SEARCH OR ACCELERATE: CONFIDENCE-SWITCHED POSITION BEAM SEARCH FOR DIFFUSION LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion Language Models (DLMs) generate text by iteratively denoising a masked sequence, repeatedly deciding *which positions* to commit at each step. Standard decoding follows a greedy rule, unmask the most confident positions, yet this local choice can lock the model into a suboptimal unmasking order, especially on reasoning-heavy prompts. We present **S**earch **O**r **A**ccele**R**ate (`SOAR`), a **training-free** decoding algorithm that adapts its behavior to the model's uncertainty. When confidence is low, `SOAR` briefly widens the search over alternative unmasking decisions to avoid premature commitments; when confidence is high, it collapses the search and decodes many positions in parallel to reduce the number of denoising iterations. Across mathematical reasoning and code generation benchmarks (GSM8K, MBPP, HumanEval) on DREAM-7B and LLADA-8B, `SOAR` improves generation quality while maintaining competitive inference speed, offering a practical way to balance quality and efficiency in DLM decoding.

1 INTRODUCTION

Diffusion Language Models (DLMs) (Ye et al., 2025; Nie et al., 2025; Zhu et al., 2025a) have recently emerged as a promising alternative to autoregressive (AR) generation (Touvron et al., 2023; Yang et al., 2025a). Unlike AR models that decode tokens sequentially in a fixed left-to-right order, DLMs iteratively refine a partially masked sequence and can decode multiple token positions in parallel. In practice, most mask-based DLM decoders follow a simple greedy rule: at each denoising step, they unmask the positions with the highest prediction confidence and keep the remaining positions as `[MASK]`.

This greedy unmasking rule is attractive for efficiency, but it also makes a strong local decision about *which positions to commit next*. Importantly, for DLMs this unmasking order is not fixed (unlike left-to-right AR decoding), and different unmasking schedules can lead to different final generations. This motivates our central question: **Can we improve DLM decoding by explicitly exploring alternative unmasking orders beyond greedy selection?**

A natural signal for guiding such exploration is the model's prediction confidence, which has been leveraged in prior work for adaptive-parallel decoding (Chen et al., 2025). In our setting, we observe the same qualitative pattern: on GSM8K (Cobbe et al., 2021), examples decoded with higher average token confidence tend to be more accurate (Figure 1). This suggests that searching for decoding trajectories that maintain higher confidence may improve downstream quality, especially on reasoning-heavy prompts.

Inspired by the success of beam search in AR decoding (Freitag & Al-Onaizan, 2017), we adapt this paradigm to DLMs. A key difference is that, while AR beam search expands candidates primarily in the token/vocabulary space, DLM decoding admits a distinct search dimension: the *position space*, i.e., the order in which masked positions are unmasked. We find that widening search in this position space consistently improves output quality across benchmarks, but it also increases computation roughly linearly with beam width. This leads to a practical challenge: **How can we retain the gains from position-space search without incurring prohibitive inference latency?**

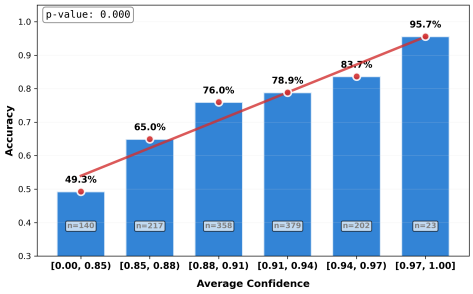

Figure 1: Accuracy and average confidence on GSM8K with DREAM-7B-Base. All questions are divided into 6 bins by their average decoding confidence, with n indicating the sample size in each bin. The red solid line represents the trend line.

To address this, we propose SOAR, a **training-free** confidence-switched position beam search procedure that dynamically balances exploration and speed based on the model's confidence at each step. When the model is uncertain, SOAR widens the beam to explore multiple unmasking choices and avoid premature commitments; when the model is confident, it decodes many high-confidence positions in parallel and collapses the beam to accelerate inference. This enables the model to spend compute when needed and decode quickly when possible, yielding a practical quality–speed trade-off for DLM inference.

Our contributions are summarized as follows:

- We introduce **P**osition **B**eam **S**earch (PBS), a training-free decoding method that explores alternative unmasking sequences in the position space and consistently improves output quality across multiple benchmarks.
- We propose SOAR, a training-free adaptive inference algorithm that switches between parallel decoding and position-space search based on per-step confidence, effectively balancing speed and quality.
- We demonstrate that SOAR integrates naturally with variable-length DLM decoding methods and multiple unmasking metrics, highlighting its flexibility and compatibility with advanced decoding strategies.

2 RELATED WORKS

2.1 DIFFUSION LANGUAGE MODELS (DLMS)

DLMs have emerged as a competitive alternative to autoregressive models (Touvron et al., 2023; Yang et al., 2025a) for text generation (Nie et al., 2025; Ye et al., 2025; Gong et al., 2025a). Prominent architectures include those initialized from pre-trained models, such as DREAM (Ye et al., 2025) and DiffuLLaMA (Gong et al., 2025a), and those trained from scratch like LLADA (Nie et al., 2025). Unlike sequential generation, these mask-based DLMs typically employ an iterative "predict-all, then commit-confident" loop, leveraging bidirectional context to refine predictions across denoising steps.

2.2 OPTIMIZED INFERENCE FOR DLMS

The iterative nature of DLM decoding poses unique efficiency challenges. Recent research has focused on: (1) **Architectural acceleration**, such as specialized KV-cache mechanisms and block-autoregressive structures (Wu et al., 2025; Nguyen-Tri et al., 2025; Huang et al., 2025a; Liu et al., 2025; Ma et al., 2025); and (2) **Decoding strategy optimization**, including confidence-based parallel decoding and early-commit methods to reduce denoising steps (Wu et al., 2025; Yang et al., 2025b; Gao et al., 2025; Li et al., 2025b;a). To address the lack of revision in standard DLMs, some works explore self-correction or latent refinement (Huang et al., 2025b; Mounier & Idehpour, 2025; Zhu et al., 2025b). A concurrent work, Order-Token Search (Anonymous, 2025), explores

both position and token dimensions but increases latency. In contrast, SOAR dynamically switches between position-space search and parallel decoding, achieving superior performance without sacrificing speed.

3 METHODOLOGY

3.1 PRELIMINARIES OF DLM INFERENCE

Let $x = [x_1, \ldots, x_L] \in \mathcal{V}^L$ be a text sequence. Mask-based DLMs (Nie et al., 2025) generate text by iteratively unmasking a fully masked sequence $x_T = [\text{[MASK]}]^L$ over T steps. At each step t, the model predicts the probability distribution $\mathbf{p}_{t,i} = P(x_{0,i}|x_t)$ for each masked position $i \in M_t = \{j : x_{t,j} = \text{[MASK]}\}$.

The decoding strategy selects a subset of positions $\mathcal{I}_t \subseteq M_t$ to unmask. A standard greedy approach selects $|\mathcal{I}_t| = k$ positions with the highest confidence scores:

$$\mathcal{I}_t = \text{top-k}_{i \in M_t} \left(\max_{v \in \mathcal{V}} \mathbf{p}_{t,i}[v] \right), \quad x_{t-1,i} = \begin{cases} \arg\max \mathbf{p}_{t,i} & \text{if } i \in \mathcal{I}_t \\ \text{[MASK]} & \text{otherwise} \end{cases} \tag{1}$$

The core challenge in DLM inference is optimizing the selection of $\mathcal{I}_t$ (the unmasking order) to balance generation quality and efficiency.

3.2 POSITION BEAM SEARCH (PBS)

However, the overall confidence of sequences generated through greedy decoding may not be optimal. To obtain decoding paths with higher confidence, we introduce a training-free method that adapts the classic beam-search paradigm to a new dimension—the position space. The PBS is initialized with a single sequence consisting only of masked tokens and a score of zero: $\mathcal{B}_0 = \{(x_0, 0)\}$, where x_0 contains only `[MASK]` tokens.

Let $\mathcal{B}_t = \{(x_t^{(1)}, s_t^{(1)}), (x_t^{(2)}, s_t^{(2)}), \ldots, (x_t^{(K)}, s_t^{(K)})\}$ be defined as the beam of size K at step t, where each $x_t^{(j)}$ is a candidate sequence and $s_t^{(j)}$ is its cumulative score, with j indexing the candidate in the beam.

3.2.1 CANDIDATE GENERATION

For each candidate $(x_t^{(j)}, s_t^{(j)})$ in the beam, we generate new candidates by exploring different ways to unmask its currently masked tokens. Specifically, for each possible set of n masked positions (denoted by $\mathcal{I}$) that we choose to unmask in this step, we obtain a new sequence $x_{t-1}^{(j)}$ and compute its cumulative score as:

$$s_t^{(j)} + \phi(\mathcal{I}, \mathbf{p}_t^{(j)}), \tag{2}$$

where $\mathbf{p}_t^{(j)}$ represents the predicted token distributions for candidate j, and $\phi(\mathcal{I}, \mathbf{p})$ is the average confidence score of the selected positions:

$$\phi(\mathcal{I}, \mathbf{p}) = \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \max_{v \in \mathcal{V}} \mathbf{p}_i[v]. \tag{3}$$

Using the average instead of the sum in the scoring function aims to prevent sequences with more remaining masked tokens from being unfairly penalized.

3.2.2 POSITION SELECTION STRATEGIES

Let $\mathcal{P}_t^{(j)}$ denote the set of candidate positions selected from $M_t^{(j)}$ for exploration. The positions are always ranked by confidence:

$$\begin{aligned} \mathcal{P}_t^{(j)} &= \{i_1, i_2, \ldots, i_{|\mathcal{P}_t^{(j)}|}\}, \\ \text{where } &\max_v \mathbf{p}_{t,i_1}^{(j)}[v] \geq \max_v \mathbf{p}_{t,i_2}^{(j)}[v] \geq \cdots \end{aligned} \tag{4}$$

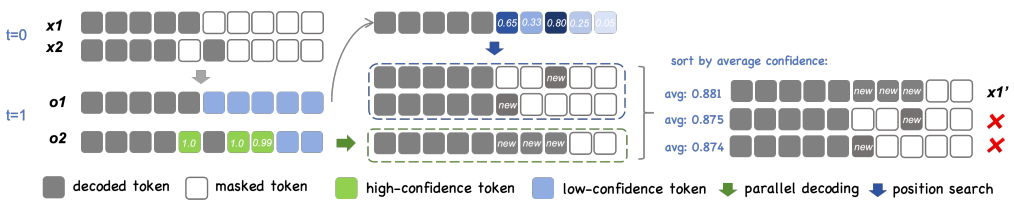

Figure 2: Overview of SOAR. When t=0, there are two sequences in the beam. Based on the confidence of decoded tokens, one undergoes parallel decoding (green arrow), while the other undergoes position search (blue arrow). After reordering, since the sequence with the highest average confidence is obtained through parallel decoding, the beam size is reduced to 1.

Case 1: Single-token Decoding ($n = 1$)

When only one token is unmasked per step, which is a boundary case of parallel decoding, candidates are generated by selecting only one token from $\mathbf{p}_t^{(j)}$, and the tokens selected by different candidates are mutually exclusive.

Case 2: Parallel Decoding ($n > 1$)
When n tokens are unmasked per step, firslty, the minimum number of candidate tokens M required to generate K distinct combinations is determined:

$$M = \min \left\{ m : \binom{m}{n} \geq K, m \leq |M_t^{(j)}| \right\} \tag{5}$$

If $|M_t^{(j)}| < M$, all masked positions are selected, so $M = |M_t^{(j)}|$. Then $\mathcal{P}_t^{(j)}$ contains the top-M positions by confidence. To avoid the combinatorial explosion of generating all $\binom{M}{n}$ subsets, we efficiently select the top-K n-element subsets from $\mathcal{P}_t^{(j)}$ by employing a best-first search strategy over the confidence-ordered positions, ensuring that only the most promising combinations are explored. These K subsets are then used as candidate position sets for parallel unmasking.

3.2.3 Beam Update

All candidates from all beams are pooled, sorted by their scores, and the top K form the new beam:

$$\mathcal{B}_{t-1} = \arg \max_{\mathcal{C} \subseteq \bigcup_{j=1}^{K} \mathcal{C}_t^{(j)}, |\mathcal{C}|=K} \sum_{(x,s)\in\mathcal{C}} s \tag{6}$$

Table 1 presents experimental results for single-token decoding and parallel decoding settings. In the single-token decoding setting, PBS consistently improves performance across multiple benchmarks. However, this improvement comes at a computational cost: computation scales linearly with beam size, resulting in proportionally slower decoding.

Enabling PBS with parallel decoding makes inference speed comparable to greedy decoding, yet leads to a noticeable drop in accuracy relative to the single-token-per-step setup. We hypothesize that this degradation stems from the confidence-agnostic nature of parallel decoding: by forcing the model to decode a fixed number of tokens at each step, it may be compelled to generate low-confidence tokens, thereby compromising output quality. We examine this hypothesis in more detail in Section 4.3.

To navigate this trade-off between decoding speed and output confidence, we introduce SOAR, a method that dynamically decides when to accelerate via parallel decoding and when to refine via confidence-based search.

3.3 SOAR: Confidence-Switched Position Beam Search

SOAR addresses this trade-off by dynamically adapting the decoding strategy according to prediction confidence, enabling the model to explore more when uncertain and decode faster when confident.

Table 1: Main Results. The upper section presents results on LLaDA-8B-Base, while the lower section shows results on Dream-7B-Base.

Benchmark	HumanEval		MBPP		GSM8K		Avg.
Max Length	256	512	256	512	256	512	
Greedy	32.3	32.9	40.8	39.2	70.4	70.9	47.8
Greedy (Adaptive Parallel)	32.3	32.9	40.8	39.2	70.4	71.0	47.8
SpeedUp	*×2.25*	*×2.79*	*×2.06*	*×2.42*	*×1.69*	*×1.92*	*×2.19*
PBS (Single Token)	34.2	34.8	41.4	39.2	71.6	72.1	48.9
SpeedUp	*×0.48*	*×0.48*	*×0.48*	*×0.48*	*×0.47*	*×0.48*	*×0.48*
PBS (Parallel)	31.1	29.3	35.2	36.1	69.5	68.1	44.9
SpeedUp	*×0.98*	*×0.98*	*×0.97*	*×0.97*	*×0.98*	*×0.99*	*×0.98*
SOAR	32.9(+0.6)	39.0(+6.1)	40.8	39.4(+0.2)	71.3(+0.9)	71.5(+0.6)	49.2(+1.4)
SpeedUp	*×1.63*	*×2.16*	*×1.49*	*×1.85*	*×1.18*	*×1.43*	*×1.62*
Greedy	50.0	53.7	53.4	55.4	73.7	74.5	60.1
Greedy (Adaptive Parallel)	50.0	51.8	53.8	55.6	73.2	74.6	59.8
SpeedUp	*×1.63*	*×1.60*	*×2.20*	*×2.86*	*×2.19*	*×1.83*	*×2.05*
PBS (Single Token)	57.3	58.1	54.0	56.9	75.2	76.4	63.0
SpeedUp	*×0.57*	*×0.52*	*×0.50*	*×0.52*	*×0.57*	*×0.56*	*×0.54*
PBS (Parallel)	53.7	54.3	53.6	54.0	73.5	75.1	60.7
SpeedUp	*×1.02*	*×0.98*	*×1.01*	*×0.96*	*×1.03*	*×1.01*	*×1.00*
SOAR	55.5(+5.5)	55.1(+1.4)	57.0(+3.6)	56.2(+0.8)	74.7(+1.0)	75.7(+1.2)	62.4(+2.3)
SpeedUp	*×1.13*	*×1.07*	*×1.87*	*×2.47*	*×0.95*	*×1.07*	*×1.43*

Let $\mathcal{C}_t = \{(x_t^{(1)}, s_t^{(1)}), \ldots, (x_t^{(N)}, s_t^{(N)})\}$ denote the set of candidate sequences at step t, where $N_t = |\mathcal{C}_t|$ is the candidate size, each $x_t^{(j)}$ represents a candidate sequence, and $s_t^{(j)}$ is its cumulative score, with j indexing candidates in the beam.

3.3.1 Confidence-Guided Adaptive Decoding

At each decoding step t, for each candidate sequence $x_t^{(j)}$ in the candidate set, we compute confidence scores for all masked positions:

$$c_{t,i} = \max_{v \in \mathcal{V}} \mathbf{p}_{t,i}[v] \tag{7}$$

$x_t^{(j)}$ switches between two decoding modes based on the presence of high-confidence positions for each candidate sequence:

(a) **Parallel Decoding Mode**: When there exists at least one high-confidence position, SOAR directly decodes all such positions in parallel:

$$\mathcal{I}_t = \{i : c_{t,i} > \tau\} \tag{8}$$

All tokens in $\mathcal{I}_t$ are unmasked simultaneously, where τ is a hyper-parameter. This mode is similar to the design of Fast-dLLM (Wu et al., 2025), with the difference that when confidence falls below the threshold, SOAR will switch to beam search mode to explore other possible decoding paths.

(b) **Beam Search Mode**: When no masked position exceeds the confidence threshold ($c_{t,i} \leq \tau$ for all $i \in M_t^{(j)}$), SOAR falls back to position beam search with $k = 1$ token per step. The beam explores the top-K most confident positions individually, trading speed for more thorough exploration.

3.3.2 Dynamic Candidate Size Adjustment

Sequences in the candidate set generate new candidate sequences, which are then sorted in descending order based on their average decoding confidence. After sorting, only the top N_t sequences are retained, where N_t denotes the size of the candidate set at step t.

To further optimize efficiency, SOAR dynamically adjusts N_t after each decoding step based on the origin of the new candidate with the highest score. The decision rule is as follows: if the highest score new candidate originates from the parallel decoding mode, the model is considered sufficiently confident and N_t is reduced to 1 for the next step; otherwise, if the best new candidate comes from the PBS mode, N_t is reset to the preset size K, which corresponds to the maximum beam width used in the PBS mode. Formally,

$$N_t = \begin{cases} 1 & \text{if best new candidate from parallel decoding} \\ K & \text{if best new candidate from PBS} \end{cases} \tag{9}$$

This adaptive strategy ensures that when the model exhibits high confidence, computational resources are concentrated on the most promising path; when uncertainty arises, multiple decoding hypotheses are retained to avoid local optima.

The time complexity of SOAR is $O(T \cdot \bar{N})$, where T is the number of decoding steps, and $\bar{N}$ is the average candidate size. In practice, because a significant portion of the model's tokens have confidence above the threshold, $\bar{N}$ remains well below K, and the overall step count T is also reduced.

SOAR thus provides a principled way to balance the exploration-exploitation tradeoff in DLM decoding, adapting to the model's uncertainty at each step to optimize both quality and efficiency. Appendix C presents a visual analysis of the algorithm's behavior.

4 Experiments

4.1 Experiment Setup

We conduct experiments on two representative diffusion language models: LLaDA-8B (Nie et al., 2025) and Dream-7B (Ye et al., 2025), both using the Base version. To ensure reproducibility, all experiments are conducted with a temperature of 0, using softmax probabilities as confidence scores. Unless otherwise specified, the threshold $\tau = 0.95$ for LLaDA-8B and $\tau = 0.90$ for Dream-7B and the maximum beam size *K* is set to 2. The maximum sequence length is set to 256 and 512, respectively, to validate the robuestness of the results.

To comprehensively assess the effectiveness of our approach, experiments are conducted across three benchmarks covering code generation and mathematical reasoning. For mathematical reasoning, we employ GSM8K (Cobbe et al., 2021), a dataset of grade-school math word problems. For code generation, we evaluate on MBPP (Austin et al., 2021), which contains entry-level Python programming tasks, and HumanEval (Chen et al., 2021), a collection of handwritten programming challenges designed for synthesis evaluation.

We adhere to standard few-shot evaluation protocols for each benchmark: 0-shot for HumanEval, 3-shot for MBPP and 4-shot for GSM8K. Accuracy is used as the evaluation metric for mathematical reasoning, while pass@1 is adopted for code generation benchmarks. All experiments are conducted based on open-source lm-evaluation-harness with a single NVIDIA A100 80GB GPU. The total inference time of the benchmark is used to caculate the SpeedUp ratio compared with standard greedy decoding.

4.2 Main Results on Accuracy and Speed

Table 1 summarizes the performance of different decoding methods across both models and three benchmarks. In addition, we plot the variation of average accuracy and SpeedUp among these decoding methods in Figure 3 for better illustration. In most settings, PBS consistently improves accuracy over greedy decoding, with Dream-7B achieving gains of up to $+7.3\%$ on HumanEval. However, this quality improvement comes at the cost of nearly doubled inference latency (average SpeedUp $\times 0.54$), as exploring multiple decoding paths increases computation.

PBS with parallel token decoding ($n = 2$) restores decoding speed to the same level as greedy decoding, yet the average accuracy also decreases significantly compared to PBS with single token. We attribute this to the fact that forcing a fixed number of tokens to be decoded in parallel can

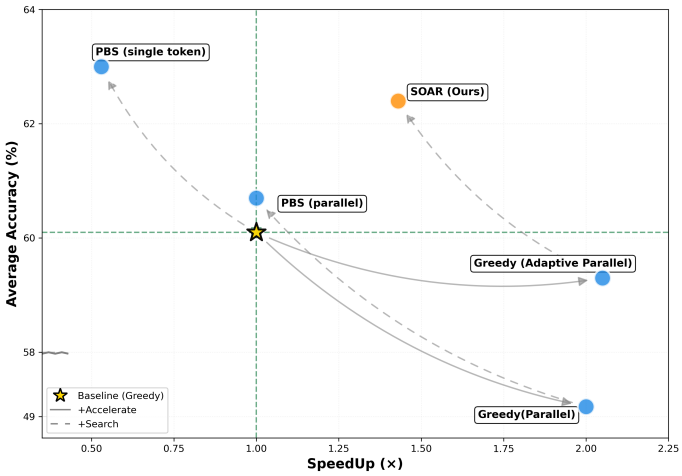

Figure 3: Pareto frontier on DREAM-7B-Base. Solid arrows indicate adding parallel decoding (acceleration), and dashed arrows indicate adding position beam search. This plot is generated using the average accuracy and average speedup from Table 1. For better visual presentation, the Y-axis range 49–58 is compressed.

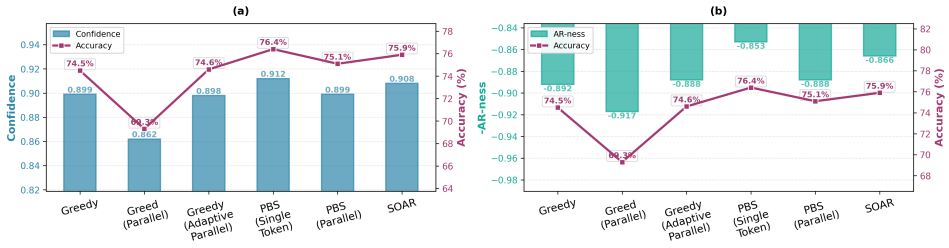

Figure 4: Left: Accuracy with average confidence. Right: Accuracy with negative Global AR-ness

undermine the confidence-based selection that underlies PBS's effectiveness. In contrast, SOAR switches between search and parallel decoding based on confidence, avoiding excessive search that affects decoding speed while also preventing mandatory parallel decoding from harming global confidence, thereby achieving a balance between speed and quality improvement.

4.3 ANALYSIS OF CONFIDENCE AND AR-NESS

Confidence of Decoding Sequence. To test the hypothesis proposed in Section 1—*Can we improve decoding quality by exploring alternative unmasking sequences with higher confidence beyond greedy selection?*—we analyze the average confidence of decoded sequences.

Average Confidence: For each sample, we extract the reasoning tokens that precede the answer keyword ("answer" token in GSM8K). Each token's confidence score is defined as the model's maximum softmax probability. The sample-level average confidence is computed as the mean of these token-wise scores. We then average across all samples for each decoding method to obtain the method-level average confidence.

Figure 4(a) illustrates the relationship between average confidence and accuracy for various decoding methods on the GSM8K benchmark using DREAM-7B. Here, "Parallel" refers to decoding exactly two tokens per step, "Adaptive Parallel" decides whether to perform parallel decoding based on confidence thresholds, and the maximum sequence length is 512.

Our results show a positive correlation between accuracy and average confidence. PBS achieves an average accuracy 1.9% higher than Greedy Search, confirming our hypothesis. In contrast, Parallel decoding forces the generation of two tokens per step regardless of confidence, which significantly

lowers average confidence and leads to a notable accuracy drop. When Parallel decoding is combined with PBS, the confidence gains from PBS are negated by the confidence loss from Parallel decoding, resulting in a 1.3% accuracy reduction compared to PBS alone. SOAR, however, dynamically switches between Parallel decoding and PBS based on confidence, preserving higher average confidence and maintaining competitive accuracy.

AR-ness of Decoding Sequence. We further investigate why SOAR yields better decoding sequences using another metric: the Global AR-ness proposed by (Gong et al., 2025b), which quantifies how much the unmasking schedule of a diffusion model resembles an autoregressive pattern, specifically, whether it follows a "left-first" pattern.

Global AR-ness: At each decoding step t, we examine whether the predicted token lies within the first k masked positions. The Global AR-ness@k is defined as the average ratio of such steps across the entire decoding process, measuring the tendency to always unmask the earliest remaining token and thus capturing a left-to-right filling strategy. This ratio increases with k, as the criterion becomes easier to satisfy when more early positions are allowed. *A higher value indicates that the generation is more autoregressive.*

We set $k = 5$ and compute the mean Global AR-ness across all samples. Figure 4(b) illustrates the relationship between the negative Global AR-ness and accuracy, revealing a **negative** correlation: methods with lower Global AR-ness (i.e., less left-to-right bias) tend to achieve higher accuracy. We hypothesize that this may be because SOAR can mitigate the "entropy sink" issue in DLMs. Specifically, the model is inherently biased toward tokens immediately to the right of the given prefix, as these positions receive stronger positional signals and closer context, leading to disproportionately high confidence. This bias may limit the DLM's ability to explore potentially better decoding paths. Although SOAR is not explicitly designed to address this problem, by enabling position beam search, it can alleviate the issue.

4.4 Ablation and Robustness Analysis

Ablation Study. We investigate the impact of confidence threshold τ and beam size K on the quality-efficiency trade-off (detailed plots in Appendix A). For the confidence threshold, we find that increasing τ steadily improves accuracy but reduces decoding speed; notably, setting $\tau > 0.8$ consistently yields better accuracy than standard decoding while maintaining competitive latency. Regarding beam size, while larger beams offer modest accuracy gains, the computational overhead grows linearly. To balance performance and efficiency, we adopt a beam size of $K = 2$ and set $\tau = 0.90$ for DREAM-7B and $\tau = 0.95$ for LLaDA-8B as default settings.

Robustness Analysis. We further validate the robustness of SOAR from two perspectives: confidence metrics and variable-length decoding. Detail results are represented in Appendix B First, beyond softmax probability, we evaluate SOAR using *Margin* (gap between top-1 and top-2 probabilities) and *Negative Entropy* metrics on HumanEval. As shown in Table 2, SOAR consistently outperforms greedy baselines across both metrics while maintaining speedups. Second, we integrate SOAR with DreamOn (Wu et al.), a variable-length DLM that dynamically predicts *expand*/*delete* tokens. Experiments on HumanEval-Infilling (Table 3) show that SOAR significantly boosts performance without retraining, demonstrating its seamless compatibility with advanced decoding mechanisms.

5 Conclusion

We introduce **S**earch **O**r **A**ccele**R**ate (SOAR), an adaptive inference framework that dynamically balances generation quality and speed for Diffusion Language Models. By allowing the model to widen its search when uncertain and accelerate when confident, SOAR consistently improves generation quality across coding and mathematical reasoning tasks while maintaining competitive inference speed. Extensive experiments confirm its robustness to different unmask metrics, sequence lengths, and base models. As a training-free method, SOAR offers a practical and effective solution to the quality-speed trade-off in DLM inference.

REFERENCES

Anonymous. Improving diffusion language model reasoning through joint search in generation order and token space. In *Submitted to The Fourteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=AaAbeUp7O4. under review.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021. URL https://arxiv.org/abs/2108.07732.

Mohammad Bavarian, Heewoo Jun, Nikolas Tezak, John Schulman, Christine McLeavey, Jerry Tworek, and Mark Chen. Efficient training of language models to fill in the middle, 2022. URL https://arxiv.org/abs/2207.14255.

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

Zigeng Chen, Gongfan Fang, Xinyin Ma, Ruonan Yu, and Xinchao Wang. dparallel: Learnable parallel decoding for dllms, 2025. URL https://arxiv.org/abs/2509.26488.

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

Markus Freitag and Yaser Al-Onaizan. Beam search strategies for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pp. 56–60. Association for Computational Linguistics, 2017. doi: 10.18653/v1/w17-3207. URL http://dx.doi.org/10.18653/v1/W17-3207.

Yifeng Gao, Ziang Ji, Yuxuan Wang, Biqing Qi, Hanlin Xu, and Linfeng Zhang. Self speculative decoding for diffusion large language models, 2025. URL https://arxiv.org/abs/2510.04147.

Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, Chenxin An, Peilin Zhao, Wei Bi, Jiawei Han, Hao Peng, and Lingpeng Kong. Scaling diffusion language models via adaptation from autoregressive models, 2025a. URL https://arxiv.org/abs/2410.17891.

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, 2025b. URL https://arxiv.org/abs/2506.20639.

Jianuo Huang, Yaojie Zhang, Yicun Yang, Benhao Huang, Biqing Qi, Dongrui Liu, and Linfeng Zhang. Mask tokens as prophet: Fine-grained cache eviction for efficient dllm inference, 2025a. URL https://arxiv.org/abs/2510.09309.

Zemin Huang, Yuhang Wang, Zhiyang Chen, and Guo-Jun Qi. Don't settle too early: Self-reflective remasking for diffusion language models, 2025b. URL https://arxiv.org/abs/2509.23653.

Jinsong Li, Xiaoyi Dong, Yuhang Zang, Yuhang Cao, Jiaqi Wang, and Dahua Lin. Beyond fixed: Training-free variable-length denoising for diffusion large language models, 2025a. URL https://arxiv.org/abs/2508.00819.

Pengxiang Li, Yefan Zhou, Dilxat Muhtar, Lu Yin, Shilin Yan, Li Shen, Yi Liang, Soroush Vosoughi, and Shiwei Liu. Diffusion language models know the answer before decoding, 2025b. URL https://arxiv.org/abs/2508.19982.

Zhiyuan Liu, Yicun Yang, Yaojie Zhang, Junjie Chen, Chang Zou, Qingyuan Wei, Shaobo Wang, and Linfeng Zhang. dllm-cache: Accelerating diffusion large language models with adaptive caching, 2025. URL https://arxiv.org/abs/2506.06295.

Xinyin Ma, Runpeng Yu, Gongfan Fang, and Xinchao Wang. dkv-cache: The cache for diffusion language models, 2025. URL https://arxiv.org/abs/2505.15781.

Nikita Mounier and Parsa Idehpour. Review, remask, refine (r3): Process-guided block diffusion for text generation, 2025. URL https://arxiv.org/abs/2507.08018.

Quan Nguyen-Tri, Mukul Ranjan, and Zhiqiang Shen. Attention is all you need for kv cache in diffusion llms, 2025. URL https://arxiv.org/abs/2510.14973.

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, 2025. URL https://arxiv.org/abs/2502.09992.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023. URL https://arxiv.org/abs/2302.13971.

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, 2025. URL https://arxiv.org/abs/2505.22618.

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

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025a. URL https://arxiv.org/abs/2505.09388.

Yicun Yang, Cong Wang, Shaobo Wang, Zichen Wen, Biqing Qi, Hanlin Xu, and Linfeng Zhang. Diffusion llm with native variable generation lengths: Let [eos] lead the way, 2025b. URL https://arxiv.org/abs/2510.24605.

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

Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Hu, Jun Zhou, Jianfei Chen, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Llada 1.5: Variance-reduced preference optimization for large language diffusion models, 2025a. URL https://arxiv.org/abs/2505.19223.

Qinglin Zhu, Yizhen Yao, Runcong Zhao, Yanzheng Xiang, Amrutha Saseendran, Chen Jin, Philip Teare, Bin Liang, Yulan He, and Lin Gui. Latent refinement decoding: Enhancing diffusion-based language models by refining belief states, 2025b. URL https://arxiv.org/abs/2510.11052.

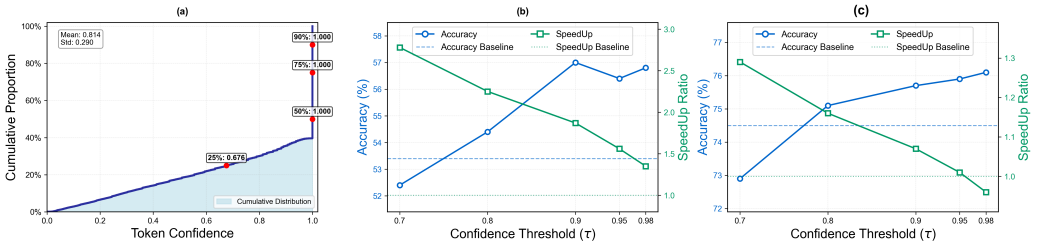

Figure 5: The threshhold study on DREAM-7B: (a) Cumulative distribution of token confidence scores on GSM8K; (b) Trade-off between accuracy and SpeedUp under varying confidence thresholds on MBPP; (c) Trade-off between accuracy and SpeedUp under varying confidence thresholds on GSM8K.

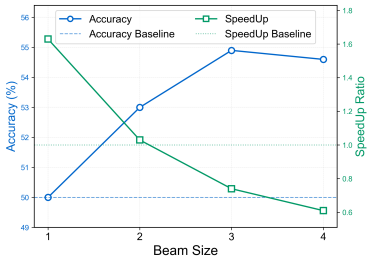

Figure 6: The beam size study on DREAM-7B: Trade-off between accuracy and SpeedUp under increasing beam size on HumanEval.

A ABLATION STUDY DETAILS

Confidence Threshold. We examine how the confidence threshold τ affects decoding quality and speed. Figure. 5(a) plots the cumulative distribution of token confidence scores on GSM8K under standard decoding, showing a long tail—about 50% of tokens have probability > 0.95. We vary $\tau \in [0.8, 0.98]$ and evaluate accuracy and SpeedUp with coding (MBPP) and math (GSM8K) tasks on DREAM-7B. As shown in Figure. 5(b)-(c), raising τ from 0.8 to 0.98 steadily improves accuracy but reduces decoding speed. Notably, for $\tau > 0.8$, SOAR consistently beats standard decoding in accuracy while matching or exceeding its speed, demonstrating robustness. We conducted the same analysis on LLADA-8B, and ultimately set $\tau = 0.95$ for LLADA-8B and $\tau = 0.9$ for DREAM-7B.

Beam Size. We further investigate whether expanding the beam size can yield additional performance gains. Figure 6 illustrates the trade-off between generation quality and inference speed under different beam sizes (ranging from 1 to 4) on the HumanEval benchmark using the DREAM-7B model. Setting beam size to 1 corresponds to using only confidence-based parallel decoding. While increasing the beam size may offer modest improvements in accuracy, the computational overhead grows linearly with beam width, resulting in significant slowdowns in inference speed. Therefore, we select a beam size of 2 as the default setting to balance quality and efficiency in our experiments.

B ANALYSIS OF THE ROBUSTNESS OF SOAR

Unmask Metric. Although softmax probability is the mainstream metric for selecting unmask tokens, several alternative metrics are also viable options: (1) the margin between the top-1 and top-2 token probabilities and (2) the negative entropy of the probability distribution. Formally, given the token probability distribution $\mathbf{p} \in \mathbb{R}^V$, the two metrics are defined as

$$\text{Margin}(\mathbf{p}) = p_{(1)} - p_{(2)},$$

$$\text{NegEntropy}(\mathbf{p}) = \sum_{i=1}^{V} p_i \log p_i,$$

Table 2: Results of other unmask metrics on HumanEval with DREAM-7B.

Metric	Method	Length=256	Length=512
Margin	Greedy	47.0	49.4
Margin	SOAR	51.2	51.2
	SpeedUp	*×1.04*	*×1.07*
NegEntropy	Greedy	51.2	56.1
NegEntropy	SOAR	53.0	57.9
	SpeedUp	*×1.07*	*×1.14*

where $p_{(1)}$ and $p_{(2)}$ denote the largest and second-largest probabilities in $\mathbf{p}$, respectively.

Table 2 presents the results on HumanEval under two sequence length settings. We use $\tau = 0.9$ for margin-based metric and $\tau = -0.1$ for negative-entropy-based metric. For both, SOAR consistently improves accuracy over the corresponding greedy decoding baseline while maintaining a faster decoding speed.

Table 3: Results on HumanEval-Infilling with DREAMON under different initial sequence lengths.

	Initial Mask Length			
Method	**8**	**16**	**32**	**64**
Greedy	58.4	57.4	58.0	58.0
SOAR	63.0	66.1	67.2	67.7
SpeedUp	*x1.02*	*x1.02*	*x1.03*	*x1.04*

Results with Variable-Length Decoding Method. To assess the robustness of SOAR when combined with variable-length decoding strategies, we evaluate it on DREAMON (Wu et al.)—a model post-trained upon DREAM-7B that dynamically adjusts sequence length during decoding by predicting *expand* and *delete* tokens. Specifically, when an *expand* token is generated, it is replaced with two *mask* tokens, thereby extending the sequence; conversely, a *delete* token triggers the removal of the corresponding token from the output sequence.

Following the DreamOn experimental setup, we conduct evaluations on the HumanEval-Infilling benchmark (Bavarian et al., 2022). We set the maximum decoding length to 64, employ negative entropy ($\tau = -0.1$) as the confidence metric, and use exact match as the evaluation criterion.

Table 3 compares the performance of standard greedy decoding and SOAR across different initial sequence lengths. The results demonstrate that SOAR, as a training-free method, consistently maintains or improves performance when integrated with DreamOn's variable-length decoding mechanism. This experiment confirms SOAR 's flexibility and robustness, highlighting its potential to be effectively combined with other carefully designed decoding strategies without requiring architectural modifications or additional training.

C VISUALIZATION OF SOAR

Increasingly Parallel Decoding. We analyzed the usage patterns of Beam Search and Parallel Decoding modes across decoding steps on the Dream-7B-Base model using GSM8K. Consistent with the processing in Subsection 4.3, for all sample decoding sequences, we retain only the valid portion — i.e., the tokens preceding the keyword "answer". Since different samples have varying effective decoding lengths S_{sample}, we record the ratio of each token's decoding step to its effective sequence length as a measure of the Decoding Process(%). Figure 7 illustrates how the two decoding modes evolve as decoding progresses. In the early stages, SOAR allocates a larger proportion of steps (55%) to PBS to explore more promising sequences. As decoding advances, the model eventually favors parallel decoding in 85% of the cases. This trend demonstrates the model's transition from initial uncertainty toward greater confidence. Throughout the inference process, SOAR dynamically switches between the two decoding modes based on confidence scores in an adaptive manner.

Decoding Path Illustration. Figure 8 illustrates the decoding trajectories of a GSM8K sample using Greedy and SOAR decoding, based on the Dream-7B-Base model. The Greedy decoder exhibits an AR-style decoding path, while SOAR, through parallel decoding, decodes nearly the same number of positions in fewer steps. Moreover, by employing PBS, SOAR retains earlier positions as masked and defers their decoding to later steps, aligning with the analysis in Subsection 4.3. For tokens that appear early but lack sufficient model confidence, decoding them later with richer contextual information yields an overall higher-confidence sequence compared to Greedy decoding.

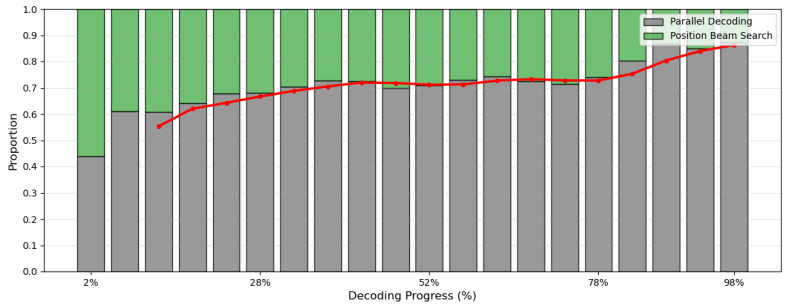

Figure 7: Variation of Decoding Modes Across the Decoding Process. The x-axis represents the progress of decoding (quantized into 20 bins), while the y-axis indicates the proportion of tokens decoded by each mode across all samples. The stacked bars are partitioned into gray (Parallel Decoding) and green (PBS) segments, reflecting their relative usage at each stage. A red trend line highlights the overall transition pattern between the two modes.

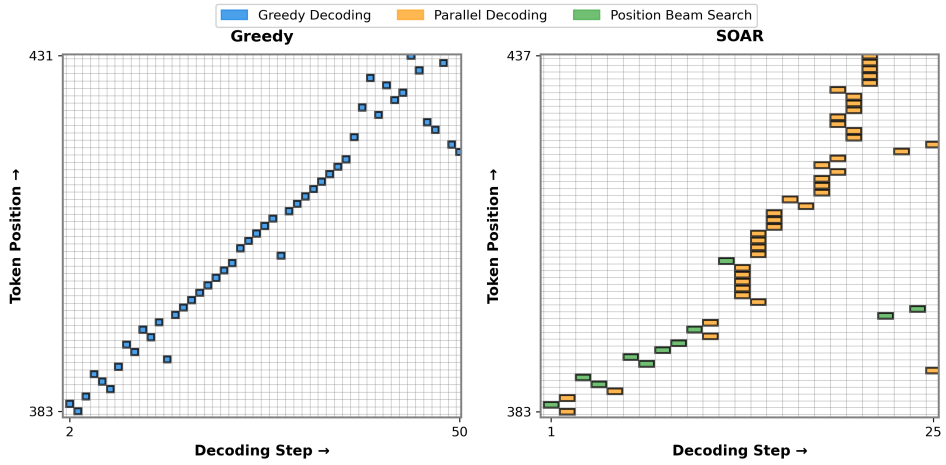

Figure 8: Decoding Path Comparison. The left figure shows Greedy Decoding, and the right figure shows Our method. The X-axis represents steps, and the Y-axis represents positions.