# Guiding Diffusion Language Models with Denoising Entropy

**Ziyu Chen**[1,2*] **Xinbei Jiang**[1*] **Peng Sun**[1,3] **Tao Lin**[3]
[1]Zhejiang University [2]University of Chicago [3]Westlake University
{ziyu, xinbei}@zju.edu.cn; {sunpeng, lintao}@westlake.edu.cn

## Abstract

Diffusion Language Models (DLMs) offer flexible, non-autoregressive generation, but this freedom introduces a challenge: final output quality is highly sensitive to the decoding order. We are the first to formalize this issue, attributing the variability in output quality to the cumulative predictive uncertainty along a generative path. To quantify this uncertainty, we introduce Denoising Entropy, a computable metric that serves as an internal signal for evaluating generative process. Leveraging this metric, we propose two algorithms designed to optimize the decoding path: a post-hoc selection method and a real-time guidance strategy. Experiments demonstrate that our entropy-guided methods significantly improve generation quality, consistently boosting accuracy on challenging reasoning, planning, and code benchmarks. Our work establishes Denoising Entropy as a principled tool for understanding and controlling generation, effectively turning the uncertainty in DLMs from a liability into a key advantage for discovering high-quality solutions.

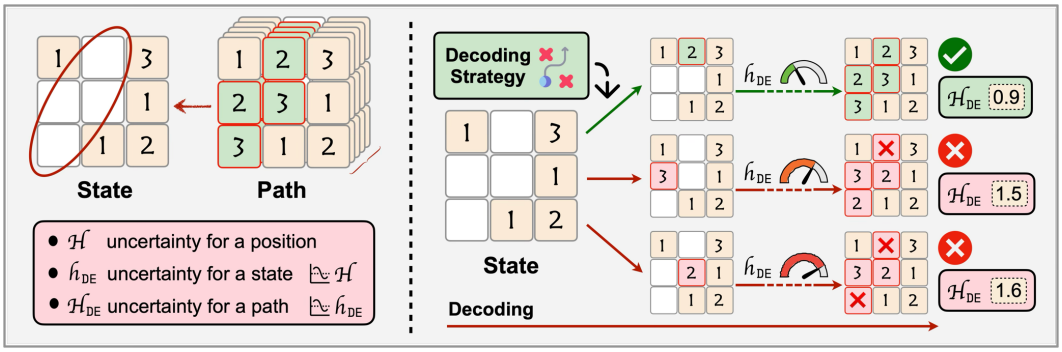



Figure 1: **Quantifying Path Uncertainty in DLMs with Denoising Entropy. Left**: State Entropy ($h_{\text{DE}}$) measures per-state uncertainty, computed as the mean Shannon Entropy over the predictive distributions for all masked positions. $h_{\text{DE}}$ is then aggregated over the entire path to form the Path Entropy ($H_{\text{DE}}$). **Right**: Decoding process shows how different paths yield outputs of varying quality. Our key insight is that the lower $H_{\text{DE}}$ indicates path yielding better output, providing a potent internal signal for generation quality.

## 1 Introduction

In language modeling, Diffusion Language Models (DLMs) [4, 30, 42] are rapidly emerging as powerful counterparts to the dominant Auto-Regressive Language Models (AR LMs). While AR LMs rely on the next-token prediction paradigm and are constrained to a fixed left-to-right generation path [7, 47], DLMs learn to denoise sequences via random-order masking, enabling them to construct

*Equal contribution. Code available at https://github.com/LINs-lab/DenoisingEntropy.

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

sequences in any order in principle. Each unique sequence construction process can be viewed as a *decoding path*, and different paths typically entail varying generation difficulty that shapes the final output. The generation flexibility of DLMs thus has theoretical potential: within the vast space of possible decoding paths, there must exist an optimal path that yields results no worse than the single, rigid sequential path of AR LMs.

However, this theoretical potential often remains untapped in practice, as DLMs always underperform compared to AR LMs [15]. Beyond the model capability, decoding strategy acts as a critical bottleneck determining the path taken [26]. The default random-order strategy, for instance, treats all paths as equally probable [4], reducing the search for an optimal path to mere chance. More sophisticated approaches employ greedy token-level heuristics, such as unmasking the token with the highest confidence [8], but are still myopic: a sequence of locally optimal steps provides no guarantee of a globally optimal path. The fundamental limitation underlying these strategies is their lack of a global perspective, as they operate without a mechanism to assess the quality of the overall generative path.

We draw inspiration from uncertainty quantification in AR LMs, where metrics like entropy reliably link predictive uncertainty to generative quality [28, 50], and uncertainty-aware decoding has shown effect on enhancing global coherence [1, 56]. Adapting this insight to DLMs, we formalize **Path Uncertainty**: DLMs' cumulative predictive uncertainty along a complete decoding path, capturing the global quality of entire generative process. We attribute DLM quality variability to path uncertainty. High cumulative uncertainty harms output consistency, while low one indicates reliable paths [38].

We introduce **Denoising Entropy**, a novel and internally computable metric designed to quantify the inherent uncertainty of generative paths in DLMs. As illustrated in Figure 1, Denoising Entropy comprises two complementary variants: State Entropy ($h_{\texttt{DE}}$), which captures uncertainty at a single decoding state, and Path Entropy ($H_{\texttt{DE}}$), which integrates this uncertainty across an entire path to measure cumulative generative uncertainty. This formulation provides a principled framework for evaluating the reliability of different generative paths.

Minimizing Denoising Entropy offers a powerful new objective for steering DLM decoding towards more reliable outcomes. We operationalize this principle through two search algorithms: a post-hoc selection method that reranks sampled paths to find the one with the minimum Path Entropy ($H_{\texttt{DE}}$), and a real-time guidance method that uses State Entropy ($h_{\texttt{DE}}$) to actively steer generation towards lower-uncertainty regions of the state space. By shifting the focus from myopic, token-level decisions to holistic, path-level optimization, our approach consistently boosts the performance of DLMs on complex open-ended generation and reasoning tasks. **We summarize our key findings below:**

- **Formalizing Path Uncertainty** (Section 3.1): We identify and formalize Path Uncertainty, the cumulative predictive uncertainty of a DLM along a single decoding path, which drives the variability in output quality.
- **Denoising Entropy as Uncertainty Proxy** (Section 3.2): We introduce two novel and computable metrics, $h_{\texttt{DE}}(t)$ and $H_{\texttt{DE}}$, as a theoretically-grounded toolkit to evaluate entire generative states and paths, providing signals for path-aware guidance.
- **Path Search Algorithms** (Section 3.3): We propose two path search algorithms, Entropy-based Best-of-N and Entropy-guided Sequential Monte Carlo, that leverage Denoising Entropy as an internal signal to iteratively optimize the decoding process and improve the generation quality.

## 2 Preliminaries

Let $\mathcal{V}$ be a vocabulary of size $K$, and consider a sequence $\mathbf{x}_0$ of $L$ tokens $\mathbf{x}_0 = (x_0^1, \ldots, x_0^L)$. For any position $\ell \in \{1, \ldots, L\}$, the token $x_0^\ell$ is represented by a one-hot vector $\mathbf{x}_0^\ell \in \{0, 1\}^K$, whose component corresponding to the index of $x_0^\ell$ is 1.

### 2.1 Forward Process and Training Objective

The training of DLMs relies on a continuous-time forward corruption process. For any time $t \in [0, 1]$, a noisy state $\mathbf{x}_t$ is generated from the clean sequence $\mathbf{x}_0$. This is achieved probabilistically: each token $x_0^\ell$ is independently replaced by a special `[MASK]` token with probability $1 - \alpha_t$, and remains unchanged with probability $\alpha_t$. Here $\alpha_t \in [0, 1]$ is a strictly decreasing noise schedule with $\alpha_0 \approx 1$ and $\alpha_1 \approx 0$ [42]. Let $\mathcal{M}_t$ be the set of indices that are masked at time $t$.

A DLM with denoising network $p_\theta$, parameterized by $\theta$, is trained to reverse this process. Given the corrupted variable $\mathbf{x}_t$ and the time $t$, DLM predicts the probability distribution $\hat{\mathbf{p}}_0$ for each token in the original sequence as an estimate of $\mathbf{x}_0$. For any position $\ell$, the predicted distribution $\hat{\mathbf{p}}_0^\ell$ is:

$$\hat{\mathbf{p}}_0^\ell = p_\theta(X_0^\ell \mid \mathbf{x}_t, t) \in \Delta^K, \quad \forall \ell \in \{1, \ldots, L\}\,, \tag{1}$$

where $X_0^\ell$ denotes the random variable for the *original* token at position $\ell$, and $\Delta^K$ is the $K$-simplex.

Parameters $\theta$ are optimized by minimizing the Negative Evidence Lower Bound (NELBO) [30, 42, 44]. We define the reconstruction loss at time $t$ as the negative log-likelihood summed over masked tokens:

$$\mathcal{L}_t(\mathbf{x}_0, \mathbf{x}_t) = \sum_{\ell \in \mathcal{M}_t} -\log p_\theta(x_0^\ell \mid \mathbf{x}_t, t)\,.$$

The training objective is computed as the expectation of loss integrated over the diffusion time, where $q(\mathbf{x}_0)$ is the posterior data distribution, $1-\alpha_t$ is the unmasking probability at time $t$, and $w(t) = \left|\frac{d\alpha_t}{dt}\right| \frac{1}{1-\alpha_t}$ is the weighting function.

$$\mathcal{L}(\theta) = \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0)} \left[ \int_0^1 w(t) \mathbb{E}_{\mathbf{x}_t \mid \mathbf{x}_0} \Big[ \mathcal{L}_t(\mathbf{x}_0, \mathbf{x}_t) \Big] dt \right]. \tag{2}$$

### 2.2 Reverse Process for Generation

The inference process generates a sequence $\mathbf{x}_0$ by iteratively denoising from a fully masked sequence. The procedure starts with $\mathbf{z}_1$, where all tokens are `[MASK]`. It then proceeds backward in time along a discrete schedule $1 = t_N > t_{N-1} > \cdots > t_0 = 0$.

At each step $i$ (from $N$ down to 1), the DLM $p_\theta$ performs prediction and transition. First, it takes the current state $\mathbf{x}_{t_i}$ and time $t_i$ as input to predict the original token probability distribution $\hat{\mathbf{p}}_0^\ell$ for each position $\ell$. Then, the next state $\mathbf{x}_{t_{i-1}}$ is sampled from the reverse transition kernel $p(\mathbf{x}_{t_{i-1}} \mid \mathbf{x}_{t_i})$. This distribution determines the value of tokens in the next state: a masked token is either unmasked to a content token $\hat{x} \sim \hat{\mathbf{p}}_0^\ell$ or remains masked, governed by the noise levels $\alpha_{t_i}$ and $\alpha_{t_{i-1}}$. This iterative refinement continues until $t_0 = 0$, yielding the final generated sequence.

## 3 Path Uncertainty and Denoising Entropy

### 3.1 Path Uncertainty

A decoding path $\tau = (\mathbf{x}_{t_N}, \ldots, \mathbf{x}_{t_0})$ is a sequence of states traversed during the generative process, guided by a decoding strategy $\pi$. The strategy $\pi$ determines the transition from $\mathbf{x}_{t_i}$ to $\mathbf{x}_{t_{i-1}}$ at each step, based on the model's reverse transition kernel $p(\cdot|\cdot)$.

We define **Path Uncertainty** as the model's cumulative predictive uncertainty along a *single* decoding path $\tau$, which reflects how *confident* the model is throughout the entire generation of a sequence.

### 3.2 Denoising Entropy

To quantify path uncertainty, we introduce **Denoising Entropy**. We define *Oracle State Uncertainty* first as the uncertainty of the joint posterior distribution over missing information at any given state.

**Definition 1 (Oracle State Uncertainty).** *Let $\mathbf{X}_{\mathcal{M}_t} = \{X_0^\ell\}_{\ell \in \mathcal{M}_t}$ be the random vector representing the true original tokens at all masked positions $\mathcal{M}_t$. Oracle State Uncertainty is the Shannon entropy of the model's joint predictive distribution over this vector, conditioned on the current state $\mathbf{x}_t$:*

$$H_{oracle}(\mathbf{x}_t) \triangleq H(\mathbf{X}_{\mathcal{M}_t} \mid \mathbf{x}_t). \tag{3}$$

Computing joint entropy requires summing over an exponentially large outcome space $K^{|\mathcal{M}_t|}$, which is computationally intractable. We define *State Entropy* as a proxy based on marginal distributions.

**Definition 2 (State Entropy).** *For a given state $\mathbf{x}_t$ at time $t \in [0,1]$, State Entropy ($h_{DE}$) is defined as the average Shannon entropy of the model's predictive distributions over the set of masked positions $\mathcal{M}_t$:*

$$h_{DE}(\mathbf{x}_t) \triangleq \frac{1}{|\mathcal{M}_t|} \sum_{\ell \in \mathcal{M}_t} H\left(p_\theta(X_0^\ell \mid \mathbf{x}_t, t)\right) , \tag{4}$$

*where $H(\cdot)$ denotes the Shannon entropy and the definition is valid for $|\mathcal{M}_t| > 0$.*

The integral of *State Entropy* across the entire path yields *Path Entropy*, which quantifies the cumulative uncertainty of the generative process.

**Definition 3 (Path Entropy).** *Path Entropy ($H_{DE}$) provides a measure for the total uncertainty of a denoising process with a decoding path $\tau$. It is the integral of the $h_{DE}$ over the generation time. In a discrete-time setting with $N$ steps, it is approximated by averaging the $h_{DE}$ across all steps:*

$$H_{DE}(\tau) \triangleq \int_0^1 h_{DE}(\mathbf{x}_t) dt \approx \frac{1}{N} \sum_{i=1}^{N} h_{DE}(\mathbf{x}_{t_i}) . \tag{5}$$

### 3.3 Entropy-Guided Decoding

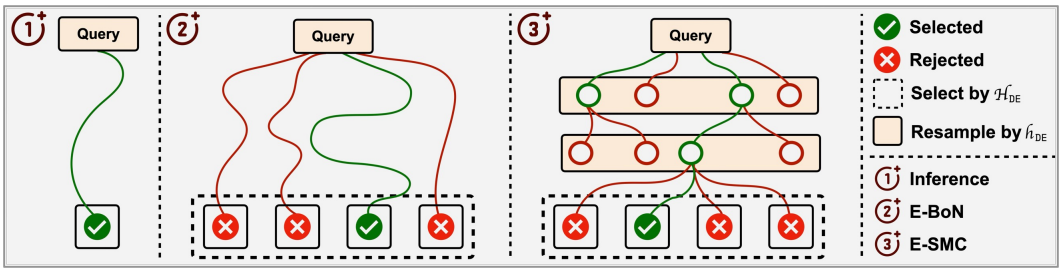



Figure 2: **An overview of our entropy-guided decoding algorithms.** While ***standard inference*** in subfigure (1) generates a single decoding path, our methods explore multiple candidate paths guided by Denoising Entropy. **E-BoN** in subfigure (2) performs **post-hoc selection**, choosing the best path from multiple independent candidates based on path entropy $H_{\text{DE}}$. **E-SMC** in subfigure (3) provides **real-time guidance**, iteratively pruning high-entropy paths and replicating low-entropy ones based on interim state entropy $h_{\text{DE}}$.

A decoding strategy $\pi$ induces a distribution $P(\tau|\pi)$ over paths and their corresponding path entropies $H_{\text{DE}}(\tau)$. This implies two properties: (i) a single stochastic strategy yields a variance in path uncertainty, $\mathbb{V}_{\tau \sim P(\cdot|\pi)}[H_{\text{DE}}(\tau)] > 0$; and (ii) distinct strategies, $\pi_a$ and $\pi_b$, produce different expected uncertainties, $\mathbb{E}_{\pi_a}[H_{\text{DE}}(\tau)] \neq \mathbb{E}_{\pi_b}[H_{\text{DE}}(\tau)]$.

Hypothesizing that low path uncertainty lead to higher-quality outputs [50], generative decoding can be framed as an optimization problem. For a generative task, the objective is to find an optimal strategy $\pi^\star$ that minimizes the expected path entropy:

$$\pi^\star = \arg\min_{\pi} \mathbb{E}_{\tau \sim P(\cdot|\pi)}[H_{\text{DE}}(\tau)] . \tag{6}$$

While finding the globally optimal strategy $\pi^\star$ is intractable due to the vast search space, we introduce two algorithms that leverage denoising entropy as an active guidance signal: (i) *a post-hoc selection method,* ***Entropy-based Best-of-N* (E-BoN)**, and (ii) *a real-time path optimization strategy,* ***Entropy-guided Sequential Monte Carlo* (E-SMC)**, both designed to effectively approximate this objective. Figure 2 provides a conceptual overview of E-BoN, E-SMC, and standard single-path inference.

#### 3.3.1 Entropy-Based Best-of-N

$H_{\text{DE}}$ can be used for post-hoc selection from a set of candidates. E-BoN formalizes this principle. Given a population of $M$ candidate decoding paths, $\{\tau^{(1)}, \ldots, \tau^{(M)}\}$, E-BoN identifies the single

best path with minimum $H_{\text{DE}}$:

$$\tau^\star = \underset{\tau^{(m)} \in \{\tau^{(1)}, \ldots, \tau^{(M)}\}}{\arg\min} H_{\text{DE}}(\tau^{(m)}) \,. \tag{7}$$

E-BoN is straightforward, requiring the full generation of $M$ samples before selection. However, it expends the full computational budget uniformly across all paths, without any mechanism to redirect resources during generation [9].

#### 3.3.2 Entropy-Guided Sequential Monte Carlo

To actively optimize the decoding path in real-time, we introduce **E-SMC**, a variant of Sequential Monte Carlo [14] adapted for DLM reverse process [45]. E-SMC employs a guidance mechanism operating on $M$ parallel particles. By evaluating partial paths via State Entropy $h_{\text{DE}}$, E-SMC can dynamically reallocate computational budget, pruning high-entropy paths and replicating low-entropy ones. This iterative search process comprises three steps: *Propagation, Evaluation, and Resampling:*

- **Propagation.** At each reverse step $i \in \{N, \ldots, 1\}$, we advance each particle $\mathbf{x}_{t_i}^{(m)}$ in the set $\mathcal{X}_{t_i}$ to a new state $\mathbf{x}_{t_{i-1}}^{(m)}$ by sampling from the proposal distribution given by the DLM reverse kernel, $\mathbf{x}_{t_{i-1}}^{(m)} \sim p(\cdot|\mathbf{x}_{t_i}^{(m)})$. This process generates the subsequent particle population $\mathcal{X}_{t_{i-1}}$.
- **Evaluation.** To actively guide the search, evaluations are performed periodically at fixed intervals of $\Delta i_r$ steps. During evaluation, each particle $\mathbf{x}_{t_{i-1}}^{(m)}$ in the current set $\mathcal{X}_{t_{i-1}}$ is assessed using a potential function $\Phi(\mathbf{x}; \lambda)$. $\Phi$ is designed to assign higher scores to particles with lower $h_{\text{DE}}(\mathbf{x})$. Temperature $\lambda$ modulates the sharpness of the resulting score distribution. $\Phi$ is detailed in Appendix B.1.
- **Resampling.** Potential scores computed during evaluation are normalized to form categorical distribution over the current particle set $\mathcal{X}_{t_{i-1}}$. This distribution assigns selection probability $w^{(m)}$ to each particle. A new population is formed by drawing $M$ particles with replacement from $\mathcal{X}_{t_{i-1}}$ according to $w^{(m)}$. The resampling mechanism prunes high-entropy paths and replicate low-entropy ones, steering the search towards more promising path space [31]. The full process is detailed in Algorithm 1.

### 3.4 Theoretical Justification for Denoising Entropy

The effectiveness of E-BoN and E-SMC is predicated on Denoising Entropy as indicator for generation quality. This section validates the use of the metric by establishing theoretical basis through three key properties: **(i)** we prove that State Entropy $h_{\text{DE}}$ is a computable upper bound on the intractable *oracle uncertainty*. **(ii)** we relate $h_{\text{DE}}$ to the training objective, showing it serves as a close proxy for the instantaneous loss. **(iii)** we demonstrate that the gap in expected Path Entropy $H_{\text{DE}}$ lower-bounds the divergence between the generative and optimal reference distributions. These results justify minimizing Denoising Entropy for guiding generation toward more consistent and higher-quality outputs. We also provide further analysis of its properties in Appendix A.6.

**$h_{\text{DE}}$ as a Bound on Ideal Uncertainty.** $h_{\text{DE}}$ is a well-posed measure of uncertainty by relating it to *Oracle State Uncertainty*, the joint entropy over all masked tokens. $h_{\text{DE}}$ sums the uncertainty of each masked token independently, thereby ignoring contextual constraints that tokens impose on each other and forming a computable upper bound to this oracle. Proof relying on the subadditivity of Shannon entropy [43], provided in Appendix A.1.

**Proposition 1 ($h_{\text{DE}}$ as an upper bound).** *Oracle State Uncertainty $H_{oracle}(\mathbf{x}_t)$ is upper-bounded by the sum of marginal entropies, which is directly proportional to State Entropy $h_{DE}(\mathbf{x}_t)$:*

$$H_{oracle}(\mathbf{x}_t) \leq \sum_{\ell \in \mathcal{M}_t} H\left(p_\theta(X_0^\ell \mid \mathbf{x}_t, t)\right) = |\mathcal{M}_t| \cdot h_{DE}(\mathbf{x}_t). \tag{8}$$

**$h_{\text{DE}}$ as a Proxy for DLM Loss.** We assume an $\epsilon$-accurate DLM, whose reverse transition kernel approximates the true posterior one within step-wise error $\epsilon$ (see Definition 4 in Appendix A.2). For well-trained $\epsilon$-accurate DLMs, $h_{\text{DE}}$ serves as direct proxy to approximate the instantaneous per-token training loss within an $\mathcal{O}(\epsilon)$ error. We state the approximation with state entropy as follows:

**Proposition 2 (Approximation of Normalized Loss by $h_{\mathrm{DE}}$).** *Considering an $\epsilon$-accurate DLM. $h_{DE}(\mathbf{x}_t)$ provides an approximation to the instantaneous, per-token normalized training loss:*

$$\frac{1}{|\mathcal{M}_t|}\sum_{\ell\in\mathcal{M}_t}\left(-\log p_\theta(x_0^\ell \mid \mathbf{x}_t, t)\right) - h_{DE}(\mathbf{x}_t) \le \epsilon + \sqrt{\frac{\epsilon}{2}}\log K\,, \tag{9}$$

*where $K$ is the vocabulary size.*

Proof in Appendix A.2. While $h_{\mathrm{DE}}$ measures instantaneous difficulty, $H_{\mathrm{DE}}$ extends this by integrating it throughout the path, measuring the total generation challenge.

The full objective $\mathcal{L}$ over masked tokens can be approximated by $h_{\mathrm{DE}}$:

$$\mathcal{L}(\mathbf{x}_0) = \int_0^1 w(t)\left[\sum_{\ell\in\mathcal{M}_t} -\log p_\theta(x_0^\ell \mid \mathbf{x}_t, t)\right]dt \approx \int_0^1 w(t)\cdot|\mathcal{M}_t|\cdot h_{\mathrm{DE}}(\mathbf{x}_t)dt,$$

where the weighting function is $w(t) = \left|\frac{d\alpha_t}{dt}\right|\frac{1}{1-\alpha_t}$. While Path Entropy is the unweighted integral $H_{\mathrm{DE}}(\tau) = \int_0^1 h_{\mathrm{DE}}(\mathbf{x}_t)dt$, $\mathcal{L}$ provides justification for using $H_{\mathrm{DE}}$ as a proxy for generation quality.

**$H_{\mathrm{DE}}$ as a Bound on Output Quality.** Let $\mathcal{P}_{\text{ref}}$ denote the reference path distribution induced by the optimal reverse transition kernel $q(\cdot|\cdot)$ conditioned on real data, and $\mathcal{P}_\theta$ denote the generative path distribution induced by the learned kernel $p_\theta(\cdot|\cdot)$. We define $\mu_{\text{ref}} = \mathbb{E}_{\tau\sim\mathcal{P}_{\text{ref}}}[H_{\mathrm{DE}}(\tau)]$ and $\mu_\theta = \mathbb{E}_{\tau\sim\mathcal{P}_\theta}[H_{\mathrm{DE}}(\tau)]$ as the expected Path Entropy evaluated over reference and generative distributions.

**Proposition 3 (Path Entropy Gap as Quality Bound).** *The KL divergence between the reference distribution $\mathcal{P}_{ref}$ and the generative one $\mathcal{P}_\theta$ is lower-bounded by the squared error in their expected Path Entropies:*

$$N\cdot\epsilon \ge D_{\mathrm{KL}}(\mathcal{P}_{ref}\|\mathcal{P}_\theta) \ge \frac{1}{2B^2}\left(\mu_\theta - \mu_{ref}\right)^2\,, \tag{10}$$

*$B$ is a positive constant, where $H_{DE}(\tau) \le B$.*

The total KL divergence between the reference distribution $\mathcal{P}_{\text{ref}}$ and the generative distribution $\mathcal{P}_\theta$ of an $\epsilon$-accurate DLM is bounded by $N\cdot\epsilon$, as local approximation errors accumulate over denoising steps. Details in Appendix A.3.

Making $\mu_\theta$ closer to $\mu_{\text{ref}}$ is necessary for lower KL divergence, which represents higher output quality. For E-BoN, we demonstrate that finite-sample selection reduces the deviation from $\mu_{\text{ref}}$ at a rate of $\sqrt{\ln N}$. For E-SMC, we prove that an optimal temperature $\lambda^*$ exists to calibrate the distribution, ensuring its entropy precisely matches $\mu_{\text{ref}}$. Details in Appendix A.4 & A.5.

## 4 Experiments

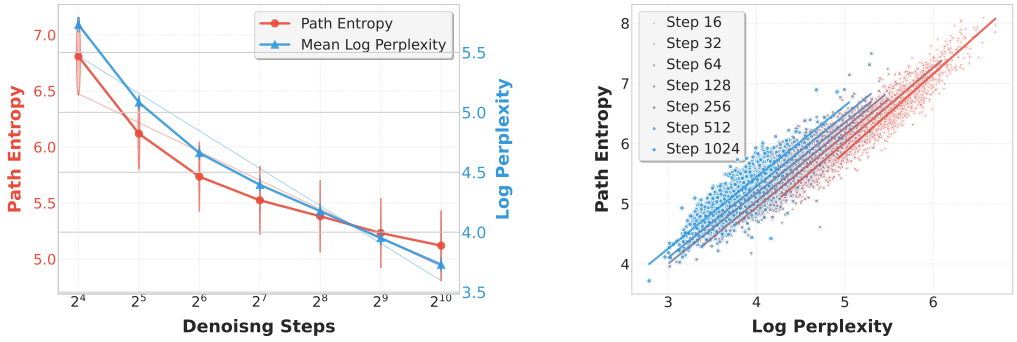
This section empirically studies Path Uncertainty and Denoising Entropy in DLMs through three stages: **(i)** validating Denoising Entropy as a quality-aligned internal metric on DLMs; **(ii)** improving generation by using E-BoN and E-SMC to optimize decoding paths; **(iii)** scaling our method to larger models and complex reasoning and planning tasks.

### 4.1 Validating Denoising Entropy as a Quality Metric

We study whether the internal metric $H_{\mathrm{DE}}$, computed online during generation as a proxy for path uncertainty, aligns with external text quality evaluations.

**Experimental Setting.** We use `MDLM` trained on OpenWebText [17] with approximately 130 million non-embedding parameters for evaluation [42]. `MDLM` generates sequences of length 1,024 under random-order unmasking. We vary the number of denoising steps $S = 2^i$ for $i \in [5, 10]$ and use

a substantially larger model, `GPT2-Large` [39], as AR LM evaluator to compute Perplexity (PPL). Perplexity is a standard evaluation metric for generative uncertainty and text quality quantification. For each generated sample, we record $H_{\text{DE}}$ and $\ln(\text{PPL})$.




(a) Trend of $H_{\text{DE}}$ vs. $\ln(\text{PPL})$ with Denoising Steps.

(b) Correlation between $H_{\text{DE}}$ and $\ln(\text{PPL})$.

Figure 3: **Empirical validation of Path Entropy ($H_{\text{DE}}$) as an internal proxy for generation quality.** The aggregate trend in ***(a)*** shows that a more refined generation process reduces both the model's internal uncertainty ($H_{\text{DE}}$, red) and the output perplexity ($\ln(\text{PPL})$, blue). ***(b)*** provides a per-sample analysis, revealing a strong positive linear correlation between $H_{\text{DE}}$ and $\ln(\text{PPL})$. Details in Appendix B.4.

**Results.** *Across thousands of unconditional generations, we observe a clear, near-linear relationship between $H_{\text{DE}}$ and $\ln(\text{PPL})$ as shown in Figure 3.* Lower $H_{\text{DE}}$ aligns with lower PPL, indicating outputs with higher quality. Both $H_{\text{DE}}$ and $\ln(\text{PPL})$ decrease as the number of denoising steps $S$ increases, reflecting a reduction in path uncertainty and a improvement in sample quality. This case shows that *$H_{\text{DE}}$ is a reliable internal proxy for generation quality.*

### 4.2 Improving Generation with Denoising Entropy

Building upon finding that $H_{\text{DE}}$ serves as a internal proxy for text quality, we now investigate whether Denoising Entropy can be actively employed to mitigate Path Uncertainty and improve generation.

**Experimental Setting.** We evaluate three decoding types: vanilla single path uniform sampling, with E-BoN and E-SMC. We utilize the same `MDLM`, while adopting `GPT2-Large` and `Llama-3-8B` as AR LM evaluators. We analyze performance measured by perplexity and diversity [53] across different numbers of denoising steps ($S$), particles ($K$), and resampling intervals ($\Delta i_r$). In detail:

- For E-BoN, we generate $K$ particles and return the one with the minimum $H_{\text{DE}}$.
- For E-SMC, we steer $K$ particles using weighted resampling with $h_{\text{DE}}$ at intervals of $\Delta i_r$ steps, and finally return the one with the minimum $H_{\text{DE}}$.

Table 1: **Comparison of decoding strategies via ablation studies on hyperparameters.** Perplexity evaluated on `GPT2-Large` and `Llama-3-8B`.

| Configuration | | | Perplexity ↓ | | |
|---|---|---|---|---|---|
| $S$ | $K$ | $\Delta i_r$ | Vanilla | E-BoN | E-SMC |
| **128** | 4 | 32 | 85.3 / 84.5 | 66.4 / 66.1 | **63.4 / 61.9** |
| **256** | 4 | 32 | 68.5 / 66.9 | 52.3 / 55.2 | **47.7 / 51.5** |
| 128 | **4** | 32 | 85.3 / 84.5 | 66.4 / 66.1 | **63.4 / 61.9** |
| 128 | **8** | 32 | 85.3 / 84.5 | 60.3 / 59.8 | **56.0 / 59.0** |
| 256 | **2** | 32 | 68.5 / 66.9 | 60.0 / 59.1 | **55.5 / 57.0** |
| 256 | **4** | 32 | 68.5 / 66.9 | 52.3 / 55.2 | **47.7 / 51.5** |
| 256 | **8** | 32 | 68.5 / 66.9 | 47.5 / 51.1 | **40.4 / 45.1** |
| 256 | 4 | **8** | 68.5 / 66.9 | 52.3 / 55.2 | **43.9 / 44.4** |
| 256 | 4 | **16** | 68.5 / 66.9 | 52.3 / 55.2 | **45.1 / 48.1** |
| 256 | 4 | **32** | 68.5 / 66.9 | 52.3 / 55.2 | **47.7 / 51.5** |
| 256 | 4 | **64** | 68.5 / 66.9 | 52.3 / 55.2 | **50.2 / 51.8** |
| 256 | 4 | **128** | 68.5 / 66.9 | 52.3 / 55.2 | **53.4 / 54.1** |

We also record Diversity of generated content and compare it with Greedy Search, indicating that both E-BoN and E-SMC reduce PPL without compromising diversity. Details in Table 7 and Table 8.

**Results.** As shown in Table 1, both entropy-guided strategies substantially outperform the vanilla sampler, *confirming the practical utility of Denoising Entropy as a control signal.* E-SMC *consistently achieves the best performance, as it benefits from the active online guidance mechanism.*

E-BoN *and* E-SMC *reduce perplexity while maintaining diversity comparable to the vanilla sampler. Under specific configurations,* E-SMC *can improve generation quality without compromising diversity.* Table 8 in Appendix B.4 shows optimizing Denoising Entropy with greedy search, which achieve significantly lower perplexity by sacrificing diversity. This aligns with the analysis in Proposition 3 that over-minimizing $\mathcal{P}_\theta$ reduces generation quality. We also measure generation quality and diversity jointly with MAUVE [37]. With 5,000 samples and four particles, E-BoN and E-SMC raise the MAUVE of `MDLM` from 0.059 $\rightarrow$ 0.094 and 0.087, respectively, and E-SMC lifts the stronger ReMDM [48] from 0.427 $\rightarrow$ 0.456 (E-BoN: 0.412). Therefore, the perplexity reduction does not come at the cost of distributional collapse.

The results also affirm the scalability of our methods, as performance improves with the computational budget directly. Increasing the number of particles ($K$) significantly enhances outcomes, particularly for E-SMC, which better exploits the expanded search space (e.g., perplexity drops from 47.7 to 36.1 as $K$ increases from 4 to 12). Furthermore, for E-SMC, more frequent resampling (smaller $\Delta i_r$) provides a complementary and computationally efficient means of improvement by pruning high-uncertainty paths.

### 4.3 Scaling to Reasoning and Planning

Experiments in Section 4.1 & 4.2 validated Denoising Entropy as a proxy for generation quality and demonstrated the efficacy of our proposed algorithms, E-BoN and E-SMC, on a small-scale DLM using perplexity as the evaluation metric. To assess the generalizability and practical impact of our approach, we now scale to advanced DLMs and evaluate on a suite of challenging reasoning benchmarks. This shift enables evaluation beyond abstract quality to task-specific accuracy, testing whether entropy guidance improves problem-solving in large-scale DLMs.

**Models and Datasets.** We conduct experiments on three large DLMs: `LLaDA-Instruct-8B` [32], `LLaDA-1.5-8B` [55], and `Open-dCoder-0.5B` [36]. `LLaDA` are evaluated across five diverse benchmarks spanning three reasoning domains: mathematical reasoning with GSM8K [11] and MATH500 [29], scientific reasoning with GPQA [40], and planning with Sudoku [33] and Countdown [51]. `Open-dCoder-0.5B` is a DLM for code generation, we evaluate its performance on HumanEval/HumanEval+ [10] and MBPP/MBPP+ [5].

Table 2: **Accuracy (%) of our path search algorithms on `LLaDA` models across five reasoning and planning benchmarks.** When applied to the strong PC-Sampler baseline, both E-BoN and E-SMC improve performance.

| | LLaDA-Instruct-8B | | | | | | LLaDA-1.5-8B | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Decoding strategy** | **GSM.** | **Math.** | **GPQA.** | **Count.** | **Sudoku.** | **Avg.** | **GSM.** | **Math.** | **GPQA.** | **Count.** | **Sudoku.** | **Avg.** |
| *Existing decoding strategies* | | | | | | | | | | | | |
| Uniform | 48.8 | 15.0 | 29.0 | 14.4 | 2.2 | 21.9 | 52.7 | 20.0 | 28.1 | 15.8 | 3.4 | 24.0 |
| Confidence | 6.8 | 3.4 | 27.9 | 34.0 | 23.8 | 19.2 | 19.2 | 5.4 | 29.0 | 33.8 | 24.8 | 22.4 |
| Entropy | 2.2 | 3.8 | 28.4 | 33.8 | 1.6 | 14.0 | 12.1 | 5.0 | 28.8 | 34.7 | 0.2 | 16.2 |
| Margin | 11.1 | 1.8 | 28.4 | 33.9 | 26.6 | 20.4 | 27.9 | 6.4 | 28.6 | 31.8 | 33.6 | 25.7 |
| P2 | 11.6 | 3.4 | 28.2 | 34.2 | 24.0 | 20.3 | 25.4 | 5.6 | 28.8 | 33.6 | 27.8 | 24.2 |
| EB-Sampler | 1.6 | 3.6 | 29.9 | 34.1 | 24.2 | 18.7 | 12.3 | 4.8 | 28.6 | 34.6 | 1.6 | 16.4 |
| Semi-AR | 77.9 | 27.6 | 27.7 | 32.6 | 0.0 | 33.2 | 80.7 | 34.2 | 26.1 | 32.4 | 0.0 | 34.7 |
| Fast-dLLM | 78.2 | 28.4 | 28.6 | 11.4 | 0.3 | 29.4 | 80.8 | 31.2 | 27.9 | 32.9 | 0.4 | 34.6 |
| *Strongest baseline and our path search* | | | | | | | | | | | | |
| PC-Sampler | 79.3 | 34.0 | 28.6 | 36.3 | 27.6 | 41.2 | 82.2 | 37.4 | 28.8 | 35.0 | 33.4 | 43.4 |
| *w/* E-BoN | 79.8 | 35.0 | 28.8 | **42.2** | 28.2 | 42.8 | 82.9 | 38.2 | 28.9 | **40.2** | **34.2** | **44.9** |
| *w/* E-SMC | **81.2** | **35.6** | **28.9** | 40.4 | **29.2** | **43.1** | **83.2** | **38.6** | **29.0** | 39.3 | 33.6 | 44.8 |

**Decoding Strategies.** We evaluate our path search algorithms with a spectrum of established DLM decoding strategies. These include the standard uniform sampler [4], uncertainty-based samplers (e.g., confidence [8], entropy [6], and margin [26]), efficient samplers (Semi-AR [32], EB-Sampler [6], and Fast-dLLM [49]), and two recently proposed advanced strategies: P2 [35] and PC-Sampler [24]. For block-wise strategy including Semi-AR and Fast-dLLM, we set the number of decoding blocks to 8 for all datasets. A detailed description of each baseline strategy is provided in Appendix B.2.

***Path Search Algorithms Enhance Decoding Strategies.*** We demonstrate the efficacy of E-BoN and E-SMC through two experiments. First, to establish their maximum potential, we apply them to

PC-Sampler [24] on LLaDA models. This combination sets new state-of-the-art, achieving substantial accuracy gains across five challenging benchmarks as shown in Table 2. Second, to validate their broad applicability, we integrate our methods with five different baseline samplers on Open-dCoder model. As shown by the average accuracy improvements in Figure 4 (Left), both algorithms consistently boost each tested sampler, acting as versatile enhancers for a wide array of decoding strategies. Thus, *denoising entropy optimization generalizes across models, decoding strategies, and tasks*. E-SMC also stays effective when multi tokens are unmasked per step, as shown in Appendix B.6.

***Path-Level Optimization Is Effective for Complex Reasoning and Planning.*** The gains are most pronounced on benchmarks requiring multi-step reasoning and planning. For example, on LLaDA-Instruct-8B, E-SMC improves GSM8K accuracy from 79.3% to 81.2% (+1.9%) and the Countdown planning from 36.3% to 40.4% (+4.1%). These results demonstrate that performing well on such tasks depends on adopting a global view of the entire solution. Our path search algorithms enable this perspective by utilizing Denoising Entropy to evaluate the full generation path, rather than assessing only a single step.

### 4.4 Other Results and Ablation Studies

**Path, Entropy and Accuracy.** We validate Denoising Entropy as a proxy for task performance on Sudoku, a benchmark sensitive to decoding order. We use the PC-Sampler hyperparameter $\lambda$ to control decoding sequentiality, thus generating a spectrum of distinct paths. Smaller $\lambda$ encourage more random decoding orders, which benefit Sudoku solving, while larger $\lambda$ values lead to more sequential decoding, harming performance. Figure 4 (Middle) reveals a negative correlation between $H_{\text{DE}}$ and final accuracy, confirming that higher entropy signals a lower-quality generation path.

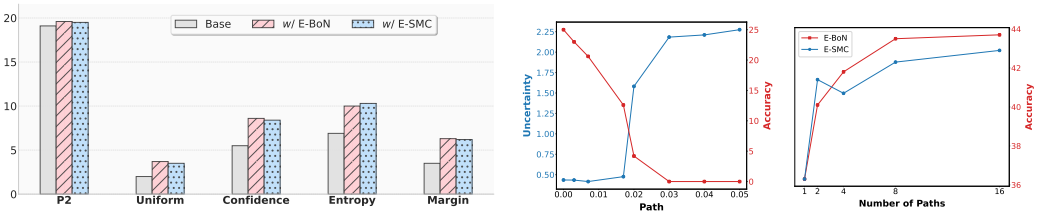



Figure 4: **Left**: Average accuracy on code benchmarks. E-BoN and E-SMC enhance the performance of various baseline decoding strategies consistently, demonstrating their broad applicability. **Middle**: Relationship between decoding path, path uncertainty, and accuracy on Sudoku. **Right**: Relationship between number of paths and accuracy on Countdown.

**Number of Paths and Accuracy.** We show accuracy versus number of particle paths on Countdown. Similar to results observed on MDLM, scaling the number of particles improves performance.

**Budget-Efficiency.** We disentangle the metric effectiveness from computational budget by comparing strategies under a fixed cost (N=5). We benchmark against *Majority Voting* and *Likelihood-based* baselines (using sum of log-probabilities). Table 3 shows that our methods outperform Likelihood-based counterparts. Random selection and majority voting perform on par with or below the single-path PC-Sampler, while E-BoN and E-SMC improve the average accuracy over the strongest non-entropy baseline, Likelihood-BoN, by 1.47 and 1.80 points on LLaDA-Instruct-8B, and by 0.97 and 0.89 points on LLaDA-1.5-8B. It confirms that the gains come specifically from *Denoising Entropy*, which provides a better signal than standard confidence or consensus. We also add runtime and memory analysis in Table 10 and 11.

## 5 Related Work

**Diffusion Language Models.** Recent years have seen the extension of diffusion models, originally successful in continuous domains [13, 23, 41, 46], to discrete generation [4]. A key branch is masked diffusion, formalized in D3PM [4] and explored for text generation in DiffusionBERT [22]. The field

Table 3: Accuracy (%) of path selection criteria LLaDA. All methods share the PC-Sampler backbone and decode five candidate paths. Random picks one path uniformly, Majority Voting returns the most frequent answer, and Likelihood-BoN/-SMC replace the Denoising Entropy signal of E-BoN/E-SMC with the sum of token log-likelihoods. **Bold** marks the best result in each column among the five-path methods.

| Selection Criterion | GSM. | Math. | GPQA. | Count. | Sudoku. | Avg. |
|---|---|---|---|---|---|---|
| **LLaDA-Instruct-8B** | | | | | | |
| PC-Sampler | 79.30 | 34.00 | 28.57 | 36.30 | 27.60 | 41.15 |
| Random | 79.26 ± 0.37 | 33.92 ± 0.54 | 28.48 ± 0.41 | 36.40 ± 0.65 | 27.52 ± 0.73 | 41.12 ± 0.54 |
| Majority Voting | 79.00 ± 0.43 | 33.68 ± 0.56 | 28.26 ± 0.46 | 36.90 ± 0.46 | 26.96 ± 0.64 | 40.96 ± 0.48 |
| Likelihood-BoN | 79.45 ± 0.45 | 34.20 ± 0.49 | 28.35 ± 0.45 | 37.30 ± 0.63 | 27.76 ± 0.79 | 41.41 ± 0.54 |
| Likelihood-SMC | 79.15 ± 0.55 | 33.80 ± 0.73 | 28.21 ± 0.46 | 37.50 ± 0.57 | 25.80 ± 0.81 | 40.89 ± 0.56 |
| E-BoN | 79.92 ± 0.06 | 35.08 ± 0.18 | 28.84 ± 0.19 | **42.30 ± 0.41** | 28.28 ± 0.23 | 42.88 ± 0.10 |
| E-SMC | **81.35 ± 0.12** | **35.80 ± 0.20** | **29.02 ± 0.16** | 40.50 ± 0.50 | **29.40 ± 0.32** | **43.21 ± 0.17** |
| **LLaDA-1.5-8B** | | | | | | |
| PC-Sampler | 82.20 | 37.40 | 28.80 | 35.00 | 33.40 | 43.36 |
| Random | 82.11 ± 0.48 | 37.32 ± 0.67 | 28.71 ± 0.41 | 35.10 ± 0.70 | 33.32 ± 0.86 | 43.31 ± 0.62 |
| Majority Voting | 81.80 ± 0.43 | 36.92 ± 0.54 | 28.48 ± 0.46 | 34.70 ± 0.54 | 32.96 ± 0.61 | 42.97 ± 0.51 |
| Likelihood-BoN | 82.64 ± 0.43 | 37.88 ± 0.52 | 28.79 ± 0.50 | 36.30 ± 0.56 | 33.68 ± 0.73 | 43.86 ± 0.54 |
| Likelihood-SMC | 82.03 ± 0.41 | 37.40 ± 0.58 | 28.75 ± 0.48 | 35.90 ± 0.64 | 33.08 ± 0.83 | 43.43 ± 0.57 |
| E-BoN | 82.87 ± 0.12 | 38.12 ± 0.23 | 28.88 ± 0.26 | **40.14 ± 0.50** | **34.12 ± 0.23** | **44.83 ± 0.07** |
| E-SMC | **83.25 ± 0.12** | **38.60 ± 0.32** | **29.02 ± 0.35** | 39.30 ± 0.45 | 33.60 ± 0.32 | 44.75 ± 0.14 |

matured with theoretical progress simplifying the training objective, making the paradigm more stable and efficient [16, 34, 42, 44, 54]. The progress has culminated in the development of large-scale DLMs [18, 19, 32, 52, 55] that achieve performance competitive with autoregressive counterparts, establishing DLMs as a viable and powerful paradigm for language generation.

**Decoding Strategies for Language Models.** Decoding strategy is critical for language models. Unlike AR LMs, which only select the next token [20], DLMs face a more complex two-dimensional problem: choosing both which position to unmask and what token to generate. Uncertainty-based greedy approaches are most common, where the next position is selected based on local confidence signals like maximum probability [8], minimum entropy [6, 27], or largest margin [26], often building upon a simple random-selection baseline [4]. Other approaches impose structure through a semi-autoregressive block-wise order [2, 21, 32], combining positional bias with content-aware confidence scores [24], or explicitly training a planner [25] to achieve more global control. Some methods improve flexibility by allowing already-decoded tokens to be remasked [35, 48].

## 6 Conclusion and Discussion

Generation uncertainty has remained a critical but unquantified aspect for DLMs. Our work is the first to address this gap, introducing Denoising Entropy as a metric designed to explicitly quantify the path-level uncertainty of DLM outputs. We theoretically establish this metric as a bound on generation quality and proposed E-BoN and E-SMC, two algorithms that leverage Denoising Entropy to actively guide the generation process toward paths with lower uncertainty. Our experiments demonstrate that these methods significantly improve both the general quality of DLM outputs and performance on challenging benchmarks. Beyond the specific algorithms proposed, we believe Denoising Entropy serves as a foundational tool for uncertainty quantification in DLMs, opening directions such as developing more calibrated decoding strategies and providing internal reward signals for reinforcement learning.

## Acknowledgements

We thank the anonymous reviewers for their helpful comments and suggestions.

## References


[1] Kushal Arora, Timothy J O'Donnell, Doina Precup, Jason Weston, and Jackie CK Cheung. The stable entropy hypothesis and entropy-aware decoding: An analysis and algorithm for robust natural language generation. *arXiv preprint arXiv:2302.06784*, 2023.

[2] Marianne Arriola, Aaron Gokaslan, Justin T. Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*, 2025. URL https://openreview.net/forum?id=tyEyYT267x.

[3] Koenraad MR Audenaert. A sharp fannes-type inequality for the von neumann entropy. *arXiv preprint quant-ph/0610146*, 2006.

[4] Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in neural information processing systems*, 34:17981–17993, 2021.

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

[6] Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas Nolte, and Brian Karrer. Accelerated sampling from masked diffusion models via entropy bounded unmasking. *ArXiv preprint*, 2025.

[7] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[8] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T. Freeman. Maskgit: Masked generative image transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 11305–11315, 2022. URL https://doi.org/10.1109/CVPR52688.2022.01103.

[9] Sourav Chatterjee and Persi Diaconis. The sample size required in importance sampling. *The Annals of Applied Probability*, 28(2):1099–1135, 2018.

[10] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[11] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[12] Imre Csiszár and János Körner. *Information theory: coding theorems for discrete memoryless systems*. Cambridge University Press, 2011.

[13] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[14] Arnaud Doucet, Nando De Freitas, and Neil Gordon. An introduction to sequential monte carlo methods. In *Sequential Monte Carlo methods in practice*, pages 3–14. Springer, 2001.

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

[16] Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky TQ Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. *Advances in Neural Information Processing Systems*, 37:133345–133385, 2024.

[17] Aaron Gokaslan, Vanya Cohen, Ellie Pavlick, and Stefanie Tellex. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019.

[18] Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, Chenxin An, Peilin Zhao, Wei Bi, Jiawei Han, et al. Scaling diffusion language models via adaptation from autoregressive models. In *The Thirteenth International Conference on Learning Representations*, 2025.

[19] Shansan Gong, Ruixiang Zhang, Huangjie Zheng, Jiatao Gu, Navdeep Jaitly, Lingpeng Kong, and Yizhe Zhang. Diffucoder: Understanding and improving masked diffusion models for code generation. *arXiv preprint arXiv:2506.20639*, 2025.

[20] Alex Graves. Sequence transduction with recurrent neural networks. *arXiv preprint arXiv:1211.3711*, 2012.

[21] Xiaochuang Han, Sachin Kumar, and Yulia Tsvetkov. SSD-LM: semi-autoregressive simplex-based diffusion language model for text generation and modular control. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 11575–11596, 2023. URL https://doi.org/10.18653/v1/2023.acl-long.647.

[22] Zhengfu He, Tianxiang Sun, Qiong Tang, Kuanning Wang, Xuanjing Huang, and Xipeng Qiu. Diffusionbert: Improving generative masked language models with diffusion models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 4521–4534, 2023. URL https://doi.org/10.18653/v1/2023.acl-long.248.

[23] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arxiv:2006.11239*, 2020.

[24] Pengcheng Huang, Shuhao Liu, Zhenghao Liu, Yukun Yan, Shuo Wang, Zulong Chen, and Tong Xiao. Pc-sampler: Position-aware calibration of decoding bias in masked diffusion models. *arXiv preprint arXiv:2508.13021*, 2025.

[25] Zemin Huang, Zhiyang Chen, Zijun Wang, Tiancheng Li, and Guo-Jun Qi. Reinforcing the diffusion chain of lateral thought with diffusion language models. *arXiv preprint arXiv:2505.10446*, 2025.

[26] Jaeyeon Kim, Kulin Shah, Vasilis Kontonis, Sham Kakade, and Sitan Chen. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. *arXiv preprint arXiv:2502.06768*, 2025.

[27] Hyukhun Koh, Minha Jhang, Dohyung Kim, Sangmook Lee, and Kyomin Jung. Plm-based discrete diffusion language models with entropy-adaptive gibbs sampling. *arXiv prints arXiv:2411.06438*, 2024.

[28] Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. *arXiv preprint arXiv:2302.09664*, 2023.

[29] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

[30] Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*, 2023.

[31] Pierre Moral. *Feynman-Kac formulae: genealogical and interacting particle systems with applications*. Springer, 2004.

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

[33] Niklas Nolte, Ouail Kitouni, Adina Williams, Mike Rabbat, and Mark Ibrahim. Transformers can navigate mazes with multi-step prediction. *arXiv preprint arXiv:2412.05117*, 2024.

[34] Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. In *The Thirteenth International Conference on Learning Representations*, 2024.

[35] Fred Zhangzhi Peng, Zachary Bezemek, Sawan Patel, Jarrid Rector-Brooks, Sherwood Yao, Avishek Joey Bose, Alexander Tong, and Pranam Chatterjee. Path planning for masked diffusion model sampling. *arXiv preprint arXiv:2502.03540*, 2025.

[36] Fred Zhangzhi Peng, Shuibai Zhang, Alex Tong, and contributors. Open-dllm: Open diffusion large language models. https://github.com/pengzhangzhi/Open-dLLM, 2025. Model: https://huggingface.co/fredzzp/open-dcoder-0.5B.

[37] Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. MAUVE: Measuring the gap between neural text and human text using divergence frontiers. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[38] Ori Press, Ravid Shwartz-Ziv, Yann LeCun, and Matthias Bethge. The entropy enigma: Success and failure of entropy minimization. *arXiv preprint arXiv:2405.05012*, 2024.

[39] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[40] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

[41] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[42] Subham Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184, 2024.

[43] Claude E Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3): 379–423, 1948.

[44] Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. *Advances in neural information processing systems*, 37:103131–103167, 2024.

[45] Raghav Singhal, Zachary Horvitz, Ryan Teehan, Mengye Ren, Zhou Yu, Kathleen McKeown, and Rajesh Ranganath. A general framework for inference-time scaling and steering of diffusion models. *arXiv preprint arXiv:2501.06848*, 2025.

[46] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[47] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

[48] Guanghan Wang, Yair Schiff, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Remasking discrete diffusion models with inference-time scaling. *arXiv preprint arXiv:2503.00307*, 2025.

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

[50] Jiacheng Xu, Shrey Desai, and Greg Durrett. Understanding neural abstractive summarization models via uncertainty. *arXiv preprint arXiv:2010.07882*, 2020.

[51] Jiacheng Ye, Jiahui Gao, Shansan Gong, Lin Zheng, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Beyond autoregression: Discrete diffusion for complex reasoning and planning. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*, 2025. URL https://openreview.net/forum?id=NRYgUzSPZz.

[52] 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*, 2025.

[53] Kaiwen Zheng, Yongxin Chen, Hanzi Mao, Ming-Yu Liu, Jun Zhu, and Qinsheng Zhang. Masked diffusion models are secretly time-agnostic masked models and exploit inaccurate categorical sampling. *arXiv preprint arXiv:2409.02908*, 2024.

[54] Lin Zheng, Jianbo Yuan, Lei Yu, and Lingpeng Kong. A reparameterized discrete diffusion model for text generation. In *First Conference on Language Modeling*, 2023.

[55] Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Hu, Jun Zhou, Jianfei Chen, Yankai Lin, Ji-Rong Wen, et al. Llada 1.5: Variance-reduced preference optimization for large language diffusion models. *arXiv preprint arXiv:2505.19223*, 2025.

[56] Wenhong Zhu, Hongkun Hao, Zhiwei He, Yiming Ai, and Rui Wang. Improving open-ended text generation via adaptive decoding. *arXiv preprint arXiv:2402.18223*, 2024.

# A Theoretical Analysis

## A.1 Proof of Proposition 1

As the sample mean of marginal entropy over the masked set $\mathcal{M}_t$, $h_{\texttt{DE}}(\mathbf{x}_t)$ is the expected entropy for the single position $\ell$ chosen uniformly from $\mathcal{M}_t$:

$$h_{\texttt{DE}}(\mathbf{x}_t) \triangleq \frac{1}{|\mathcal{M}_t|} \sum_{\ell \in \mathcal{M}_t} H\left(p_\theta(X_0^\ell | \mathbf{x}_t, t)\right) = \mathbb{E}_{\ell \sim \text{Unif}(\mathcal{M}_t)} \left[H\left(p_\theta(X_0^\ell | \mathbf{x}_t, t)\right)\right].$$

This establishes $h_{\texttt{DE}}$ as a measure of average uncertainty over masked tokens.

Our proof relies on the subadditivity property of Shannon entropy. We state it here as a lemma.

**Lemma 1 (Subadditivity of Conditional Entropy).** *For any set of random variables $\{Y_1, \ldots, Y_n\}$ and a conditioning variable $Z$, the joint conditional entropy is bounded by the sum of the individual conditional entropy:*

$$H(Y_1, \ldots, Y_n | Z) \leq \sum_{i=1}^{n} H(Y_i | Z).$$

*Equality holds if and only if the variables $\{Y_i\}$ are conditionally independent given $Z$.*

We now restate and prove Proposition 1.

**Proposition 1 ($h_{\texttt{DE}}$ as an upper bound).** *Oracle State Uncertainty $H_{oracle}(\mathbf{x}_t)$ is upper-bounded by the sum of the marginal entropies, which is directly proportional to State Entropy $h_{DE}(\mathbf{x}_t)$:*

$$H_{oracle}(\mathbf{x}_t) \leq |\mathcal{M}_t| \cdot h_{DE}(\mathbf{x}_t).$$

*Proof.* By definition, $H_{\text{oracle}}(\mathbf{x}_t)$ is the conditional joint entropy $H(\mathbf{X}_{\mathcal{M}_t} | \mathbf{x}_t)$. We can derive the bound as follows:

$$\begin{aligned}
H_{\text{oracle}}(\mathbf{x}_t) &= H(\mathbf{X}_{\mathcal{M}_t} | \mathbf{x}_t) && \text{by Definition 1} \\
&\leq \sum_{\ell \in \mathcal{M}_t} H(X_0^\ell | \mathbf{x}_t) && \text{by Lemma 1} \\
&= \sum_{\ell \in \mathcal{M}_t} H\left(p_\theta(X_0^\ell | \mathbf{x}_t, t)\right) && \text{by definition of DLM output} \\
&= |\mathcal{M}_t| \cdot \left( \frac{1}{|\mathcal{M}_t|} \sum_{\ell \in \mathcal{M}_t} H\left(p_\theta(X_0^\ell | \mathbf{x}_t, t)\right) \right) \\
&= |\mathcal{M}_t| \cdot h_{\texttt{DE}}(\mathbf{x}_t). && \text{by Definition 2}
\end{aligned}$$

The inequality in the second step is strict if the random variables $\{X_0^\ell\}_{\ell \in \mathcal{M}_t}$ are not conditionally independent given the state $\mathbf{x}_t$. In the context of natural language, where token occurrences are highly correlated, this condition for strict inequality is almost always met. □

## A.2 Proof and Corollary of Proposition 2

We first introduce the $\epsilon$-accurate condition, which characterizes the approximation error of the learned reverse kernel relative to the true posterior transition kernel.

**Definition 4 ($\epsilon$-Accurate Denoising Model).** *We say a DLM is $\epsilon$-accurate if, for any time step $t_i$, the Kullback-Leibler (KL) divergence between the true posterior transition kernel $q$ (conditioned on ground truth) and the model's reverse kernel $p_\theta$ is bounded:*

$$\mathbb{E}_{\mathbf{x}_{t_i} \sim q} \left[ D_{\text{KL}} \Big( q(\mathbf{x}_{t_{i-1}} \mid \mathbf{x}_{t_i}) \parallel p_\theta(\mathbf{x}_{t_{i-1}} \mid \mathbf{x}_{t_i}) \Big) \right] \leq \epsilon \,, \tag{11}$$

*where $\epsilon$ is a small constant reflecting the training loss.*

Here, we provide the detailed proof for Proposition 2 and discuss the correlation between negative evidence lower bound $\mathcal{L}$ and $H_{\texttt{DE}}$.

The proof relies on the decomposition of cross-entropy loss:

**Lemma 2 (Decomposition of Cross-Entropy Loss).** *Let $q(X)$ be the true data distribution and $p_\theta(X)$ be the model's estimation distribution. For any random variable $X$, the cross-entropy loss can be decomposed as:*

$$\mathbb{E}_{X_0^\ell \sim q(\cdot|\mathbf{x}_t)}[-\log p_\theta(X_0^\ell|\mathbf{x}_t, t)] = H(q) + D_{KL}(q \parallel p_\theta(X_0^\ell|\mathbf{x}_t, t)),$$

*where $q = q(X_0^\ell|\mathbf{x}_t)$ denotes the true posterior distribution, $H(\cdot)$ is the Shannon entropy and $D_{KL}(\cdot \parallel \cdot)$ is the Kullback-Leibler divergence.*

The proposition states that with the assumption of $\epsilon$-accurate model, the gap between $h_{\texttt{DE}}$ and training loss can be upper-bounded:

$$\frac{1}{|\mathcal{M}_t|} \sum_{\ell \in \mathcal{M}_t} \left(-\log p_{\boldsymbol{\theta}}(x_0^\ell|\mathbf{x}_t, t)\right) - h_{\texttt{DE}}(\mathbf{x}_t) \leq \epsilon + \sqrt{\frac{\epsilon}{2}} logK\,. \tag{12}$$

*Proof.* For an $\epsilon$-accurate model, the KL divergence between the posterior distribution $q(x_0^\ell|\mathbf{x}_t, t)$ and the estimation distribution $p_\theta(x_0^\ell|\mathbf{x}_t, t)$, which is less than $D_{\mathrm{KL}}\Big(q(\mathbf{x}_{t_{i-1}}|\mathbf{x}_{t_i}, t_i) \parallel p_\theta(\mathbf{x}_{t_{i-1}}|\mathbf{x}_{t_i}, t_i)\Big)$, is upper-bounded by $\epsilon$:

$$D_{\mathrm{KL}}(q(x_0^\ell|\mathbf{x}_t, t) \parallel p_\theta(x_0^\ell|\mathbf{x}_t, t)) \leq \epsilon\,.$$

With the upper bound as above, we turn to bound the gap between entropy $H(p_\theta(x_0^\ell|\mathbf{x}_t, t))$ and $H(q(x_0^\ell|\mathbf{x}_t, t))$. We use $q$, $p_\theta$ for simplification.

Let $\delta(q, p_\theta) \triangleq \frac{1}{2}\sum_{x \in \mathcal{V}} |q(x) - p_\theta(x)|$ be the total variation distance on the size-$K$ vocabulary set $\mathcal{V}$. Since $D_{\mathrm{KL}}(q \parallel p_\theta) \leq \epsilon$, we obtain $\delta(q, p_\theta) \leq \sqrt{\frac{\epsilon}{2}}$ based on Pinsker's Inequality [12].

Using Audenaert-Fannes Inequality [3], we obtain the bound of shannon entropy gap:

$$|H(q) - H(p_\theta)| \leq \delta \log(K - 1) + h(\delta),$$

where $h(\delta)$ is binary entropy function $h(\delta) = -\delta log\delta - (1 - \delta)log(1 - \delta)$. Since $h(\delta)$ (bounded with $\delta \leq \sqrt{\frac{\epsilon}{2}}$) is small term that approches 0 when $\epsilon$ approaches 0, we omit this term in the following inequality. Thus, we have $|H(q) - H(p_\theta)| \leq \delta \log(K)$.

Putting the above inequalities together, we average the expected loss over the masked positions with upper bounds $h_{\texttt{DE}}(\mathbf{x}_t) + \epsilon + \sqrt{\frac{\epsilon}{2}} logK$. For any fixed $\mathbf{x}_t$ with $|\mathcal{M}_t| > 0$, linearity of conditional expectation gives

$$\begin{aligned}
&\mathbb{E}_{\mathbf{x}_0 \sim q(\cdot|\mathbf{x}_t)}\left[\frac{1}{|\mathcal{M}_t|}\sum_{\ell \in \mathcal{M}_t} -\log p_\theta(x_0^\ell \mid \mathbf{x}_t, t)\right] - h_{\texttt{DE}}(\mathbf{x}_t) \\
&= \frac{1}{|\mathcal{M}_t|}\sum_{\ell \in \mathcal{M}_t}\Big(\mathbb{E}_{X_0^\ell \sim q(\cdot|\mathbf{x}_t)}\left[-\log p_\theta(X_0^\ell \mid \mathbf{x}_t, t)\right] - H\big(p_\theta(X_0^\ell \mid \mathbf{x}_t, t)\big)\Big)\,.
\end{aligned}$$

Applying the above bound to each position gives

$$\mathbb{E}_{\mathbf{x}_0 \sim q(\cdot|\mathbf{x}_t)}\left[\frac{\mathcal{L}_t(\mathbf{x}_0, \mathbf{x}_t)}{|\mathcal{M}_t|}\right] - h_{\texttt{DE}}(\mathbf{x}_t) \leq \epsilon + \sqrt{\epsilon/2}\log(K - 1) + h(\sqrt{\epsilon/2}).$$

□

**Justification for the Correlation between $\mathcal{L}$ and $H_{\text{DE}}$.** The full objective (negative evidence lower bound, NELBO) $\mathcal{L}$ and our metric $H_{\text{DE}}(\tau)$ are strongly correlated, which is direct corollary of Proposition 2.

We begin with the definition of the NELBO objective:

$$\mathcal{L}(\mathbf{x}_0) = \int_0^1 w(t) \left[ \sum_{\ell \in \mathcal{M}_t} -\log p_\theta(x_0^\ell | \mathbf{x}_t, t) \right] dt, \tag{13}$$

where the weighting function is $w(t) = |\frac{d\alpha_t}{dt}| \frac{1}{1-\alpha_t}$. Assume that $\epsilon \to 0$, we take the approximation by $h_{\text{DE}}$ from Proposition 2:

$$\sum_{\ell \in \mathcal{M}_t} -\log p_\theta(x_0^\ell | \mathbf{x}_t, t) \approx |\mathcal{M}_t| \cdot h_{\text{DE}}(\mathbf{x}_t).$$

Substituting this back into the $\mathcal{L}(\mathbf{x}_0)$:

$$\mathcal{L} \approx \int_0^1 w(t) \cdot |\mathcal{M}_t| \cdot h_{\text{DE}}(\mathbf{x}_t) dt. \tag{14}$$

In expectation, $|\mathcal{M}_t| \approx L(1-\alpha_t)$, where $L$ is the sequence length. Thus, the term $w(t) \cdot |\mathcal{M}_t| \approx L \cdot |\frac{d\alpha_t}{dt}|$ is a strictly positive weighting function of time $t$, which we denote as $w'(t)$. Then the objective is approximated as follows:

$$\mathcal{L} \approx \int_0^1 w'(t) \cdot h_{\text{DE}}(\mathbf{x}_t) dt. \tag{15}$$

In parallel, our metric is the unweighted integral:

$$H_{\text{DE}}(\tau) = \int_0^1 h_{\text{DE}}(\mathbf{x}_t) dt. \tag{16}$$

Since both $\mathcal{L}$ and $H_{\text{DE}}(\tau)$ are integrals of the same underlying function, $h_{\text{DE}}(\mathbf{x}_t)$, with one being uniformly weighted and the other weighted by a positive function $w'(t)$, a strong positive correlation between their values is mathematically expected. This provides rigorous justification for the remark.

In this section, we provide a rigorous theoretical justification for minimizing Path Uncertainty as a proxy for improving generation quality. We establish a direct link between our proposed metric and the divergence from the true data distribution.

### A.3 Distributions over Paths

To rigorously compare the uncertainty of generated samples against real data, we first align their metrics. While $H_{\text{DE}}(\tau)$ is defined over a decoding path $\tau = (\mathbf{x}_{t_N}, \ldots, \mathbf{x}_{t_0})$, real data only consists of static sample sequence $\mathbf{x}_0$. We bridge this gap by defining the **True Path Distribution** via the forward diffusion process.

**Definition 5 (Path Distributions).** *We consider two distributions over the space of paths:*

- ***Model Distribution ($\widehat{\mathcal{T}}$):*** *The distribution of paths generated by the model's reverse process, starting from Gaussian noise:*

$$\mathcal{P}_\theta(\tau) = p(\mathbf{x}_{t_N}) \prod_{i=N}^{1} p_\theta(\mathbf{x}_{t_{i-1}} | \mathbf{x}_{t_i}), \tag{17}$$

  *where the starting state is sampled from the prior $\mathbf{x}_{t_N} \sim p(\cdot)$.*
- ***Reference Distribution ($\mathcal{T}^*$):*** *The distribution of paths induced by applying the forward corruption process $q(\mathbf{x}_t|\mathbf{x}_0)$ to ground-truth data $\mathbf{x}_0 \sim \mathcal{D}_{real}$:*

$$\mathcal{P}_{ref}(\tau) = q(\mathbf{x}_0) \prod_{i=1}^{N} q(\mathbf{x}_{t_i} | \mathbf{x}_{t_{i-1}}, \mathbf{x}_0), \tag{18}$$

*where $\mathbf{x}_0 \sim q(\cdot)$ is sampled from the real dataset $\mathcal{D}_{real}$.*

Here, $\mathcal{P}_{\text{ref}}(\tau)$ represents the distributions of the *ideal* paths that stay perfectly on the manifold of the forward process. For a well-trained model, evaluating $H_{\text{DE}}$ on the reversed reference paths should yield low uncertainty, as the states are consistent with the training objective. Conversely, generated paths $\tau \sim \mathcal{P}_\theta$ that deviate from this manifold (e.g., via hallucinations) are expected to induce higher model uncertainty.

The discrepancy between the generated paths and the true data manifold originates from imperfect approximation at each denoising step. Assume that DLM is $\epsilon$-accurate. Although $\epsilon$ is small locally, the generation process involves a sequence of $N$ transitions. For the full path distributions defined as $\mathcal{P}_{\text{ref}}(\tau)$ (Reference) and $\mathcal{P}_\theta(\tau)$ (Model), the chain rule of KL divergence implies that errors accumulate additively:

$$D_{\text{KL}}(\mathcal{P}_{\text{ref}}(\tau)\|\mathcal{P}_\theta(\tau)) = \sum_{i=N}^{1} \mathbb{E}_{\mathbf{x}_{t_i}\sim q}\left[D_{\text{KL}}(q(\cdot|\mathbf{x}_{t_i})\|p_\theta(\cdot|\mathbf{x}_{t_i}))\right] \leq N \cdot \epsilon\,. \tag{19}$$

Equation 19 establishes that even a well-trained model ($\epsilon \to 0$) can exhibit a non-negligible path divergence $N \cdot \epsilon$ due to the cumulative nature of the generative process. This macroscopic divergence is the root cause of the drift.

### A.4 Entropy Drift and the Divergence Bounds

#### A.4.1 Proof of Proposition 3

We formally quantify the entropy discrepancy as **Entropy Drift**. Let $\mu_{\text{ref}} = \mathbb{E}_{\tau\sim\mathcal{P}_{\text{ref}}}[H_{\text{DE}}(\tau)]$ be the expected Path Entropy evaluated on reference paths (representing the model's uncertainty when guided by ground truth), and let $\mu_\theta = \mathbb{E}_{\tau\sim\mathcal{P}_\theta}[H_{\text{DE}}(\tau)]$ be the expected Path Entropy of the model's autonomous generations.

*Proof.* We now show that the gap between the two Path Entropy quantities lower-bounds the divergence between the generated distribution and the true distribution.

**Lemma 3 (Pinsker's Inequality).** *Let $P$ and $Q$ be two distributions over the path space. For any bounded measurable functional $f(\tau)$ such that $|f(\tau)| \leq B$, the difference in expectations is bounded by the KL divergence:*

$$|\mathbb{E}_{\tau\sim P}[f(\tau)] - \mathbb{E}_{\tau\sim Q}[f(\tau)]| \leq B\sqrt{2D_{\text{KL}}(P\|Q)}\,. \tag{20}$$

Applying Lemma 3 with $f(\tau) := H_{\text{DE}}(\tau)$, we obtain the following bound:

$$D_{\text{KL}}(\mathcal{P}_{\text{ref}}\|\mathcal{P}_\theta) \geq \frac{1}{2B^2}\left(\mu_\theta - \mu_{\text{ref}}\right)^2. \tag{21}$$

□

**Remark.** Equation 21 establishes a necessary condition for high-fidelity generation. Ideally, we expect the generated distribution to match the reference distribution ($D_{\text{KL}} \to 0$), but Proposition 3 implies that it is impossible unless the entropy of the generated paths matches the entropy of the reference paths ($\mu_\theta \to \mu_{\text{ref}}$). Since reference paths $\mathcal{P}_{\text{ref}}$ correspond to ground-truth data evolution where the model is typically confident (low $\mu_{\text{ref}}$), and autonomous generations often drift into high-uncertainty regions (high $\mu_\theta$), minimizing the expected Path Entropy $\mu_\theta$ effectively reduces the gap, implicitly constraining the generative process closer to the true data manifold.

#### A.4.2 Entropy Gap Lower Bounds Divergence

Since computing the high-dimensional divergence $D_{\text{KL}}(\mathcal{P}_{\text{ref}}\|\mathcal{P}_\theta)$ directly is intractable, we propose using Path Entropy as a measurable proxy. We formally show that the KL divergence is upper-bounded with $\epsilon$-accurate condition and lower-bounded with the measurable gap in expected path entropy.

**Theorem 1 (Bounds of KL Divergence).** *Let $\mathcal{P}_{ref}$ and $\mathcal{P}_\theta$ be the reference and generated distributions over path space; $\mu_{ref} = \mathbb{E}_{\tau\sim\mathcal{P}_{ref}}[H_{DE}(\tau)]$ and $\mu_\theta = \mathbb{E}_{\tau\sim\mathcal{P}_\theta}[H_{DE}(\tau)]$ be the Path Entropy expectation on the reference and model distributions, respectively.*
*Assume that DLM is $\epsilon-$ accurate for the model's reverse kernel $p_\theta$ and true transition kernel $q$, the KL-divergence of distributions $\mathcal{P}_{ref}$ and $\mathcal{P}_\theta$ is upper-bounded as in Eq. 19. By applying Pinsker's Inequality to path functionals, the divergence is lower-bounded by the squared entropy gap:*

$$N \cdot \epsilon \geq D_{\mathrm{KL}}(\mathcal{P}_{ref} \| \mathcal{P}_\theta) \geq \frac{1}{2B^2} \left(\mu_\theta - \mu_{ref}\right)^2 . \tag{22}$$

**Implication:** This inequality chain reveals the full mechanism:

- **Source:** Local approximation errors ($\epsilon$) accumulate over $N$ steps.
- **State:** This accumulation creates a distributional shift $D_{\mathrm{KL}}(\mathcal{P}_{\text{ref}} \| \mathcal{P}_\theta)$.
- **Observation:** This shift forces drift in observable statistics, specifically creating a gap $|\mu_\theta - \mu_{\text{ref}}|$.

Therefore, if we observe a large entropy gap (Entropy Drift), it implies that the model has deviated significantly from the reference path. Conversely, our algorithms aim to reduce this entropy gap $|\mu_\theta - \mu_{\text{ref}}| \to 0$, which is a necessary condition for tightening the divergence bound and mitigating the accumulated error.

### A.5 Entropy Drift in Decoding Algorithms

Motivated by Proposition 3, our algorithms, E-BoN and E-SMC, aim to bridge the divergence gap by aligning the expected path entropy of the generated distribution $\mu_\theta$ with that of the reference $\mu_{\text{ref}}$.

**E-BoN.** E-BoN approximates a truncated distribution $\mathcal{P}_{\text{E-BoN}}$ by rejecting paths with high $H_{\text{DE}}$. A potential concern is that the algorithm might retain path samples with extremely low uncertainty (e.g., repetition). For a well-designed model $\mathcal{P}_\theta$, the sampling probability of such paths is low. The model tends to sample problematic sequences with high entropy or sequences with normal entropy; then the finite candidate path set $S = \{\tau_1, ...\tau_N\}$ is unlikely to contain path samples with extremely low entropy. E-BoN reduces the expectation $\mathbb{E}_{\tau\sim\mathcal{P}_\theta}[H_{\text{DE}}(\tau)]$. Considering that $S$ consists solely of samples from the region where $\mu_\theta \leq \mu_{\text{ref}}$, the selection process brings it closer to the reference baseline $\mu_{\text{ref}}$. By Proposition 3, bridging this path entropy gap is a prerequisite for reducing the distributional divergence.

Assume $H_{\text{DE}}(\tau_i) \sim \mathcal{N}(\mu_\theta, \sigma^2)$ with an initial uncertainty gap $\mu_\theta \geq \mu_{\text{ref}}$, the E-BoN strategy identifies the optimal candidate via $\tau_{\text{E-BoN}} = \arg\min_{\tau\in S} H_{\text{DE}}(\tau)$. According to Extreme Value Theory, the expected entropy of this selected path is $\mu_{\text{E-BoN}} = H_{\text{DE}}(\tau_{\text{E-BoN}}) = \mu_\theta - \sigma\sqrt{2\ln N} + \sigma\mathcal{O}(\frac{\ln\ln N}{\sqrt{\ln N}})$. Based on this approximation, if the sample size satisfies $N \leq \exp\left\{\frac{1}{2}\left(\frac{\mu_\theta - \mu_{\text{ref}}}{\sigma}\right)^2\right\}$, the condition $\mu_{\text{E-BoN}} \geq \mu_{\text{ref}}$ holds.

**E-SMC.** E-SMC modifies the transition kernel by re-weighting particles with $w \propto \exp(-\lambda h_{\text{DE}})$ at fixed intervals of $\Delta_{i_r}$ steps, which can be viewed as constructing a calibrated distribution $\mathcal{P}_{\text{E-SMC}}$ that penalizes transitions into high-entropy states. By actively avoiding regions where the model is uncertain (which are characteristic of $\mathcal{P}_\theta$ but rare in $\mathcal{P}_{\text{ref}}$), E-SMC promotes that the accumulated Path Entropy remains low. A similar potential concern is that when the entropy of $\mu_{\text{E-SMC}}$ is lower than the reference $\mu_{\text{ref}}$, reducing the entropy $\mu_{\text{E-SMC}}$ would increase the gap. There we can set the temperature $\lambda$ near the optimal one $\lambda^*$, which enables $\mu_{\text{E-SMC}} = \mu_{\text{ref}}$ and the entropy gap is zero. The existence of $\lambda^*$ can be verified as follows:

**Theorem 2 (Existence and Uniqueness of Optimal Temperature).** *Let $\mu_{\text{E-SMC}}(\lambda)$ denote the expected path entropy of the generative process under the* E-SMC *algorithm with temperature parameter $\lambda \in [0, \infty)$. There exists a unique optimal temperature $\lambda^* \in (0, \infty)$ such that the expected generated entropy matches the reference entropy:*

$$\mu_{\text{E-SMC}}(\lambda^*) = \mu_{ref} . \tag{23}$$

*Proof.* Consider a set of $M$ particles with state entropies $\{H_1, \ldots, H_M\}$ before evaluation, the $m$-th particle is retained with probability $\pi_m(\lambda) = \frac{\exp(-\lambda H_m)}{\sum_{j=1}^{M} \exp(-\lambda H_j)}$ in the E-SMC resampling step. The expectated entropy after resampling is given by the function $\tilde{H} : [0, \infty) \to \mathbb{R}$:

$$\tilde{H}(\lambda) = \sum_{m=1}^{M} \pi_m(\lambda) H_m = \frac{\sum_{m=1}^{M} H_m \exp(-\lambda H_m)}{\sum_{j=1}^{M} \exp(-\lambda H_j)} . \tag{24}$$

**Continuity.** Since the denominator $\sum_{j=1}^{M} \exp(-\lambda H_j)$ is a sum of strictly positive terms for all $\lambda \in \mathbb{R}$, $\tilde{H}(\lambda)$ is continuous and differentiable on $[0, \infty)$.

**Monotonicity.** Since $\frac{d}{d\lambda} e^{-\lambda H} = -H e^{-\lambda H}$, we obtain:

$$\frac{d}{d\lambda} \tilde{H}(\lambda) = -\left( \sum_{m=1}^{M} \pi_m(\lambda) H_m^2 - \left( \sum_{m=1}^{M} \pi_m(\lambda) H_m \right)^2 \right) = -\text{Var}\pi(\lambda)(H).$$

Since the varianceVar$\pi(\lambda)(H)$ is non-negative, the derivative $\frac{d}{d\lambda}$ is non-positive. Assuming the particle entropies $\{H_m\}$ are not all identical, the variance is strictly positive, implying $\tilde{H}(\lambda)$ is strictly decreasing with respect to $\lambda$.

- **Case $\lambda = 0$:** The weights become uniform with $\pi_m(0) = 1/M$. The output distribution is consistent with the original model. Given the Entropy Drift phenomenon, we have $\mu_{\text{E-SMC}} > \mu_{\text{ref}}$ when $\lambda = 0$.
- **Case $\lambda \to \infty$:** The retaining probability of resampling steps concentrates on the particle with the minimum entropy, that is,

$$\lim_{\lambda \to \infty} \pi_m(\lambda) = \begin{cases} 1, & \text{if } H_m = \min_i H_i , \\ 0, & \text{otherwise} . \end{cases}$$

  This corresponds to Model Collapse, where $\lim_{\lambda \to \infty} \mu_{\text{E-SMC}}(\lambda) < \mu_{\text{ref}}$.

The function $\mu_{\text{E-SMC}}(\lambda)$ is continuous and strictly monotonically decreasing on the domain $[0, \infty)$. The reference entropy $\mu_{\text{ref}}$ lies strictly between the boundary values $\lim_{\lambda \to \infty} \mu_{\text{E-SMC}}(\lambda)$ and $\mu_{\text{E-SMC}}(0)$. By the **Intermediate Value Theorem**, there exists a $\lambda^*$ such that $\mu_{\text{E-SMC}}(\lambda^*) = \mu_{\text{ref}}$. Furthermore, due to the strict monotonicity of the function, this $\lambda^*$ is unique. □

**Remark.** While Theorem 2 establishes $\lambda$ as a primary control for entropy, the resampling interval $\Delta i_r$ also governs the final uncertainty. Since the autonomous generative process inherently exhibits *Entropy Drift*, uncertainty accumulates between resampling steps. Consequently, a larger $\Delta i_r$ allows greater accumulation of drift before resampling occurs, leading to a higher expectation of entropy. Thus, $\lambda$ (resampling strength) and $\Delta i_r$ (resampling frequency) jointly determine the accumulated entropy of the generation.

### A.6 Properties of State Entropy

We present theoretical analysis to verify the validity of the state entropy $h_{\text{DE}}(\mathbf{x}_t)$. We begin by verifying its asymptotic behavior at the diffusion process boundaries. Then, we establish its monotonicity with respect to context information.

**Theorem 3 (Asymptotic Behavior of State Entropy).** *Let $\hat{\mathbf{p}}_\theta(\cdot|\mathbf{x}_t, t)$ be a denoising model that approximates the true posterior distribution $q(\mathbf{x}_t|x_0)$. Assume the following limits exist, then the expected state entropy $\mathbb{E}[h_{DE}(\mathbf{x}_t)]$ satisfies:*

- ***Fully-Noised State** ($t \to 1$): As $t \to 1$, $\alpha_t \to 0$ and $\mathbf{x}_1$ becomes independent of the original sequence $\mathbf{x}_0$. The entropy converges to the marginal distribution of tokens in the training data, $p_{data}$*

$$\lim_{t \to 1} \mathbb{E}_{\mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{x}_0)}[h_{DE}(\mathbf{x}_t)] = H(p_{data}) . \tag{25}$$

- ***Noise-Free State*** *($t \to 0$): As $t \to 0$, $\alpha_t \to 1$ and the input $z_0$ is the clean data $x_0$. The model performs reconstruction, and the entropy converges to 0:*

$$\lim_{t \to 0} \mathbb{E}_{\mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{x}_0)}[h_{\texttt{DE}}(\mathbf{x}_t)] = 0\,. \tag{26}$$

*Proof.* The proof examines the two temporal boundaries of the diffusion process.

**Asymptotic behavior as $t \to 1$.** As $t \to 1$, the noise schedule satisfies $\lim_{t\to 1} \alpha_t = 0$. The forward process marginal, $q(\mathbf{x}_t|\mathbf{x}_0) = \text{Cat}(\mathbf{x}_t; \alpha_t \mathbf{x}_0 + (1-\alpha_t)\mathbf{m})$, converges to a categorical distribution concentrated entirely on the mask token:

$$\lim_{t \to 1} q(\mathbf{x}_t|\mathbf{x}_0) = \text{Cat}(\mathbf{x}_t; \mathbf{m})\,,$$

where $\text{Cat}(\cdot; \boldsymbol{\pi})$ denotes the categorical distribution with probability vector $\boldsymbol{\pi}$, and $\mathbf{m}$ is the one-hot vector representing the `[MASK]` token. Consequently, $\mathbf{x}_1 = [\mathbf{m}, \ldots, \mathbf{m}]$ is deterministic and independent of $\mathbf{x}_0$, with all positions masked ($\mathcal{M}_1 = \{1, \ldots, L\}$). For an optimally trained denoising model, the prediction at every masked position converges to the marginal data distribution:

$$\forall \ell \in \mathcal{M}_1, \quad \lim_{t \to 1} \mathbf{p}_0^\ell(\mathbf{x}_t, t) = \mathbf{p}_{\text{data}}\,.$$

The state entropy therefore satisfies:

$$h_{\texttt{DE}}(\mathbf{x}_1) = \frac{1}{L} \sum_{\ell=1}^{L} H(\mathbf{p}_{\text{data}}) = H(\mathbf{p}_{\text{data}})\,.$$

As $\mathbf{x}_1$ is a deterministic state, the expectation is equal to the value itself.

**Asymptotic behavior as $t \to 0$.** In the opposite limit $t \to 1$, $\alpha_t \to 1$. The forward process marginal converges to the clean data:

$$\lim_{t \to 0} q(\mathbf{x}_t|\mathbf{x}_0) = Cat(\mathbf{x}_t, \mathbf{x}_0)\,, \implies \mathbf{x}_0 = \mathbf{x}\,.$$

In this regime, the model has access to nearly the entire ground-truth context. For any masked position $\ell \in \mathcal{M}_t$ where $t$ is small, an optimal model's prediction $\mathbf{p}_0^\ell(\mathbf{x}_0|\mathbf{x}_t, t)$ is conditioned on a context that is almost entirely the ground truth, $\{x_0^j \mid j \notin \mathcal{M}_t\}$. Formally, the predictive distribution converges to a point mass on the true token:

$$\lim_{t \to 0} \hat{\mathbf{p}}_0^\ell(\mathbf{x}_t, t) = \delta_{k, x_0^\ell}\,.$$

The Shannon entropy of a Dirac delta distribution is zero:

$$\lim_{t \to 0} H\left(\hat{\mathbf{x}}_0^\ell(\mathbf{x}_t, t)\right) = H(\delta_{k, x_0^\ell}) = 0\,.$$

As this holds for every masked position, the average entropy $h_{\texttt{DE}}(\mathbf{x}_t)$ converges to zero. The convergence of the expectation follows from the deterministic nature of the limit and the boundedness of the entropy function.

This completes the verification of both asymptotic behaviors. □

The asymptotic analysis confirms that $h_{\texttt{DE}}$ behaves intuitively at process boundaries. We now establish its monotonicity with respect to context information.

**Lemma 4 (Concavity of Entropy).** *The Shannon entropy $H(p)$ is a concave function on the probability simplex $\Delta^{K-1}$.*

*Proof.* The entropy function $H(p) = -\sum_{i=1}^{K} p_i \log p_i$ is a sum of terms $f(p_i) = -p_i \log p_i$. Since $f''(p_i) = -1/p_i < 0$ for $p_i > 0$, each $f(p_i)$ is strictly concave. The sum of concave functions remains concave. □

We now turn to investigating the context-sensitivity property of the state entropy. Let $\mathcal{O}_t \subseteq \{1, \ldots, L\}$ denote the set of indices corresponding to unmasked tokens a time $t$.

**Algorithm 1** Entropy-guided Sequential Monte Carlo

**Input:** DLM $\theta$, number of particles $M$, temperature $\lambda$, resampling interval $\Delta i_r$.
**Initialize:** Sample initial particle set $\mathcal{X}_{t_N} = \{\mathbf{x}_{t_N}^{(m)}\}_{m=1}^{M}$ where $\mathbf{x}_{t_N}^{(m)} \sim p(\mathbf{x}_{t_N})$.
**for** $i = N, N-1, \ldots, 1$ **do**
  **Propagate:** For each particle $m \in \{1, \ldots, M\}$, sample $\mathbf{x}_{t_{i-1}}^{(m)} \sim p_\theta(\cdot|\mathbf{x}_{t_i}^{(m)})$.
  **if** $(N - i + 1) \pmod{\Delta i_r} = 0$ **and** $i > 1$ **then**
    **Evaluate:** For each particle $m$, compute potential $G^{(m)} \leftarrow \Phi(\mathbf{x}_{t_{i-1}}^{(m)}; \lambda)$.
    Normalize to get probabilities: $w^{(m)} \leftarrow G^{(m)} / \sum_{j=1}^{M} G^{(j)}$ for all $m$.
    **Resample:** Draw ancestor indices $\{a^{(m)}\}_{m=1}^{M}$ where $a^{(m)} \sim$ Categorical$(\{w^{(j)}\}_{j=1}^{M})$.
    Update particle set: $\mathcal{X}_{t_{i-1}} \leftarrow \{\mathbf{x}_{t_{i-1}}^{(a^{(m)})}\}_{m=1}^{M}$.
  **end if**
**end for**
**Return:** Final population $\mathcal{X}_{t_0}$.

**Theorem 4 (Context-Sensitivity of State Entropy).** *Let $\mathbf{x}_t$ and $\mathbf{x}'_t$ be variables with observed(unmasked) position sets $\mathcal{O}_t$ and $\mathcal{O}'_t$ respectively, where $\mathcal{O}_t \subset \mathcal{O}'_t$. For any masked position $\ell \notin \mathcal{O}'_t$, revealing additional context does not increase the prediction entropy:*

$$\mathbb{E}[H(\hat{\mathbf{p}}_0^\ell(\mathbf{x}'_t, t))] \leq \mathbb{E}[H(\hat{\mathbf{p}}_0^\ell(\mathbf{x}_t, t))]\,. \tag{27}$$

*Proof.* For an optimal denoising model, the prediction $\hat{\mathbf{p}}_0^\ell(\mathbf{x}_t, t)$ approximates the true posterior distribution $p(x_0^\ell | \{x_0^j\}_{j \in \mathcal{O}_t})$, where $\mathcal{O}_t$ is the set of observed indices in $\mathbf{x}_t$. Let $X = x_0^\ell$ be the target token at position $\ell$, $Y = \{x_0^j\}_{j \in \mathcal{O}_t}$ be the initial context, and $Z = \{x_0^j\}_{j \in \mathcal{O}'_t \setminus \mathcal{O}_t}$ be the additional context revealed in $z'_t$.

By the information-theoretic property that conditioning reduces entropy:

$$H(X|Y, Z) \leq H(X|Y),$$

the entropy of an optimal model's prediction should converge to the true conditional entropy of the data:

$$\mathbb{E}[H(\hat{\mathbf{p}}_0^\ell(\mathbf{x}_t, t))] \to H(X|Y)\,,$$
$$\mathbb{E}[H(\hat{\mathbf{p}}_0^\ell(\mathbf{x}'_t, t))] \to H(X|Y, Z)\,.$$

Combining these inequalities yields the desired result Equation 27. □

Theorem 4 demonstrates that $h_{\text{DE}}$ properly decreases with increasing context information, validating its use as a measure of uncertainty in diffusion language models.

# B Details of Experiments

## B.1 E-SMC Implementation

Here we provide algorithm details and specific formulas for the **Evaluation** step of E-SMC.

**Reward Definition.** The raw State Entropy, $h_{\text{DE}}(\mathbf{x}) \in [0, \log_2 V]$ where $V$ is the vocabulary size, is first converted into a normalized reward signal $r(\mathbf{x}) \in [0, 1]$. This is done by inverting and scaling the entropy:

$$r(\mathbf{x}) \triangleq \frac{\log_2 V - h_{\text{DE}}(\mathbf{x})}{\log_2 V} = 1 - \frac{h_{\text{DE}}(\mathbf{x})}{\log_2 V}. \tag{28}$$

This formulation ensures that a state with minimum entropy (0) receives the maximum reward (1), and a state with maximum entropy ($\log_2 V$) receives the minimum reward (0).

**Potential Function.** The reward $r(\mathbf{x})$ is then used to compute the potential score $G(\mathbf{x})$ for each particle, which determines its fitness for resampling. We employ a Gibbs potential function, controlled by a temperature parameter $\lambda > 0$:

$$G(\mathbf{x}) \triangleq \exp(\lambda \cdot r(\mathbf{x})). \tag{29}$$

The parameter $\lambda$ controls the selection pressure. A higher $\lambda$ value more strongly favors particles with high rewards (low entropy), leading to a more aggressive pruning of undesirable trajectories.

**Resampling Probabilities.** The probability $\pi^{(k)}$ of selecting particle $\mathbf{x}_{t_{i-1}}^{(k)}$ during the resampling step is given by its normalized potential score:

$$\pi^{(k)} = \frac{G(\mathbf{x}_{t_{i-1}}^{(k)})}{\sum_{j=1}^{K} G(\mathbf{x}_{t_{i-1}}^{(j)})} = \frac{\exp(\lambda \cdot r(\mathbf{x}_{t_{i-1}}^{(k)}))}{\sum_{j=1}^{K} \exp(\lambda \cdot r(\mathbf{x}_{t_{i-1}}^{(j)}))}. \tag{30}$$

This is equivalent to applying a softmax function to the scaled rewards of all particles in the population. The new population is then formed by drawing $K$ samples with replacement from the current population according to these probabilities.

### B.2 Decoding Strategies

Here we provide a detailed description of the decoding strategies we evaluate.

**Uniform [4]** At each step, this sampler unmasks a fixed number of tokens at positions chosen uniformly at random from the set of currently masked tokens. This is the vanilla sampler in DLMs.

**Confidence [8]** Sampler with confidence score selects tokens for positions where DLM prediction is most confident, defined as the position $\ell$ that maximizes the probability of the most likely token, i.e., $\arg\max_{\ell \in \mathcal{M}_t} \max(\hat{\mathbf{p}}_0^\ell)$.

**Entropy [6]** Sampler with entropy selects tokens for positions where the DLM prediction is least ambiguous, defined as the position $\ell$ that minimizes the Shannon entropy of the predicted probability distribution, i.e., $\arg\min_{\ell \in \mathcal{M}_t} H(\hat{\mathbf{p}}_0^\ell)$.

**Margin [26]** Sampler with margin selects tokens for positions with the clearest distinction between the top two candidates, defined as the position $\ell$ that maximizes the margin $p_{(1)}^\ell - p_{(2)}^\ell$, where $p_{(1)}^\ell$ and $p_{(2)}^\ell$ are the highest and second-highest probabilities in $\hat{\mathbf{p}}_0^\ell$.

**EB-Sampler [6]** Entropy-Bounded sampler is an adaptive method that unmasks a variable number of tokens, $k$, at each step. It selects the $k$ tokens with the lowest entropy, where $k$ is dynamically determined by ensuring the cumulative entropy of the selected tokens remains bounded by a predefined threshold $\gamma$.

**Semi-AR [32]** Semi-AutoRegressive sampler partitions the sequence into contiguous blocks. It generates tokens within each block in a parallel, non-autoregressive manner, while the blocks themselves are generated sequentially from left to right.

**Fast-dLLM [49]** This sampler accelerates the generation of each block by adaptively unmasking a variable number of tokens at each step. The number of tokens is determined based on whether their prediction confidence exceeds a given threshold, allowing a block to be completed in fewer steps than prescheduled.

**PC-Sampler [24]** Positional-Confidence sampler selects tokens based on a score that combines a confidence measure (derived from the cross-entropy against a background token distribution) with an exponentially decaying positional bias, thus prioritizing tokens that are confidently predicted and appear earlier in the sequence.

**P2 [35]** This is a multi-stage sampler that employs a self-correction mechanism. It first generates a high-proportion draft of the sequence by filling the most confident positions. It then enters an iterative refinement phase, where in each step it identifies the *least* confident generated tokens, re-masks them, and immediately re-predicts their content based on the updated context. This process allows DLM to revise and improve its initial predictions.

### B.3 Experimental Configurations

Here we provide a detailed description of the configurations for each experiment. Table 4 summarizes the hyperparameters used for experiments on `LLaDA` model. Table 5 summarizes the hyperparameters used for experiments on `Open-dCoder` model.

**Note on Selection Temperature Parameter.** `LLaDA` experiments utilize a selection temperature parameter ($T_{\text{sel}} = 0.1$) while `Open-dCoder` experiments do not require this parameter. This difference is because of the generation temperature settings and their impact on path diversity.

In `LLaDA` experiments, generation temperature is set to 0, making the token prediction process deterministic. Under such deterministic conditions, multiple runs would yield identical paths, preventing path exploration for E-BoN and E-SMC algorithms. The selection temperature introduces controlled randomness during the position selection phase of decoding strategies.

Specifically, when selecting the next $k$ positions to unmask based on confidence scores $\mathbf{s} = [s_1, s_2, \ldots, s_n]$, the selection temperature modifies the standard top-$k$ selection through stochastic sampling:

$$\begin{aligned} \text{Top-}2k \text{ candidates:} \quad & \mathbf{s}_{\text{top}} = \text{topk}(\mathbf{s}, 2k) \\ \text{Temperature-scaled probabilities:} \quad & p_i = \frac{\exp(s_i/T_{\text{sel}})}{\sum_{j \in \text{top-}2k} \exp(s_j/T_{\text{sel}})} \\ \text{Stochastic selection:} \quad & \text{positions} \sim \text{Multinomial}(\mathbf{p}, k) \end{aligned}$$

where $T_{\text{sel}}$ controls the randomness level: lower values favor high-confidence positions (approaching deterministic top-$k$ as $T_{\text{sel}} \to 0$), while higher values increase selection diversity.

`Open-dCoder` experiments use a generation temperature of 0.8, which introduces stochasticity in the token prediction process and the generation process already satisfies the diversity requirements for path search algorithms.

### B.4 Results

In this section, we present supplementary experimental results that provide further details supporting the findings discussed in the main text.

Table 6 reports the complete raw data from Figure 3.

Table 6 extends Table 1 with Diversity. Diversity is used to assess whether repetitive degeneration occurs in generated content. For a sequence of length $L$ that contains $K$ distinct tokens, with each token $k$ occurring $L_k$ times. Diversity $D$ is computed as $D = -\sum_{k=1}^{K} \frac{L_k}{L} \log \frac{L_k}{L}$.

Table 8 compares decoding algorithms with Greedy Search, where $c$ represents the number of candidates and $s$ represents the beam size.

Table 10 presents the runtime analysis of different decoding algorithms.

Table 9 provides the detailed numerical values underlying Figure 4.

Figure 5 visually illustrates the evolution of state entropy across different particles throughout the decoding process.

**Runtime and Memory on LLaDA.** Table 11 reports the average per-sample time and peak GPU memory on `LLaDA-Instruct-8B` (length 256, 256 steps, resampling interval 32, five particles). The overhead comes from sampling multiple paths: serial time grows linearly with the number of

Table 4: Experimental Configurations for Different Tasks of `LLaDA` models.

| Parameter | GSM. | Math. | GPQA. | Count. | Sudoku. |
|---|---|---|---|---|---|
| **Task** | | | | | |
| Few-shot examples | 4 | 4 | 5 | 3 | 5 |
| **Generation** | | | | | |
| Generation length | 256 | 1024 | 256 | 128 | 128 |
| Steps | 256 | 1024 | 256 | 128 | 128 |
| Temperature | 0 | 0 | 0 | 0 | 0 |
| CFG scale | 0 | 0 | 0 | 0 | 0 |
| **PC-Sampler** | | | | | |
| $\lambda$ | 0.25 | 0.25 | 0.25 | 0.5 | 0.0 |
| $\alpha$ | 10 | 10 | 10 | 10 | 10 |
| **E-BoN & E-SMC** | | | | | |
| Number of particles | 5 | 5 | 5 | 5 | 5 |
| $\lambda_{\text{weight}}$ | 5.0 | 5.0 | 5.0 | 10.0 | 5.0 |
| Selection temperature | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| **E-SMC** | | | | | |
| Resample interval | 64 | 256 | 64 | 32 | 32 |

Table 5: Experimental configurations for `Open-dCoder`.

| Parameter | HumanEval | HumanEval+ | MBPP | MBPP+ |
|---|---|---|---|---|
| **Generation** | | | | |
| Generation length | 128 | 128 | 128 | 128 |
| Steps | 128 | 128 | 128 | 128 |
| Temperature | 0.8 | 0.8 | 0.8 | 0.8 |
| **E-BoN & E-SMC** | | | | |
| Number of particles | 5 | 5 | 5 | 5 |
| $\lambda_{\text{weight}}$ | 5.0 | 5.0 | 5.0 | 5.0 |
| **E-SMC** | | | | |
| Resample interval | 32 | 32 | 32 | 32 |

Table 6: Mean/Std of $H_{\text{DE}}$, mean log PPL, and their correlation (Corr.) across steps.

| Step | Mean $H_{\text{DE}}$ | Std $H_{\text{DE}}$ | Log PPL | Corr. |
|---|---|---|---|---|
| 16 | 6.8080 | 0.4320 | 5.731 | 0.891 |
| 32 | 6.1183 | 0.3927 | 5.083 | 0.863 |
| 64 | 5.7352 | 0.3887 | 4.661 | 0.852 |
| 128 | 5.5242 | 0.3823 | 4.396 | 0.853 |
| 256 | 5.3815 | 0.4018 | 4.176 | 0.851 |
| 512 | 5.2318 | 0.3891 | 3.951 | 0.847 |
| 1024 | 5.1197 | 0.3924 | 3.729 | 0.854 |

particles, while parallel execution cuts wall time at the cost of higher peak memory. Because E-SMC is plug-in, it can be paired with an efficient sampler: on GSM8K, Fast-dLLM with E-SMC reaches 79.6 accuracy, above both Fast-dLLM (78.2) and PC-Sampler (79.3), with a parallel run time of 5.32 s against 10.61 s for PC-Sampler.

### B.5 Statistical Robustness

E-BoN and E-SMC are stochastic, whereas PC-Sampler is deterministic at temperature 0 and is treated as a fixed reference. Table 12 reports the mean and standard deviation over five independent seeds on all benchmarks, together with the improvement in average accuracy over PC-Sampler and its 95% confidence interval.

### B.6 Group-wise Decoding

The path definition only requires a sequence of partially denoised states, so Denoising Entropy applies directly to decoding that updates several positions at once: after each group update, State Entropy is computed over the remaining masked positions, E-BoN ranks the resulting paths, and E-SMC resamples between group updates. Table 13 unmasks two positions per step with `LLaDA-Instruct-8B`. E-SMC improves the five-task average from 37.99 to 40.77 under this coarser setting, so path uncertainty remains informative when each transition changes more than one token.

Table 7: **Comparison of decoding strategies via ablation studies on hyperparameters.** The table analyzes the impact of the number of denoising steps ($S$), number of particles ($K$), and resampling interval ($\Delta i_r$). Perplexity is evaluated on two models: `GPT2-Large` and `Llama-3-8B`. E-SMC consistently shows the best performance, which can be achieved without compromising diversity in certain configurations.

| Configuration | | | Perplexity ↓ | | | Diversity ↑ | | |
|---|---|---|---|---|---|---|---|---|
| $S$ | $K$ | $\Delta i_r$ | Vanilla | E-BoN | E-SMC | Vanilla | E-BoN | E-SMC |
| 128 | 4 | 32 | 85.3 / 84.5 | 66.4 / 66.1 | **63.4** / **61.9** | 5.53 | 5.47 | 5.48 |
| 256 | 4 | 32 | 68.5 / 66.9 | 52.3 / 55.2 | **47.7** / **51.5** | 5.45 | 5.39 | 5.40 |
| 128 | 4 | 32 | 85.3 / 84.5 | 66.4 / 66.1 | **63.4** / **61.9** | 5.53 | 5.47 | 5.48 |
| 128 | 8 | 32 | 85.3 / 84.5 | 60.3 / 59.8 | **56.0** / **59.0** | 5.53 | 5.42 | 5.43 |
| 256 | 2 | 32 | 68.5 / 66.9 | 60.0 / 59.1 | **55.5** / **57.0** | 5.45 | 5.43 | 5.43 |
| 256 | 4 | 32 | 68.5 / 66.9 | 52.3 / 55.2 | **47.7** / **51.5** | 5.45 | 5.39 | 5.40 |
| 256 | 8 | 32 | 68.5 / 66.9 | 47.5 / 51.1 | **40.4** / **45.1** | 5.45 | 5.37 | 5.31 |
| 256 | 4 | 8 | 68.5 / 66.9 | 52.3 / 55.2 | **43.9** / **44.4** | 5.45 | 5.39 | 5.31 |
| 256 | 4 | 16 | 68.5 / 66.9 | 52.3 / 55.2 | **45.1** / **48.1** | 5.45 | 5.39 | 5.35 |
| 256 | 4 | 32 | 68.5 / 66.9 | 52.3 / 55.2 | **47.7** / **51.5** | 5.45 | 5.39 | 5.40 |
| 256 | 4 | 64 | 68.5 / 66.9 | 52.3 / 55.2 | **50.2** / **51.8** | 5.45 | 5.39 | 5.42 |
| 256 | 4 | 128 | 68.5 / 66.9 | 52.3 / 55.2 | **53.4** / **54.1** | 5.45 | 5.39 | 5.46 |

Table 8: **Comparison with Greedy Search.** Perplexity on `GPT2-Large` / `Llama-3-8B`. Greedy Search lowers perplexity by restricting the search space but significantly degrades diversity compared to E-SMC.

| Experiment | Perplexity ↓ | Diversity ↑ |
|---|---|---|
| Vanilla | 68.5 / 66.9 | 5.45 |
| E-BoN ($K$=4) | 52.3 / 55.2 | 5.39 |
| E-SMC ($K$=4, $\Delta i_r$=128) | 53.4 / 54.8 | **5.46** |
| Greedy ($c$=2, $s$=1) | 32.7 / 35.9 | 5.17 |
| Greedy ($c$=4, $s$=1) | 19.9 / 24.2 | 4.83 |
| Greedy ($c$=8, $s$=1) | 14.0 / 18.7 | 4.59 |
| Greedy ($c$=16, $s$=1) | 10.6 / 15.7 | 4.40 |
| Greedy ($c$=8, $s$=2) | 11.0 / 15.7 | 4.42 |
| Greedy ($c$=8, $s$=4) | 8.8 / 13.7 | 4.09 |
| Greedy ($c$=8, $s$=8) | **7.9** / **12.1** | 3.84 |

Table 9: Accuracy gain (%) of E-BoN and E-SMC over each baseline sampler on code generation with `Open-dCoder` (HE = HumanEval).

| Strategies | HE | HE+ | MBPP | MBPP+ | Avg. |
|---|---|---|---|---|---|
| P2 | 19.3 | 17.0 | 16.3 | 23.7 | 19.1 |
| *w/* E-BoN | 0.4↑ | 0.9↑ | 0.5↑ | 0.2↑ | 0.5↑ |
| *w/* E-SMC | 0.2↑ | 0.7↑ | 0.4↑ | 0.3↑ | 0.4↑ |
| Uniform | 2.7 | 2.6 | 1.1 | 1.6 | 2.0 |
| *w/* E-BoN | 2.2↑ | 1.6↑ | 1.3↑ | 1.5↑ | 1.7↑ |
| *w/* E-SMC | 1.8↑ | 1.3↑ | 1.5↑ | 1.3↑ | 1.5↑ |
| Confidence | 7.0 | 6.3 | 2.9 | 5.9 | 5.5 |
| *w/* E-BoN | 3.4↑ | 2.7↑ | 2.2↑ | 3.8↑ | 3.1↑ |
| *w/* E-SMC | 3.1↑ | 2.4↑ | 2.1↑ | 4.0↑ | 2.9↑ |
| Entropy | 7.6 | 6.4 | 6.1 | 7.5 | 6.9 |
| *w/* E-BoN | 3.4↑ | 3.8↑ | 2.7↑ | 2.4↑ | 3.1↑ |
| *w/* E-SMC | 3.7↑ | 4.2↑ | 3.0↑ | 2.6↑ | 3.4↑ |
| Margin | 4.3 | 3.5 | 2.2 | 3.9 | 3.5 |
| *w/* E-BoN | 3.7↑ | 3.3↑ | 2.1↑ | 2.3↑ | 2.8↑ |
| *w/* E-SMC | 3.3↑ | 3.2↑ | 2.2↑ | 2.0↑ | 2.7↑ |

Table 10: **Runtime analysis comparison.** We report the computation time (in seconds) across different hyperparameter configurations ($S$, $K$, $\Delta i_r$) on a single NVIDIA L40 GPU with `MDLM`. Parallel implementations (**Par.**) reduce latency compared to sequential baselines (**Seq.**) as the number of particles $K$ increases.

| Configuration | | | Baseline | E-BoN | | E-SMC | |
|---|---|---|---|---|---|---|---|
| $S$ | $K$ | $\Delta i_r$ | Vanilla | Seq. | Par. | Seq. | Par. |
| 256 | 2 | 64 | 7.995 | 13.273 | 8.091 | 13.635 | 8.650 |
| 256 | 4 | 64 | 7.995 | 26.981 | 11.236 | 27.330 | 12.066 |
| 256 | 8 | 64 | 7.995 | 53.718 | 17.943 | 54.617 | 19.329 |
| 1024 | 2 | 256 | 20.256 | 53.848 | 32.790 | 54.652 | 34.644 |
| 1024 | 4 | 256 | 20.256 | 106.605 | 46.928 | 108.457 | 47.771 |
| 1024 | 8 | 256 | 20.256 | 213.582 | 75.134 | 215.317 | 76.474 |

Table 11: Time and peak GPU memory per sample on `LLaDA-Instruct-8B`.

| Method | Paths | Mode | Time (s) | Mem. (MB) |
|---|---|---|---|---|
| Semi-AR | 1 | serial | 10.57 | 15610.0 |
| PC-Sampler | 1 | serial | 10.61 | 15610.5 |
| *w/* E-SMC | 5 | serial | 52.81 | 15610.5 |
| *w/* E-SMC | 5 | parallel | 31.17 | 42810.0 |
| Fast-dLLM | 1 | serial | 1.80 | 16218.9 |
| *w/* E-SMC | 5 | serial | 8.82 | 16218.9 |
| *w/* E-SMC | 5 | parallel | 5.32 | 42232.5 |

Table 12: Accuracy (%) on `LLaDA` models. E-BoN and E-SMC raise the average accuracy over PC-Sampler by 1.4 to 2.1 points.

| Method | GSM. | Math. | GPQA. | Count. | Sudoku. | Avg. | $\Delta$ Avg. [95% CI] |
|---|---|---|---|---|---|---|---|
| **LLaDA-Instruct-8B** | | | | | | | |
| PC-Sampler | 79.30 | 34.00 | 28.57 | 36.30 | 27.60 | 41.15 | – |
| *w/* E-BoN | 79.92 $\pm$ 0.06 | 35.08 $\pm$ 0.18 | 28.84 $\pm$ 0.19 | **42.30 $\pm$ 0.41** | 28.28 $\pm$ 0.23 | 42.88 $\pm$ 0.10 | +1.73 [1.61, 1.85] |
| *w/* E-SMC | **81.35 $\pm$ 0.12** | **35.80 $\pm$ 0.20** | **29.02 $\pm$ 0.16** | 40.50 $\pm$ 0.51 | **29.40 $\pm$ 0.32** | **43.21 $\pm$ 0.17** | +2.06 [1.85, 2.27] |
| **LLaDA-1.5-8B** | | | | | | | |
| PC-Sampler | 82.20 | 37.40 | 28.80 | 35.00 | 33.40 | 43.36 | – |
| *w/* E-BoN | 82.87 $\pm$ 0.12 | 38.12 $\pm$ 0.23 | 28.88 $\pm$ 0.26 | **40.14 $\pm$ 0.50** | **34.12 $\pm$ 0.23** | **44.83 $\pm$ 0.07** | +1.47 [1.38, 1.56] |
| *w/* E-SMC | **83.25 $\pm$ 0.12** | **38.60 $\pm$ 0.32** | **29.02 $\pm$ 0.35** | 39.30 $\pm$ 0.45 | 33.60 $\pm$ 0.32 | 44.75 $\pm$ 0.14 | +1.40 [1.22, 1.57] |

Table 13: Accuracy (%) on `LLaDA-Instruct-8B` with one or two tokens unmasked per step (E-SMC: mean $\pm$ standard deviation). The gain over PC-Sampler grows from $+2.06$ to $+2.78$.

| Method | GSM. | Math. | GPQA. | Count. | Sudoku. | Avg. |
|---|---|---|---|---|---|---|
| PC-Sampler, 1 token/step | 79.30 | 34.00 | 28.57 | 36.30 | 27.60 | 41.15 |
| *w/* E-SMC | 81.35 $\pm$ 0.12 | 35.80 $\pm$ 0.20 | 29.02 $\pm$ 0.16 | 40.50 $\pm$ 0.50 | 29.40 $\pm$ 0.32 | **43.21 $\pm$ 0.17** |
| PC-Sampler, 2 tokens/step | 78.32 | 34.40 | 24.33 | 28.90 | 24.00 | 37.99 |
| *w/* E-SMC | 81.43 $\pm$ 0.19 | 36.00 $\pm$ 0.32 | 24.91 $\pm$ 0.26 | 37.50 $\pm$ 0.51 | 24.00 $\pm$ 0.32 | **40.77 $\pm$ 0.31** |

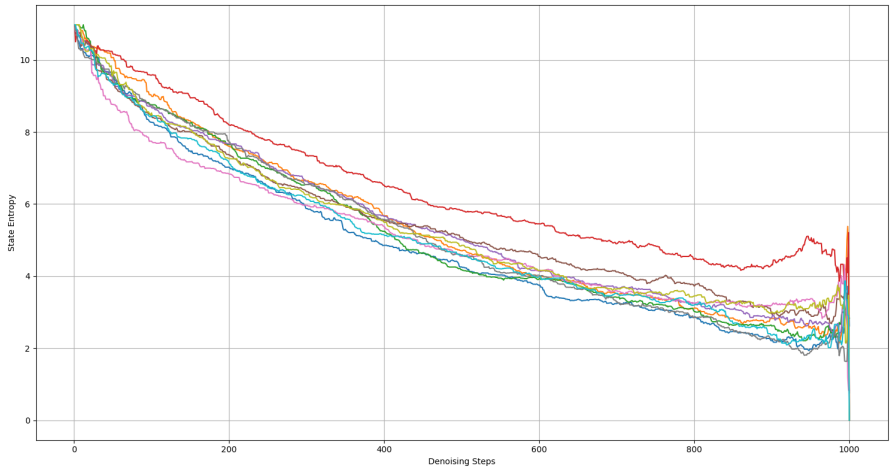



Figure 5: Paths of state entropy over denoising steps for 8 particles, where different colors represent distinct particles. It visually illustrates how the state entropy evolves across each particle throughout the decoding process.

## NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and Section 1 state our main claims: formalizing path uncertainty in DLMs, introducing Denoising Entropy, and proposing E-BoN and E-SMC for entropy-guided decoding. These claims are supported by the theoretical analysis in Section 3.4 and the experiments in Section 4.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: The paper includes a limitations discussion in Section 6.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

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

Answer: [Yes]

Justification: The assumptions for the main theoretical results are stated in Section 3.4 and Appendix A.1, including the $\epsilon$-accurate DLM assumption in Definition 4. Proofs for Propositions 1, 2, and 3 are provided.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: Section 4 and Appendix B.1, B.2, B.3, and B.4 describe the models, datasets, baselines, decoding algorithms, hyperparameters, and evaluation settings used in experiments.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [Yes]

   Justification: We have made the code available at https://github.com/LINs-lab/DenoisingEntropy.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Section 4.3 and Appendix B.3 report the evaluated models, datasets, prompts, generation steps, temperatures, particle counts, selection temperatures, and resampling intervals. Appendix B.2 and B.1 further document the baseline samplers and the E-SMC implementation.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: For the validation experiment, Appendix Table 6 reports mean values, standard deviations, and per-sample correlations across generated samples. For the benchmark evaluations, the paper reports dataset-level accuracies on the full evaluation splits.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Appendix Table 10 reports runtime and memory details.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

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

   Answer: [N/A]

   Justification: [N/A]

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: Section 6 includes a broader impacts discussion describing both positive impacts and negative risks.

Guidelines:

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: [N/A].

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Appendix B.4 summarizes the main model and dataset assets used in the paper.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [N/A]

    Justification: [N/A].

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: [N/A]

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: [N/A]

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [Yes]

    Justification: Language models are central to both the methodology and evaluation in this work. Section 2, Section 4, and Appendix B.3 describe the DLMs and AR LM evaluators used in the study.

    Guidelines:

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