Interleaved Latent Thinking and Adaptive Termination for Efficient Reasoning LLMs

Zhao Jin[1], **Rong-Cheng Tu**[1,†], **Wenhao Sun**[1], **Qi Guo**[1],
Mingyang Yu[2], **Hao Guan**[1], **Dacheng Tao**[1,†]

[1]Nanyang Technological University [2]City University of Hong Kong
{zhao.jin,rongcheng.tu,dacheng.tao}@ntu.edu.sg

Abstract

Emerging latent reasoning paradigms allow Large Language Models to "think" in latent embedding spaces, offering a more efficient alternative to explicit Chain-of-Thought. However, they generally face two critical limitations: First, relying exclusively on latent tokens amplifies *uncertainty* due to their inherently high entropy, thereby degrading solution correctness. Second, the *overthinking* problem remains prevalent, and existing approaches typically rely on rigid heuristics for stopping, often resulting in suboptimal termination. To address these challenges, we view efficient reasoning as a learnable control problem over both *how to think* and *when to stop*. We formulate this within a Reinforcement Learning framework, training lightweight per-step gates on a frozen LLM. This modulation enables the model to (i) adaptively interleave latent soft-token reasoning with explicit token generation, and (ii) predict cumulative exit probabilities for robust early termination. Experiments on diverse reasoning benchmarks across varying model scales and families demonstrate that our method surpasses both standard CoT and existing latent baselines, achieving accuracy gains of $\sim$3 percentage points while reducing total reasoning length by up to 40%.

1 Introduction

Reasoning has become a core capability of modern large language models (LLMs). This paradigm shifts from simple short-form question answering to rigorous step-by-step computation and verification, demonstrating strong performance on various complex mathematical and scientific tasks [6, 29, 16, 15, 62].

Most current reasoning pipelines rely on explicit Chain-of-Thought (CoT) [49], which externalizes intermediate computation as a verbose token sequence. While effective, CoT tightly couples reasoning with discrete decoding: it forces the model to commit to a single textual path early, incurs substantial overhead in intermediate text, and often leads to overthinking. Such long and redundant deliberation wastes tokens and may even end in truncation or degeneration (Figure 1). Moreover, this step-by-step externalization contrasts with human cognition, where reasoning often proceeds implicitly and ambiguity can persist until sufficient evidence accumulates.

To better emulate such internal computation, latent reasoning [68, 7] internalizes intermediate steps into continuous representations (e.g., hidden states or soft tokens), effectively bypassing the discrete information bottleneck and preserving higher per-step bandwidth. Existing latent-reasoning methods broadly fall into two paradigms. *Training-based* approaches fine-tune LLMs to compress explicit CoT trajectories into latent states [21, 14]. While effective, they typically require extensive

[†]Co-corresponding authors: Rong-Cheng Tu and Dacheng Tao.

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

optimization on LLMs, which increases training costs and reduces deployability across settings. In contrast, *training-free* approaches construct latent computation at inference time—for example, Soft Thinking [64] forms soft tokens via probability-weighted mixtures over vocabulary embeddings. By maintaining reasoning paths in superposition, training-free methods preserve high-dimensional bandwidth [67] while avoiding additional fine-tuning for feature alignment, making them an attractive foundation for efficient reasoning.

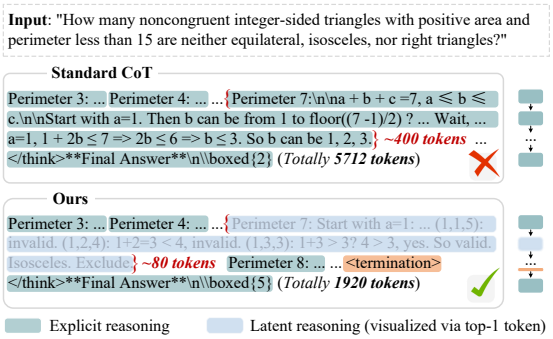


Figure 1: Standard CoT relies on verbose discrete decoding, often leading to overthinking and truncation due to token exhaustion. Our approach adaptively interleaves detailed explicit reasoning and high-bandwidth latent reasoning, and triggers early termination upon reasoning sufficiency, thereby improving inference efficiency.

However, two critical challenges remain in the soft token-based paradigm. First, relying solely on soft tokens amplifies *uncertainty* due to their inherently high entropy, which can lead to unstable reasoning behavior and degrade solution correctness. Second, *overthinking* persists in latent reasoning similarly to explicit CoT, making it crucial to regulate reasoning duration. To address these issues, prior work [41] uses entropy-based heuristics at inference time to switch between latent and explicit modes and to trigger early exit. However, entropy trajectories during reasoning are inherently noisy and non-monotonic, and their magnitudes are not consistently aligned with reasoning sufficiency. As a result, entropy-triggered switching and early exit remain suboptimal for efficient latent reasoning.

Moving beyond static rules, we view efficient reasoning as a *learnable control problem* that jointly decides *how to think* and *when to stop*. Concretely, keeping the backbone LLM frozen, we introduce a lightweight per-step modulation gate that leverages hidden states to output two parallel decisions. The first is a *mode decision* that toggles between explicit token sampling and latent soft-token mixing, enabling **Interleaved Thinking** to exploit the complementarity of these paradigms. The second is an *exit decision* that predicts reasoning sufficiency, enabling **Adaptive Termination** to eliminate redundant deliberation.

To train this control policy without expensive step-wise annotations, we formulate learning within a Reinforcement Learning framework and employ Group Relative Policy Optimization (GRPO) to optimize gate decisions directly from outcome signals. Crucially, we model termination as a *cumulative survival analysis* process: instead of isolated binary exit predictions, the model learns a survival curve that smoothly reflects accumulated reasoning evidence. To encourage robust exploration over different rollout horizons, we further introduce dynamic termination thresholds during training. Extensive experiments across multiple reasoning LLMs demonstrate that our method achieves superior performance while significantly compressing inference length.

Our main contributions are summarized as follows: 1) We propose a unified RL-based framework that dynamically orchestrates *Interleaved Thinking* and *Adaptive Termination* in LLM reasoning. 2) We formulate termination as a cumulative survival analysis process and propose a dynamic-threshold-based GRPO training strategy to enhance exploration efficiency. 3) Extensive experiments demonstrate that our method significantly compresses inference length while maintaining high reasoning accuracy.

2 Related Work

2.1 Explicit LLM Reasoning

A prevailing paradigm for enhancing LLM reasoning is to *externalize* intermediate computation via Chain-of-Thought prompting (CoT) [49, 27]. To improve robustness, subsequent works have extended single-path reasoning into structured exploration [63] (e.g., ToT [57]) or self-consistency aggregations [48]. Beyond static prompting, *Reinforcement Learning* (RL) has emerged as a critical method for internalizing these capabilities [66]. Seminal works like STaR [60] illustrate that models can bootstrap reasoning capabilities through iterative self-correction. While some approaches rely

on dense Process Reward Models (PRMs) [30] for fine-grained supervision, recent breakthroughs like DeepSeek-R1 [12] demonstrate the efficacy of outcome-based RL via Group Relative Policy Optimization (GRPO) [36]. However, these methods are constrained by discrete decoding, which commits to a single token at each step. This inherently limits the information bandwidth of the reasoning process and discards alternative hypothesis paths.

2.2 Latent LLM Reasoning

Latent reasoning shifts from discrete text to continuous representations, encoding intermediate computation in *hidden states* or *soft tokens* with significantly higher per-step bandwidth. Prior research generally falls into two paradigms. Training-based methods [21, 14, 50, 39, 47, 13, 59, 54, 65, 4] learn continuous reasoning representations through supervision, distillation, or reinforcement learning. Related pretraining methods incorporate continuous concepts alongside text [45]. Another related line explores recurrent or iterative latent computation, where models repeatedly refine hidden or latent states before decoding final answers [18, 26, 61, 3, 9, 69]. Training-free methods, conversely, modify inference dynamics without updating the backbone, typically by approximating reasoning via probability-weighted embedding mixtures [64, 70, 51].

Recent hybrid latent-explicit methods further attempt to combine the efficiency of latent reasoning with the reliability of textual reasoning. HybridCoT [37] uses a structured interleaving scheme based on predefined training rules, while ThinkRouter [53] and SwiReasoning [41] rely on confidence- or entropy-based heuristic signals for routing. Token Assorted [43] trains with VQ-VAE-derived discrete latent tokens and adapts the backbone; our method uses soft embedding mixtures and trains only gates while freezing the backbone. In contrast, our Mode Gate learns latent-explicit switching from internal reasoning representations, enabling context-aware mode decision.

2.3 Efficient Reasoning Approaches

A growing body of work targets efficient reasoning by reducing unnecessary CoT generation and mitigating overthinking [8, 44]. *Training-based* methods shorten reasoning traces by explicitly shaping model behavior, e.g., supervised fine-tuning [34, 32] on concise or variable-length rationales, preference optimization [34, 33], or reinforcement learning [11, 38, 1, 19, 25, 42] with length-sensitive objectives. While effective, these methods incur substantial retraining costs and risk distorting the model's output distribution, potentially harming generalization. *Inference-time* methods avoid retraining and instead attempt to stop reasoning dynamically, via prompting/sampling optimizations [52, 20] or early-exit mechanisms [56, 17] that rely on logits, entropy, or confidence signals to decide when to stop. Separately, attention-head norms have been explored as internal signals for factual reliability [23]. Different from prior early-exit methods, our method learns an exit policy and aggregates step-wise hazard estimates through a survival process for robust adaptive termination.

3 Method

3.1 Preliminaries

We consider a pre-trained language model $\mathcal{M}$ defined over a vocabulary $\mathcal{V}$ with hidden dimension d. The model comprises an embedding matrix $\mathbf{E} \in \mathbb{R}^{|\mathcal{V}| \times d}$, a Transformer backbone f_θ, and a language head parameterized by $W_h \in \mathbb{R}^{|\mathcal{V}| \times d}$. Given a context sequence at step t, the backbone produces a contextualized hidden state $h_t \in \mathbb{R}^d$:

$$h_t = f_\theta(x_t, h_{<t}), \tag{1}$$

where x_t represents the input embedding at the current step. Based on this formulation, we distinguish between the following two reasoning paradigms:

Explicit Chain-of-Thought (CoT). In standard explicit decoding, the model commits to a *discrete* token at each step. The hidden state h_t is projected via the language head to produce a probability distribution p_t over the vocabulary $\mathcal{V}$:

$$p_t = \mathrm{softmax}(W_h h_t) \in \Delta^{|\mathcal{V}|-1}. \tag{2}$$

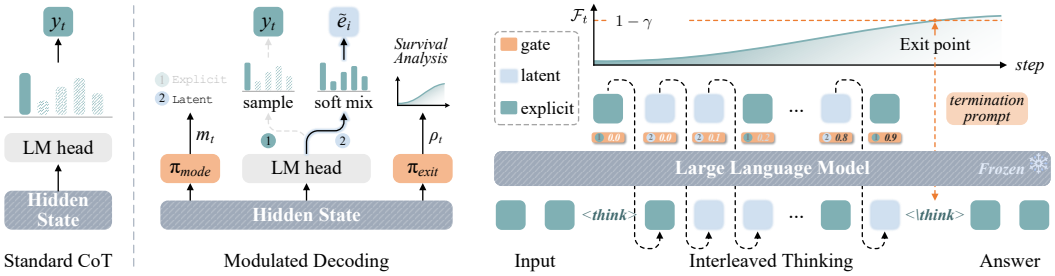


Figure 2: **Overview of our modulated adaptive reasoning framework.** Standard CoT performs discrete decoding at each step. In contrast, our modulated decoding attaches a lightweight controller to the frozen LLM hidden state, with a Mode Gate π_{mode} selecting between explicit token decoding and latent soft-token reasoning, and an Exit Gate π_{exit} estimating the instantaneous hazard rate ρ_t for termination. During inference, our method adaptively interleaves explicit tokens y_t and latent soft tokens $\tilde{e}_t$, and accumulates hazard rates into a cumulative stopping probability $\mathcal{F}_t$. The model terminates when $\mathcal{F}_t \geq 1-\gamma$, equivalently $\mathcal{S}_t \leq \gamma$, and then generates the final answer.

A discrete token y_t is then selected from p_t (e.g., via greedy search or sampling). Crucially, the input for the subsequent step is the specific embedding of the selected token, i.e., $x_{t+1} = \mathbf{E}[y_t]$. This process yields an interpretable text trajectory but explicitly collapses the uncertainty contained in p_t into a single categorical symbol at every step.

Latent Reasoning via Soft Thinking. To mitigate the information loss caused by discrete sampling, latent reasoning methods such as *Soft Thinking* [64] enable the model to think in a continuous space. Instead of committing to a single token, the model computes a **soft token** $\tilde{e}_t$ as the expectation of embeddings under the distribution p_t:

$$\tilde{e}_t = \mathbb{E}_{v\sim p_t}[\mathbf{E}[v]] = \sum_{v\in\mathcal{V}} p_t(v)\mathbf{E}[v]. \tag{3}$$

By setting the next-step input as $x_{t+1} = \tilde{e}_t$, the full distributional information is propagated through the reasoning chain in a continuous form. The resulting soft token retains information from competing local next-step alternatives in one embedding, rather than executing multiple complete reasoning chains in parallel.

3.2 Problem Formulation

Explicit CoT provides interpretability but often leads to information loss due to discrete sampling at each step. Conversely, latent reasoning maintains high-bandwidth distributional information, yet it inevitably introduces noise and increases uncertainty, which may cause reasoning instability. To harness their *complementary strengths*, we formulate the inference process of a frozen LLM $\mathcal{M}$ as a sequential decision-making problem governed by a lightweight controller π_ϕ ($|\phi| \ll |\theta|$). At each step t, given the hidden state h_t, the controller steers the reasoning trajectory via two simultaneous signals:

Mode Modulation (m_t). The controller determines the reasoning representation for the next step, switching between latent and explicit modes:

$$m_t \sim \pi_\phi(\cdot \mid h_t), \quad m_t \in \{\texttt{Latent}, \texttt{Explicit}\}. \tag{4}$$

Duration Modulation (e_t). Simultaneously, the controller decides whether to terminate the reasoning process based on the accumulated evidence:

$$e_t \sim \pi_\phi(\cdot \mid h_t), \quad e_t \in \{\texttt{Continue}, \texttt{Stop}\}. \tag{5}$$

3.3 Latent-Explicit Interleaved Reasoning

To enable dynamic switching between latent and explicit reasoning, we introduce a lightweight reasoning mode controller π_{mode}. As illustrated in Figure 2, this module operates on the frozen LLM's hidden states to determine the decoding paradigm at each step.

The Reasoning Mode Policy. At each decoding step t, given the hidden state $h_t \in \mathbb{R}^d$ generated by the backbone, the controller predicts the probability of entering the Latent Mode using a Multi-Layer Perceptron (MLP) with a Sigmoid activation σ:

$$P(m_t = \texttt{Latent} \mid h_t) = \sigma(\text{MLP}_\phi(h_t)). \tag{6}$$

During training, the mode m_t is sampled from this Bernoulli distribution for exploration. During inference, we use deterministic selection based on the mode probability, which acts as a switching signal governing the next-step input embedding x_{t+1}.

Specifically, the system executes one of the following operations to construct the reasoning trajectory:

- **Latent Mode** ($m_t = \texttt{Latent}$): The model bypasses the discrete sampling process. It directly utilizes the **soft token** $\tilde{e}_t$ (Eq. 3) as the next input, i.e., $x_{t+1} = \tilde{e}_t$.
- **Explicit Mode** ($m_t = \texttt{Explicit}$): A discrete token y_t is sampled from p_t, and its specific embedding is used as: $x_{t+1} = \mathbf{E}[y_t]$.

Based on this mechanism, we employ Reinforcement Learning to empower the controller to learn adaptive switching strategies, as detailed in Section 3.5.

3.4 Survival-Based Early Termination

While the interleaved reasoning mechanism optimizes *how* the model thinks, an efficient system must also decide *when* to stop. Naive early-exit approaches often treat each step as an isolated binary classification problem (stop vs. continue), which ignores the temporal accumulation of semantic information and often leads to premature or delayed exits.

To address this, we reformulate the termination decision as a **Cumulative Survival Analysis** process. Instead of making decisions based solely on the current step, the controller models the reasoning process as a lifespan, continuously estimating the probability of completion.

Hazard Rate Estimation. We formulate the termination signal e_t not as a binary classification label, but as a probabilistic *hazard rate*. Specifically, we parameterize the exit policy π_{exit} (instantiated as an MLP with parameters ψ) to predict a reference conditional probability of stopping at step t, given that the process has survived up to step $t-1$:

$$\rho_t = P_{\text{ref}}(e_t = \texttt{Stop} \mid h_t) = \sigma(\text{MLP}_\psi(h_t)), \tag{7}$$

where $\rho_t \in [0, 1]$ represents the instantaneous intensity of stopping. This value reflects the confidence that the current hidden state h_t contains sufficient information to derive the final answer.

Survival Function and Evidence Accumulation. To capture the cumulative nature of reasoning, we compute the survival function $\mathcal{S}_t$, which is the probability of continuing past step t under sequential Bernoulli sampling with hazards ρ_k. By the chain rule of survival analysis, this is the product of continuing probabilities across the trajectory:

$$\mathcal{S}_t = \prod_{k=1}^{t} (1 - \rho_k). \tag{8}$$

Correspondingly, the **Cumulative Stopping Probability** (or Failure CDF) $\mathcal{F}_t$ is the corresponding reference stopping CDF:

$$\mathcal{F}_t = 1 - \mathcal{S}_t = 1 - \prod_{k=1}^{t} (1 - \rho_k). \tag{9}$$

Unlike isolated predictions, $\mathcal{F}_t$ is structurally monotonic ($\mathcal{F}_t \geq \mathcal{F}_{t-1}$). This acts as a temporal integrator that filters out local fluctuations, providing theoretical guarantees on decision stability (as detailed in Appendix B) and ensuring that termination is driven by the global reasoning history. We define $\gamma \in (0, 1)$ as a *survival threshold* and $\mathcal{L}_t = \log \mathcal{S}_t$. Termination occurs when $\mathcal{L}_t \leq \log \gamma$, equivalently $\mathcal{S}_t \leq \gamma$ or $\mathcal{F}_t \geq 1 - \gamma$, followed by a termination prompt for final-answer generation. The stopping-time distribution induced by randomized thresholds is derived in Appendix A.

Crucially, our design enables *process-aware adaptivity*. By continuously monitoring the evolving reasoning state, it ensures that termination is dictated by the actual semantic sufficiency of the current trajectory. This design accommodates the stochastic nature of generation, where even identical queries may require varying compute paths.

3.5 Policy Optimization

Group Relative Policy Optimization (GRPO) [36] is employed to optimize our proposed modules, eliminating the need for a critic model by directly estimating baselines from group statistics.

Objective and Reward Formulation. For each query, we sample a group of trajectories $\{\tau_1, \ldots, \tau_G\}$ and optimize the policy to maximize the group-relative advantage. To incentivize concise reasoning without compromising accuracy, we design a group-relative length reward. Specifically, we strictly prioritize correctness, awarding an efficiency bonus only when the answer is correct:

$$R(\tau_i) = \begin{cases} 1 + \lambda \cdot (\bar{L}_{\text{correct}} - L(\tau_i)) & \text{if } \tau_i \text{ is correct} \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

where $L(\tau_i)$ is the length of the i-th trajectory, $\bar{L}_{\text{correct}}$ denotes the average length of correct samples within the group, and λ is the coefficient for length bonus. This formulation ensures that the model is driven to discover the most efficient path to the *correct* solution, dynamically penalizing redundancy based on the group's own performance baseline.

Sparse Control at Structural Boundaries. Given the extended nature of reasoning trajectories, step-wise decision-making is susceptible to local token noise, potentially disrupting the reasoning flow. To address this, we restrict the controller's activation to structural triggers $\mathcal{T}$ (e.g., "`\n\n`", "`.\n\n`"), which naturally demarcate *atomic reasoning segments*. This design effectively prevents the interruption of coherent thought processes and mitigates the instability arising from frequent mode switching.

Survival Exploration via Dynamic Thresholds. For each training trajectory i, we sample one log-survival threshold $\log \gamma_i \sim U(\alpha, \beta)$, with $\alpha < \beta \leq 0$, to explore different stopping horizons. The stop step is determined by:

$$t_{\text{stop}}^{(i)} = \min\{t \mid \mathcal{L}_t \leq \log \gamma_i\} = \min\{t \mid \mathcal{F}_t \geq 1 - \gamma_i\}. \tag{11}$$

Appendix A derives this log-uniform stopping distribution and the Bernoulli-equivalent uniform-threshold reference case.

Joint gate optimization. Using outcome-based Monte Carlo credit assignment, both gates share the trajectory-level group-relative advantage from our correctness-first length reward; we do not split the reward. Each gate uses its own realized actions: the Mode Gate's selected modes, and the Exit Gate's preceding `Continue` decisions and terminal `Stop`. Their clipped GRPO losses are summed without an additional coefficient:

$$\mathcal{L}_{\text{joint}} = \mathcal{L}_{\text{GRPO}}^{\text{mode}} + \mathcal{L}_{\text{GRPO}}^{\text{exit}}. \tag{12}$$

Their trainable parameters are disjoint, while on-policy trajectories and advantages are shared.

4 Experiments

4.1 Experimental Settings

Backbone Models. We evaluate our method on multiple recent reasoning models spanning different families and scales, including Qwen3-8B [55], DeepSeek-R1-Distill-Llama-8B [12], and Qwen3-1.7B [55]. By evaluating across these diverse backbones, we aim to validate that our method generalizes effectively regardless of the underlying architecture or parameter scale.

Evaluation Benchmarks. We adopt a multi-dimensional evaluation protocol covering diverse task domains and difficulty regimes. For foundational mathematical proficiency, we utilize **GSM8K** [10] and **MATH-500** [22], which assess arithmetic and high-school level problem-solving. In the domain of advanced logical reasoning, we incorporate the latest Olympiad-level datasets, **AIME 2024** [24] and **AIME 2025** [58], to test the model's capacity for handling extended reasoning chains on novel problems. Complementing these, the **GPQA Diamond** [35] is employed to evaluate expert-level scientific knowledge and reasoning robustness. Detailed statistics are provided in Appendix I.3.

Table 1: Accuracy (pass@1) (ACC) and total decoding length (LEN) comparison across GSM8K, MATH-500, GPQA-Diamond (GPQA), AIME 2024, and AIME 2025. LEN counts both explicit tokens and latent soft-token steps. Ours achieves the best accuracy–efficiency trade-off. Accuracy gaps are in percentage points; length gaps are relative percentages. Color encoded gaps indicate improvement or degradation relative to the baseline.

METHOD	GSM8K		MATH-500		GPQA		AIME 2024		AIME 2025		AVERAGE	
	ACC↑	LEN↓	ACC↑	LEN↓	ACC↑	LEN↓	ACC↑	LEN↓	ACC↑	LEN↓	ACC↑	LEN↓
Qwen3-8B												
STANDARD COT	95.15	2378	95.80	5571	56.57	9658	73.33	15752	63.33	19559	76.84 (+0.00)	10584 (+0.00%)
SOFT THINKING	95.07	2423	95.20	5754	54.04	12750	68.89	16016	65.56	18997	75.75 (−1.09)	11188 (+5.71%)
DEER	95.38	1045	93.80	3213	55.56	5819	76.67	11369	68.89	11637	78.06 (+1.22)	6617 (−37.48%)
SWIREASONING	95.30	2305	96.60	5467	57.41	8473	75.56	14971	68.89	18330	78.75 (+1.91)	9909 (−6.38%)
OURS	95.68	1189	96.00	3241	59.76	5675	76.67	10378	71.11	11260	**79.84** (+3.00)	**6349** (−40.01%)
Qwen3-1.7B												
STANDARD COT	90.47	2529	90.60	6134	39.39	10759	45.56	17289	30.00	18962	59.20 (+0.00)	11135 (+0.00%)
SOFT THINKING	89.36	2682	90.20	6274	34.34	14427	38.89	17468	33.33	19103	57.22 (−1.98)	11991 (+7.69%)
DEER	89.92	1040	89.33	3158	37.37	8529	43.33	11537	30.00	12110	57.99 (−1.21)	7275 (−34.66%)
SWIREASONING	90.30	2342	90.80	5642	41.50	10092	48.89	17228	36.67	18209	61.63 (+2.43)	10703 (−3.88%)
OURS	90.57	1267	90.87	3348	41.91	7834	48.89	11389	37.78	11770	**62.00** (+2.80)	**7122** (−36.04%)
DeepSeek-R1-Distill-Llama-8B												
STANDARD COT	74.88	1810	80.00	4704	43.77	8823	34.44	16143	26.67	16590	51.95 (+0.00)	9614 (+0.00%)
SOFT THINKING	74.75	1922	78.00	4961	42.93	8978	34.44	17326	24.44	16309	50.91 (−1.04)	9899 (+2.97%)
DEER	75.07	740	80.20	3480	40.40	8324	33.33	9952	27.78	11291	51.36 (−0.60)	6757 (−29.71%)
SWIREASONING	75.79	1650	82.40	4698	45.45	8409	36.67	15897	31.11	14187	54.28 (+2.33)	8968 (−6.72%)
OURS	75.94	816	82.87	3264	46.80	7659	36.67	9205	32.22	10857	**54.90** (+2.95)	**6360** (−33.84%)

We report Accuracy (pass@1) to measure reasoning correctness and token Length which counts both explicit text tokens and latent soft tokens to quantify computational cost. To mitigate the variance inherent in long-chain reasoning, all reported results are averaged over six independent runs.

Implementation Details. We train on DeepScaleR [31] with the backbone LLM frozen, and optimize only the lightweight gates using GRPO with group size $G = 8$ on a single NVIDIA B200 GPU. Our implementation is based on `vLLM` 0.10.2 [28] and `verl` 0.6.0 [40]. We set the length penalty coefficient to $\lambda = 5\text{e}{-4}$. For adaptive termination, we sample the log-survival threshold $\log \gamma \sim U(-20.0, 0.0)$ during training to encourage diverse stopping behaviors. A low LLM sampling temperature ($1\text{e}{-4}$) is used during training to make exploration mainly depend on gate actions. Importantly, this temperature does not affect soft-token computation in Eq. (3), where $p_t = \text{softmax}(W_h h_t)$ is computed from the original logits. For inference, we use a fixed log-survival threshold $\log \gamma = -15.0$ for all datasets, selected as a conservative operating point within the training range rather than tuned per benchmark. The inference decoding temperature is set to 0.7. Further details are provided in Appendix I.

4.2 Main Results

We compare against explicit and latent reasoning baselines in Table 1, as well as training-based methods in Table 2. Specifically, we benchmark our method against representative approaches from the following four paradigms:

(i) Standard CoT. We use standard explicit chain-of-thought decoding as a strong and widely adopted baseline under the same prompting template and decoding budget.

(ii) Latent reasoning. Soft Thinking [64] replaces discrete reasoning tokens with soft tokens computed as probability-weighted mixtures of vocabulary embeddings. We also compare with the recent work SwiReasoning [41], which employs heuristic rules to govern both the switching between latent/explicit modes and the early termination of the reasoning process.

(iii) Inference-time Early Exit. We include DEER [56] as a representative baseline, which shares our setting of keeping LLMs frozen and determines when to exit based on output statistics.

(iv) Training-based Baselines. LoRA-RL is a control baseline that trains LoRA adapters instead of our gates using the same RL pipeline, with a comparable trainable-parameter budget. We also compare with Token Assorted [43] and RL with Length Penalty [1], using matched Qwen3-8B training data and evaluation settings (Table 2).

Accuracy Analysis. Our method achieves the highest average accuracy across all three backbone models, consistently outperforming **Standard CoT** with improvements of +3.00, +2.80, and +2.95 points respectively (Table 1). This confirms that integrating latent dimensions enhances reasoning without compromising symbolic rigor. Crucially, our analysis highlights two key advantages over existing baselines: **(1)** Our approach consistently surpasses purely latent methods like Soft Thinking across all benchmarks. For instance, on the Qwen3-8B backbone, while Soft Thinking suffers a performance drop (-1.09 percentage points vs. Standard CoT) due to lack of precision, our method achieves a robust gain of +3.00 points. This advantage stems from our interleaved reasoning mechanism, which synergistically combines latent and explicit modes. By dynamically anchoring reasoning in discrete symbols when necessary, we effectively mitigate the semantic drift and precision loss inherent in purely continuous reasoning. **(2)** We demonstrate clear advantages over SwiReasoning, which relies on fixed rules for mode switching. This is particularly evident on complex tasks requiring deep generalization, such as AIME 2025, where our method achieves 71.11% accuracy on the Qwen3-8B backbone, surpassing SwiReasoning's 68.89%. This result validates that our method successfully learns a switching policy superior to empirical heuristic designs, enabling more adaptive and optimal navigation of complex solution spaces. Compared with the training-based baselines in Table 2, our method improves accuracy over Token Assorted and RL with Length Penalty by 2.02 and 1.11 percentage points, respectively, while generating shorter sequences and updating only 33.6M gate parameters.

Table 2: **Comparison with training-based baselines on Qwen3-8B.** ACC (pass@1) and LEN (total decoding length) follow Table 1 and are averaged over the five benchmarks. PARAMS denotes updated parameters. Methods use matched backbone, training data, and evaluation settings.

METHOD	PARAMS	ACCURACY ↑	AVG. LENGTH ↓
Standard CoT	–	76.84	10584
LoRA-RL	32.74M	77.13	10232
RL with Length Penalty [1]	8.19B	78.73	6783
Token Assorted [43]	8.20B	77.82	9326
Ours	**33.6M**	**79.84**	**6349**

Efficiency Analysis. Regarding computational cost, our framework improves the accuracy–efficiency trade-off across all evaluated backbones. As shown in Table 1, our method reduces total reasoning length by 33.84% to 40.01% on average while maintaining or improving accuracy. This advantage is particularly clear when compared with efficiency-oriented baselines. For example, on Qwen3-8B, SwiReasoning improves accuracy by 1.91 percentage points but only reduces length by 6.38%, whereas our method achieves a 40.01% length reduction with higher average accuracy (79.84% vs. 78.75%). Compared with the early-exit method DEER, our method achieves comparable or better length reduction while providing larger accuracy gains (+3.00 points vs. +1.22 points on Qwen3-8B). These accuracy and efficiency gains stem from the complementary roles of the two gates: the *Mode Gate* mainly enhances reasoning quality through selective latent-explicit modulation, while the *Exit Gate* mainly improves efficiency by terminating redundant reasoning once sufficient evidence has accumulated.

4.3 Ablation Study

Component contribution analysis. We first disentangle the contributions of the two gates under comparable trainable parameter budgets. As shown in Table 3 (a), the LoRA-RL control brings only marginal improvement over Standard CoT, suggesting that *the gain does not come from merely introducing extra trainable parameters or RL itself*. By contrast, adding only the Mode Gate improves the average accuracy by 2.93, but yields limited length reduction. Adding only the Exit Gate exhibits a complementary effect: it substantially reduces the average length by 38.0%, while maintaining accuracy close to Standard CoT. These results indicate that the two gates are **both essential** and specialize in different dimensions: the Mode

Table 3: **Ablation study on Qwen3-8B. (a)** decomposes the two gate contributions under matched trainable parameter budgets, while **(b)** validates key design choices. Gaps in (a) are computed relative to Standard CoT, while gaps in (b) are computed relative to the full model. Darker green indicates better values for each metric.

	METHOD	PARAMS	ACCURACY ↑	AVG. LENGTH ↓
(a)	Standard CoT	–	76.84	10,584
	+ LoRA-RL	32.74M	77.13 (+0.29)	10,232 (−3.3%)
	+ Mode Gate	33.58M	79.77 (+2.93)	10,066 (−4.9%)
	+ Exit Gate	33.62M	76.98 (+0.14)	6,562 (−38.0%)
	+ Both Gates (Ours)	**33.60M**	**79.84** (+3.00)	**6,349** (−40.0%)
(b)	Force Latent	33.60M	75.75 (−4.09)	7,693 (+21.2%)
	w/o Survival Analysis	33.60M	66.73 (−13.11)	5,072 (−20.1%)

Gate mainly improves reasoning quality through selective latent-explicit modulation, whereas the Exit Gate mainly improves efficiency by removing redundant reasoning steps. Only by combining both gates can the model achieve simultaneous gains in accuracy and efficiency. Further Exit Gate comparisons against DEER, with and without the Mode Gate, are provided in Appendix J. To better understand the Mode Gate, we analyze the latent-explicit ratio in Appendix E.

Design choice validation. We further validate two key design choices in Table 3 (b). First, forcing the model to always use latent reasoning significantly degrades accuracy compared with our full model and also increases the average length, indicating that purely latent reasoning is unstable and may accumulate uncertainty. This supports the necessity of selective latent-explicit modulation rather than replacing explicit reasoning with latent steps entirely. Second, removing the survival accumulation and using a naive exit rule leads to a severe accuracy collapse. Although this variant produces shorter outputs, the reduction mainly comes from premature termination rather than reliable reasoning compression. This confirms that survival-based accumulation is crucial for filtering noisy local stopping signals and ensuring that early termination occurs only after sufficient reasoning evidence has been accumulated.

Wall-clock inference time. To verify whether the token reduction leads to actual runtime acceleration, we further compare the inference wall-clock time on Qwen3-8B. For each benchmark, we report the average per-sample inference time under the same setting. As shown in Table 4, our method consistently accelerates inference across all datasets, achieving up to $1.69\times$ speedup over standard CoT. This indicates that our efficiency gain is not merely reflected by fewer generated tokens, but also translates into practical inference acceleration.

Table 4: Wall-clock time comparison.

Benchmark	CoT (s)	Ours (s)	Speedup ↑
GSM8K	3.64	2.16	1.69×
MATH-500	9.37	7.15	1.31×
GPQA	11.59	8.12	1.43×
AIME 2024	17.67	11.83	1.49×
AIME 2025	20.21	12.70	1.59×

Sensitivity and Effectiveness of Termination. We analyze the robustness of our termination mechanism by varying the negative log-survival threshold $-\log\gamma$ on GSM8K, as shown in Figure 3. The default value $-\log\gamma = 15$ is not tuned on GSM8K, but is used as a universal setting across all main experiments. As the threshold becomes more conservative, accuracy quickly stabilizes while token length increases smoothly, suggesting that our survival-based termination is not sensitive to a narrow threshold choice. At the default operating point, our method reduces token consumption by 47% with only a negligible accuracy drop of about 0.2% compared with the no-early-exit baseline. This indicates that the accumulated stopping probability can reliably identify early convergence points and remove redundant reasoning steps without sacrificing solution quality. We further provide cross-dataset threshold sweeps in Appendix D, which exhibit similar trends and show that our method is robust across a relative broad range of threshold choices.

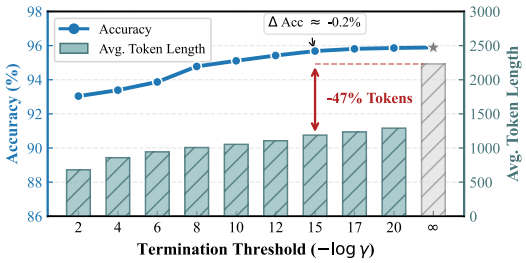


Figure 3: Accuracy-efficiency trade-off across varying negative log-survival thresholds $-\log\gamma$ on GSM8K.

Qualitative Analysis of Survival Dynamics. Figure 4 visualizes a GSM8K reasoning trajectory together with the cumulative log-survival score $\mathcal{L}_t = \log\mathcal{S}_t$. The case study provides two key observations. **(1) Adaptive evidence accumulation.** The log-survival score changes according to the reasoning process rather than decreasing at a fixed rate. It remains relatively high when the answer is not fully verified, but drops rapidly once subsequent steps consistently support the same result, eventually reaching $\mathcal{L}_t \leq \log\gamma$ and triggering final-answer generation. **(2) Context-aware stopping.** The exit decision is not merely based on next-token keywords, such as "wait" or explicit conclusion markers. For instance, around Step 8, the next output starts with "Total CDs", which is not a common stopping cue; nevertheless, the gate predicts a high hazard score. This suggests that the gate relies on the preceding reasoning trajectory, where a plausible answer has already been derived and partially verified, rather than simple keyword detection.

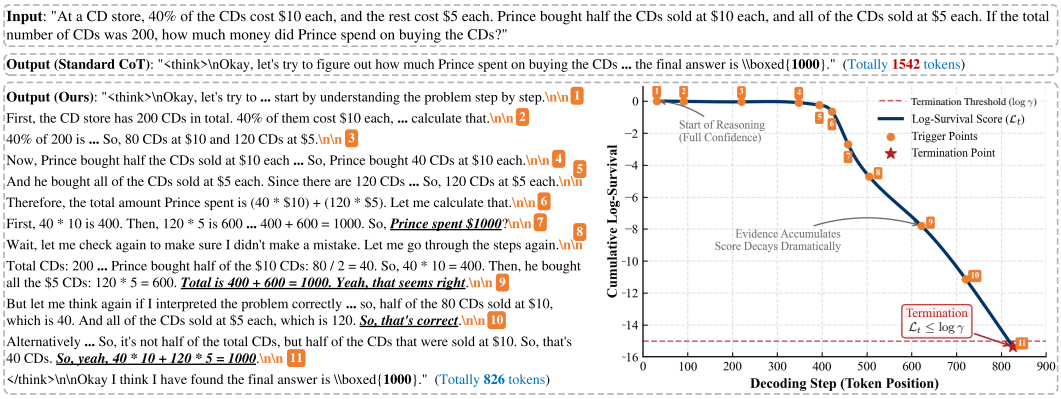


Figure 4: **Case study of the log-survival mechanism.** The left panel shows the segmented reasoning trace, where the orange markers correspond to the trigger positions on the log-survival curve (right).

5 Conclusion

We present a lightweight adaptive reasoning framework for frozen LLMs that jointly controls how to reason and when to stop. The proposed Mode Gate selectively interleaves latent and explicit reasoning, while the Survival-Based Exit Gate accumulates step-wise hazard estimates to trigger robust early termination. Across multiple backbones and reasoning benchmarks, our method improves accuracy while substantially reducing reasoning length and time cost. These results suggest that adaptive inference-time control is a practical way to regulate the computation of frozen LLMs without modifying the backbone. Future work will explore integrating this lightweight control mechanism with LLM fine-tuning for stronger latent reasoning and termination policies.

Acknowledgments and Disclosure of Funding

This research / project is supported by the National Research Foundation, Singapore, and Cyber Security Agency of Singapore under its National Cybersecurity R&D Programme and CyberSG R&D Cyber Research Programme Office. Any opinions, findings and conclusions or recommendations expressed in these materials are those of the author(s) and do not reflect the views of National Research Foundation, Singapore, Cyber Security Agency of Singapore as well as CyberSG R&D Programme Office, Singapore.

References

[1] Daman Arora and Andrea Zanette. Training language models to reason efficiently. *arXiv preprint arXiv:2502.04463*, 2025.

[2] 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. *arXiv preprint arXiv:2108.07732*, 2021.

[3] Sangmin Bae, Yujin Kim, Reza Bayat, Sungnyun Kim, Jiyoun Ha, Tal Schuster, Adam Fisch, Hrayr Harutyunyan, Ziwei Ji, Aaron Courville, and Se-Young Yun. Mixture-of-recursions: Learning dynamic recursive depths for adaptive token-level computation. *arXiv preprint arXiv:2507.10524*, 2025.

[4] Natasha Butt, Ariel Kwiatkowski, Ismail Labiad, Julia Kempe, and Yann Ollivier. Soft tokens, hard truths. *arXiv preprint arXiv:2509.19170*, 2025.

[5] 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.

[6] Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *arXiv preprint arXiv:2503.09567*, 2025.

[7] Xinghao Chen, Anhao Zhao, Heming Xia, Xuan Lu, Hanlin Wang, Yanjun Chen, Wei Zhang, Jian Wang, Wenjie Li, and Xiaoyu Shen. Reasoning beyond language: A comprehensive survey on latent chain-of-thought reasoning. *arXiv preprint arXiv:2505.16782*, 2025.

[8] Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, et al. Do not think that much for 2+ 3=? on the overthinking of o1-like llms. *arXiv preprint arXiv:2412.21187*, 2024.

[9] Yilong Chen, Junyuan Shang, Zhenyu Zhang, Yanxi Xie, Jiawei Sheng, Tingwen Liu, Shuohuan Wang, Yu Sun, Hua Wu, and Haifeng Wang. Inner thinking transformer: Leveraging dynamic depth scaling to foster adaptive internal thinking. *arXiv preprint arXiv:2502.13842*, 2025.

[10] 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.

[11] Muzhi Dai, Chenxu Yang, and Qingyi Si. S-grpo: Early exit via reinforcement learning in reasoning models. *arXiv preprint arXiv:2505.07686*, 2025.

[12] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. URL https://arxiv.org/abs/2501.12948v1.

[13] Jingcheng Deng, Liang Pang, Zihao Wei, Shichen Xu, Zenghao Duan, Kun Xu, Yang Song, Huawei Shen, and Xueqi Cheng. Latent reasoning in llms as a vocabulary-space superposition. *arXiv preprint arXiv:2510.15522*, 2025. URL https://arxiv.org/abs/2510.15522v1.

[14] Yuntian Deng, Yejin Choi, and Stuart Shieber. From explicit cot to implicit cot: Learning to internalize cot step by step. *arXiv preprint arXiv:2405.14838*, 2024.

[15] Wenkai Fang, Shunyu Liu, Yang Zhou, Kongcheng Zhang, Tongya Zheng, Kaixuan Chen, Mingli Song, and Dacheng Tao. Serl: Self-play reinforcement learning for large language models with limited data. In *Advances in Neural Information Processing Systems*, 2025.

[16] Sicheng Feng, Gongfan Fang, Xinyin Ma, and Xinchao Wang. Efficient reasoning models: A survey. *arXiv preprint arXiv:2504.10903*, 2025.

[17] Yichao Fu, Junda Chen, Siqi Zhu, Zheyu Fu, Zhongdongming Dai, Yonghao Zhuang, Yian Ma, Aurick Qiao, Tajana Rosing, Ion Stoica, et al. Efficiently scaling llm reasoning with certaindex. *arXiv preprint arXiv:2412.20993*, 2024.

[18] Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. *arXiv preprint arXiv:2502.05171*, 2025.

[19] Jinyi Han, Ying Huang, Ying Liao, Zishang Jiang, Xikun Lu, Haiquan Zhao, Xinyi Wang, Guanghao Zhou, Sihang Jiang, Jiaqing Liang, Weikang Zhou, Zeye Sun, Fei Yu, and Yanghua Xiao. Your models have thought enough: Training large reasoning models to stop overthinking. *arXiv preprint arXiv:2509.23392*, 2025.

[20] Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. Token-budget-aware llm reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 24842–24855, 2025.

[21] Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024.

[22] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

[23] Zheng Yi Ho, Siyuan Liang, Sen Zhang, Yibing Zhan, and Dacheng Tao. NoVo: Norm voting off hallucinations with attention heads in large language models. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/db6461eaf0eaeaad1d9c4a70e4818cbd-Abstract-Conference.html.

[24] HuggingFaceH4. AIME 2024 (american invitational mathematics examination 2024). Hugging Face dataset, 2024. URL https://huggingface.co/datasets/HuggingFaceH4/aime_2024.

[25] Shuyang Jiang, Yusheng Liao, Ya Zhang, Yanfeng Wang, and Yu Wang. Overthinking reduction with decoupled rewards and curriculum data scheduling. *arXiv preprint arXiv:2509.25827*, 2025.

[26] Yeskendir Koishekenov, Aldo Lipani, and Nicola Cancedda. Encode, think, decode: Scaling test-time reasoning with recursive latent thoughts. *arXiv preprint arXiv:2510.07358*, 2025.

[27] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.

[28] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

[29] Zhong-Zhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, et al. From system 1 to system 2: A survey of reasoning large language models. *arXiv preprint arXiv:2502.17419*, 2025.

[30] 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*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/aca97732e30bcf1303bc22ac3924fd16-Abstract-Conference.html.

[31] Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl, 2025. URL https://huggingface.co/agentica-org/DeepScaleR-1.5B-Preview. Notion Blog.

[32] Xinyin Ma, Guangnian Wan, Runpeng Yu, Gongfan Fang, and Xinchao Wang. Cot-valve: Length-compressible chain-of-thought tuning. *arXiv preprint arXiv:2502.09601*, 2025.

[33] Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235, 2024.

[34] Ziqing Qiao, Yongheng Deng, Jiali Zeng, Dong Wang, Lai Wei, Guanbo Wang, Fandong Meng, Jie Zhou, Ju Ren, and Yaoxue Zhang. ConCISE: Confidence-guided compression in step-by-step efficient reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 8021–8040, 2025. doi: 10.18653/v1/2025.emnlp-main.405. URL https://aclanthology.org/2025.emnlp-main.405/.

[35] 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.

[36] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[37] Shannon Zejiang Shen, Rulin Shao, Chenyu Wang, Songlin Yang, Vincent-Pierre Berges, Gargi Ghosh, Pang Wei Koh, Luke Zettlemoyer, Yoon Kim, Jason E Weston, et al. Hybridcot: Interleaving latent and text chain-of-thought for efficient reasoning. In *NeurIPS 2025 Workshop on Efficient Reasoning*, 2025. URL https://openreview.net/forum?id=NRGRrHmq1H.

[38] Yi Shen, Jian Zhang, Jieyun Huang, Shuming Shi, Wenjing Zhang, Jiangze Yan, Ning Wang, Kai Wang, Zhaoxiang Liu, and Shiguo Lian. Dast: Difficulty-adaptive slow-thinking for large reasoning models. *arXiv preprint arXiv:2503.04472*, 2025.

[39] Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. Codi: Compressing chain-of-thought into continuous space via self-distillation. *arXiv preprint arXiv:2502.21074*, 2025.

[40] Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

[41] Dachuan Shi, Abedelkadir Asi, Keying Li, Xiangchi Yuan, Leyan Pan, Wenke Lee, and Wen Xiao. Swireasoning: Switch-thinking in latent and explicit for pareto-superior reasoning llms. *arXiv preprint arXiv:2510.05069*, 2025.

[42] Mingyang Song and Mao Zheng. Walk before you run! concise llm reasoning via reinforcement learning. *arXiv preprint arXiv:2505.21178*, 2025.

[43] Dijia Su, Hanlin Zhu, Yingchen Xu, Jiantao Jiao, Yuandong Tian, and Qinqing Zheng. Token assorted: Mixing latent and text tokens for improved language model reasoning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 57144–57163. PMLR, 2025. URL https://proceedings.mlr.press/v267/su25g.html.

[44] Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, et al. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*, 2025.

[45] Jihoon Tack, Jack Lanchantin, Jane Yu, Andrew Cohen, Ilia Kulikov, Janice Lan, Shibo Hao, Yuandong Tian, Jason Weston, and Xian Li. Llm pretraining with continuous concepts. *arXiv preprint arXiv:2502.08524*, 2025.

[46] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158. Association for Computational Linguistics, 2019. doi: 10.18653/v1/N19-1421. URL https://aclanthology.org/N19-1421/.

[47] Wenhui Tan, Jiaze Li, Jianzhong Ju, Zhenbo Luo, Jian Luan, and Ruihua Song. Think silently, think fast: Dynamic latent compression of llm reasoning chains. *arXiv preprint arXiv:2505.16552*, 2025. URL https://arxiv.org/abs/2505.16552v1.

[48] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023.

[49] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[50] Xilin Wei, Xiaoran Liu, Yuhang Zang, Xiaoyi Dong, Yuhang Cao, Jiaqi Wang, Xipeng Qiu, and Dahua Lin. Sim-cot: Supervised implicit chain-of-thought. *arXiv preprint arXiv:2509.20317*, 2025.

[51] Junhong Wu, Jinliang Lu, Zixuan Ren, Gangqiang Hu, Zhi Wu, Dai Dai, and Hua Wu. Llms are single-threaded reasoners: Demystifying the working mechanism of soft thinking. *arXiv preprint arXiv:2508.03440*, 2025.

[52] Yuxi Xie, Kenji Kawaguchi, Yiran Zhao, James Xu Zhao, Min-Yen Kan, Junxian He, and Michael Xie. Self-evaluation guided beam search for reasoning. *Advances in Neural Information Processing Systems*, 36:41618–41650, 2023.

[53] Xin Xu, Tong Yu, Xiang Chen, Haoliang Wang, Julian McAuley, and Saayan Mitra. Thinkrouter: Efficient reasoning via routing thinking between latent and discrete spaces. *arXiv preprint arXiv:2602.11683*, 2026.

[54] Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. Softcot: Soft chain-of-thought for efficient reasoning with llms. *arXiv preprint arXiv:2502.12134*, 2025.

[55] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[56] Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Minghui Chen, Zheng Lin, and Weiping Wang. Dynamic early exit in reasoning models. *arXiv preprint arXiv:2504.15895*, 2025.

[57] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

[58] Yentinglin. AIME 2025 (american invitational mathematics examination 2025). Hugging Face dataset, 2025. URL https://huggingface.co/datasets/yentinglin/aime_2025.

[59] Zhenrui Yue, Bowen Jin, Huimin Zeng, Honglei Zhuang, Zhen Qin, Jinsung Yoon, Lanyu Shang, Jiawei Han, and Dong Wang. Hybrid latent reasoning via reinforcement learning. *arXiv preprint arXiv:2505.18454*, 2025.

[60] Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488, 2022.

[61] Boyi Zeng, Shixiang Song, Siyuan Huang, Yixuan Wang, He Li, Ziwei He, Xinbing Wang, Zhiyu Li, and Zhouhan Lin. Ponderlm: Pretraining language models to ponder in continuous space. *arXiv preprint arXiv:2505.20674*, 2025.

[62] Kongcheng Zhang, Qi Yao, Shunyu Liu, Yingjie Wang, Baisheng Lai, Jieping Ye, Mingli Song, and Dacheng Tao. Consistent paths lead to truth: Self-rewarding reinforcement learning for llm reasoning. In *Advances in Neural Information Processing Systems*, 2025.

[63] Kongcheng Zhang, Qi Yao, Baisheng Lai, Jiaxing Huang, Wenkai Fang, Dacheng Tao, Mingli Song, and Shunyu Liu. Incentivizing llm reasoning via reinforcement learning with functional monte carlo tree search. In *International Conference on Learning Representations*, 2026.

[64] Zhen Zhang, Xuehai He, Weixiang Yan, Ao Shen, Chenyang Zhao, Shuohang Wang, Yelong Shen, and Xin Eric Wang. Soft thinking: Unlocking the reasoning potential of llms in continuous concept space. *arXiv preprint arXiv:2505.15778*, 2025.

[65] Zhi Zheng and Wee Sun Lee. Soft-grpo: Surpassing discrete-token llm reinforcement learning via gumbel-reparameterized soft-thinking policy optimization. *arXiv preprint arXiv:2511.06411*, 2025. URL https://arxiv.org/abs/2511.06411v1.

[66] Yang Zhou, Sunzhu Li, Shunyu Liu, Wenkai Fang, Kongcheng Zhang, Jiale Zhao, Jingwen Yang, Yihe Zhou, Jianwei Lv, Tongya Zheng, Hengtong Lu, Chen Wei, Xie Yan, and Mingli Song. Breaking the exploration bottleneck: Rubric-scaffolded reinforcement learning for open-ended LLM reasoning. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 165222–165251. PMLR, 2026. URL https://proceedings.mlr.press/v306/zhou26y.html.

[67] Hanlin Zhu, Shibo Hao, Zhiting Hu, Jiantao Jiao, Stuart Russell, and Yuandong Tian. Reasoning by superposition: A theoretical perspective on chain of continuous thought. *arXiv preprint arXiv:2505.12514*, 2025.

[68] Rui-Jie Zhu, Tianhao Peng, Tianhao Cheng, Xingwei Qu, Jinfa Huang, Dawei Zhu, Hao Wang, Kaiwen Xue, Xuanliang Zhang, Yong Shan, et al. A survey on latent reasoning. *arXiv preprint arXiv:2507.06203*, 2025.

[69] Rui-Jie Zhu, Zixuan Wang, Kai Hua, Tianyu Zhang, Ziniu Li, Haoran Que, Boyi Wei, Zixin Wen, Fan Yin, He Xing, Lu Li, Jiajun Shi, Kaijing Ma, Shanda Li, Taylor Kergan, Andrew Smith, Xingwei Qu, Mude Hui, Bohong Wu, Qiyang Min, Hongzhi Huang, Xun Zhou, Wei Ye, Jiaheng Liu, Jian Yang, Yunfeng Shi, Chenghua Lin, Enduo Zhao, Tianle Cai, Ge Zhang, Wenhao Huang, Yoshua Bengio, and Jason Eshraghian. Scaling latent reasoning via looped language models. *arXiv preprint arXiv:2510.25741*, 2025.

[70] Yufan Zhuang, Liyuan Liu, Chandan Singh, Jingbo Shang, and Jianfeng Gao. Mixture of inputs: Text generation beyond discrete token sampling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

A Theoretical Analysis of Dynamic Thresholding

Section 3.5 samples a trajectory-level survival threshold with $\log \gamma \sim U(\alpha, \beta)$, where $(\alpha, \beta) = (-20, 0)$ in our experiments. We show that sampling such a threshold corresponds to a sequential stochastic stopping policy, providing a probabilistic interpretation of dynamic thresholding.

A.1 Stopping Distribution for a General Survival Threshold

Let t index eligible decision boundaries and let T denote the first threshold-induced stopping boundary. Condition on a potential continuation trajectory, fixing all non-threshold randomness. Assume that γ is sampled independently and affects this trajectory only through stopping, not through the continuation dynamics. The resulting hazard sequence $\rho_t \in [0, 1)$ defines

$$\mathcal{S}_0 = 1, \qquad \mathcal{S}_t = \prod_{k=1}^{t} (1 - \rho_k), \qquad \mathcal{L}_t = \log \mathcal{S}_t.$$

All probabilities below are conditional on this continuation trajectory. Let $H(s) = P(\gamma \le s)$ be the continuous CDF of a survival threshold supported in $(0, 1)$. Since $\mathcal{S}_t$ is non-increasing,

$$T = t \quad \Longleftrightarrow \quad \mathcal{S}_t \le \gamma < \mathcal{S}_{t-1}. \tag{13}$$

The equality cases have probability zero. Therefore,

$$P_H(T = t) = H(\mathcal{S}_{t-1}) - H(\mathcal{S}_t), \qquad P_H(T > t) = H(\mathcal{S}_t). \tag{14}$$

Accordingly, $1 - H(\mathcal{S}_t)$ describes the stopping-time CDF induced by the threshold distribution.

A.2 Equivalence for a Uniform Survival Threshold

If $\gamma \sim U(0, 1)$, then $H(s) = s$ for $s \in [0, 1]$, and Eq. (14) becomes

$$P_U(T = t) = \mathcal{S}_{t-1} - \mathcal{S}_t = \mathcal{S}_{t-1}\rho_t = \left[\prod_{k<t} (1 - \rho_k)\right] \rho_t. \tag{15}$$

This matches the stopping-time distribution of sequential Bernoulli decisions with stopping probability ρ_t at each boundary.

A.3 Distribution Induced by Log-Uniform Thresholds

For $\log \gamma \sim U(\alpha, \beta)$ with $\alpha < \beta \le 0$, the threshold CDF is

$$H_{\alpha,\beta}(s) = \begin{cases} 0, & s \le e^{\alpha}, \\ \dfrac{\log s - \alpha}{\beta - \alpha}, & e^{\alpha} < s < e^{\beta}, \\ 1, & s \ge e^{\beta}. \end{cases} \tag{16}$$

Equivalently, the probability mass assigned to a stopping boundary is the overlap of its log-survival interval with the sampled threshold range:

$$P_{\log U}(T = t) = \frac{\left[\min(\mathcal{L}_{t-1}, \beta) - \max(\mathcal{L}_t, \alpha)\right]_+}{\beta - \alpha}, \qquad [z]_+ = \max(z, 0). \tag{17}$$

When both log-survival endpoints lie within $[\alpha, \beta]$, this simplifies to $-\log(1 - \rho_t)/(\beta - \alpha)$. Thus log-uniform thresholding allocates stopping mass according to increments in log-survival, while preserving the threshold-crossing rule.

The equivalent conditional stopping probability for a general threshold CDF is

$$q_t^H = P_H(T = t \mid T \ge t) = 1 - \frac{H(\mathcal{S}_t)}{H(\mathcal{S}_{t-1})}, \qquad H(\mathcal{S}_{t-1}) > 0. \tag{18}$$

For a uniform threshold, $q_t^H = \rho_t$; for log-uniform thresholds, the same formula gives the calibrated hazard by taking $H = H_{\alpha,\beta}$.

Equivalence with sequential stopping. Consider a sequential stochastic policy that, at each eligible boundary t, stops with probability q_t^H from Eq. (18), conditional on having continued to that boundary. Its stopping-time probability telescopes:

$$\begin{aligned} P_{\text{seq}}^H(T=t) &= \left[\prod_{k<t}(1-q_k^H)\right] q_t^H \\ &= H(\mathcal{S}_{t-1}) - H(\mathcal{S}_t) = P_H(T=t). \end{aligned} \tag{19}$$

Both mechanisms also assign probability $H(\mathcal{S}_K)$ to continuation beyond a finite boundary budget K. When the sampled threshold influences generation through the stopping decision, the correspondence extends to the generated trajectories and their expected rewards:

$$\mathbb{E}_{\gamma\sim H}[R(\tau_T)] = \mathbb{E}_{\text{seq}(q^H)}[R(\tau_T)]\,. \tag{20}$$

Therefore, **dynamic thresholding admits an equivalent sequential-stopping interpretation**: sampling one threshold for the trajectory corresponds to sequential decisions governed by q_t^H. The uniform-threshold case gives $q_t^H = \rho_t$, while our log-uniform training scheme uses the corresponding hazards $q_t^{H_{\alpha,\beta}}$.

B Theoretical Analysis on Decision Robustness

In this section, we provide a theoretical justification for replacing standard step-wise binary classification with our cumulative survival mechanism. We follow the main-paper notation: the exit head predicts a hazard rate ρ_t, the survival function is $S_t = \prod_{k=1}^{t}(1-\rho_k)$, and the cumulative stopping probability (failure CDF) is $F_t = 1 - S_t$, with inference terminating when $F_t \geq 1-\gamma$, equivalently $S_t \leq \gamma$ or $\log S_t \leq \log\gamma$, where γ is the survival threshold.

Consider the true (latent) "reasoning completeness" at step t as a monotonic variable $F_t^\star \in [0,1]$. A standard step-wise classifier directly predicts a noisy completeness score $\hat{F}_t^{\text{step}}$ from the hidden state h_t:

$$\hat{F}_t^{\text{step}} = F_t^\star + \epsilon_t, \quad \text{where } \epsilon_t \sim \mathcal{N}(0,\sigma^2), \tag{21}$$

where ϵ_t represents local aleatoric noise.

Instability of Step-wise Decision. In standard approaches, termination triggers when $\hat{F}_t^{\text{step}} \geq \eta$, where $\eta \in (0,1)$ is a separate threshold on the classifier output, not a survival threshold. Since ϵ_t is independent across steps, a single large positive noise spike (i.e., $\epsilon_t \gg 0$) can cause a **premature exit** even when the reasoning is still incomplete ($F_t^\star \ll \eta$). Thus, the false-trigger probability at each step is directly governed by the raw variance σ^2.

Robustness of Cumulative Survival. Our survival mechanism makes decisions based on the cumulative stopping probability $F_t = 1 - S_t$, which is structurally monotone ($F_t \geq F_{t-1}$ by construction). In the low-hazard regime (as commonly observed at structural boundaries), we can use the first-order approximation

$$F_t \;=\; 1 - \prod_{k=1}^{t}(1-\rho_k) \;\approx\; \sum_{k=1}^{t}\rho_k. \tag{22}$$

If we view each per-step evidence increment as $\rho_k \approx \Delta F_k^\star + \epsilon_k$ with $\Delta F_k^\star := F_k^\star - F_{k-1}^\star$, then

$$F_t \;\approx\; \sum_{k=1}^{t}(\Delta F_k^\star + \epsilon_k) \;=\; F_t^\star + \sum_{k=1}^{t}\epsilon_k. \tag{23}$$

Crucially, because F_t cannot decrease, the model cannot "regret" previously accumulated completeness due to a localized negative fluctuation. Therefore, termination is driven by **consistent accumulation over time** rather than isolated step-wise spikes, yielding a more stable stopping behavior under local noise.

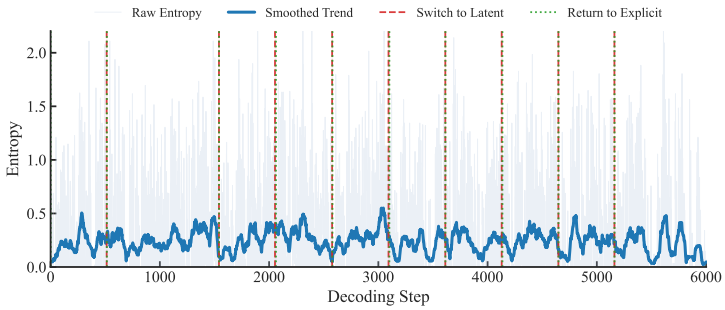


Figure 5: **Entropy dynamics of SwiReasoning** [41]. The curve tracks uncertainty during decoding, with vertical lines marking mode switches. The quasi-periodic switching pattern reveals that mode transitions are largely dictated by the fixed context window size (e.g., every 512 tokens) rather than semantic closure. This reliance on a "switch count" limit prevents true adaptive early termination.

C Comparison of Termination Mechanisms

To highlight the superior adaptability of our learned controller, we contrast our termination dynamics with **SwiReasoning** [41], a representative heuristic baseline.

Limitations of Heuristic Switching (SwiReasoning). Figure 5 visualizes the entropy dynamics of SwiReasoning during the decoding process. Although it aims to switch modes based on uncertainty (entropy), the trajectory reveals a critical limitation: the switching behavior is heavily constrained by the preset context window (e.g., 512 tokens). As observed in Figure 5, the transition points (vertical dashed lines) often occur at regular intervals corresponding to this enforced window size, rather than at moments of true semantic necessity. Consequently, its "early termination" is effectively a rigid truncation governed by a hyperparameter: the model terminates after a fixed number of switches (N_{max_switch}) rather than when the problem is actually solved. This results in a generation length that approximates $Length \approx N_{max_switch} \times L_{window}$, rendering the mechanism a parameter-controlled heuristic rather than a genuinely adaptive stopping policy.

Adaptivity via Log-Survival (Ours). In contrast, our method decouples termination from fixed token counts or predefined switching windows. As shown in Figure 4, the cumulative log-survival score $\mathcal{L}_t = \log \mathcal{S}_t$ changes according to the reasoning trajectory: it remains relatively stable during intermediate exploration and verification, and decreases more rapidly once consistent evidence for the answer has accumulated. The model terminates when $\mathcal{L}_t \leq \log \gamma$, indicating sufficient stopping evidence. This shows that survival-based termination adapts to the evolving reasoning state rather than relying on fixed-length truncation or local heuristic signals.

Table 5: **Cross-dataset threshold sensitivity on Qwen3-8B.** Each entry reports accuracy with average length in parentheses. Across benchmarks, accuracy remains stable once the threshold becomes sufficiently conservative, while reasoning length changes smoothly.

$-\log \gamma$	GSM8K	MATH-500	GPQA	AIME 2024	AIME 2025
8	94.78 (1007)	95.40 (3056)	57.16 (5143)	73.33 (9742)	68.34 (10180)
12	95.42 (1106)	95.73 (3138)	58.59 (5463)	73.33 (10037)	70.56 (10713)
15	95.68 (1189)	96.00 (3241)	59.76 (5675)	76.67 (10378)	71.11 (11260)
20	95.86 (1291)	96.07 (3459)	59.93 (6138)	76.67 (11049)	71.11 (11847)
∞	95.89 (2231)	96.20 (5327)	60.52 (10284)	76.11 (15825)	71.11 (17734)

D Additional Analysis on Threshold Sensitivity

To further examine the robustness of the survival-based termination mechanism, we conduct threshold sweeps across all evaluation benchmarks. Table 5 reports the accuracy and average reasoning length

Table 6: **Latent-token ratio across benchmarks on Qwen3-8B.** The latent ratio increases on harder and longer-reasoning benchmarks, suggesting that the learned controller adapts its mode decisions to task complexity rather than following a fixed switching pattern.

Benchmark	Latent Ratio	Avg. Length
GSM8K	10.9%	1,189
MATH-500	11.8%	3,241
GPQA	14.5%	5,675
AIME 2024	17.5%	10,378
AIME 2025	19.3%	11,260

Table 7: **Latent-token ratio for correct and incorrect samples.** Incorrect samples show a much higher latent ratio, suggesting that excessive latent usage may hurt reasoning stability.

Sample Type	Avg. Latent Ratio
Correct	10.5%
Incorrect	26.7%

under different values of $-\log\gamma$. Here, $-\log\gamma = \infty$ denotes disabling the early-exit mechanism, i.e., the model performs full reasoning without survival-based termination.

As shown in Table 5, the proposed termination mechanism exhibits consistent behavior across datasets. When the threshold is too aggressive, the model may terminate earlier and slightly sacrifice accuracy. As the threshold becomes more conservative, accuracy quickly stabilizes, while the average length increases gradually. Notably, fully disabling early exit leads to substantially longer reasoning traces, but provides only marginal accuracy gains. These results suggest that our survival-based termination is not sensitive to a specific threshold value and remains robust across a broad range of threshold choices.

E Analysis of Learned Mode Modulation

To better understand the behavior of the learned Mode Gate, we analyze the latent-token ratio during inference on Qwen3-8B. The latent ratio is defined as the percentage of reasoning steps in which the controller selects the Latent Mode. Table 6 reports the latent ratio across different benchmarks.

As shown in Table 6, the controller uses more latent steps on more challenging benchmarks. For instance, the latent ratio increases from 10.9% on GSM8K to 19.3% on AIME 2025. This indicates that the Mode Gate learns a task-adaptive modulation strategy: easier problems require fewer latent steps, while harder problems benefit from more compact latent computation.

We further compare the latent ratio between correct and incorrect samples in Table 7. Interestingly, incorrect samples exhibit a substantially higher latent ratio than correct samples. This suggests that latent reasoning is beneficial when used selectively, but excessive latent usage may increase instability or semantic drift. Therefore, the role of the Mode Gate is not to replace explicit reasoning with latent computation entirely, but to deploy latent steps sparsely while periodically anchoring the reasoning trajectory with explicit tokens.

Matched-ratio random mode selection. To isolate the effect of learning where latent tokens are useful, we compare the learned policy with random mode selection at the same latent-token ratio for each benchmark. Table 8 shows that the learned policy improves both accuracy and average decoding length.

Table 8: Learned versus random mode selection on Qwen3-8B. The random baseline matches the latent-token ratio per benchmark; results are averaged over the five main benchmarks.

Mode selection	Accuracy (%) ↑	Avg. Length ↓
Random (matched latent ratio)	77.29	6,841
Learned (ours)	**79.84**	**6,349**

F Variance Across Independent Runs

To quantify the variance of long-form reasoning evaluation, we report the standard deviation of our method on Qwen3-8B across six independent runs. Table 9 summarizes the mean accuracy and standard deviation on each benchmark.

As shown in Table 9, the variance differs across benchmarks. On relatively stable mathematical benchmarks such as GSM8K and MATH-500, the standard deviations are small. In contrast, GPQA and AIME exhibit larger variance, which is expected because these benchmarks involve harder problems, longer reasoning trajectories, and fewer evaluation examples. In particular, small changes in the number of correctly solved AIME problems can lead to noticeable changes in pass@1 accuracy. Overall, the standard deviations remain consistent with the difficulty and scale of each benchmark, and the main trends reported in the paper are stable across independent runs.

G Generalization Across Model Scales

To further examine whether the proposed controller generalizes across different model sizes, we evaluate it on Qwen3 models with 1.7B, 8B, and 14B parameters. Table 10 reports the average accuracy and reasoning length across the evaluated benchmarks. Across all three scales, our method consistently improves accuracy over Standard CoT while substantially reducing reasoning length. This suggests that the proposed adaptive control mechanism is not specific to a single model size, but provides consistent accuracy–efficiency improvements across Qwen3 backbones.

The results show a consistent pattern across model scales. Pure soft-token reasoning tends to reduce accuracy and even increase reasoning length, indicating that directly replacing explicit reasoning with latent steps is unstable. In contrast, our adaptive controller selectively uses latent computation while applying survival-based termination, leading to consistent accuracy gains of 2.43–3.00 points and length reductions of 31.22%–40.01% across Qwen3 backbones.

H Limitations and Broader Impact

Our framework has several limitations. First, although the backbone LLM remains frozen, the controller still requires RL training and customized inference support for soft-token decoding and survival-state tracking. Thus, our method is not a purely plug-and-play heuristic, but rather provides an alternative path to improve reasoning efficiency beyond directly fine-tuning the LLM itself. It may also be complementary to fine-tuning-based latent reasoning methods: a stronger latent-capable LLM could potentially be further enhanced by our adaptive controller. Exploring such combinations between backbone-level latent reasoning training and lightweight inference-time control is an important direction for future work. Scaling RL-based training to substantially larger models remains computationally expensive, which is a common challenge for RL-based reasoning methods. Finally, the Mode Gate primarily improves accuracy, while its efficiency gain is less pronounced than that of the Exit Gate. This reflects the complementary roles of the two gates rather than a conflict: latent modulation improves reasoning quality, whereas survival-based termination removes redundant computation.

By reducing redundant reasoning computation, our method can lower inference cost and improve the accessibility of reasoning-capable LLMs. However, cheaper reasoning may also lower the barrier for misuse, and the proposed controller inherits the biases and safety risks of the frozen backbone, calling for deployment with appropriate safety alignment and monitoring.

Table 9: **Accuracy variance of our method on Qwen3-8B.** Results are averaged over six runs.

Benchmark	Accuracy (%)	Std.
GSM8K	95.68	±0.41
MATH-500	96.00	±0.97
GPQA	59.76	±2.23
AIME 2024	76.67	±4.71
AIME 2025	71.11	±4.04

Table 10: **Results across Qwen3 model scales.** Accuracy and average length are averaged over the evaluated benchmarks. Gaps are computed relative to Standard CoT under the same backbone.

Backbone	Method	Accuracy (%) ↑	Avg. Length ↓
Qwen3-1.7B	Standard CoT	59.20	11,135
	Soft Thinking	57.22 (−1.98)	11,991 (+7.69%)
	Ours	**62.00** (+2.80)	**7,122** (−36.04%)
Qwen3-8B	Standard CoT	76.84	10,584
	Soft Thinking	75.75 (−1.09)	11,188 (+5.71%)
	Ours	**79.84** (+3.00)	**6,349** (−40.01%)
Qwen3-14B	Standard CoT	81.41	8,472
	Soft Thinking	80.13 (−1.28)	9,207 (+8.68%)
	Ours	**83.84** (+2.43)	**5,827** (−31.22%)

I Implementation Details

I.1 Gate Architectures

The modulation module consists of two lightweight Multi-Layer Perceptrons (MLPs), designed to introduce minimal computational overhead.

- **Mode Gate:** A simple projection network: Linear($d_{\text{model}} \to d_{\text{hidden}}$) → GELU → Linear($d_{\text{hidden}} \to 2$).
- **Exit Gate:** To enable time-aware decision making, we augment the hidden state with multi-scale positional encodings. We define a temporal feature vector $\mathbf{p}_t = [t, \ln(t), \sqrt{t}]^\top$, covering linear, logarithmic, and sub-linear scales. The input to the Exit Gate is the concatenation $[\mathbf{h}_t; \mathbf{p}_t]$. The network structure mirrors the Mode Gate but projects to a single scalar logit representing the hazard rate.

I.2 Termination Prompt

Upon triggering the exit action, we append a standardized transition string to guide the backbone model into the answer generation phase:

```
</think>\n\nOkay I think I have found the final answer is
\boxed{
```

I.3 Evaluation Benchmarks

We evaluate our method on five standard benchmarks spanning diverse difficulty levels, from foundational arithmetic to expert-level STEM reasoning:

- **GSM8K** [10]: We use the standard test set consisting of 1,319 grade-school math word problems. This dataset serves as the baseline for evaluating foundational multi-step arithmetic reasoning.
- **Math500** [22]: A curated collection of 500 challenging problems drawn from the MATH dataset. It covers seven mathematical disciplines including algebra, geometry, and number theory, designed to test the model's ability to handle complex problem constraints.

Table 11: **Parameter breakdown on Qwen3-8B.** The proposed controller introduces only 33.60M trainable parameters, accounting for about 0.4% of the total model size.

Component	Parameters	Ratio
LLM Backbone	8190.7M	99.6%
Gates	33.60M	0.4%
- Mode Gate	16.79M	0.2%
- Exit Gate	16.81M	0.2%

- **AIME 2024** [24]: This set contains 30 problems from the 2024 American Invitational Mathematics Examination. As a rigorous competition benchmark, it evaluates the capability to perform deep symbolic manipulation and maintain logical consistency over long reasoning chains.
- **AIME 2025** [58]: We also include the latest 30 problems from the 2025 edition of the AIME. Testing on this recently released dataset helps assess the model's generalization to unseen tasks and mitigates potential contamination issues.
- **GPQA Diamond** [35]: For broader scientific reasoning, we utilize the high-quality Diamond subset containing 198 expert-level multiple-choice questions. This benchmark evaluates factual knowledge and reasoning across physics, chemistry, and biology, where even human experts often struggle.

I.4 Hyperparameters

Table 12 summarizes the training configurations used in our experiments.

Table 12: Hyperparameter Settings

Hyperparameter	Value
Batch Size	64
GRPO Group Size (G)	8
Actor Learning Rate	1e-4
Length Penalty (λ)	5e-4
Max Response Length (Training)	8192
Max Response Length (Inference)	38912
Temperature (Inference)	0.7
Exit Gate Training	
Training Log-Survival Threshold	$\log \gamma \sim U(-20.0, 0.0)$
Inference Log-Survival Threshold	$\log \gamma = -15.0$

J Further Analysis of the Exit Gate

Termination on the same Mode-Gated policy. We compare DEER with our Exit Gate on the same independently trained Mode Gate checkpoint (Table 13). The Mode Gate is held fixed while the Exit Gate is trained sequentially. Both termination methods use the same structural decision boundaries and maximum decoding budget. The learned Exit Gate reaches 79.92% accuracy with 6237 tokens, compared with 78.49% and 6831 tokens for DEER. This controlled comparison uses sequential training, whereas the main results in Table 1 use joint training.

Table 13: **Termination methods on a fixed Mode Gate policy.** Results use Qwen3-8B and are averaged over the five main benchmarks. Our Exit Gate is trained sequentially with the Mode Gate held fixed.

Setting	**Accuracy ↑**	**Avg. Length ↓**
Mode Gate only	79.77	10066
Mode Gate + DEER	78.49	6831
Mode Gate + Exit Gate (sequential)	**79.92**	**6237**

Standalone termination without the Mode Gate. Table 14 compares both termination methods without the Mode Gate across three backbones. The standalone Exit Gate achieves a mean accuracy of 62.98% with 6726 tokens, compared with 62.47% and 6883 tokens for DEER. DEER has higher accuracy on Qwen3-8B, while our Exit Gate has higher accuracy on the other two models.

Table 14: **Standalone Exit Gate versus DEER.** Both methods are evaluated without the Mode Gate. Accuracy and length are averaged over the five main benchmarks for each backbone.

Backbone	**Method**	**Accuracy ↑**	**Avg. Length ↓**
Qwen3-8B	DEER	78.06	6617
	Exit Gate	76.98	6562
Qwen3-1.7B	DEER	57.99	7275
	Exit Gate	59.51	7402
R1-Distill-Llama-8B	DEER	51.36	6757
	Exit Gate	52.46	6213
Average	DEER	62.47	6883
	Exit Gate	**62.98**	**6726**

K Additional Evaluation Results

We evaluate sampling stability on AIME, generalization to an instruction-tuned backbone and additional task domains, and sensitivity to structural decision boundaries.

AIME sampling stability. We report mean ± standard deviation of the avg@k estimate, for $k = 16, 32$ samples per problem (avg@16/avg@32), rather than pass@k or majority-vote accuracy.

Table 15: AIME accuracy (%) with larger sampling budgets (Qwen3-8B).

Benchmark	Method	avg@16 ↑	avg@32 ↑
AIME 2024	Standard CoT	74.17 ± 1.16	74.38 ± 0.83
	Soft Thinking	70.00 ± 1.29	70.21 ± 0.90
	DEER	74.58 ± 1.39	74.27 ± 0.96
	SwiReasoning	75.21 ± 1.22	75.31 ± 0.85
	Ours	**76.25** ± 1.17	**76.35** ± 0.82
AIME 2025	Standard CoT	64.17 ± 1.51	64.38 ± 1.04
	Soft Thinking	66.46 ± 1.41	66.56 ± 0.98
	DEER	66.88 ± 1.54	66.67 ± 1.07
	SwiReasoning	69.17 ± 1.38	69.27 ± 0.95
	Ours	**71.46** ± 1.25	**71.35** ± 0.87

Generalization to an instruction-tuned model. We train and evaluate the controller on Qwen3-4B-Instruct-2507 using the same protocol as in the main experiments. For this backbone, the termination prompt omits the `</think>` marker. Table 16 reports accuracy and average response length on each benchmark. Our method achieves the highest accuracy and shortest responses on four of the five benchmarks, with DEER performing best on MATH-500.

Table 16: **Evaluation on Qwen3-4B-Instruct-2507.** Each cell reports accuracy (%) / average response length in tokens. Results are shown separately for each benchmark.

Benchmark	Standard CoT	Soft Thinking	DEER	SwiReasoning	Ours
GSM8K	91.89 / 384	90.70 / 377	91.48 / 343	92.00 / 374	**92.13 / 321**
MATH-500	88.73 / 1325	87.83 / 1315	**88.79 / 1236**	88.63 / 1330	88.77 / 1249
GPQA Diamond	40.99 / 2860	37.54 / 2959	40.68 / 2773	41.33 / 2910	**42.42 / 2694**
AIME 2024	52.78 / 5746	53.33 / 5707	53.33 / 5460	53.33 / 5800	**55.00 / 5315**
AIME 2025	43.89 / 5854	42.22 / 5607	42.22 / 5571	44.44 / 5660	**47.22 / 5360**

Generalization to code generation and commonsense reasoning. We evaluate Qwen3-8B on HumanEval [5] and MBPP [2] for code generation, and CommonsenseQA [46] for commonsense reasoning. All methods use matched evaluation settings on each benchmark. Table 17 reports task performance and average response length for each dataset. Our method achieves the highest task score and shortest average response length on all three benchmarks.

Table 17: **Cross-domain evaluation on Qwen3-8B.** Each cell reports task performance (%) / average response length in tokens.

Benchmark	Standard CoT	Soft Thinking	DEER	SwiReasoning	Ours
HumanEval	92.29 / 2967	92.08 / 3149	93.47 / 1768	94.37 / 2749	**95.14 / 1639**
MBPP	87.95 / 4322	87.66 / 4569	86.13 / 2742	88.67 / 4268	**89.20 / 2531**
CommonsenseQA	83.78 / 1128	83.95 / 1144	84.44 / 675	84.34 / 1156	**85.17 / 636**

Structural trigger ablation. Our default *paragraph delimiters* mark boundaries of coherent reasoning segments using three newline variants: `\n\n`, `.\n\n`, and a space followed by `\n\n`. We detect these boundaries by matching their tokenized representations with the model's tokenizer. In contrast, *discourse markers* are lexical transitions (e.g., "Wait", "However", and "Therefore") that signal a new reasoning move; the *union* includes both types. We train two additional Qwen3-8B variants using discourse markers and their union with paragraph delimiters, respectively, under the same experimental protocol. Table 18 shows that paragraph delimiters yield higher average accuracy and shorter responses. Combining both sets does not improve performance, suggesting that paragraph boundaries more reliably identify completed reasoning units than lexical transitions alone.

Table 18: Structural trigger-set ablation on Qwen3-8B, averaged over the five main benchmarks.

Trigger set	Accuracy (%) ↑	Avg. Length ↓
Paragraph delimiters (ours)	**79.84**	**6,349**
Discourse markers	77.43	6,944
Union of both sets	78.67	6,596

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 introduction state that the paper proposes a lightweight controller for frozen LLMs with mode modulation and survival-based termination. These claims are supported by the method formulation and the experiments across multiple backbones and benchmarks.

   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 discusses limitations and future directions in Appendix H.

   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 paper provides theoretical analyses for dynamic thresholding and the robustness of cumulative survival-based termination in the appendix. The assumptions and derivations are stated in the corresponding theoretical sections.

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: The paper describes the backbone models, datasets, evaluation metrics, training objective, gate architectures, termination prompt, hyperparameters, threshold protocol, and implementation framework. These details provide a reasonable path to reproduce the main experimental results.

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: [No]

   Justification: The evaluation datasets and backbone models used in the paper are publicly available and properly cited, but the codes are not released at submission time. We provide implementation details and hyperparameters to facilitate reproduction.

   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: The paper specifies the training dataset, backbone models, evaluation benchmarks, metrics, group size, learning rate, length penalty, response lengths, threshold range, inference threshold, decoding temperatures, and software framework. Additional implementation details are provided in the appendix.

   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: The paper reports that all main results are averaged over six independent runs, and the Appendix F provides standard deviations for the main experimental results.

   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: The computer resources information is provided in Section 4.1.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research uses publicly available models and benchmark datasets and does not involve human subjects, private data, or data collection from users.

   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: The paper discusses broader impacts in Appendix H.

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: The paper does not release a new pretrained foundation model, high-risk dataset, or deployed system.

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: The paper cites the original sources for all backbone models, datasets, and software frameworks used in the experiments.

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: The paper does not introduce or release a new dataset, benchmark, or pretrained model asset.

    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: The paper does not involve crowdsourcing, user studies, or research with human participants.

    Guidelines:

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

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

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

    Answer: [N/A]

Justification: The paper does not involve human subjects, user studies, or collection of personal data, so IRB approval is not applicable.

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: LLMs are a central component of the proposed method. The paper explicitly describes the frozen backbone LLMs used in the method and experiments.

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.