Thinking Inside the Mask: In-Place Prompting in Diffusion LLMs

Anonymous submission

Abstract

Despite large language models (LLMs) have achieved remarkable success, their *prefix-only* prompting paradigm and sequential generation process offer limited flexibility for bidirectional information. Diffusion large language models (dLLMs) present new opportunities through their bidirectional attention mechanisms and iterative refinement processes, enabling more flexible *in-place* prompting strategies. We introduce ICE (**I**n-Place **C**hain-of-Thought Prompting with **E**arly Exit), a novel framework that transforms prefix-only prompting into in-place prompting specifically designed for dLLMs. ICE integrates in-place prompts directly within masked token positions during iterative refinement and employs a confidence-aware early exit mechanism to significantly reduce computational overhead. Extensive experiments demonstrate ICE's effectiveness, achieving up to 17.29% accuracy improvement with 4.12× speedup on GSM8K, and up to 276.67× acceleration on MMLU while maintaining competitive performance. *Codes are available in the supplementary materials and will be released on Github.*

Introduction

Large language models (LLMs) (Zhao et al. 2023) have revolutionized natural language processing, with autoregressive (AR) models dominating the landscape through their sequential, left-to-right token generation paradigm (Brown et al. 2020; Touvron et al. 2023). AR models are fundamentally constrained by *prefix-only* prompting and sequential generation. Diffusion large language models (dLLMs) (Ye et al. 2025; Nie et al. 2025; Zhu et al. 2025; Yang et al. 2025) offer non-autoregressive alternatives through iterative masked token refinement (Austin et al. 2021a; Lou, Meng, and Ermon 2023). Crucially, dLLMs present new opportunities through their bidirectional attention mechanisms and iterative refinement processes, enabling more flexible *in-place* prompting strategies that can embed information directly within masked token positions (Wen et al. 2025). LLaDA matches GPT-3.5 performance (Nie et al. 2025) and Mercury achieves 1000+ tokens/second (Liu et al. 2025a), yet reasoning capabilities in dLLMs remain underexplored.

While Chain-of-Thought (CoT) prompting has proven highly effective for AR models by decomposing complex problems into intermediate reasoning steps (Wei et al. 2022), the bidirectional and iterative nature of dLLMs enables fundamentally different approaches. Unlike AR models that treat reasoning as sequential prefix conditioning, dLLMs can embed reasoning directly within the generation process itself with in-place prompting (Figure 1). Moreover, AR models exhibit *sequential answer emergence*, where answers remain inaccessible until the completion of sequential generation, while dLLMs enable *concurrent answer accessibility* through their bidirectional context modeling, allowing intermediate visibility of answer content during the iterative refinement process. This architectural distinction creates opportunities for novel confidence-aware optimization strategies that can monitor answer during generation.

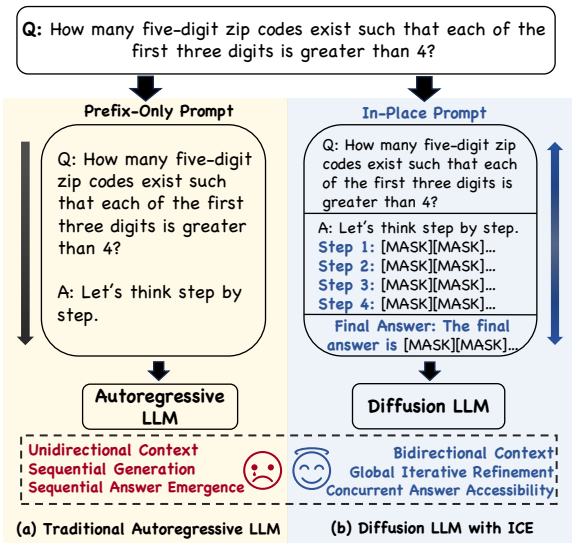

Figure 1: Prompt construction in **(a) Autoregressive LLMs** vs. **(b) Diffusion LLMs**. Autoregressive LLMs employ unidirectional context with prefix-only prompting, while dLLMs leverage bidirectional context modeling, enabling in-place prompting and concurrent answer accessibility.

To address these opportunities, we propose ICE (**I**n-Place **C**hain-of-Thought Prompting with **E**arly Exit), a novel framework that enhances both reasoning capabilities and inference efficiency in dLLMs (Figure 3). Our central insight is that the iterative generation process of dLLMs provides a unique opportunity to embed reasoning steps directly within

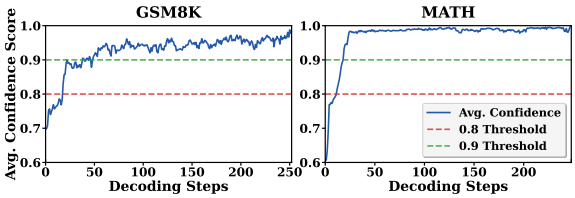

Figure 2: Average confidence of answer section on GSM8K and MATH during generation. The model's confidence in the answer section rapidly converges to a high level and remains stable throughout subsequent iterations, indicating that the model internally determines the correct answer early in the process while continuing to refine the reasoning trace.

the generation process, transforming reasoning from external preprocessing into an integral component of the generation mechanism. ICE introduces two key innovations:

In-Place Chain-of-Thought Prompting: This approach integrates reasoning steps directly into masked token positions during iterative refinement. It exploits the bidirectional nature of dLLMs by structuring the generation sequence into distinct thinking and answer sections, with explicit step-by-step reasoning templates embedded within the thinking section. This enables enhanced reasoning performance while preserving parallel generation advantages.

Two-Phase Decoding with Early Exit Mechanism: Motivated by a crucial empirical observation, we design a confidence-aware inference strategy that capitalizes on the distinct refinement patterns between reasoning and answer components. Through systematic analysis of iterative refinement dynamics, we uncover a distinctive behavioral pattern in dLLMs: model confidence in answer tokens converges rapidly to high levels and maintains stability, while the reasoning section continues undergoing refinement (Figure 2). This observation reveals that models often internally determine correct answers significantly earlier than the completion of explicit reasoning traces. Leveraging this insight, ICE implements a two-phase decoding approach that enables parallel decoding of all answer tokens while effectively reducing redundant computation.

Extensive experimental validation demonstrates ICE's effectiveness across diverse reasoning benchmarks. On mathematical reasoning tasks, ICE achieves up to 17.29% accuracy improvement with 4.12× speedup on GSM8K, alongside consistent gains on MATH. For knowledge-intensive tasks, ICE delivers up to 276.67× acceleration on MMLU and speedups exceeding 40× on GPQA while maintaining competitive accuracy. Notably, ICE is compatible with existing acceleration techniques, such as dLLM-Cache (Liu et al. 2025c), enabling cumulative benefits when combined.

Our contributions are threefold:

- We introduce the first in-place prompting framework for dLLMs, embedding prompts directly in masked tokens to improve both accuracy and efficiency.
- We develop a two-phase decoding strategy with early exit mechanism that significantly reduces inference latency while maintaining generation quality.
- We provide comprehensive empirical evidence demonstrating ICE's effectiveness, establishing that architectural alignment between reasoning patterns and generation mechanisms can yield synergistic benefits.

Related Work

Diffusion LLMs. Recent diffusion large language models (dLLMs) (Nie et al. 2025; Austin et al. 2021b; Lou, Meng, and Ermon 2023; Shi et al. 2024; Liu et al. 2025b), notably LLaDA, represent a paradigm shift from autoregressive generation by employing bidirectional attention mechanisms and iterative refinement processes. Unlike autoregressive models that generate tokens sequentially from left to right, dLLMs leverage masked token prediction with full sequence context awareness, enabling them to overcome fundamental limitations such as the reversal curse (Berglund et al. 2023). LLaDA, an 8-billion parameter model trained from scratch, rivals LLaMA3 8B performance. The masked diffusion process inherently supports in-context learning and instruction following through its non-autoregressive architecture, opening unique opportunities for embedding structured reasoning directly within the sequence, rather than relying solely on prefix prompting in autoregressive models.

Chain-of-Thought Reasoning. The introduction of Chain-of-Thought (CoT) prompting (Wei et al. 2022; Nakamura and et al. 2021) has substantially enhanced reasoning accuracy in LLMs by facilitating systematic step-by-step problem decomposition. However, traditional CoT approaches are inherently constrained by their reliance on autoregressive generation and prompt-level guidance, which limits their effective integration with diffusion-based architectures. Recent advances predominantly target autoregressive models (Kojima et al. 2022; Gao and et al. 2023), thereby failing to leverage the distinctive bidirectional context modeling capabilities inherent in dLLMs. To address this gap, we introduce in-place CoT prompting that seamlessly embeds reasoning steps directly into the sequence during iterative refinement, enabling fine-grained control throughout the generation.

Efficient Inference for dLLMs. Diffusion LLMs inherently suffer from high computational overhead due to their iterative generation process. Current acceleration approaches for dLLMs predominantly focus on computational optimization strategies at the algorithmic level (Wu et al. 2025; Ma et al. 2025; Hu et al. 2025; Luxembourg, Permuter, and Nachmani 2025). dLLM-Cache (Liu et al. 2025c) addresses this challenge through a training-free adaptive caching framework that strategically reuses stable prompt computations and employs similarity-guided partial response updates. SlowFast Sampling (Wei et al. 2025) further enhances inference speed by dynamically alternating between an exploratory 'Slow' phase for uncertainty resolution and an aggressive 'Fast' phase for confident generation, guided by certainty, convergence, and positional principles. In contrast to these algorithmic approaches, our method introduces a content-aware acceleration framework that leverages the unique bidirectional nature of dLLMs for confidence-based early exit.

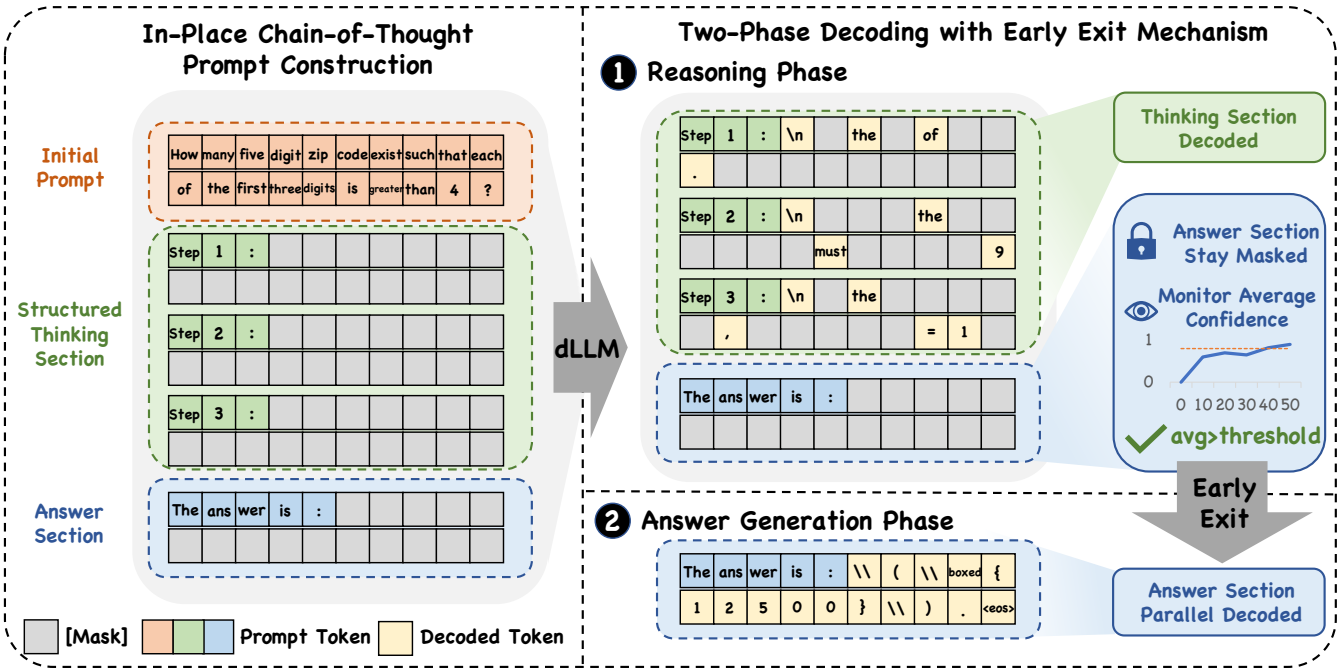

Figure 3: Overview of the ICE framework. ICE integrates two key components: **(1) In-Place Chain-of-Thought Prompting**, which embeds structured step-by-step reasoning templates directly into the prompt, and **(2) Two-Phase Decoding with Confidence-Aware Early Exit Mechanism**. During the *Reasoning Phase*, the model iteratively decodes the thinking section while monitoring the average confidence of the masked answer section. Once the confidence threshold is reached, the framework transitions to the *Answer Generation Phase*, decoding all answer tokens in parallel to produce the final response.

Methodology

Preliminaries

Masked Diffusion Large Language Models. Masked Diffusion Large Language Models (dLLMs) implement a forward process that incrementally corrupts an input sequence x_0 by introducing a special `[MASK]` token. This process is controlled by a continuous time parameter $t \in [0, 1]$. At each timestep t, the resulting sequence x_t is partially masked, with each token independently replaced by `[MASK]` with probability $1 - \alpha_t$ or retained with probability α_t. The noise schedule α_t, which decreases monotonically with t, determines the corruption rate. At $t = 1$, the sequence x_1 is entirely masked, consisting solely of `[MASK]` tokens.

Training a masked dLLM involves a bidirectional predictor f_θ that reconstructs the original sequence x_0 from its corrupted counterpart x_t. During each training step, a timestep $t \in [0, 1)$ is uniformly sampled, and tokens are masked based on the forward process defined by α_t. The objective is to minimize the negative evidence lower bound (NELBO), an upper bound on the negative log-likelihood of the data. For masked dLLMs, the NELBO reduces to a weighted log-likelihood loss, with weights derived from α_t (Sahoo et al. 2024). The popular LLaDA models use a linear noise schedule $\alpha_t = 1 - t$, in which the resulting NELBO is given by:

$$-\mathbb{E}\left[\frac{1}{t}\sum_{k=1}^{|x_t|}\mathbb{I}[x_t^k = \texttt{[MASK]}]\log f_\theta(x_0^k \mid x_t)\right], \quad (1)$$

where $t \sim \mathcal{U}[0, 1)$, $x_0 \sim p_{\text{data}}$, $x_t \sim q_{t|0}(x_t|x_0)$, $|x_t|$ represents the sequence length, x_t^k denotes the k-th token of x_t, and $\mathbb{I}[\cdot]$ is an indicator function that restricts the loss to masked tokens. In contrast to BERT (Devlin 2018), which relies on a static masking ratio and single-step token prediction, masked dLLMs employ dynamic masking probabilities and support iterative decoding from a fully masked state, enabling generative modeling.

Inference Process of Masked dLLMs. The inference procedure reverses the forward corruption through iterative refinement, progressively transforming a fully masked sequence into coherent output. Given an inference sequence $\mathbf{y} = (\mathbf{y}_{\text{prompt}}, \mathbf{y}_{\text{gen}})$, where $\mathbf{y}_{\text{prompt}}$ represents the initial prompt and $\mathbf{y}_{\text{gen}}$ denotes the sequence to be generated, the model maintains an intermediate state $\mathbf{y}^{(k)} = (\mathbf{y}_{\text{prompt}}, \mathbf{y}_{\text{gen}}^{(k)}) \in \mathcal{T}^L$ across N discrete refinement steps, progressing from $k = N$ to $k = 0$. Here, $\mathcal{T}$ denotes the token vocabulary and L represents the total sequence length. The process initializes with a fully masked generation sequence:

$$\mathbf{y}^{(N)} = (\mathbf{y}_{\text{prompt}}, \underbrace{\texttt{[MASK]}, \ldots, \texttt{[MASK]}}_{L_{\text{gen}} \text{ times}}) \quad (2)$$

At each step $k \in \{N, N-1, \ldots, 1\}$, the bidirectional predictor f_θ estimates the original sequence $\mathbf{y}_0$ from the current noisy state $\mathbf{y}^{(k)}$:

$$f_\theta(\mathbf{y}_0 \mid \mathbf{y}^{(k)}) \quad (3)$$

The model obtains an estimate of the clean sequence at step k, denoted $\hat{\mathbf{y}}_0^{(k)}$, through greedy decoding:

$$\hat{\mathbf{y}}_{0,i}^{(k)} = \arg\max_{v \in \mathcal{T}} f_\theta(\mathbf{y}_{0,i} = v \mid \mathbf{y}^{(k)}) \quad \forall i \in \{1, \ldots, L\} \tag{4}$$

Subsequently, a transition function S generates the next state $\mathbf{y}^{(k-1)}$ by selectively updating tokens in $\mathbf{y}^{(k)}$ based on the current estimate $\hat{\mathbf{y}}_0^{(k)}$:

$$\mathbf{y}^{(k-1)} = S(\hat{\mathbf{y}}_0^{(k)}, \mathbf{y}^{(k)}, k) \tag{5}$$

The implementation of S typically employs strategies such as confidence-based unmasking or stochastic unmasking.

In-Place Chain-of-Thought Prompting

The iterative, non-autoregressive generation paradigm of dLLMs, coupled with their inherent bidirectional attention mechanisms, enables a fundamental departure from the conventional prefix-only prompting strategies employed in autoregressive models. While autoregressive models are constrained by sequential, left-to-right generation processes, dLLMs possess the capability to consider entire sequence contexts simultaneously and enable *concurrent answer accessibility*. This architectural advantage unlocks novel prompting paradigms as it allows the model to simultaneously refine reasoning steps while maintaining awareness of answer regions throughout the generation process.

Our approach leverages this distinctive capability by structuring the generation sequence $\mathbf{y}_{\text{gen}}$ into two semantically distinct sections: a *thinking* section $\mathbf{y}_{\text{thinking}}$ and an *answer* section $\mathbf{y}_{\text{answer}}$. This structural division is uniquely enabled by dLLMs' bidirectional nature: Unlike autoregressive models where reasoning must be sequentially generated before any answer content becomes available, dLLMs can simultaneously consider both reasoning and answer contexts during iterative refinement. Formally, we define the generation sequence as:

$$\mathbf{y}_{\text{gen}} = (\mathbf{y}_{\text{thinking}}, \mathbf{y}_{\text{answer}}) \tag{6}$$

Consequently, the complete input sequence is structured as:

$$\mathbf{y} = (\mathbf{y}_{\text{prompt}}, \mathbf{y}_{\text{thinking}}, \mathbf{y}_{\text{answer}}) \tag{7}$$

To establish clear demarcation between these functional sections, we introduce a task-specific *answer indicator* positioned between the thinking and answer sections. This indicator serves as an explicit signal for the model to transition from reasoning elaboration to final response formulation.

Building upon this foundation, we further guide the model toward systematic reasoning decomposition by partitioning the thinking section $\mathbf{y}_{\text{thinking}}$ into multiple explicit reasoning steps $T = (T_1, T_2, \ldots, T_{N_t})$. This is achieved by embedding explicit step-by-step reasoning templates among the masked tokens. The thinking section is initialized as:

$$\mathbf{y}_{\text{thinking}}^{(N)} = (\underbrace{T_1, T_2, \ldots, T_{N_t}}_{N_t \text{ reasoning steps}}) \tag{8}$$

This methodology of embedding structural reasoning cues directly within the generation space effectively guides the dLLM to produce explicit, traceable reasoning sequences prior to answer formulation, thereby enhancing both transparency and accuracy of the reasoning process. Crucially, this in-place approach leverages the concurrent answer accessibility of dLLMs, enabling the model to maintain holistic awareness of both reasoning development and answer formation throughout the iterative refinement process.

Two-Phase Decoding with Early Exit Mechanism

While our in-place chain-of-thought prompting significantly enhances reasoning capabilities, the iterative nature of dLLM inference introduces substantial computational overhead. To address this challenge, we leverage the concurrent answer accessibility of dLLMs for confidence-aware optimization. We observe a critical pattern in the confidence dynamics during iterative refinement: the model's confidence in answer tokens exhibits rapid convergence to high levels and maintains remarkable stability throughout subsequent iterations, while the thinking section continues undergoing refinement (Figure 2). This asymmetric convergence pattern suggests that continuing refinement beyond early confidence stabilization yields diminishing returns in answer quality while incurring unnecessary computational costs.

This phenomenon reveals that models often internally converge on correct answers substantially earlier than the completion of explicit reasoning traces. Continuing iterative refinement beyond this confidence convergence point yields diminishing returns in answer quality while incurring unnecessary computational costs, as subsequent iterations primarily serve to elaborate and refine reasoning steps rather than improve final answer accuracy.

Leveraging this insight, we introduce an efficient inference strategy comprising a two-phase decoding process with a confidence-based early-exit mechanism. The core innovation lies in using a confidence threshold τ to dynamically control the transition between phases, enabling adaptive computation based on the model's internal state.

Confidence-Based Phase Switching. Throughout decoding, we continuously monitor the average confidence of the masked answer tokens. We first compute the confidence score for each individual answer token:

$$\text{confidence}_i^{(k)} = \max_{v \in \mathcal{T}} f_\theta(\mathbf{y}_{0,i} = v \mid \mathbf{y}^{(k)}) \tag{9}$$

Then, we calculate the average confidence across all masked answer tokens:

$$\text{avg\_confidence}_{\text{answer}}^{(k)} = \frac{1}{L_{\text{answer}}} \sum_{i \in \text{answer indices}} \text{confidence}_i^{(k)} \tag{10}$$

This confidence threshold τ serves as the decisive criterion for phase transition: when $\text{avg\_confidence}_{\text{answer}}^{(k)} \geq \tau$, we trigger an early exit from the reasoning phase and immediately transition to answer generation.

Phase 1: Reasoning Phase. During this initial phase, we focus on generating the thinking trace ($\mathbf{y}_{\text{thinking}}$) while maintaining all answer tokens in their masked state. The transition function S (Equation 5) unmasks tokens solely within $\mathbf{y}_{\text{thinking}}$, guided by model confidence scores. Crucially, we

simultaneously monitor the answer confidence for early exit detection. When the confidence threshold is reached, this phase terminates immediately.

Phase 2: Answer Generation Phase. This phase is triggered exclusively when $\text{avg\_confidence}_{\text{answer}}^{(k)} \geq \tau$. Upon transition, we perform a single-step decoding operation to reveal the complete answer sequence $\mathbf{y}_{\text{answer}}$. This dynamic phase switching eliminates redundant computation while preserving accuracy, as the model has demonstrated enough confidence in the final answer. A detailed algorithm overview of the ICE framework is provided in the appendix.

Experiments

Experimental Setup

We evaluate ICE on diverse benchmarks: GSM8K (Cobbe et al. 2021) and MATH (Hendrycks et al. 2021) for mathematical reasoning, and MMLU (Hendrycks et al. 2020) and GPQA (Rein et al. 2023) for knowledge-intensive reasoning. We conduct comprehensive evaluations using two representative dLLMs: LLaDA-8B-Instruct (Nie et al. 2025) and LLaDA-1.5 (Zhu et al. 2025), with the Language Model Evaluation Harness framework (Gao et al. 2024) on 8 NVIDIA H100 GPUs. We compare ICE against two baseline approaches: (1) *Vanilla*: the standard autoregressive generation approach, and (2) *Prefix CoT*: traditional chain-of-thought prompting as input prefixes. ICE offers two operational modes: ICE-SP (speed-prioritized) for maximum acceleration and ICE-PP (performance-prioritized) for superior accuracy, which are configured by the hyperparameters.

Main Results

Our main results are summarized in Table 1.

Mathematical Reasoning Tasks. For complex mathematical reasoning problems, ICE-PP achieves substantial accuracy improvements: +17.29% and +13.03% improvements on GSM8K with 1.67× and 1.52× speedups, and +3.00% and +2.54% improvements on MATH while maintaining computational efficiency. ICE-SP delivers exceptional efficiency (4.12× and 3.21× on GSM8K, 1.67× and 1.70× on MATH) with near-zero accuracy loss, demonstrating versatile optimization capabilities.

Knowledge-Intensive Tasks. For knowledge-intensive reasoning tasks, ICE achieves simultaneous accuracy improvements and remarkable efficiency gains. GPQA shows substantial improvements of +4.91% and +5.57% with extraordinary speedups of 19.24× and 42.08×, demonstrating effectiveness on tasks requiring deep domain expertise. MMLU achieves +13.10% and +0.74% improvements with exceptional speedups up to 133.08× and 276.67×, with the larger accuracy gain for LLaDA Instruct indicating particular benefits for non-preference-optimized models. These results suggest that diverse reasoning queries can be effectively resolved through early-layer representations without requiring explicit speed-accuracy trade-offs.

Latency-Accuracy Trade-off Comparison. Figure 4 compares ICE with vanilla baselines that achieve different trade-off points by fixing output length and adjusting generation steps. ICE establishes superior Pareto frontiers on both

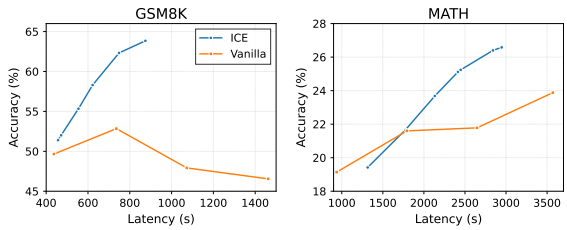

Figure 4: Latency-accuracy trade-off comparison between ICE and vanilla baselines. ICE demonstrates superior Pareto frontiers on both GSM8K and MATH datasets.

datasets: reducing latency by 2-4× on GSM8K while maintaining accuracy, and achieving both lower latency and higher accuracy on MATH.

Compatibility with dLLM-Cache. To validate ICE's compatibility with existing optimization techniques, we evaluate its integration with dLLM-Cache. Table 2 demonstrates that ICE maintains its effectiveness when combined with caching mechanisms, achieving substantial additional speedups while preserving accuracy. For ICE-SP, cache acceleration enhances speedup from 4.12× to 6.10× on GSM8K and from 1.67× to 2.02× on MATH. The ICE-PP similarly benefits from cache integration, achieving 2.47× speedup on GSM8K and 1.33× on MATH. The accuracy degradation introduced by caching is minimal (typically $<2\%$), confirming that our structured reasoning approach remains robust with complementary optimization strategies.

Ablation Study

To gain deeper insights into ICE's design choices, we conduct ablation studies focusing on the key hyperparameters and architectural decisions that drive the performance.

Impact of Core Components. Table 3 demonstrates the incremental contributions of ICE's architectural components. The initial thinking/answer segmentation mechanism (+ Segment) provides consistent improvements across all benchmarks, yielding +9.40% and +7.35% gains on GSM8K for LLaDA-8B-Instruct and LLaDA-1.5 respectively, validating the fundamental importance of explicit reasoning structure in discrete masked language models. The introduction of structured thinking subdivision further enhances performance, contributing an additional +8.42% and +2.50% on GSM8K, demonstrating that fine-grained decomposition of reasoning processes enables more effective utilization of the masked generation paradigm. The confidence-based early exit mechanism (ICE) exhibits task-dependent behavior: while showing marginal accuracy trade-offs on GSM8K (-0.53% for LLaDA-8B-Instruct), it provides substantial improvements on knowledge-intensive tasks like GPQA (+0.67% and +3.57%), indicating that the early exit strategy is particularly effective when reasoning complexity varies significantly across tasks.

Effect of Reasoning Steps (N_t). Figure 5 demonstrates the critical relationship between reasoning step granularity and model performance. Our systematic analysis reveals distinct optimal operating points for different task domains:

Task	Method	LLaDA-8B-Instruct			LLaDA-1.5		
		Accuracy ↑	Latency (s) ↓	Speedup ↑	Accuracy ↑	Latency (s) ↓	Speedup ↑
GSM8K	Vanilla	46.55	1461.83	1.00×	45.19	3376.21	1.00×
	+ Prefix CoT	$40.49_{-6.06}$	$1443.02_{-18.81}$	1.01×	$34.95_{-10.24}$	$3555.10_{+178.89}$	0.95×
	ICE-SP	$46.01_{-0.54}$	$\mathbf{354.86}_{-1106.97}$	**4.12**×	$45.56_{+0.37}$	$\mathbf{1050.48}_{-2325.73}$	**3.21**×
	ICE-PP	$\mathbf{63.84}_{+17.29}$	$874.53_{-587.30}$	1.67×	$\mathbf{58.22}_{+13.03}$	$2221.24_{-1154.97}$	1.52×
MATH	Vanilla	23.88	3570.13	1.00×	22.74	10018.06	1.00×
	+ Prefix CoT	$22.26_{-1.62}$	$3517.61_{-52.52}$	1.01×	$19.76_{-2.98}$	$10034.73_{+16.67}$	1.00×
	ICE-SP	$23.68_{-0.20}$	$\mathbf{2132.79}_{-1437.34}$	**1.67**×	$22.74_{0.00}$	$\mathbf{5885.87}_{-4132.19}$	**1.70**×
	ICE-PP	$\mathbf{26.88}_{+3.00}$	$3155.99_{-414.14}$	1.13×	$\mathbf{25.28}_{+2.54}$	$8774.40_{-1243.66}$	1.14×
GPQA	Vanilla	27.46	970.84	1.00×	28.13	2156.62	1.00×
	+ Prefix CoT	$27.68_{+0.22}$	$1043.77_{+72.93}$	0.93×	$27.90_{-0.23}$	$2234.75_{+78.13}$	0.97×
	ICE	$\mathbf{32.37}_{+4.91}$	$\mathbf{50.45}_{-920.39}$	**19.24**×	$\mathbf{33.70}_{+5.57}$	$\mathbf{51.24}_{-2105.38}$	**42.08**×
MMLU	Vanilla	49.67	19396.79	1.00×	60.35	47795.97	1.00×
	+ Prefix CoT	$51.22_{+1.55}$	$19469.64_{+72.85}$	1.00×	$60.11_{-0.24}$	$48298.11_{+502.14}$	0.99×
	ICE	$\mathbf{62.77}_{+13.10}$	$\mathbf{145.75}_{-19251.04}$	**133.08**×	$\mathbf{61.09}_{+0.74}$	$\mathbf{172.79}_{-47623.18}$	**276.67**×

Table 1: Performance comparison of ICE on different tasks. Experiments are conducted using LLaDA-8B-Instruct (length 256) and LLaDA-1.5 (length 512), with block length and generation steps matching generation length. *Prefix CoT* refers to the vanilla with CoT prompting as prefixes. ICE operates in SP mode for inference acceleration and PP mode for enhanced accuracy. For MMLU and GPQA, a unified configuration achieves both superior accuracy improvements and efficiency gains.

Task	Method	Acc.	Latency	Speedup
GSM8K	Vanilla	46.55	1461.83	1.00×
	ICE-SP	46.01	354.86	4.12×
	+ Cache	**46.17**	**239.83**	**6.10**×
	ICE-PP	**63.84**	874.53	1.67×
	+ Cache	61.56	**592.73**	**2.47**×
MATH	Vanilla	23.88	3570.13	1.00×
	ICE-SP	23.68	2132.79	1.67×
	+ Cache	**23.72**	**1771.15**	**2.02**×
	ICE-PP	**26.88**	3155.99	1.13×
	+ Cache	25.94	**2684.77**	**1.33**×

Table 2: Compatibility evaluation of ICE with dLLM-Cache on LLaDA-8B-Instruct across GSM8K and MATH.

Task	Vanilla	+ Segment (T/A)	+ Structured Thinking	+ Early Exit (ICE)
LLaDA-8B-Instruct				
GSM8K	46.55	55.95	**64.37**	63.84
MATH	23.88	24.18	**26.88**	**26.88**
GPQA	27.68	29.91	31.70	**32.37**
LLaDA-1.5				
GSM8K	45.19	52.54	55.04	**58.22**
MATH	22.74	23.74	24.64	**25.28**
GPQA	27.90	29.91	30.13	**33.70**

Table 3: Ablation study on the core components of ICE.

GSM8K achieves peak performance at $N_t = 3$ with approximately 58-60% accuracy, while MATH demonstrates optimal results at $N_t = 4$ with 25-26% accuracy. This finding suggests that mathematical reasoning tasks benefit from moderate decomposition depths that balance reasoning granularity with computational overhead. Notably, excessive subdivision ($N_t > 6$) consistently degrades performance across all benchmarks, indicating that overly fine-grained reasoning steps may introduce noise and computational inefficiency. For knowledge-intensive tasks like GPQA, the framework maintains stable performance across a broader range of thought numbers (28-31% accuracy), demonstrating robustness to hyperparameter variations.

Mask Token Allocation Strategies. We evaluate three distinct strategies for distributing mask tokens across reasoning steps: *uniform* allocation maintains equal token budgets for each thought, *front-heavy* allocation prioritizes initial reasoning steps, and *back-heavy* allocation concentrates computational resources on final reasoning phases. The experimental results demonstrate that back-heavy and front-heavy strategies consistently outperform uniform allocation, particularly in lower thought number regimes ($N_t \leq 4$). This finding suggests that strategic token concentration yields superior performance compared to equal resource distribution, highlighting the importance of adaptive computational allocation in structured reasoning frameworks.

Confidence Threshold Analysis. The confidence threshold τ serves as a pivotal hyperparameter governing the speed-accuracy trade-off in our early exit mechanism. Figure 6 systematically illustrates this relationship across mathematical reasoning benchmarks (GSM8K and MATH). Our analysis reveals a clear pattern: lower thresholds aggressively prioritize computational efficiency at the expense of accuracy, while higher thresholds emphasize accuracy preservation

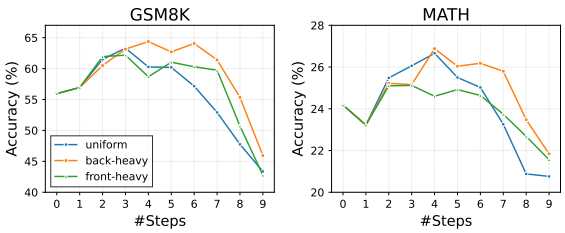

Figure 5: Ablation study on reasoning steps (N_t).

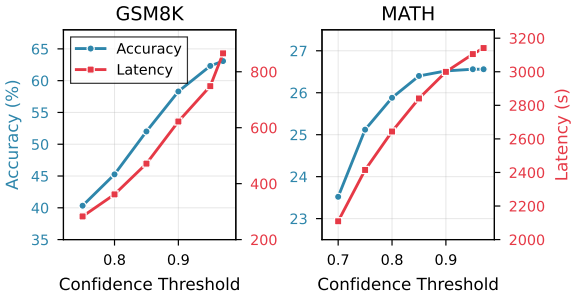

Figure 6: Ablation study on the confidence threshold τ.

with reduced speedup benefits. Critically, moderate thresholds achieve the optimal balance, delivering substantial accuracy improvements while maintaining significant computational efficiency gains, thereby validating the effectiveness of our confidence-based early exit strategy. For knowledge-intensive tasks (GPQA and MMLU), we observe that excessively high confidence thresholds yield diminishing returns, with accuracy plateauing rather than improving further. This finding underscores the importance of task-adaptive threshold selection, where reasoning complexity should inform the optimal confidence calibration for maximum effectiveness.

Discussion

In-Place Prompting: A New Paradigm for dLLMs

Our work repositions chain-of-thought from a sequential pre-computation into a dynamic, symbiotic component of the dLLM's iterative refinement process. By embedding prompts directly within the generation canvas, we move beyond the prefix-only constraints of autoregressive models. This integration fosters a co-refinement dynamic where the reasoning trace and the final answer evolve in parallel, mutually informing one another throughout generation steps.

This paradigm shift is unique to dLLMs. Unlike autoregressive models where a generated reasoning step is immutable, our approach allows the model to revisit and refine its entire line of thought in light of the emerging answer. The reasoning templates $(T_1, \ldots, T_{N_t})$ act less like rigid instructions and more like a flexible scaffold that guides the refinement process while adapting to the model's evolving understanding of the problem. This suggests a fundamental compatibility between structured problem decomposition and the core mechanics of diffusion models, transforming reasoning from a static prerequisite into a concurrently optimized process. The implications suggest that dLLMs could be architected with internal scaffolds for other complex tasks like constrained generation and planning.

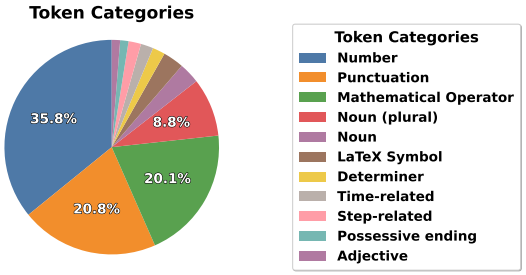

Figure 7: Statistics of token categories decoded when confidence rapidly changes during generation on GSM8K.

Internal Dynamics in dLLMs

Decoupling Solution Finding from Explanation Generation. Our findings reveal that dLLMs effectively decouple solution finding from explanation generation. The rapid confidence convergence in answer sections, while reasoning traces remain unstable, indicates that models stabilize final answers before completing narrative justifications. This contrasts with AR models where solutions and explanations form sequential chains, which enables our early-exit mechanism to leverage a fundamental property of dLLM rather than merely providing computational efficiency.

Token-Level Confidence Dynamics. Token-level analysis reveals that confidence maturation occurs through decisive jumps rather than gradual increases. Figure 7 presents the distribution of token categories that are decoded when confidence rapidly changes during GSM8K generation. The statistics show these critical shifts are primarily driven by numerical token stabilization. The model first anchors quantitative results, then refines punctuation and mathematical operators to solidify reasoning syntax. This hierarchical convergence process locks core results before constructing explanatory scaffolding. Interestingly, this finding echoes observations from SepLLM (Chen et al. 2025), which identified high attention scores for punctuation tokens in autoregressive LLMs, suggesting that structural tokens play crucial roles across different model architectures.

Conclusion

We introduce ICE, a novel framework that enhances both reasoning capabilities and inference efficiency in dLLMs through in-place CoT prompting and two-phase decoding with confidence-aware early exit mechanism. Our approach leverages dLLMs' natural advantages of bidirectional attention, achieving significant improvements with up to 17.29% accuracy gains and 4.12× speedup on GSM8K, and up to 276.67× acceleration on MMLU. This work demonstrates that architectural alignment between reasoning patterns and generation mechanisms can yield synergistic benefits, transforming iterative refinement from computational burden into architectural advantage and establishing a new paradigm for efficient inference in non-autoregressive language models.

References

Austin, J.; Johnson, D. D.; Ho, J.; Tarlow, D.; and van den Berg, R. 2021a. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34: 17981–17993.

Austin, J.; Odena, A.; Nye, M.; Bosma, M.; Michalewski, H.; Dohan, D.; Jiang, E.; Cai, C.; Terry, M.; Le, Q.; et al. 2021b. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Berglund, L.; Tong, M.; Kaufmann, M.; Balesni, M.; Stickland, A. C.; Korbak, T.; and Evans, O. 2023. The Reversal Curse: LLMs trained on" A is B" fail to learn" B is A". *arXiv preprint arXiv:2309.12288*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Chen, G.; Shi, H.; Li, J.; Gao, Y.; Ren, X.; Chen, Y.; Jiang, X.; Li, Z.; Liu, W.; and Huang, C. 2025. SepLLM: Accelerate Large Language Models by Compressing One Segment into One Separator. In *International Conference on Machine Learning*. Also available at arXiv:2412.12094.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Devlin, J. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Gao, L.; Tow, J.; Abbasi, B.; Biderman, S.; Black, S.; DiPofi, A.; Foster, C.; Golding, L.; Hsu, J.; Le Noac'h, A.; Li, H.; McDonell, K.; Muennighoff, N.; Ociepa, C.; Phang, J.; Reynolds, L.; Schoelkopf, H.; Skowron, A.; Sutawika, L.; Tang, E.; Thite, A.; Wang, B.; Wang, K.; and Zou, A. 2024. A framework for few-shot language model evaluation.

Gao, Y.; and et al. 2023. Synthesizing and Debugging Chain-of-Thought Prompts with Large Language Models. *arXiv*.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; and Steinhardt, J. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Hu, Z.; Meng, J.; Akhauri, Y.; Abdelfattah, M. S.; Seo, J.-s.; Zhang, Z.; and Gupta, U. 2025. Accelerating diffusion language model inference via efficient kv caching and guided diffusion. *arXiv preprint arXiv:2505.21467*.

Kojima, T.; Hashimoto, D.; Shimbo, T.; and Inui, K. 2022. Large Language Models Are Strong Reasoners. *arXiv*.

Liu, K.; et al. 2025a. Mercury: Ultra-Fast Language Models Based on Diffusion. *arXiv preprint arXiv:2506.17298*.

Liu, X.; Liu, Z.; Huang, Z.; Guo, Q.; He, Z.; and Qiu, X. 2025b. LongLLaDA: Unlocking Long Context Capabilities in Diffusion LLMs. *arXiv preprint arXiv:2506.14429*.

Liu, Z.; Yang, Y.; Zhang, Y.; Chen, J.; Zou, C.; Wei, Q.; Wang, S.; and Zhang, L. 2025c. dllm-cache: Accelerating diffusion large language models with adaptive caching. *arXiv preprint arXiv:2506.06295*.

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

Luxembourg, O.; Permuter, H.; and Nachmani, E. 2025. Plan for Speed–Dilated Scheduling for Masked Diffusion Language Models. *arXiv preprint arXiv:2506.19037*.

Ma, X.; Yu, R.; Fang, G.; and Wang, X. 2025. dkv-cache: The cache for diffusion language models. *arXiv preprint arXiv:2505.15781*.

Nakamura, Y.; and et al. 2021. WebGPT: Browser-assisted question-answering with human feedback. In *NeurIPS*.

Nie, S.; Zhu, F.; You, Z.; Zhang, X.; Ou, J.; Hu, J.; Zhou, J.; Lin, Y.; Wen, J.-R.; and Li, C. 2025. Large Language Diffusion Models. *arXiv preprint arXiv:2502.09992*.

Rein, D.; Hou, B. L.; Stickland, A. C.; Petty, J.; Pang, R. Y.; Dirani, J.; Michael, J.; and Bowman, S. R. 2023. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*.

Sahoo, S. S.; Arriola, M.; Schiff, Y.; Gokaslan, A.; Marroquin, E.; Chiu, J. T.; Rush, A.; and Kuleshov, V. 2024. Simple and effective masked diffusion language models. *arXiv preprint arXiv:2406.07524*.

Shi, J.; Han, K.; Wang, Z.; Doucet, A.; and Titsias, M. K. 2024. Simplified and Generalized Masked Diffusion for Discrete Data. *arXiv preprint arXiv:2406.04329*.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Wei, J.; Chen, X.; Yang, Y.; Klein, D.; Polyak, A.; Chandra, S.; M., S.; and Tan. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *arXiv*.

Wei, Q.; Zhang, Y.; Liu, Z.; Liu, D.; and Zhang, L. 2025. Accelerating Diffusion Large Language Models with SlowFast: The Three Golden Principles. *arXiv preprint arXiv:2506.10848*.

Wen, Z.; Qu, J.; Liu, D.; Liu, Z.; Wu, R.; Yang, Y.; Jin, X.; Xu, H.; Liu, X.; Li, W.; et al. 2025. The Devil behind the mask: An emergent safety vulnerability of Diffusion LLMs. *arXiv preprint arXiv:2507.11097*.

Wu, C.; Zhang, H.; Xue, S.; Liu, Z.; Diao, S.; Zhu, L.; Luo, P.; Han, S.; and Xie, E. 2025. Fast-dLLM: Training-free Acceleration of Diffusion LLM by Enabling KV Cache and Parallel Decoding. arXiv:2505.22618.

Yang, L.; Tian, Y.; Li, B.; Zhang, X.; Shen, K.; Tong, Y.; and Wang, M. 2025. Mmada: Multimodal large diffusion language models. *arXiv preprint arXiv:2505.15809*.

Ye, J.; Xie, Z.; Zheng, L.; Gao, J.; Wu, Z.; Jiang, X.; Li, Z.; and Kong, L. 2025. Dream 7B.

Zhao, W. X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Zhu, F.; Wang, R.; Nie, S.; Zhang, X.; Wu, C.; Hu, J.; Zhou, J.; Chen, J.; Lin, Y.; Wen, J.-R.; and Li, C. 2025. LLaDA 1.5: Variance-Reduced Preference Optimization for Large Language Diffusion Models. *ArXiv*, abs/2505.19223.

Reproducibility Checklist

Instructions for Authors:

This document outlines key aspects for assessing reproducibility. Please provide your input by editing this `.tex` file directly.

For each question (that applies), replace the "yes" text with your answer.

Example: If a question appears as

```
\question{Proofs of all novel claims
are included} {(yes/partial/no)}
yes
```

you would change it to:

```
\question{Proofs of all novel claims
are included} {(yes/partial/no)}
yes
```

Please make sure to:

- Replace ONLY the "yes" text and nothing else.
- Use one of the options listed for that question (e.g., **yes**, **no**, **partial**, or **NA**).
- **Not** modify any other part of the `\question` command or any other lines in this document.

You can `\input` this .tex file right before `\end{document}` of your main file or compile it as a stand-alone document. Check the instructions on your conference's website to see if you will be asked to provide this checklist with your paper or separately.

---

1. General Paper Structure

1.1. Includes a conceptual outline and/or pseudocode description of AI methods introduced (yes/partial/no/NA) yes

1.2. Clearly delineates statements that are opinions, hypothesis, and speculation from objective facts and results (yes/no) yes

1.3. Provides well-marked pedagogical references for less-familiar readers to gain background necessary to replicate the paper (yes/no) yes

2. Theoretical Contributions

2.1. Does this paper make theoretical contributions? (yes/no) no

If yes, please address the following points:

2.2. All assumptions and restrictions are stated clearly and formally (yes/partial/no) NA

2.3. All novel claims are stated formally (e.g., in theorem statements) (yes/partial/no) NA

2.4. Proofs of all novel claims are included (yes/partial/no) NA

2.5. Proof sketches or intuitions are given for complex and/or novel results (yes/partial/no) NA

2.6. Appropriate citations to theoretical tools used are given (yes/partial/no) NA

2.7. All theoretical claims are demonstrated empirically to hold (yes/partial/no/NA) NA

2.8. All experimental code used to eliminate or disprove claims is included (yes/no/NA) NA

3. Dataset Usage

3.1. Does this paper rely on one or more datasets? (yes/no) yes

If yes, please address the following points:

3.2. A motivation is given for why the experiments are conducted on the selected datasets (yes/partial/no/NA) yes

3.3. All novel datasets introduced in this paper are included in a data appendix (yes/partial/no/NA) yes

3.4. All novel datasets introduced in this paper will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no/NA) yes

3.5. All datasets drawn from the existing literature (potentially including authors' own previously published work) are accompanied by appropriate citations (yes/no/NA) yes

3.6. All datasets drawn from the existing literature (potentially including authors' own previously published work) are publicly available (yes/partial/no/NA) yes

3.7. All datasets that are not publicly available are described in detail, with explanation why publicly available alternatives are not scientifically satisficing (yes/partial/no/NA) yes

4. Computational Experiments

4.1. Does this paper include computational experiments? (yes/no) yes

If yes, please address the following points:

4.2. This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting (yes/partial/no/NA) yes

4.3. Any code required for pre-processing data is included in the appendix (yes/partial/no) yes

4.4. All source code required for conducting and analyzing the experiments is included in a code appendix (yes/partial/no) yes

4.5. All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no) yes

4.6. All source code implementing new methods have comments detailing the implementation, with references to the paper where each step comes from (yes/partial/no) yes

4.7. If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results (yes/partial/no/NA) yes

4.8. This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks (yes/partial/no) yes

4.9. This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics (yes/partial/no) yes

4.10. This paper states the number of algorithm runs used to compute each reported result (yes/no) yes

4.11. Analysis of experiments goes beyond single-dimensional summaries of performance (e.g., average; median) to include measures of variation, confidence, or other distributional information (yes/no) yes

4.12. The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank) (yes/partial/no) yes

4.13. This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments (yes/partial/no/NA) yes