al., 2025), and compare Trajectory-MBR against full decoding, Fast-dLLM (Wu et al., 2026), Sparse-dLLM (Song et al., 2026), SparseD (Wang et al., 2026b) and Focus-dLLM (Long et al., 2026). Full decoding runs the unmodified sampler to the terminal checkpoint. Per-task metrics follow Long et al. (2026) and are listed in Table 5, and the average score is the unweighted macro-average over all 13 tasks. The task utility is computed only between trajectory candidates, while gold references are used for final evaluation.

Multimodal dataset. To test whether the same decision rule transfers across modalities, we conduct experiments on nine multimodal benchmarks, following the evaluation protocol of Hong et al. (2026). We use LLaDA-V (You et al., 2026) and LaViDa (Li et al., 2025b), and compare Trajectory-MBR against full decoding, Fast-dLLM (Wu et al., 2026), Sparse-dLLM (Song et al., 2026), and SparseD (Wang et al., 2026b). Per-task metrics follow Hong et al. (2026) and are listed in Table 6. The average score is the unweighted macro-average over all nine benchmarks, with the task utility computed only among trajectory candidates as before.

Extension to token-level early commitment. To test whether trajectory consensus transfers to a finer decision granularity, we evaluate Trajectory-Token-MBR on GSM8K (Cobbe et al., 2021), MMLU (Hendrycks et al., 2021), HellaSwag (Zellers et al., 2019) and HumanEval (Chen et al., 2021). We use Dream-7B-Instruct (Ye et al., 2025) and LLaDA-8B-Instruct (Nie et al., 2025), and compare against full decoding, JoT (Kohut et al., 2026), KLASS (Kim et al., 2025) and Prophet (Li et al., 2025a). Per-benchmark metrics follow Kohut et al. (2026) and are listed in Table 7.

Efficiency. We report denoiser-evaluation speedup for sequence-level experiments and step-based speedup for token-level experiments, as described in Appendix B.4.

6 Results

6.1 Quantitative Analysis

Language dataset Table 1 reports results on 13 LongBench tasks. Trajectory-MBR achieves the highest average score with the fewest denoiser evaluations on both backbones, improving UltraLLaDA from 44.90 to 48.37 and Dream-7B-Instruct from 43.03 to 44.56 at average denoiser-evaluation speedups of 3.76× and 1.94×, respectively. It matches or exceeds vanilla decoding on all 13 tasks for UltraLLaDA and 12 for Dream-7B-Instruct, while three of the four baselines reduce average quality on both backbones. Figure 7 further shows that fewer denoiser evaluations translate into lower latency on 200 UltraLLaDA/LSHT examples.

Multimodal dataset. Table 2 reports the performance on nine multimodal benchmarks. Trajectory-MBR achieves the highest average score on both backbones while executing the fewest denoiser evaluations, raising the average from 281.54 to 285.66 at 1.97× on LLaDA-V and from 228.62

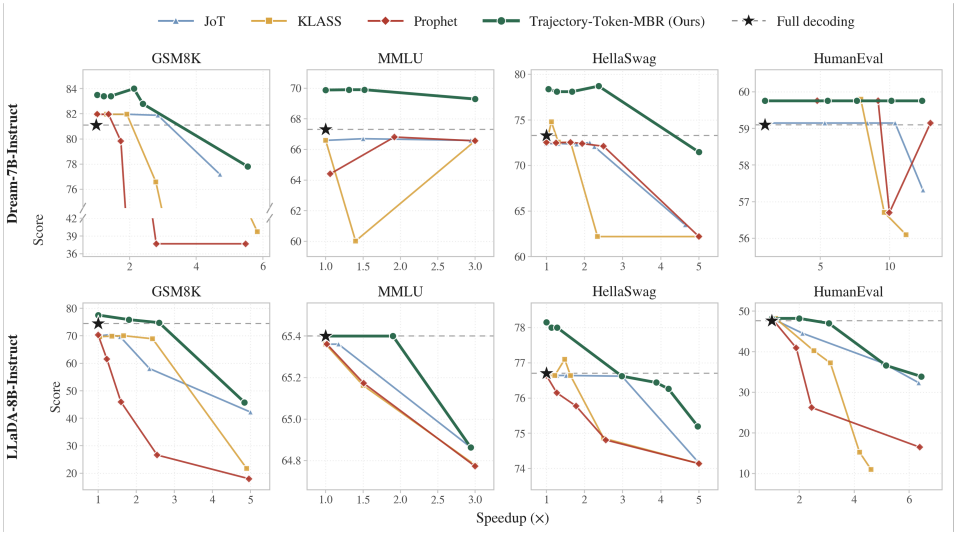

Figure 3: **Higher HumanEval pass@1 with a** $12.34\times$ **step-based speedup.** Trajectory-Token-MBR improves Dream-7B-Instruct pass@1 from 59.10% to 59.76% at this speedup and matches or exceeds full-decoding quality at selected accelerated operating points across all four benchmarks and both backbones. Top and bottom rows show Dream-7B-Instruct and LLaDA-8B-Instruct, respectively. Stars mark full decoding. Settings follow Kohut et al. (2026) and are detailed in Appendix B.3.

to 235.16 at $1.13\times$ on LaViDa, both above the strongest baseline. It also matches or exceeds full decoding on all nine benchmarks for both backbones, whereas on each backbone two of the three baselines fall below full decoding on average. The largest gains appear on long-form generation, where LLaVA-Bench rises from 61.3 to 75.8 on LLaDA-V and from 39.5 to 52.6 on LaViDa, and MIA improves by more than six points on both backbones.

Extension to token-level early commitment. At positions that pass the JoT gate, Trajectory-Token-MBR selects the token with the lowest weighted risk from two to K recent top-1 proposals and commits it early if the risk is at most ϵ. Other positions remain subject to the base transfer schedule. Figure 3 shows accelerated operating points that match or exceed full-decoding quality across all benchmarks and both backbones. On Dream-7B-Instruct, GSM8K accuracy rises from 81.10% to 82.79% at $2.40\times$ and HumanEval pass@1 from 59.10% to 59.76% at $12.34\times$. Implementation and evaluation details appear in Appendix B.3.

6.2 Qualitative Analysis

Table 3: **Higher LSHT accuracy with a** $12.94\times$ **denoiser-evaluation speedup.** The default setting raises UltraLLaDA accuracy from 41.00% to 42.50%. Intermediate window sizes and risk thresholds attain the highest accuracy in their sweeps. Each sweep varies one factor on the same 200 examples and 64-step trajectories. The defaults are $K = 4$, $\epsilon = 0.05$, and exact match, shown in bold.

UltraLLaDA on LSHT

Window size K (fixed ϵ, $u(\mathbf{y}, \mathbf{y}')$)			Risk threshold ϵ (fixed K, $u(\mathbf{y}, \mathbf{y}')$)			Task utility $u(\mathbf{y}, \mathbf{y}')$ (fixed K, ϵ)		
K	Score	Speedup	ϵ	Score	Speedup	Utility	Score	Speedup
2	41.50	24.11×	0	41.00	1.28×	Exact match	**42.50**	**12.94**×
4	**42.50**	**12.94**×	0.01	42.50	12.28×	BLEU	42.50	10.80×
6	42.50	9.54×	**0.05**	**42.50**	**12.94**×	Character 3-gram F1	41.50	13.63×
8	42.00	7.68×	0.10	42.00	13.28×	Token F1	41.50	12.30×
16	41.00	3.93×	0.50	41.00	15.94×	ROUGE-L	41.50	12.28×
64	41.00	1.00×	1.00	41.00	16.00×	Embedding similarity	41.00	16.00×

Sensitivity to window size, risk threshold, and task utility. Table 3 reports a one-factor-at-a-time analysis of K, ϵ, and $u(\mathbf{y}, \mathbf{y}')$ on all 200 LSHT examples using the same 64-step UltraLLaDA trajectories. Window size and risk threshold exhibit interior optima, with $K = 64$ and both threshold extremes yielding the full-decoding score of 41.00. A narrow window limits the candidate set, while a wide one delays stopping toward the end of the trajectory. Likewise, $\epsilon = 0$ requires exact agreement

and $\epsilon = 1.00$ stops at the first complete window, so the two extremes reach the same score despite different stopping behavior. Exact match and BLEU reach 42.50, while other overlap measures reach 41.50 and embedding similarity matches the baseline. These results suggest selecting the window size, threshold, and task utility jointly to balance response quality and decoding cost.

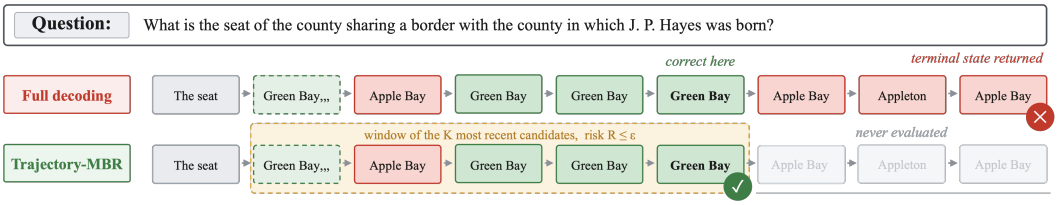

Figure 4: **Preserving a correct answer before it is lost.** On this MuSiQue example with Dream-7B-Instruct, Trajectory-MBR stops and returns the correct "Green Bay," while full decoding ends with the incorrect "Apple Bay." Both share the same trajectory up to stopping. Grayed-out checkpoints are skipped by Trajectory-MBR.

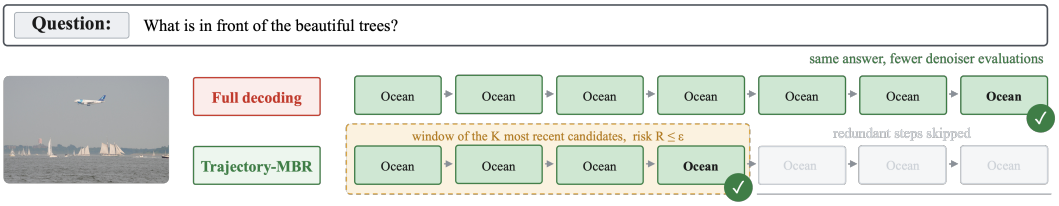

Figure 5: **Preserving the same correct answer with fewer steps.** On this GQA example with LLaDA-V, "Ocean" is correct from the first checkpoint and remains unchanged. Trajectory-MBR stops once weighted agreement is sufficient, skipping the steps that full decoding spends reproducing the answer. Both share the same trajectory up to stopping.

Qualitative examples. Figures 4 and 5 illustrate the two failure modes in Section 1, with both methods sharing the same trajectory in each example. In Figure 4, Trajectory-MBR stops before continued denoising corrupts a correct intermediate answer. In Figure 5, response-level agreement enables early stopping despite low token confidence, avoiding redundant steps that reproduce an unchanged correct answer. Figure 6 demonstrates checkpoint selection. Despite oscillation and an incorrect terminal answer, Trajectory-MBR selects an earlier correct candidate with the lowest weighted risk. These decisions require neither future checkpoints nor ground-truth labels. Table 9 shows a long-form MIA example where Trajectory-MBR satisfies every requirement while the baselines degrade the response.

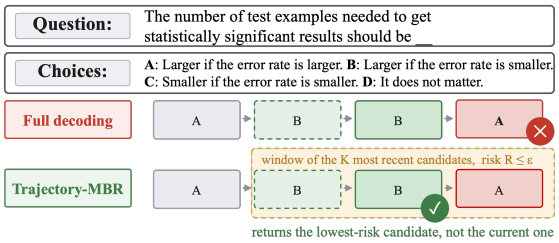

Figure 6: **Recovering an earlier correct answer when the current one is wrong.** On this MMLU example with LLaDA-8B-Instruct, full decoding returns the incorrect choice. Trajectory-MBR selects the earlier correct choice with the lowest weighted risk in the window, demonstrating the benefit of checkpoint selection.

7 Conclusion

We proposed **Trajectory-MBR**, a training-free method that decouples when to stop from which response to return. It uses trajectory-score-weighted agreement to preserve useful intermediate responses and avoid redundant denoising without additional denoiser evaluations. Experiments across language and multimodal benchmarks, together with a token-level extension, demonstrate improvements in quality and efficiency. Our analysis provides conditional selection guarantees within the candidate window and an offline sufficient condition for improvement over full decoding. These findings highlight denoising trajectories as evidence for both better response selection and earlier stopping.

AI USE STATEMENT

We used large language models (LLMs) to refine the language of the manuscript and as judges for response evaluation. Editorial assistance included correcting grammatical errors, rephrasing awkward sentences, and improving stylistic consistency. LLM-based judges assessed generated responses according to the evaluation protocols described in the paper. The authors independently developed the core research ideas and analytical methods, interpreted the findings, and formulated the conclusions. The authors take responsibility for the accuracy and integrity of the work.

REFERENCES

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, volume 34, pp. 17981–17993, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/958c530554f78bcd8e97125b70e6973d-Abstract.html.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3119–3137, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.172. URL https://aclanthology.org/2024.acl-long.172/.

Amanda Bertsch, Alex Xie, Graham Neubig, and Matthew Gormley. It's MBR all the way down: Modern generation techniques through the lens of minimum Bayes risk. In *Proceedings of the Big Picture Workshop*, pp. 108–122, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.bigpicture-1.9. URL https://aclanthology.org/2023.bigpicture-1.9/.

Maosong Cao, Kai Chen, Haodong Duan, Yixiao Fang, Tong Gao, Ge Jiaye, Mo Li, Hongwei Liu, Junnan Liu, Yuan Liu, Chengqi Lyu, Han Lyu, Ningsheng Ma, Zerun Ma, Yu Sun, Zhiyong Wu, Linchen Xiao, Jun Xu, Haochen Ye, Zhaohui Yu, Yike Yuan, Songyang Zhang, Yufeng Zhao, Fengzhe Zhou, Peiheng Zhou, Dongsheng Zhu, Lin Zhu, and Jingming Zhuo. OpenCompass: A universal evaluation platform for large language models. *arXiv preprint arXiv:2605.19276*, May 2026. doi: 10.48550/arXiv.2605.19276. URL https://arxiv.org/abs/2605.19276.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, July 2021. doi: 10.48550/arXiv.2107.03374. URL https://arxiv.org/abs/2107.03374.

Julius Cheng and Andreas Vlachos. Faster minimum Bayes risk decoding with confidence-based pruning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12473–12480, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.767. URL https://aclanthology.org/2023.emnlp-main.767/.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, October 2021. doi: 10.48550/arXiv.2110.14168. URL https://arxiv.org/abs/2110.14168.

Hongyuan Dong, Jiawen Li, Bohong Wu, Jiacong Wang, Yuan Zhang, and Haoyuan Guo. Benchmarking and improving detail image caption. *arXiv preprint arXiv:2405.19092*, May 2024. doi: 10.48550/arXiv.2405.19092. URL https://arxiv.org/abs/2405.19092.

Bryan Eikema and Wilker Aziz. Is MAP decoding all you need? the inadequacy of the mode in neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 4506–4520, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.398. URL https://aclanthology.org/2020.coling-main.398/.

Markus Freitag, David Grangier, Qijun Tan, and Bowen Liang. High quality rather than high model probability: Minimum Bayes risk decoding with neural metrics. *Transactions of the Association for Computational Linguistics*, 10:811–825, 2022. doi: 10.1162/tacl_a_00491. URL https://aclanthology.org/2022.tacl-1.47/.

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, Yunsheng Wu, Rongrong Ji, Caifeng Shan, and Ran He. MME: A comprehensive evaluation benchmark for multimodal large language models. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-4899. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/d79a27cf2772fe00be7f341efc0eb517-Abstract-Datasets_and_Benchmarks_Track.html.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness. Zenodo, July 2024. URL https://zenodo.org/records/12608602.

Guangxin He, Shen Nie, Fengqi Zhu, Yuankang Zhao, Tianyi Bai, Ran Yan, Jie Fu, Chongxuan Li, and Binhang Yuan. UltraLLaDA: Scaling the context length to 128K for diffusion large language models. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/448444518637da106d978ae7409d9789-Abstract-Conference.html.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Sujung Hong, Chanyong Yoon, and Seongjae Hwang. Mitigating mask prior drift and positional attention collapse in large diffusion vision-language models. *arXiv preprint arXiv:2605.14530*, May 2026. doi: 10.48550/arXiv.2605.14530. URL https://arxiv.org/abs/2605.14530.

Drew A. Hudson and Christopher D. Manning. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6700–6709, June 2019. doi: 10.1109/CVPR.2019.00686. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Hudson_GQA_A_New_Dataset_for_Real-World_Visual_Reasoning_and_Compositional_CVPR_2019_paper.html.

Yuki Ichihara, Yuu Jinnai, Kaito Ariu, Tetsuro Morimura, and Eiji Uchibe. Theoretical guarantees for minimum Bayes risk decoding. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 16262–16284, Vienna, Austria, July 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.acl-long.793. URL https://aclanthology.org/2025.acl-long.793/.

Yuu Jinnai, Tetsuro Morimura, Ukyo Honda, Kaito Ariu, and Kenshi Abe. Model-based minimum Bayes risk decoding for text generation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 22326–22347. PMLR, July 2024. URL https://proceedings.mlr.press/v235/jinnai24a.html.

Seo Hyun Kim, Sunwoo Hong, Hojung Jung, Youngrok Park, and Se-Young Yun. KLASS: KL-guided fast inference in masked diffusion models. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-3087. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/8514c1f12dcdd061ac8521feb78f28ca-Abstract-Conference.html.

Zakhar Kohut, Severyn Shykula, Mykola Vysotskyi, Serhii Dmytryshyn, Dmytro Khamula, Michal Zakrzewski, Damian Rynczak, Jacek Małecki, Taras Rumezhak, and Volodymyr Karpiv. Just on time: Token-level early stopping for diffusion language models. *arXiv preprint arXiv:2602.11133*, February 2026. doi: 10.48550/arXiv.2602.11133. URL https://arxiv.org/abs/2602.11133.

Pengxiang Li, Yefan Zhou, Dilxat Muhtar, Lu Yin, Shilin Yan, Li Shen, Soroush Vosoughi, and Shiwei Liu. Diffusion language models know the answer before decoding. *arXiv preprint arXiv:2508.19982*, August 2025a. doi: 10.48550/arXiv.2508.19982. URL https://arxiv.org/abs/2508.19982.

Shufan Li, Konstantinos Kallidromitis, Hritik Bansal, Akash Gokul, Yusuke Kato, Kazuki Kozuka, Jason Kuen, Zhe Lin, Kai-Wei Chang, and Aditya Grover. LaViDa: A large diffusion language model for multimodal understanding. In *Advances in Neural Information Processing Systems*, volume 38, 2025b. doi: 10.52202/085713-3511. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/975affbe7b5f3b55fba62247b6877b1c-Abstract-Conference.html.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 34892–34916, 2023. doi: 10.52202/075280-1516. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/6dcf277ea32ce3288914faf369fe6de0-Abstract-Conference.html.

Sulin Liu, Juno Nam, Andrew Campbell, Hannes Stärk, Yilun Xu, Tommi Jaakkola, and Rafael Gómez-Bombarelli. Think while you generate: Discrete diffusion with planned denoising. In *International Conference on Learning Representations*, 2025a. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/cbf883a744952d4a40591271a58ab9d0-Abstract-Conference.html.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. MMBench: Is your multi-modal model an all-around player? In *Computer Vision – ECCV 2024*, volume 15064 of *Lecture Notes in Computer Science*, pp. 216–233, Cham, Switzerland, 2025b. Springer. doi: 10.1007/978-3-031-72658-3_13. URL https://link.springer.com/chapter/10.1007/978-3-031-72658-3_13.

Lingkun Long, Yushi Huang, Shihao Bai, Ruihao Gong, Jun Zhang, Ao Zhou, and Jianlei Yang. Focus-dLLM: Accelerating long-context diffusion LLM inference via confidence-guided context focusing. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12146–12160, San Diego, California, United States, July 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.556. URL https://aclanthology.org/2026.acl-long.556/.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-1689. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/48b383b24230e0e6e649d9c98dae4d8c-Abstract-Conference.html.

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/a365e37c18fb91af547a2f0012a89e98-Abstract-Conference.html.

Fred Zhangzhi Peng, Zachary Bezemek, Sawan Patel, Jarrid Rector-Brooks, Sherwood Yao, Alexander Tong, and Pranam Chatterjee. Path planning for masked diffusion model sampling. *arXiv preprint arXiv:2502.03540*, February 2025. doi: 10.48550/arXiv.2502.03540. URL https://arxiv.org/abs/2502.03540.

Yusu Qian, Hanrong Ye, Jean-Philippe Fauconnier, Peter Grasch, Yinfei Yang, and Zhe Gan. MIA-Bench: Towards better instruction following evaluation of multimodal LLMs. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/571082ea18d30060177dfcaf662ff0e5-Abstract-Conference.html.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 130136–130184, 2024. doi: 10.52202/079017-4135. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/eb0b13cc515724ab8015bc978fdde0ad-Abstract-Conference.html.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. Simplified and generalized masked diffusion for discrete data. In *Advances in Neural Information Processing Systems*, volume 37, pp. 103131–103167, 2024. doi: 10.52202/079017-3277. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/bad233b9849f019aead5e5cc60cef70f-Abstract-Conference.html.

Yuerong Song, Xiaoran Liu, Ruixiao Li, Zhigeng Liu, Zengfeng Huang, Qipeng Guo, Ziwei He, and Xipeng Qiu. Sparse-dLLM: Accelerating diffusion LLMs with dynamic cache eviction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(39):33038–33046, March 2026. doi: 10.1609/aaai.v40i39.40586. URL https://ojs.aaai.org/index.php/AAAI/article/view/40586.

Guanghan Wang, Yair Schiff, Subham Sahoo, and Volodymyr Kuleshov. Remasking discrete diffusion models with inference-time scaling. In *Advances in Neural Information Processing Systems*, volume 38, 2025a. doi: 10.52202/085713-4927. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/d877ea0dd78bdbe54830670618c1de09-Abstract-Conference.html.

Guanghan Wang, Gilad Turok, Yair Schiff, Marianne Arriola, and Volodymyr Kuleshov. d2: Improving reasoning in diffusion language models via trajectory likelihood estimation. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026a. URL https://icml.cc/Downloads/2026.

Wen Wang, Bozhen Fang, Chenchen Jing, Yongliang Shen, Yangyi Shen, Qiuyu Wang, Hao Ouyang, Hao Chen, and Chunhua Shen. Time is a feature: Exploiting temporal dynamics in diffusion language models. *arXiv preprint arXiv:2508.09138*, August 2025b. doi: 10.48550/arXiv.2508.09138. URL https://arxiv.org/abs/2508.09138.

Zeqing Wang, Gongfan Fang, Xinyin Ma, Xingyi Yang, and Xinchao Wang. SparseD: Sparse attention for diffusion language models. In *International Conference on Learning Representations*, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/a1598ba0aa7bd7bdf18e0ad04d993edc-Abstract-Conference.html.

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dLLM: Training-free acceleration of diffusion LLM by enabling KV cache and parallel decoding. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/5d8d4e6061c3ba96c240b7fa1ae3471d-Abstract-Conference.html.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7B: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, August 2025. doi: 10.48550/arXiv.2508.15487. URL https://arxiv.org/abs/2508.15487.

Haoxuan You, Haotian Zhang, Zhe Gan, Xianzhi Du, Bowen Zhang, Zirui Wang, Liangliang Cao, Shih-Fu Chang, and Yinfei Yang. Ferret: Refer and ground anything anywhere at any granularity. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/fd6613131889a4b656206c50a8bd7790-Abstract-Conference.html.

Zebin You, Shen Nie, Xiaolu Zhang, JUN ZHOU, Zhiwu Lu, Ji-Rong Wen, and Chongxuan Li. LLaDA-V: Large language diffusion models with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10093–10105, June 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/You_LLaDA-V_Large_Language_Diffusion_Models_with_Visual_Instruction_Tuning_CVPR_2026_paper.html.

Licheng Yu, Patrick Poirson, Shan Yang, Alexander C. Berg, and Tamara L. Berg. Modeling context in referring expressions. In *Computer Vision – ECCV 2016*, volume 9906 of *Lecture Notes in Computer Science*, pp. 69–85, Cham, Switzerland, 2016. Springer. doi: 10.1007/978-3-319-46475-6_5. URL https://link.springer.com/chapter/10.1007/978-3-319-46475-6_5.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. MMMU: A massive multi-discipline multimodal understanding and reasoning benchmark for expert AGI. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9556–9567, June 2024. doi: 10.1109/CVPR52733.2024.00913. URL https://openaccess.thecvf.com/content/CVPR2024/html/Yue_MMMU_A_Massive_Multi-discipline_Multimodal_Understanding_and_Reasoning_Benchmark_for_CVPR_2024_paper.html.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472/.

Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, and Ziwei Liu. LMMs-Eval: Reality check on the evaluation of large multimodal models. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 881–916, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-naacl.51. URL https://aclanthology.org/2025.findings-naacl.51/.

A Additional Details and Proofs for Trajectory-MBR

This appendix provides the sampler-specific qualifications, implementation details, and formal proofs for Trajectory-MBR. We first state the conditions under which the trajectory score admits a likelihood interpretation, and we record the qualifications that apply to samplers falling outside those conditions. We then describe the implementation of the stopping rule, including incremental scoring in log space, utility caching, denoiser call accounting, and the complete procedure in Algorithm 1. Finally, we restate the results of Section 4.2 and prove them. The proof of Theorem 1 follows the trajectory-likelihood factorization of d2 (Wang et al., 2026a), and the proofs of Theorem 2 and Proposition 1 adapt the MBR regret decomposition of Ichihara et al. (2025) to our trajectory-score-weighted utility.

A.1 Sampler Scope and Trajectory Interpretation

Checkpoints and noise level. The denoiser may require a noise level input. Any such input is supplied by the base sampler and is absorbed into the checkpoint index of $p^i_{\theta,k}$. The method therefore operates only on the discrete checkpoints $k = T, \dots, 0$ and never queries the denoiser between them.

Reveal rule. The likelihood interpretation of Theorem 1 requires the reveal rule shared by $\mathcal{A}$ and $\bar{\mathcal{A}}$ to be a deterministic function of the current state. The rule may consult the denoiser output at that state, as a confidence rule does, but it must not depend on the token values drawn at the same transition. For a confidence-based rule we therefore define the confidence of a masked position as the maximum of its sampler-side categorical distribution, which depends on the state alone, so that $\mathcal{A}$ and $\bar{\mathcal{A}}$ share the rule even when $\bar{\mathcal{A}}$ draws a token other than the top-1 one. Under this condition, the next reveal set is fixed by the state and contributes no probability factor to the trajectory. If reveal positions are instead selected stochastically, the score in Equation 2 is interpreted conditionally on the realized reveal path. An unconditional trajectory likelihood would also require the probabilities of the reveal-set choices. Under the absorbing sampler scope of Section 3, the reveal sets satisfy $\mathcal{M}_{r-1} \subseteq \mathcal{M}_r$, the sets $\mathcal{U}_1, \dots, \mathcal{U}_T$ partition $\{1, \dots, L\}$, and every run terminates with $\mathcal{M}_0 = \varnothing$.

Induced response distribution. The distribution used in the analysis is $p_{\theta,\bar{\mathcal{A}}}$, the response distribution induced by the stochastic ancestral counterpart defined in Section 4.2. A deployed sampler $\mathcal{A}$ that assigns revealed values greedily has a degenerate output distribution of its own, for which likelihood comparison across candidate responses is vacuous. In that case, s_k scores the realized greedy trajectory under $\bar{\mathcal{A}}$. It is not interpreted as a non-degenerate likelihood under $\mathcal{A}$. If the deployed sampler already samples revealed values under the assumptions of Theorem 1, then $\bar{\mathcal{A}} = \mathcal{A}$.

Transformed sampling distributions. The categorical probabilities entering the score and the eventual-reveal factorization must be the distributions from which $\bar{\mathcal{A}}$ samples the revealed token values. By the definition in Section 3, $p^i_{\theta,r}$ denotes the sampler-side categorical distribution after any deterministic transformation applied before token assignment. Thus, $\bar{\mathcal{A}}$ samples from the same distribution that is used by the deployed assignment rule. If the deployed rule uses another transformation not represented by $p^i_{\theta,r}$, the theorem concerns the counterpart defined by $p^i_{\theta,r}$ rather than that deployed rule. Similarly, if values revealed at the same transition are sampled jointly rather than independently, the corresponding product in Equation 2 must be replaced by their joint probability. Truncation can assign zero probability to a token, in which case the required log probability is not finite. We therefore restrict the analysis to candidates for which every score factor and every eventual-reveal factor in the induced response factorization is strictly positive. A zero-support extension requires separate treatment.

Duplicate window entries. The window Y_k is an indexed tuple and may contain the same response at several checkpoints. The weights of Equation 4 and the normalized likelihood $\overline{p}_{k+j}$ of Equation 9 assign mass to entries rather than to distinct responses, so the total mass of a repeated response is the sum of the masses of its occurrences. All statements below are proved entrywise and therefore remain valid in the presence of duplicates.

Algorithm 1 Trajectory-MBR

Require: Prompt $\mathbf{c}$, sampler-side denoiser distribution p_θ, base sampler $\mathcal{A}$
Require: Response length L, transition count T, task utility u
Require: Window size $K \in \{2, \dots, T+1\}$, threshold $\epsilon \in [0, 1]$

Set $\mathbf{x}_T \leftarrow [\texttt{MASK}]^L$ and $a_T \leftarrow 0$
Initialize an empty recent candidate list
for $k = T, T-1, \dots, 0$ **do**
  Compute $\mathcal{M}_k$ from $\mathbf{x}_k$ and keep a copy for this iteration
  if $k > 0$ **then**
    Obtain $\{\log p^i_{\theta,k}(\cdot)\}_{i \in \mathcal{M}_k}$ from the checkpoint prediction of $\mathcal{A}$
    Form $\mathbf{y}_k$ with Equation 1
    Compute $\log s_k$ with Equation 12
  else
    Set $\mathbf{y}_0 \leftarrow \mathbf{x}_0$ and $\log s_0 \leftarrow a_0$
  end if
  Add $(\mathbf{y}_k, \log s_k)$ to the front of the list
  Remove the oldest pair if the list has more than K pairs
  if $T - k + 1 \geq K$ **then**
    Form Y_k with Equation 3
    Compute the window weights with Equation 4 and log-sum-exp
    Compute $R_{k,\ell}$ for all ℓ with Equation 5
    Compute $\ell^\star_k$ and $\mathbf{y}^\star_k$ with Equation 6
    if $k = 0$ **or** $R_{k,\ell^\star_k} \leq \epsilon$ **then**
      return $\mathbf{y}^\star_k$
    end if
  end if
  if $k > 0$ **then**
    Let $\mathcal{A}$ complete its usual transition and obtain $\mathbf{x}_{k-1}$
    Compute $\mathcal{M}_{k-1}$ from $\mathbf{x}_{k-1}$
    Set $\mathcal{U}_k \leftarrow \mathcal{M}_k \setminus \mathcal{M}_{k-1}$
    Update a_{k-1} with Equation 11
    Discard the saved denoiser output for checkpoint k
  end if
end for

A.2 Implementation Details

Incremental scoring in log space. For fast scoring, define the prefix log score for $k \in \{0, \dots, T\}$ as

$$a_k \coloneqq \sum_{r=k+1}^{T} \sum_{i \in \mathcal{U}_r} \log p^i_{\theta,r}\big(x^i_{r-1}\big)\,,$$

so that the empty sum gives $a_T = 0$. After transition $k \to k-1$ the prefix score is updated by

$$a_{k-1} = a_k + \sum_{i \in \mathcal{U}_k} \log p^i_{\theta,k}\big(x^i_{k-1}\big)\,, \qquad k = 1, \dots, T, \tag{11}$$

and for $k > 0$ the score of Equation 2 becomes

$$\log s_k = a_k + \sum_{i \in \mathcal{M}_k} \log p^i_{\theta,k}\big(y^i_k\big)\,. \tag{12}$$

At $k = 0$ the second term is empty and Equation 2 gives $\log s_0 = a_0$. All scores are held in log space, and the normalized window weights are computed with log-sum-exp.

Denoiser output reuse and call counting. At checkpoint $k > 0$ the implementation saves the denoiser output used to form $\mathbf{y}_k$ and to compute s_k. This output can also serve the base transition $\mathbf{x}_k \to \mathbf{x}_{k-1}$ if and only if that transition consumes the same denoiser evaluation at the same state and noise level, without first changing the state and without requiring an additional refinement evaluation. Any correction, resampling, guidance, or refinement evaluation that the sampler actually executes is counted separately.

Under this reuse condition a full decode uses T denoiser evaluations. If Trajectory-MBR stops at checkpoint $k > 0$, it has evaluated the denoiser at checkpoints $T, T-1, \ldots, k$, for a total of $T-k+1$ evaluations, and has completed $T-k$ transitions. These counts refer to denoiser evaluations and not to the individual token predictions produced in parallel by one evaluation.

Utility caching. Because $u(\mathbf{y}, \mathbf{y}) = 1$ is fixed, only off-diagonal values are evaluated. The first complete window therefore requires $K^2 - K$ directed utility values. When the window advances by one checkpoint, the retained $(K-1)(K-2)$ values are reused and at most $2K-2$ new directed values remain. If the utility is symmetric, only the $K-1$ new unordered pairs need to be evaluated.

Utility normalization. Let $\widetilde{u}(\mathbf{y}, \mathbf{y}')$ denote the raw utility after the metric-specific scaling below, and let $\mathrm{clip}_{[0,1]}(a) \coloneqq \min\{1, \max\{0, a\}\}$. The utility used by Trajectory-MBR is

$$u(\mathbf{y}, \mathbf{y}') \coloneqq \begin{cases} 1, & \mathbf{y} = \mathbf{y}', \\ \mathrm{clip}_{[0,1]}(\widetilde{u}(\mathbf{y}, \mathbf{y}')), & \mathbf{y} \neq \mathbf{y}'. \end{cases}$$

This explicit diagonal assignment ensures $u(\mathbf{y}, \mathbf{y}) = 1$, including for an empty response or a response too short to contain all n-grams used by a metric. For the six utilities in Table 8, $\widetilde{u}$ is defined as follows.

- **Exact match:** $\widetilde{u}(\mathbf{y}, \mathbf{y}') = \mathbf{1}\{\mathbf{y} = \mathbf{y}'\}$.
- **BLEU:** sentence-level BLEU is used with the scorer's fixed tokenization and smoothing configuration. A score returned on the $[0, 100]$ scale is divided by 100, and a score already on $[0, 1]$ is used unchanged.
- **Character 3-gram F1 and token F1:** precision and recall are computed from multiset overlap of character 3-grams or scorer-tokenized tokens, respectively, and $\widetilde{u} = 2PR/(P+R)$. We set $\widetilde{u} = 0$ when $P + R = 0$, and the identical-response case is still assigned one by the definition above.
- **ROUGE-L:** the ROUGE-L F-measure is used. A value returned on the $[0, 100]$ scale is divided by 100, and a value already on $[0, 1]$ is used unchanged.
- **Embedding similarity:** if $c(\mathbf{y}, \mathbf{y}') \in [-1, 1]$ is the cosine similarity between the two response embeddings, then $\widetilde{u}(\mathbf{y}, \mathbf{y}') = \big(1 + c(\mathbf{y}, \mathbf{y}')\big)/2$.

The final clipping handles only numerical roundoff or an out-of-range scorer output. It does not otherwise change metrics already defined on $[0, 1]$. These definitions make the task utility assumptions used in Equation 5 and Appendix A.3.3 explicit for every task utility evaluated in the paper.

A.3 Theoretical Analysis of Trajectory-MBR

We now present the formal analysis behind the results of Section 4.2. The first concerns the trajectory score and shows that the window weights approximate likelihoods under $\bar{\mathcal{A}}$ restricted and normalized to the indexed window entries. The second is a deterministic conditional regret bound comparing the selected response with the best response available in a fixed window. The third, Proposition 1, is an offline sufficient condition for improvement over full decoding. We state the shared assumptions once in Appendix A.3.1, and each theorem is restated immediately before its proof so that this appendix can be read on its own.

A.3.1 Setting and Assumptions

Fix a prompt $\mathbf{c}$, a deployed sampler $\mathcal{A}$, and its stochastic ancestral counterpart $\bar{\mathcal{A}}$ as defined in Section 4.2. Under the conditions of Appendix A.1, their shared reveal rule is a deterministic function of the current state, every position is revealed exactly once, and $\bar{\mathcal{A}}$ draws values revealed at a transition independently from the sampler-side categorical distributions of Equation 2. For every candidate considered below, every score factor and every eventual-reveal factor in the induced response factorization is strictly positive. We write $p_{\theta,\bar{\mathcal{A}}}(\mathbf{y} \mid \mathbf{c})$ for the response distribution induced by $\bar{\mathcal{A}}$.

For each masked position $i \in \mathcal{M}_k$, $\tau_{k,i}(\mathbf{y}_k)$ is the checkpoint at which position i would be revealed if $\bar{\mathcal{A}}$ were continued consistently with $\mathbf{y}_k$, and the state $\widetilde{\mathbf{x}}_{k,i}(\mathbf{y}_k)$ of Theorem 1 is the corresponding reveal state. Its categorical factor is $p^i_{\theta,\tau_{k,i}(\mathbf{y}_k)}(y^i_k \mid \widetilde{\mathbf{x}}_{k,i}(\mathbf{y}_k), \mathbf{c})$. Equation 8 defines $\gamma_{k,i}$ as the discrepancy between the candidate-token log probability at checkpoint k and at this eventual reveal checkpoint and state, and $\varepsilon^{\mathrm{traj}}_k$ sums these discrepancies over the masked positions. Finally, $\overline{p}_{k+j}$ in Equation 9 is the likelihood under $\bar{\mathcal{A}}$ normalized over the indexed entries of the window Y_k.

Remark. The quantity $\varepsilon^{\mathrm{traj}}_k$ is a counterfactual analysis quantity rather than a tunable constant or an online stopping signal. At the terminal checkpoint, $\mathcal{M}_0 = \varnothing$ and hence $\varepsilon^{\mathrm{traj}}_0 = 0$. At intermediate checkpoints, however, its value depends jointly on the remaining masked positions and their individual current-to-reveal discrepancies, so monotonic decrease is not guaranteed. Moreover, $\overline{\varepsilon}_k$ is a uniform upper bound over the indexed entries of the current window. Even when it is chosen as the tight window maximum, it need not vanish for the terminal window at $k = 0$.

A.3.2 Trajectory Score Approximation

Theorem 1 (Trajectory Score Approximation). *Under the setting of Appendix A.3.1, the trajectory score of Equation 2 satisfies* $\left|\log s_k - \log p_{\theta,\bar{\mathcal{A}}}(\mathbf{y}_k \mid \mathbf{c})\right| \le \varepsilon^{\mathrm{traj}}_k$ *at every checkpoint k. If in addition* $\max_{0\le j<K} \varepsilon^{\mathrm{traj}}_{k+j} \le \overline{\varepsilon}_k$*, the window weights of Equation 4 satisfy the ratio bound of Equation 9.*

Proof. Fix a checkpoint k and its candidate $\mathbf{y}_k$. Because the reveal rule of $\bar{\mathcal{A}}$ is a deterministic function of the state and every position is revealed exactly once, the response $\mathbf{y}_k$ determines a unique reveal trajectory under $\bar{\mathcal{A}}$. The portion of that trajectory up to $\mathbf{x}_k$ is already observed, since $y^i_k = x^i_k$ for every $i \notin \mathcal{M}_k$. Conditional independence of the values revealed at each transition therefore gives

$$p_{\theta,\bar{\mathcal{A}}}(\mathbf{y}_k \mid \mathbf{c}) = \underbrace{\prod_{r=k+1}^{T} \prod_{i\in\mathcal{U}_r} p^i_{\theta,r}\big(x^i_{r-1}\big)}_{\text{observed prefix}} \underbrace{\prod_{i\in\mathcal{M}_k} p^i_{\theta,\tau_{k,i}(\mathbf{y}_k)}\big(y^i_k \mid \widetilde{\mathbf{x}}_{k,i}(\mathbf{y}_k), \mathbf{c}\big)}_{\text{remaining reveals}}. \tag{13}$$

The first product collects the probabilities of the token values already revealed along the observed prefix. The second collects the probabilities that $\bar{\mathcal{A}}$ assigns to the remaining candidate tokens at their eventual reveal states. By Equation 2, the score s_k has exactly the same prefix factors as Equation 13 but evaluates every remaining candidate token at checkpoint k instead. Taking logarithms of the two expressions cancels every prefix term and leaves

$$\log s_k - \log p_{\theta,\bar{\mathcal{A}}}(\mathbf{y}_k \mid \mathbf{c}) = \sum_{i\in\mathcal{M}_k} \Big[\log p^i_{\theta,k}\big(y^i_k \mid \mathbf{x}_k, \mathbf{c}\big) - \log p^i_{\theta,\tau_{k,i}(\mathbf{y}_k)}\big(y^i_k \mid \widetilde{\mathbf{x}}_{k,i}(\mathbf{y}_k), \mathbf{c}\big)\Big].$$

Applying the triangle inequality and the definition of $\gamma_{k,i}$ in Equation 8 gives the first claim, with the convention that the empty sum vanishes. At $k = 0$, the set $\mathcal{M}_0$ is empty and s_0 consists of observed reveal factors alone, so the first claim holds with $\varepsilon^{\mathrm{traj}}_0 = 0$.

For the second claim, write $p_j := p_{\theta,\bar{\mathcal{A}}}(\mathbf{y}_{k+j} \mid \mathbf{c})$. The first claim together with the uniform bound $\varepsilon^{\mathrm{traj}}_{k+j} \le \overline{\varepsilon}_k$ gives the entrywise sandwich

$$e^{-\overline{\varepsilon}_k} p_j \le s_{k+j} \le e^{\overline{\varepsilon}_k} p_j, \qquad j = 0, \dots, K-1, \tag{14}$$

and summing Equation 14 over the K entries of the window gives the same sandwich for the normalizer,

$$e^{-\overline{\varepsilon}_k} \sum_{r=0}^{K-1} p_r \le \sum_{r=0}^{K-1} s_{k+r} \le e^{\overline{\varepsilon}_k} \sum_{r=0}^{K-1} p_r. \tag{15}$$

By Equation 4 and the definition of $\overline{p}_{k+j}$, the ratio of the two normalized weights factors as

$$\frac{w_{k+j}}{\overline{p}_{k+j}} = \frac{s_{k+j}}{p_j} \cdot \frac{\sum_{r=0}^{K-1} p_r}{\sum_{r=0}^{K-1} s_{k+r}}.$$

The first factor lies in $[e^{-\overline{\varepsilon}_k}, e^{\overline{\varepsilon}_k}]$ by Equation 14, and the second lies in the same interval by Equation 15. Multiplying the two bounds proves Equation 9. □

Remark. The argument is entrywise throughout, so it does not require the entries of Y_k to be distinct. When a response occurs at several checkpoints, each occurrence is bounded separately and the bound on the total mass of that response follows by summation.

Remark. The cancellation of the prefix factors is what makes the bound useful in practice. The two quantities being compared share every factor associated with an already revealed token, so the error is confined to the positions that the candidate completes at the current checkpoint. In particular the bound does not accumulate over the length of the trajectory. It is controlled only by the number of positions still masked and by how much the denoiser changes its prediction at those positions as the remaining context becomes visible.

A.3.3 From Score Approximation to Utility Approximation

The conditional decision-regret result is stated in terms of an approximation error η_k between the target expected utility and the trajectory-score-weighted utility. We now express η_k explicitly in terms of the quantities bounded above. Recall from Section 4.2 the trajectory-score-weighted utility $\widehat{U}_k$, its counterpart U_k^{win} under the likelihood of $\bar{\mathcal{A}}$ normalized over the current window, and the target window mismatch δ_k^{win}.

Let Q_k and W_k be the distributions over the indexed window entries with masses $\overline{p}_{k+j}$ and w_{k+j}, and let $\mathrm{TV}(Q_k, W_k) \coloneqq \frac{1}{2}\sum_{j=0}^{K-1} |\overline{p}_{k+j} - w_{k+j}|$ be their total variation distance. Since u takes values in $[0,1]$, the two weighted utilities of any candidate $\mathbf{y}$ differ by at most this distance,

$$\left|U_k^{\mathrm{win}}(\mathbf{y}) - \widehat{U}_k(\mathbf{y})\right| \le \mathrm{TV}(Q_k, W_k). \tag{16}$$

It remains to convert the ratio bound of Equation 9 into a bound on this distance. For two distributions whose likelihood ratio lies in $[e^{-a}, e^{a}]$, the total variation distance is at most

$$\frac{e^a - 1}{e^a + 1} = \tanh\Big(\frac{a}{2}\Big). \tag{17}$$

To see this, let A collect the entries on which W_k has at least as much mass as Q_k and write $q = Q_k(A)$. The excess mass on A is at most $(e^a - 1)q$ by the upper ratio bound, and the deficit off A is at most $(1 - e^{-a})(1 - q)$ by the lower bound, so

$$\mathrm{TV}(Q_k, W_k) \le \min\big\{(e^a - 1)q,\ (1 - e^{-a})(1 - q)\big\}.$$

The right hand side is largest where its two arguments coincide, which yields Equation 17. Setting $a = 2\overline{\varepsilon}_k$ therefore gives

$$\mathrm{TV}(Q_k, W_k) \le \tanh(\overline{\varepsilon}_k). \tag{18}$$

Combining the definition of δ_k^{win} with Equations 16 and 18 through the triangle inequality shows that every window entry satisfies

$$\left|U_{\mathrm{h}}(\mathbf{y}_{k+\ell}) - \widehat{U}_k(\mathbf{y}_{k+\ell})\right| \le \underbrace{\delta_k^{\mathrm{win}}}_{\text{target window mismatch}} + \underbrace{\tanh(\overline{\varepsilon}_k)}_{\text{score approximation}}, \tag{19}$$

which is the hypothesis of Theorem 2 with $\eta_k = \delta_k^{\mathrm{win}} + \tanh(\overline{\varepsilon}_k)$.

Remark. The two terms of Equation 19 have different origins.

- The mismatch term δ_k^{win} compares the target expected utility with the utility induced by the likelihood of $\bar{\mathcal{A}}$ restricted and normalized to the current window. It is not controlled by the score-approximation theorem and may be large. The present analysis does not provide a concentration guarantee for this term.
- The approximation term $\tanh(\overline{\varepsilon}_k)$ comes from Equation 9 and reflects how far the trajectory score is from the likelihood under $\bar{\mathcal{A}}$. The bound term is zero when $\overline{\varepsilon}_k = 0$. For the tight window maximum, this means that every $\varepsilon_{k+j}^{\mathrm{traj}}$ in the window is zero. It need not decrease monotonically as decoding progresses.

A.3.4 CONDITIONAL DECISION REGRET

Theorem 2 (Conditional Decision Regret for Trajectory-MBR). *Let P_{h} be a target reference distribution with expected utility U_{h}, and let $\mathbf{y}_{k+\ell_k^{\mathrm{h}}}$ be the entry of Y_k that maximizes U_{h}. If $\max_{0\le\ell<K}\left|U_{\mathrm{h}}(\mathbf{y}_{k+\ell}) - \widehat{U}_k(\mathbf{y}_{k+\ell})\right| \le \eta_k$, then the response $\mathbf{y}_k^\star$ selected by Equation 6 satisfies the regret bound of Equation 10. Under the setting of Appendix A.3.1, the hypothesis holds with $\eta_k = \delta_k^{\mathrm{win}} + \tanh(\overline{\varepsilon}_k)$.*

Proof. Since $\widehat{U}_k(\mathbf{y}_{k+\ell}) = 1 - R_{k,\ell}$, minimizing $R_{k,\ell}$ in Equation 6 is the same as maximizing $\widehat{U}_k$ over the window. The selected entry is therefore optimal for the trajectory-score-weighted utility,

$$\widehat{U}_k(\mathbf{y}_k^\star) \ge \widehat{U}_k(\mathbf{y}_{k+\ell_k^{\mathrm{h}}}). \tag{20}$$

Adding and subtracting the trajectory-score-weighted utilities of the two entries decomposes the target utility gap into three parts,

$$U_{\mathrm{h}}(\mathbf{y}_{k+\ell_k^{\mathrm{h}}}) - U_{\mathrm{h}}(\mathbf{y}_k^\star) = \underbrace{\left[U_{\mathrm{h}}(\mathbf{y}_{k+\ell_k^{\mathrm{h}}}) - \widehat{U}_k(\mathbf{y}_{k+\ell_k^{\mathrm{h}}})\right]}_{\le \eta_k} + \underbrace{\left[\widehat{U}_k(\mathbf{y}_{k+\ell_k^{\mathrm{h}}}) - \widehat{U}_k(\mathbf{y}_k^\star)\right]}_{\le 0}$$
$$+ \underbrace{\left[\widehat{U}_k(\mathbf{y}_k^\star) - U_{\mathrm{h}}(\mathbf{y}_k^\star)\right]}_{\le \eta_k}.$$

The middle bracket is nonpositive by Equation 20, and the two outer brackets are each at most η_k by hypothesis. Summing the three bounds proves Equation 10. The explicit form of η_k is established in Equation 19. □

Remark. The factor of two arises because the utility-approximation error is incurred once at the target-optimal entry and once at the entry selected by the trajectory-score-weighted utility. The bound is stated relative to the best entry of the current window rather than to the terminal output. The separate offline comparison with the terminal output is addressed next.

A.3.5 OFFLINE SUFFICIENT CONDITION FOR IMPROVEMENT OVER FULL DECODING

Let $\mathbf{y}_{\mathrm{full}} \coloneqq \mathbf{y}_0 = \mathbf{x}_0$ denote the response returned by full decoding. Its trajectory-score-weighted utility against the reference window Y_k is $\widehat{U}_k(\mathbf{y}_{\mathrm{full}}) = \sum_{j=0}^{K-1} u(\mathbf{y}_{\mathrm{full}}, \mathbf{y}_{k+j})\, w_{k+j}$, and we call

$$m_k \coloneqq \widehat{U}_k(\mathbf{y}_k^\star) - \widehat{U}_k(\mathbf{y}_{\mathrm{full}}) \tag{21}$$

the offline trajectory-score margin of the decision over full decoding.

To keep the comparison mismatch distinct from the window-only mismatch in Theorem 2, define

$$\delta_k^{\mathrm{full}} \coloneqq \max\left\{\delta_k^{\mathrm{win}}, \left|U_{\mathrm{h}}(\mathbf{y}_{\mathrm{full}}) - U_k^{\mathrm{win}}(\mathbf{y}_{\mathrm{full}})\right|\right\}, \qquad \eta_k^{\mathrm{full}} \coloneqq \delta_k^{\mathrm{full}} + \tanh(\overline{\varepsilon}_k). \tag{22}$$

Thus δ_k^{full} is the maximum target mismatch over the indexed entries of Y_k together with $\mathbf{y}_{\mathrm{full}}$, whereas δ_k^{win} concerns the window entries only. By the same triangle-inequality and total-variation argument as in Appendix A.3.3,

$$\left|U_{\mathrm{h}}(\mathbf{y}) - \widehat{U}_k(\mathbf{y})\right| \le \eta_k^{\mathrm{full}} \quad \text{for every } \mathbf{y} \in \{\mathbf{y}_k, \dots, \mathbf{y}_{k+K-1}, \mathbf{y}_{\mathrm{full}}\}. \tag{23}$$

Proposition 1 (Offline Sufficient Condition for Improvement over Full Decoding)**.** *Under the setting of Appendix A.3.1,*

$$U_{\mathrm{h}}(\mathbf{y}_k^\star) - U_{\mathrm{h}}(\mathbf{y}_{\mathrm{full}}) \ge m_k - 2\eta_k^{\mathrm{full}}.$$

Consequently, the offline condition

$$m_k > 2\left[\delta_k^{\mathrm{full}} + \tanh(\overline{\varepsilon}_k)\right] \tag{24}$$

is sufficient for Trajectory-MBR to attain strictly higher target expected utility than full decoding.

Proof. Equation 23, applied at $\mathbf{y}_k^\star$ from below and at $\mathbf{y}_{\mathrm{full}}$ from above, gives

$$U_{\mathrm{h}}(\mathbf{y}_k^\star) \geq \widehat{U}_k(\mathbf{y}_k^\star) - \eta_k^{\mathrm{full}}, \qquad U_{\mathrm{h}}(\mathbf{y}_{\mathrm{full}}) \leq \widehat{U}_k(\mathbf{y}_{\mathrm{full}}) + \eta_k^{\mathrm{full}}.$$

Subtracting the second inequality from the first and applying Equation 21 yields the claim. Under Equation 24, the right hand side is strictly positive, and therefore $U_{\mathrm{h}}(\mathbf{y}_k^\star) > U_{\mathrm{h}}(\mathbf{y}_{\mathrm{full}})$. □

Remark. The proposition is an analysis condition rather than an online test. The quantities m_k and δ_k^{full} depend on $\mathbf{y}_{\mathrm{full}}$, which is not available at the moment the method stops early, and δ_k^{full} additionally depends on the target expected utility U_{h}. Hence Equation 24 cannot be evaluated by the stopping rule and does not certify the risk threshold of Equation 7. It is only an offline sufficient condition. If the inequality fails, the proposition does not imply that full decoding has higher target expected utility.

A.4 Controlled Synthetic Evaluation of Trajectory-Score Approximation

Purpose and model. This experiment examines the magnitude of trajectory-score approximation error and the tightness of its theoretical bounds under the assumptions of Theorem 1. We construct a finite binary model whose joint distribution and single-token conditional probabilities are exactly computable.

For responses $\mathbf{y} \in \{0,1\}^6$, let $z_i = 2y^i - 1$ and define

$$q_{\alpha,c}(\mathbf{y}) = \frac{1}{Z_{\alpha,c}} \exp\left(\sum_{i=1}^{6} h_{c,i} z_i + \alpha \sum_{i=1}^{5} J_{c,i} z_i z_{i+1}\right), \tag{25}$$

where $Z_{\alpha,c}$ is the normalizing constant, $h_{c,i}$ controls individual token preferences, and $J_{c,i}$ determines interactions between adjacent tokens. We generate 256 contexts using independent draws

$$h_{c,i} \sim \mathrm{Uniform}(-0.5, 0.5), \qquad J_{c,i} \sim \mathrm{Uniform}(-1, 1), \tag{26}$$

with random seed 20260924. The same fields and couplings are reused across $\alpha \in \{0, 0.05, 0.1, 0.25, 0.5, 1, 2\}$. Tokens are independent at $\alpha = 0$, while larger α strengthens the interaction terms. All 64 responses are enumerated to compute the normalizer and exact single-token conditionals. The predictor returns these exact conditionals.

Decoding and exact likelihood. The deployed sampler starts with six masked positions and reveals one position per transition, giving $T = L = 6$. At each transition, it chooses the masked position with the largest maximum token probability and assigns its most probable token. Position and token ties are resolved by index.

The stochastic ancestral counterpart uses the same state-dependent position selection rule but samples the revealed token from its exact conditional distribution. Because each transition reveals a single token, the chain rule along the unique response-consistent reveal trajectory gives

$$p_{\theta,\bar{\mathcal{A}}}(\mathbf{y} \mid c) = q_{\alpha,c}(\mathbf{y}). \tag{27}$$

This equality relies on the exact conditionals and single-token reveals used in this experiment.

For each intermediate candidate, we recompute its likelihood by replaying the remaining trajectory while forcing the candidate's token values. Conditional probabilities and reveal positions are recomputed at the resulting counterfactual states. The likelihood, including the observed prefix factors, is checked against direct lookup in the enumerated joint distribution. We additionally replay every response from the fully masked state to verify the induced distribution.

Window weights and response selection. We use a window size of $K = 4$. All seven checkpoint entries, including the terminal entry, are retained, yielding four eligible windows per context and interaction strength. Duplicate responses remain separate checkpoint entries.

For each indexed window, we compare the normalized trajectory-score weights W_k with exact likelihood weights Q_k, defined by

$$W_k(j) = \frac{s_{k+j}}{\sum_{r=0}^{K-1} s_{k+r}}, \tag{28}$$

$$Q_k(j) = \frac{p_{k+j}}{\sum_{r=0}^{K-1} p_{k+r}}, \qquad j = 0, \ldots, K-1, \tag{29}$$

Table 4: **Mean window-weight TV remains below 0.10 across interaction strengths.** Errors and bounds are strength-specific means. Selection agreement is computed only over the reported distinct-response windows. Values shown as approximately zero are at numerical precision.

α	Log-score error	Score bound	TV	TV bound	Distinct windows	Selection agreement
0	≈ 0	≈ 0	≈ 0	≈ 0	0	N/A
0.05	0.0167	0.0261	0.0048	0.0385	100	86.0%
0.10	0.0336	0.0526	0.0095	0.0774	158	81.6%
0.25	0.0863	0.1366	0.0261	0.1956	363	89.5%
0.50	0.1746	0.2780	0.0517	0.3603	455	89.2%
1.00	0.3171	0.4948	0.0838	0.5337	410	90.0%
2.00	0.4453	0.5748	0.0989	0.5490	207	88.9%

where $p_{k+j} = p_{\theta,\bar{A}}(\mathbf{y}_{k+j} \mid c)$. Both distributions are normalized over the same checkpoint entries, including duplicates. Their discrepancy is measured by total variation, defined as

$$\mathrm{TV}(W_k, Q_k) = \frac{1}{2} \sum_{j=0}^{K-1} |W_k(j) - Q_k(j)| \,. \tag{30}$$

We use positional Hamming similarity as the utility, defined by

$$u(\mathbf{y}, \mathbf{y}') = \frac{1}{6} \sum_{i=1}^{6} \mathbf{1}\{y^i = y'^i\}. \tag{31}$$

For the response-selection comparison, only the reference weights are replaced. The candidates, utility function, and tie-breaking rule remain fixed. When multiple entries attain the minimum risk, the most recent entry is selected.

Selection agreement means returning the same response, regardless of its checkpoint index. We compute agreement only for windows containing at least two distinct responses, since windows with identical responses yield a trivial comparison. The analysis uses all eligible windows independently of the stopping threshold. No gold-reference distribution is introduced.

Aggregation and confidence intervals. Absolute log-score error and the trajectory discrepancy bound $\varepsilon_k^{\mathrm{traj}}$ are averaged over the six nonterminal candidates in each context. Window TV and its theoretical upper bound $\tanh(\bar{\varepsilon}_k)$ are averaged over all four eligible windows, where $\bar{\varepsilon}_k$ is the maximum candidate discrepancy bound within the window. Each interaction strength therefore contributes 1,536 nonterminal candidates and 1,024 windows.

Pointwise 95% confidence intervals are computed using 5,000 percentile bootstrap resamples of the 256 contexts with seed 20260924. All candidates and windows belonging to a context are resampled together, and the same resamples are used across interaction strengths. For selection agreement, each bootstrap replicate divides the total number of matching selections by the total number of distinct-response windows in that replicate. This procedure preserves dependence among checkpoints and overlapping windows from the same trajectory.

Results. Table 4 reports the full sweep. The independent control has maximum absolute log-score error at most 1.78×10^{-15}, confirming agreement with exact likelihood up to numerical precision. It contains no distinct-response windows, so its selection agreement is undefined and omitted from panel c of Figure 2.

Across the dependent settings, mean log-score error and mean window-weight TV increase with interaction strength in this sweep. Mean TV ranges from 0.0048 to 0.0989, while the corresponding mean theoretical upper bounds range from 0.0385 to 0.5490. For example, at $\alpha = 2$, the observed mean TV is 0.0989 and the mean bound is 0.5490. The bounds are therefore conservative relative to the observed errors.

Replacing trajectory-score weights with exact likelihood weights preserves the selected response in 81.6% to 90.0% of distinct-response windows. The number and composition of these windows vary

across interaction strengths, as shown in Table 4. The agreement curve therefore does not compare a fixed subset of windows across all strengths.

Numerical verification and scope. Across 114,688 full-response replays, the maximum absolute difference between the induced and prescribed probabilities is at most 2.23×10^{-16}. Across 12,544 candidate entries, the maximum difference between replay-based and directly enumerated log-likelihoods is at most 8.89×10^{-16}. No trajectory-error or TV inequality violation exceeds the numerical tolerance of 10^{-10}.

B Experimental Setup

This appendix documents the full experimental configuration behind Section 5. All three evaluation protocols are inherited from prior work rather than designed by us, so that each comparison is made under exactly the setting in which the corresponding baselines were reported. Appendix B.1 describes the sequence-level protocol of Focus-dLLM (Long et al., 2026), Appendix B.2 the multimodal protocol of MPD-PAC (Hong et al., 2026), and Appendix B.3 the token-level protocol of JoT (Kohut et al., 2026). The three subsections are organized identically. Appendix B.4 states how efficiency is measured under each protocol.

B.1 Language dataset

This setting applies the sequence-level decision rule of Section 4.1 to long-context language generation and follows the protocol of Focus-dLLM (Long et al., 2026) throughout, so that Table 1 compares against the numbers reported there under an identical configuration.

Benchmarks. We evaluate on LongBench (Bai et al., 2024), a bilingual multitask benchmark for long-context understanding, using the 13 tasks reported by Long et al. (2026). These are Qasper and MultiFieldQA-en for single-document QA, HotpotQA, 2WikiMultihopQA and MuSiQue for multi-document QA, GovReport and QMSum for summarization, TREC, TriviaQA and LSHT for few-shot learning, PassageRetrieval-en for synthetic retrieval, and LCC and RepoBench-P for code completion. The two remaining tasks of the protocol, MultiNews and SAMSum, are not reported by Long et al. (2026) and are likewise excluded here, so that every baseline column is populated under an identical task set. Per-task input length, generation length, denoising-step budget, language and evaluation sample count are taken unchanged from Long et al. (2026) and listed in Table 5.

Evaluation metrics. We follow the metric assignment of Long et al. (2026). Token-level F1 is used for Qasper, MultiFieldQA-en, HotpotQA, 2WikiMultihopQA, MuSiQue and TriviaQA, ROUGE-L for GovReport and QMSum, accuracy for TREC, LSHT and PassageRetrieval-en, and edit similarity for LCC and RepoBench-P. The average score is the unweighted macro-average over the 13 tasks. All runs are scored with OpenCompass (Cao et al., 2026), the harness used by Long et al. (2026). Gold references enter only at this final evaluation. The task utility $u(\mathbf{y}, \mathbf{y}')$ of Equation 5 is computed between trajectory candidates alone, so the stopping decision remains available at inference time.

Backbones and decoding configuration. We use two masked diffusion language models. Ultra-LLaDA (He et al., 2026) is obtained by fine-tuning LLaDA-8B (Nie et al., 2025) for long-context capability, and the second backbone is Dream-7B-Instruct (Ye et al., 2025), which is adapted from a pretrained autoregressive model. Both retain their original decoding strategies. Generation follows the semi-autoregressive remasking paradigm, in which unmasking within a block is non-autoregressive and blocks are processed from left to right, and the transfer scheduler selects the positions to unmask at each step by confidence. The block length is 32 for every method, as in Long et al. (2026), and input contexts are truncated to a maximum of 16K tokens with the drop-middle strategy, matching the evaluation setting of UltraLLaDA (He et al., 2026). Per-task generation length and step budget are those of Table 5.

Baselines. We compare against full decoding and four training-free acceleration frameworks. Every baseline number in Table 1 marked with † is the one reported by Long et al. (2026) under the configuration above, so no baseline hyperparameter is retuned here and we describe only what each method does.

Table 5: **LongBench evaluation across 13 tasks and six categories.** Generation lengths, denoising-step budgets, and sample counts follow the Focus-dLLM protocol reported by (Long et al., 2026). The table specifies each task's metric, language, and decoding configuration.

Task	Category	Metric	Language	Gen. Len.	Steps	#Samples
Qasper	Single-Doc. QA	F1	EN	32	32	200
MultiFieldQA-en	Single-Doc. QA	F1	EN	64	64	150
HotpotQA	Multi-Doc. QA	F1	EN	32	32	200
2WikiMultihopQA	Multi-Doc. QA	F1	EN	32	32	200
MuSiQue	Multi-Doc. QA	F1	EN	32	32	200
GovReport	Summarization	ROUGE-L	EN	512	512	200
QMSum	Summarization	ROUGE-L	EN	512	512	200
TREC	Few-shot Learning	Accuracy	EN	64	64	200
TriviaQA	Few-shot Learning	F1	EN	32	32	200
LSHT	Few-shot Learning	Accuracy	ZH	64	64	200
PassageRetrieval-en	Synthetic	Accuracy	EN	32	32	200
LCC	Code	Edit Sim.	Python/C#/Java	64	64	500
RepoBench-P	Code	Edit Sim.	Python/Java	64	64	500

Full decoding is the unmodified sampler of each backbone, run to the terminal checkpoint, and is the reference that every acceleration method approximates. Fast-dLLM (Wu et al., 2026) adapts key and value caching to bidirectional attention by refreshing the cache block-wise and reusing it elsewhere, and it further shortens the trajectory by committing several confident positions in the same transition. Sparse-dLLM (Song et al., 2026) exploits the temporal stability of token saliency to evict unimportant cache entries as decoding proceeds, whereas SparseD (Wang et al., 2026b) pre-computes head-specific sparse attention patterns once and reuses them across later steps, keeping attention dense during the early steps that most affect quality. Focus-dLLM (Long et al., 2026) predicts which positions will be unmasked next from the confidence of the previous step, then computes attention only over the resulting queries and the context blocks those queries select, while retaining attention sinks.

All four aim to reproduce the terminal state $\mathbf{x}_0$ of the base sampler at a lower cost, by making each transition cheaper or, in the case of Fast-dLLM, by committing several positions per transition. None of them retains an earlier checkpoint as a candidate output.

Trajectory-MBR configuration. Trajectory-MBR is applied on top of the base sampler of each backbone and leaves the reveal rule, the block schedule and the denoiser output untouched, so its trajectory coincides with full decoding up to the stopping point. It only decides when to stop and which checkpoint to return. The rule exposes three quantities, namely the candidate-window size K of Equation 3, the risk threshold ϵ of Equation 7, and the task utility $u(\mathbf{y}, \mathbf{y}')$ of Equation 5. Following the discussion in Section 4.1, the three are selected jointly per task, and the grid searched over each of them is listed in Table 8. The six candidate utilities are the ones whose individual effect is reported in Table 3. Utilities are normalized to $[0, 1]$ with $u(\mathbf{y}, \mathbf{y}) = 1$, and a raw metric that does not already satisfy this is mapped into the range by the fixed rule of Appendix A.2. All scores are accumulated in log space and the window weights of Equation 4 are computed with log-sum-exp.

B.2 Multimodal dataset

This setting applies the same sequence-level decision rule to multimodal generation. We follow established evaluation protocol of Hong et al. (2026), with modifications documented in the corresponding subsections.

Benchmarks. We evaluate on the nine benchmarks used by Hong et al. (2026). These are MME (Fu et al., 2025), MMBench (Liu et al., 2025b) and MMMU (Yue et al., 2024) for general multimodal understanding, RefCOCOg (Yu et al., 2016), Ferret (You et al., 2024) and GQA (Hudson & Manning, 2019) for visual grounding, and LLaVA-Bench (Liu et al., 2023), DetailCaps (Dong et al., 2024) and MIA (Qian et al., 2025) for long-form generation. Evaluation splits and the evaluated instances of each benchmark are taken unchanged from Hong et al. (2026), with sample counts listed in Table 6.

Table 6: **Multimodal evaluation across nine benchmarks and three categories.** Benchmarks cover general understanding, visual grounding, and long-form generation. Settings follow the protocol reported by Hong et al. (2026). Denoising-step budgets are increased to 16 for MME, MMBench, MMMU, RefCOCOg, and GQA.

Benchmark	Category	Metric	Language	Gen. Len.	Steps	#Samples
MME	General	Sum score	EN	2	16	2,374
MMBench	General	Accuracy	EN	2	16	4,329
MMMU	General	Accuracy	EN	2	16	900
RefCOCOg	Visual Grounding	CIDEr	EN	8	16	7,573
Ferret	Visual Grounding	Model-judged score	EN	96	48	100
GQA	Visual Grounding	Accuracy	EN	2	16	500
LLaVA-Bench	Long-form Generation	Model-judged score	EN	128	64	60
DetailCaps	Long-form Generation	CAPTURE	EN	256	128	100
MIA	Long-form Generation	Model-judged score	EN	64	32	400

Following that protocol we use the lite split of GQA and a fixed random sample of 100 instances for Ferret and DetailCaps, which keeps evaluation costs tractable.

Evaluation metrics. We follow the metric assignment of Hong et al. (2026). The summed perception and cognition score is used for MME, accuracy for MMBench, MMMU, and GQA, CIDEr for RefCOCOg, the CAPTURE score for DetailCaps, and a model-judged score for Ferret, LLaVA-Bench, and MIA. For LLaVA-Bench and MIA, we use the judges specified by Hong et al. (2026): `gpt-4-0613` and `gpt-4o-2024-08-06`, respectively. The original Ferret judge, `gpt-4-0314`, has been retired, so we re-evaluate the entire Ferret column with `gpt-4-0613`. We apply the same judge and prompt to full decoding, Fast-dLLM, Sparse-dLLM, SparseD, and Trajectory-MBR. Ferret comparisons therefore use these re-evaluated scores rather than the values reported by Hong et al. (2026). The average score is the unweighted macro-average over the nine benchmarks, with MME included on its original raw scale. Our runs are scored with LMMs-Eval (Zhang et al., 2025) using the default prompt of each benchmark, following the evaluation harness of Hong et al. (2026). Gold references enter only at this final evaluation. The task utility $u(\mathbf{y}, \mathbf{y}')$ of Equation 5 is computed between trajectory candidates alone, so the stopping decision remains available at inference time.

Backbones and decoding configuration. We use two large diffusion vision-language models. LLaDA-V (You et al., 2026) pairs a SigLIP2-SO400M vision encoder with an LLaDA-8B-Instruct language tower and is trained by visual instruction tuning under fully bidirectional attention. LaViDa (Li et al., 2025b) pairs a SigLIP-400M encoder with the same LLaDA-8B-Instruct language backbone, and we use the LaViDa-L variant so that the two models share a language tower and the comparison is controlled. Both retain their original decoding strategies. Generation follows the standard full-suffix iterative unmasking setup of these models, in which every remaining masked position is updated in parallel at each step without semi-autoregressive blocking, and LaViDa additionally caches the key and value states of the visual and prompt prefix and replaces the linear unmasking schedule with a shifting one. Per-task generation length and step budget are those of Table 6. We increase the step budget to 16 for MME, MMBench, MMMU, RefCOCOg, and GQA to match our task settings, while retaining the original budgets for the remaining benchmarks. Generation length ranges from a two-token answer on the general benchmarks to 256 tokens on DetailCaps.

Baselines. We compare Trajectory-MBR against full decoding, Fast-dLLM (Wu et al., 2026), Sparse-dLLM (Song et al., 2026), and SparseD (Wang et al., 2026b). These methods are described in Appendix B.1. We adopt the baseline hyperparameter settings reported by Long et al. (2026), using the multimodal backbones and per-task generation settings specified above. Focus-dLLM is excluded from Table 2 because its implementation was unavailable at the time of our experiments.

Trajectory-MBR configuration. Trajectory-MBR is applied on top of the base sampler of each backbone and leaves the unmasking schedule and the denoiser output untouched, exactly as in Appendix B.1.

Table 7: **Token-level evaluation across four benchmarks.** The benchmarks cover mathematical reasoning, knowledge, commonsense, and code generation. All use zero-shot prompting and full evaluation sets, with generation lengths, denoising-step budgets, and sample counts following (Kohut et al., 2026).

Benchmark	Category	Metric	Language	Gen. Len.	Steps	#Samples
GSM8K	Math Reasoning	Exact match	EN	256	256	1,319
MMLU	Multitask Knowledge	Gen. choice Acc.	EN	3	3	14,042
HellaSwag	Commonsense Reasoning	Gen. choice Acc.	EN	5	5	10,042
HumanEval	Code Generation	pass@1	Python	512	512	164

Table 8: **Hyperparameter search spaces across backbones.** Window sizes range from 2 to $T+1$, where T is the full-decoding step budget and the additional checkpoint accounts for the initial state. Sequence-level configurations are selected per task. The token-level extension additionally sweeps the JoT gate threshold to trace the trade-off between quality and speed.

Method	Hyperparameter	Search grid
Trajectory-MBR	Window size K	$\{2, \ldots, T+1\}$
	Risk threshold ϵ	$\{0, 0.01, 0.05, 0.10, 0.50, 1.00\}$
	Task utility $u(\mathbf{y}, \mathbf{y}')$	Exact match, BLEU, Character 3-gram F1, Token F1, ROUGE-L, Embedding similarity
Trajectory-Token-MBR	Window size K	$\{2, \ldots, T+1\}$
	Risk threshold ϵ	$\{0, 0.01, 0.05, 0.10, 0.50, 1.00\}$
	JoT gate threshold	30 to 300
	Task utility $u(\mathbf{y}, \mathbf{y}')$	Exact match, Embedding similarity

B.3 Extension to token-level early commitment

This setting integrates the trajectory-consensus principle with an existing token-level early stopping rule and follows the protocol of JoT (Kohut et al., 2026) throughout, so that Figure 3 compares against those baselines under an identical configuration.

Benchmarks. We evaluate on the four benchmarks used by Kohut et al. (2026). These are GSM8K (Cobbe et al., 2021) for mathematical reasoning with chain-of-thought prompting, MMLU (Hendrycks et al., 2021) for multitask language understanding, HellaSwag (Zellers et al., 2019) for commonsense reasoning, and HumanEval (Chen et al., 2021) for code generation. All four use zero-shot prompting and the full benchmark rather than a subset. Per-benchmark generation length, denoising-step budget, language and evaluation sample count are taken unchanged from Kohut et al. (2026) and listed in Table 7.

Evaluation metrics. We follow the metric assignment of Kohut et al. (2026). Exact match is used for GSM8K, generated-choice accuracy for MMLU and HellaSwag, and pass@1 for HumanEval. Each benchmark is reported separately and no average is taken across them, since the four differ in generation length by two orders of magnitude. All runs are scored with the LM-Evaluation-Harness (Gao et al., 2024), the harness used by Kohut et al. (2026). Gold answers enter only at this final evaluation.

Backbones and decoding configuration. We use Dream-7B-Instruct (Ye et al., 2025) and LLaDA-8B-Instruct (Nie et al., 2025). Both retain their original decoding strategies. Generation proceeds under the default transfer schedule of each backbone, and the step budget equals the generation length, so full decoding commits one position per step. Dream is decoded without semi-autoregressive blocking, while LLaDA uses a block length of 32 on GSM8K and HumanEval, 3 on MMLU and 5 on HellaSwag. Confidence is read from unscaled logits, which decouples the commitment decision from any temperature applied at sampling time. Per-benchmark generation length and step budget are those of Table 7.

Baselines. We compare against full decoding and three training-free early-decision methods. As above, no baseline hyperparameter is retuned here and we describe only what each method does.

Full decoding runs the base sampler for the entire step budget of Table 7 and commits one position per step, and it is the reference marked by a star in Figure 3 and the same run for every curve. Prophet (Li et al., 2025a) monitors the gap between the two highest predicted probabilities and commits all remaining masked positions at once as soon as that gap exceeds a threshold, so the decision is taken for the whole response at a single step. KLASS (Kim et al., 2025) measures the KL divergence between the predictive distributions of two consecutive steps at each position and unmasks in parallel every position whose distribution has stopped moving, whereas JoT (Kohut et al., 2026) finalizes each position independently once the ratio of its two highest probabilities exceeds a position-specific threshold, which is lowered for positions adjacent to already-resolved tokens and kept strict inside masked regions.

All three read their signal from the current step or from two adjacent steps, so a token that was correct at an earlier step cannot be recovered once decoding has moved past it. Trajectory-Token-MBR adds the trajectory of past proposals at the same position as a second signal.

Trajectory-MBR configuration. Trajectory-Token-MBR combines JoT's early commitment gate with minimum Bayes risk selection over token proposals from the trajectory. It retains the base transfer schedule and uses the denoiser probabilities without modification. At each denoising step, JoT identifies positions eligible for additional early commitment. For every marked position, Trajectory-Token-MBR retains up to the K most recent top-1 proposals produced at that position along the same trajectory. The agreement test is performed only when at least two proposals are available, allowing histories shorter than K during the initial steps.

The retained proposals serve as both candidates and references. Proposal probabilities are normalized within the available history to form reference weights. For each candidate token, we compute the probability-weighted utility loss against all retained proposals, using either exact match or embedding similarity. With exact match, the risk equals the total weight of references that differ from the candidate token. With embedding similarity, the risk incorporates graded similarity between the candidate and reference tokens. These token-level weights use proposal probabilities rather than the sequence-level trajectory scores.

We select the candidate token with the minimum weighted risk. If the position passes the JoT gate and this minimum risk is at most ϵ, the selected token is committed early. The selected token may come from an earlier step and need not match the current top-1 proposal. Positions with fewer than two proposals, those that fail the JoT gate, or those whose minimum risk exceeds ϵ remain subject to the base transfer schedule.

The rule uses the window size K, risk threshold ϵ, and ratio threshold of the JoT gate, which we call the gate threshold. The search spaces are listed in Table 8. The gate threshold is swept from 30 to 300 to trace the trade-off between quality and speed in Figure 3. The remaining three hyperparameters are tuned independently of the sequence-level method in Appendix B.1.

B.4 Efficiency

Sequence-level experiments report denoiser-evaluation speedup. For the token-level experiments in Figure 3, we report step-based speedup to match the step-count convention of JoT (Kohut et al., 2026). Wall-clock speedup is reported separately.

Denoiser-evaluation speedup. For example i, let T_i be the number of denoiser forward passes used by full decoding and E_i the number actually executed by the evaluated method. We aggregate counts across examples as

$$\mathrm{Speedup}_{\mathrm{NFE}} = \frac{\sum_{i=1}^{n} T_i}{\sum_{i=1}^{n} E_i}.$$

With a common budget T, this equals $T/(\frac{1}{n}\sum_{i=1}^{n} E_i)$. Under the output-reuse condition in Appendix A.2, full decoding uses T evaluations at checkpoints $\mathbf{x}_T, \ldots, \mathbf{x}_1$. The terminal state $\mathbf{x}_0$ contributes a candidate to the window but requires no additional forward pass. Stopping at checkpoint $k > 0$ incurs $T - k + 1$ evaluations over $T - k$ completed transitions. Cost is counted through

the stopping checkpoint, even when an earlier candidate is returned. Any correction, resampling, guidance, or refinement forward pass actually executed is counted separately.

Step-based speedup for token-level experiments. For Figure 3, we count completed decoding transitions, with the initial fully masked state at step zero. Let T_i be the full-decoding step budget and S_i the number of completed transitions for example i. We report

$$\text{Speedup}_{\text{step}} = \frac{\sum_{i=1}^{n} T_i}{\sum_{i=1}^{n} S_i}.$$

This count differs from the number of denoiser forward passes. For example, stopping after the first transition of a five-step trajectory gives a $5\times$ step-based speedup, even if two forward passes have been executed at the initial and reached states. The initial state is not an additional completed step.

Wall-clock measurement. Because a reduction in denoiser evaluations does not necessarily translate into a proportional reduction in latency, we additionally report wall-clock speedup in Figure 7. It is measured on all 200 LSHT examples with UltraLLaDA under the configuration of Table 5, on a single NVIDIA A100 GPU. Wall-clock and denoiser-evaluation speedups are computed from the same runs.

C Additional Experiments

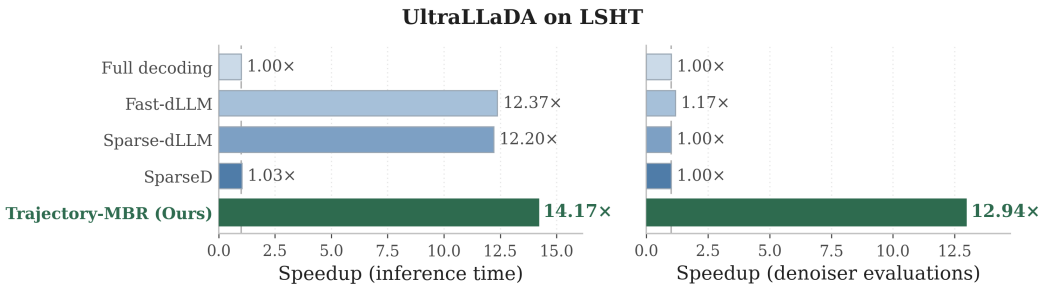

Figure 7: **Higher accuracy with $14.17\times$ faster inference.** On UltraLLaDA/LSHT, Trajectory-MBR raises accuracy from 41.00% to 42.50% with speedups of $14.17\times$ in wall-clock time and $12.94\times$ in denoiser evaluations over full decoding. Left and right panels report these measures from the same runs on all 200 examples using one NVIDIA A100 GPU.

Wall-clock efficiency on LSHT. Figure 7 compares inference cost on LSHT with UltraLLaDA over 200 examples. Trajectory-MBR reaches a wall-clock speedup of $14.17\times$ over full decoding, ahead of Fast-dLLM at $12.37\times$ and Sparse-dLLM at $12.20\times$, while SparseD stays at $1.03\times$. The gap widens once cost is measured in denoiser evaluations. All three baselines remain near $1.00\times$ in denoiser-evaluation speedup, with Fast-dLLM reaching only $1.17\times$, because they make each transition cheaper but still run the trajectory to its terminal state. Trajectory-MBR instead shortens the trajectory itself and cuts the mean number of evaluations from 64 to 4.95, yielding a denoiser-evaluation speedup of $12.94\times$. The reduction is not paid for in quality, since accuracy rises from 41.00 for full decoding to 42.50. The two panels together show that early stopping is a source of savings that is orthogonal to cheaper transitions, and that it is the source that translates most directly into latency.

Qualitative examples. Table 9 shows a long-form example on MIA. Fast-dLLM falls into repetition, and Sparse-dLLM and SparseD draw on the setting or unsupported traits instead of body language. Trajectory-MBR gives three grounded inferences in a case-summary format and receives a perfect score.

Table 9: **Complete instruction following with a 10/10 score.** On this MIA example with LLaDA-V, Trajectory-MBR provides three body-language-based inferences in case-summary format, scoring 10/10 versus scores ranging from 5/10 to 7/10 for baselines affected by repetition or unsupported inferences. Settings follow Hong et al. (2026). Each response is scored once by the official judge.



Instruction. Pretend you are a detective and infer three things about the group's relationship based on their body language, but present your findings as if it's a case summary.

Requirement: three inferences from body language in a case-summary style.

Method	Score	Response	Issue
Fast-dLLM	5/10	Case Summary: The group appears to be engaged in a serious discussion, suggesting a close relationship. The central figure's direct gaze and attentive posture indicate a leadership of, while..... and and and and and and and a a a,,,,,, a a a or or setting	Repetition and an incomplete case summary
Sparse-dLLM	7/10	The case: The seated trio. 1. The The relaxed postures suggest a friendly rapport. 2. .. The The central object,, though blurred, indicates a shared interest. 3. . The The informal setting implies a professional relationship or, at a minimum a friendly gathering	Relies on an object and the setting instead of body language
SparseD	7/10	The case: The Triangle of Three. 1. The person on the left is the leader of the group. 2.. The person in the middle is the creative genius of the group. 3.. The person on the right is the most introverted and introspective member of the group	Asserts roles and personality without visible support
Trajectory-MBR	**10/10**	The case: The Casual Conversationing. 1. The trio's relaxed posture suggests a comfortable and informal atmosphere. 2.. The speaker's direct gaze and open mouth indicate confidence and engagement. 3.. The listener's slight lean and smile imply a sense of camaraderie or friendship within the group	Meets all requirements