REASON FORWARD, REFLECT WHEN NEEDED: PREDICTIVE ANCHORING WITH SUFFIX-EVIDENCE FOR REVISABLE LLM REASONING

Anonymous authors
Paper under double-blind review

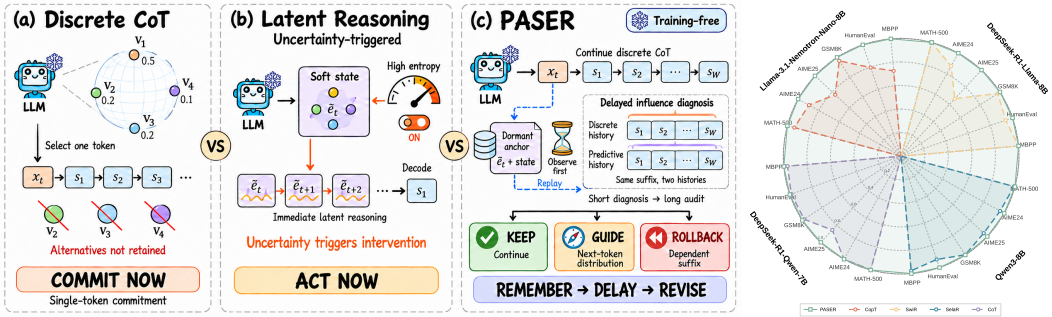

Figure 1: Comparison of LLM reasoning paradigms and performance. ***The left panel*** shows the reasoning paradigm. (a) Standard CoT commits to a single token at each decoding step. (b) Latent reasoning immediately is activated upon uncertainty. (c) **PASER** stores predictive anchors, defers judgment, and performs cognitive reflection. ***The right panel*** represents the results across model families and reasoning domains.

ABSTRACT

Human experts seldom revise uncertain intermediate steps at once. They tentatively accept a step, retain alternatives, inspect downstream effects, and revise only evidence-supported parts of the reasoning state. Existing latent-reasoning methods, however, typically activate continuous representations at high-entropy tokens, conflating uncertainty with the need for latent reasoning. Our analysis reveals three findings: ❶ Uncertainty is immediately observable but insufficient for intervention. ❷ Intervention impacts emerge later yet locally diagnosable. ❸ Revision scope should match influence scope. Building on these observations, we propose **PASER**, a training-free LLM reasoning framework with three key innovations: 1) **PASER** treats high-entropy uncertainty as a signal for caching predictive anchors, rather than immediately activating latent reasoning. 2) **PASER** restores each anchor and evaluates the same realized suffix under discrete CoT and predictive soft histories, determining whether the current latent reasoning interference has an effect. 3) By accumulating impact evidence over a longer horizon, **PASER** maps its strength to hierarchical reflective actions. Evaluated on mathematical, STEM, coding, and general reasoning benchmarks, **PASER** achieves aggregate performance over competitive training-free baselines across diverse model families and scales.

1 INTRODUCTION

Reasoning is central in large language models (Du et al., 2022; Touvron et al., 2023; Yang et al., 2025; Agarwal et al., 2025; Singh et al., 2025) to solving complex tasks in mathematics, science, programming, and multi-step decision-making. Chain-of-thought (CoT) (Wei et al., 2022; Goyal et al., 2024; Pfau et al., 2024) improves accuracy and interpretability by explicitly generating intermediate steps. However, at each decoding step, CoT collapses the next-token distribution into a single token, excluding alternative reasoning paths from subsequent computation. Latent reasoning (Cheng & Van Durme, 2024; Hao et al., 2024; Zhang et al., 2026; Xu et al., 2025; Tan et al., 2026) alleviates this premature discretization by propagating soft embeddings or hidden states that can retain multiple candidates. Existing approaches are broadly divided into training-based methods (Hao et al., 2024; Su et al., 2025; Shen et al., 2025; Tack et al., 2026), which require costly adaptation to continuous

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

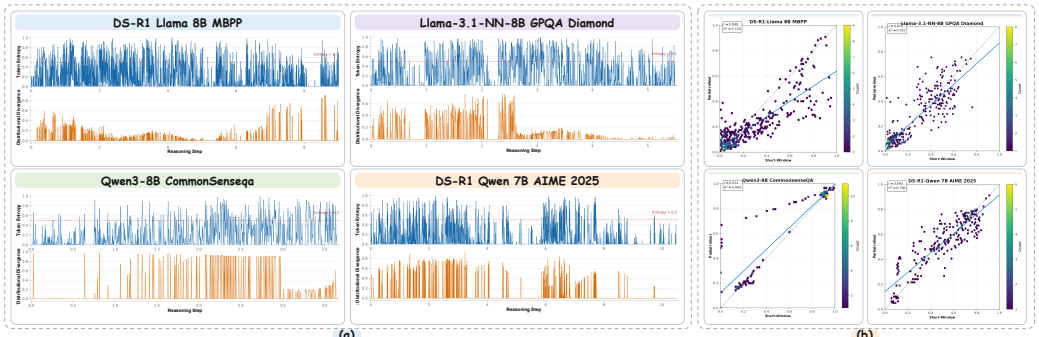

Figure 2: **Preliminary diagnostic analysis.** (a) ***The left panel*** shows token entropy and trajectory divergence. (b) ***The right panel*** reports that short-window influence correlates with partial-rollout estimates on r and $\mathcal{R}^2$.

representations, and training-free methods, which can be directly applied to pretrained reasoning models and are therefore more practical. Soft-Thinking (Zhang et al., 2026), a representative training-free method, repeatedly feeds probability-weighted soft embeddings into decoding, allowing multiple candidates to influence subsequent predictions. However, indiscriminate latent injection may perturb confident steps and induce latent-space drift or overthinking. Thus, global latentization increases representational bandwidth without determining which uncertainty warrants further computation. Recent methods introduce more selective control: SeLaR (Fu & Luo, 2026) activates soft embeddings at high-entropy exploratory positions and uses contrastive regularization to prevent premature collapse, whereas SwiReasoning (SwiR) (Shi et al., 2026a) switches between explicit and latent reasoning according to block-level entropy trends while limiting switch frequency. These methods refine where latent computation is activated and when reasoning modes are switched.

However, entropy indicates uncertainty rather than consequence: when activating latent reasoning at high-entropy token, its downstream effect is not yet observable. Our diagnostic experiments across mathematical, scientific, coding, and general reasoning tasks confirm this distinction. **Observation ❶** As shown in Fig. 2(a), *some high-entropy tokens exert limited impact on the subsequent reasoning trajectory*. Entropy is therefore useful for identifying candidate positions, but insufficient for directly triggering latent reasoning mode. CopT (Shi et al., 2026b) introduces later evidence by using generated draft tokens to reinterpret continuous embeddings as a contrastive verifier. However, its response-level reliability signal does not identify which earlier reasoning should be revised or how extensively.

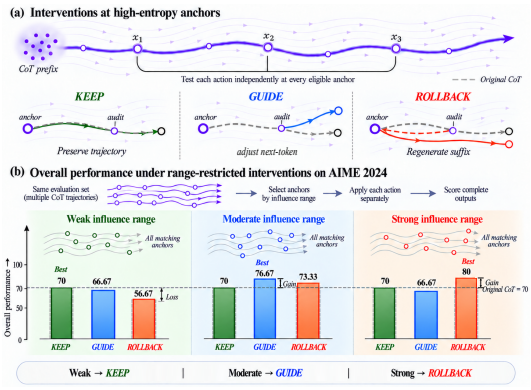

Figure 3: **Intervention effectiveness.** (a) Three actions at high-entropy anchors. (b) Overall performance of each action across specified influence ranges.

Observation ❷ Our reflection analysis in Fig. 3 further reveals that no single reflective action is universally optimal: *weak influence favors direct continuation, moderate influence can be rectified via next-token steering, and persistent influence requires backtracking*. The reflection scope should therefore match the reasoning influence footprint. **Observation ❸** Moreover, Fig. 2(b) shows that short-window and partial-rollout influence estimates are positively correlated across all four task families, suggesting that *local future trajectories can provide diagnostic evidence before reasoning is complete*. These observations align with how human experts reason under uncertainty. When proving a theorem or constructing a scientific explanation, experts may provisionally accept a step while retaining alternatives, examining its downstream consequences and restricting correction to the smallest evidence-justified scope. This process separates three cognitive operations: **Predictive thinking**, which preserves plausible alternatives. **Delayed judgment**, which waits for diagnostic consequences. **Cognitive reflection**, which revises hypotheses according to influence strength.

Inspired by this pattern, we propose ***Predictive Anchoring with Suffix Evidence for Revisable Reasoning*** (**PASER**), a training-free paradigm. As illustrated in Fig. 1, **PASER** shifts latent reasoning from immediate intervention to delayed, evidence-guided revision. At a high-entropy uncertain token position, it caches a recoverable predictive anchor while allowing the CoT trajectory to continue.

Once sufficient suffix tokens are generated, it evaluates the influence of activating latent reasoning at the high-entropy position on downstream reasoning and applies a graded reflection. This formulation raises three questions: ***I) Where to preserve predictive anchors? II) When does future evidence become diagnostic? III) What reflective action should be taken?*** **PASER** addresses them through the following mechanisms:

❶ **Predictive Anchoring.** **PASER** uses the uncertainty of high-entropy tokens only to identify potentially revisable reasoning, not to immediately switch to latent reasoning. At each such position, it caches the decoding state while the main trajectory proceeds with the sampled discrete token.

❷ **Delayed Influence Diagnosis.** After generating a short suffix, **PASER** restores anchor states and re-evaluates the realized suffix with soft embeddings. It quantifies the impact of latent intervention on downstream reasoning by measuring suffix-support discrepancy between predictive and original CoT histories. This delayed diagnosis filters inconsequential high-entropy tokens.

❸ **Evidence-Calibrated Reflection.** **PASER** accumulates divergence evidence of influence over a longer horizon and maps it to hierarchical reflective actions. **KEEP**: weak influence leaves the CoT trajectory unchanged. **GUIDE**: moderate influence redirects the next-token embedding. **ROLL BACK**: strong, persistent divergence retraces its anchoring state and regenerates its dependent trajectory under latent reasoning. The scope of reflection therefore adapts to the estimated influence footprint.

The contributions of this work are summarized as follows:

• ***Observation*:** We identify two limitations of uncertainty-driven latent reasoning: activating latent reasoning at high-entropy tokens does not necessarily affect subsequent reasoning, and no single reflective action is optimal across different influence patterns. These findings motivate a shift from immediate latent reasoning intervention to delayed, evidence-guided revisable reasoning.

• ***Paradigm*:** We propose **PASER**, a training-free revisable reasoning paradigm that integrates predictive thinking, delayed judgment, and cognitive reflection into a unified process.

• ***Performance*:** We conduct comprehensive evaluations across diverse reasoning benchmarks and multiple LLM families with different parameter scales. Experimental results demonstrate that **PASER** yields consistent performance gains over state-of-the-art training-free latent reasoning baselines.

2 Related Work

Explicit Reasoning. Explicit reasoning generates natural-language intermediate steps to improve performance on complex tasks. Chain-of-thought prompting decomposes a problem into sequential reasoning steps (Kojima et al., 2022; Wei et al., 2022; Madaan et al., 2023; Yu et al., 2024; Shao et al., 2024; Lee et al., 2025; Havrilla et al., 2024; Wei et al., 2025; Jin et al., 2025), while subsequent methods expand this paradigm through decoding and search. Self-consistency (Wang et al., 2022) aggregates multiple sampled trajectories, and Tree of Thoughts (Yao et al., 2023) searches over branching intermediate states. Despite these advances, such methods operate in discrete token space: each decoding step commits to a single token and discards the remaining candidates.

Latent Reasoning. Latent reasoning carries intermediate computation through hidden states or soft embeddings rather than discrete CoT tokens. Training-based methods (Tack et al., 2026; Zeng et al., 2026; Tan et al., 2026; Jiang et al., 2025; Shen et al., 2025; Xu et al., 2025; Geiping et al., 2026) adapt models to continuous reasoning spaces, whereas training-free methods manipulate internal representations only during inference. Soft-Thinking (Zhang et al., 2026) applies soft embeddings throughout generation, SeLaR (Fu & Luo, 2026) activates them at high-entropy positions, and SwiReasoning (Shi et al., 2026a) switches between explicit and latent modes using block-level entropy trends. These methods mainly use continuous representations to directly influence subsequent generation. In contrast, our method treats high-entropy tokens as revisable predictive anchors, using suffixes to estimate latent-intervention effects and trigger reflection only when necessary.

3 Motivation

3.1 Preliminaries

Explicit Reasoning. Let $\mathcal{V}$ be the vocabulary and $p_\theta(x_t \mid x_{<t})$ a causal language model parameterized by θ. Given a question q, the model generates a reasoning trace $r_{1:T} \in \mathcal{V}^T$ followed by an

answer $a_{1:U} \in \mathcal{V}^U$, yielding $x = [q, r_{1:T}, a_{1:U}]$. At each step, standard explicit chain-of-thought reasoning decodes a discrete token as:

$$x_t = \begin{cases} \arg\max_{v \in \mathcal{V}} p_\theta(v \mid q, x_{<t}), & \textsc{Greedy}, \\ v \sim (p_\theta(v \mid q, x_{<t}); \tau, k, p), & \textsc{Sampling}, \end{cases} \tag{1}$$

where τ, k, and p control temperature, top-k, and top-p sampling, respectively. Reasoning terminates at an end-of-reasoning delimiter, such as `</think>`, or the maximum generation length.

Training-Free Latent Reasoning. Let $\mathbf{E} \in \mathbb{R}^{|\mathcal{V}| \times d}$ be the token-embedding matrix, where $\mathbf{E}_v \in \mathbb{R}^d$ denotes the input embedding of token v. At decoding step t, the model produces logits $\ell_t \in \mathbb{R}^{|\mathcal{V}|}$ and the corresponding predictive distribution $p_t = \mathrm{softmax}(\ell_t)$. Rather than selecting a single discrete token x_t, latent reasoning maps the predictive distribution to a soft embedding $\tilde{e}_t = \sum_{v \in \mathcal{V}} p_t(v)\mathbf{E}_v$. Then, the $\tilde{e}_t$ is typically fed into the model for the next decoding step, enabling implicit exploration over multiple candidate tokens in a single forward pass.

3.2 Observation

Pilot Study. Existing training-free latent-reasoning methods typically activate latent reasoning at high-entropy tokens, implicitly equating uncertainty with need for latent reasoning. To verify this assumption, we conduct an intervention study with four representative language models (Yang et al., 2025; Guo et al., 2025; Bercovich et al., 2025) on four reasoning tasks (Yentinglin, 2025; Rein et al., 2023; Austin et al., 2021; Talmor et al., 2019). Given discrete inference trajectories $r_{1:T}$ and predictive distributions $p_\theta(\cdot \mid x_{<t})$, we analyze high-entropy token candidate positions using the normalized top-3 entropy H_t. For each candidate position t ($H_t \geq 0.5$), we build two contrasting rollout histories of length $k = 256$ branching from the identical pre-decision prefix state. The discrete history $\mathcal{S}_t^{\mathrm{d}} = \left[x_{t:t+k}, \mathbf{E}(x_{t:t+k})\right]$ corresponds to the CoT trajectory: it consumes the actually-sampled discrete tokens $x_{t:t+k}$. The predictive history $\mathcal{S}_t^{\mathrm{p}}$ reuses the same target token sequence but replaces discrete embeddings with soft embeddings $\tilde{e}_{t:t+k}$. We compute the average Jensen-Shannon divergence between the resulting next-token distributions from the two histories, which reflects how strongly latent continuous embedding alter the model's subsequent reasoning. More details are provided in Appendix B. We summarize our key findings (**Finds.**) as follows:

Finds. ❶: Uncertainty Is Immediately Observable but Not Intervention-Sufficient. Fig. 2(a) shows that entropy can cover potential uncertainties, but cannot determine whether activating latent reasoning at high-entropy token will materially alter the subsequent reasoning. It should therefore serve as an anchor-proposal signal, rather than an immediate trigger for latent reasoning.

Finds. ❷: Intervention Impacts Are Delayed but Locally Observable. If latent intervention importance can only be assessed after a complete partial rollout, reflection becomes costly post-hoc reasoning. As shown in Fig. 2(b), the short-window estimate $\mathcal{I}_t^{(k/8)}$ is strongly correlated with its remaining partial-rollout $\mathcal{I}_t$ across models and tasks ($r = 0.84$-0.93, $\mathcal{R}^2 = 0.72$-0.87). Thus, short realized suffixes provide useful diagnostic information before reasoning is complete.

Furthermore, we stratify candidate anchors by the magnitude of downstream distributional divergence and compare three actions, *e.g.,* keeping, guiding or rolling back, revealing the following:

Finds. ❸: Revision Scope Should Match Influence Scope. As shown in Fig. 3, no single reflective action is universally optimal: weak influence favors direct continuation, moderate influence can be rectified via next-token steering, and persistent influence requires rollback.

4 Methodology

4.1 Overview

As illustrated in Fig. 4, **PASER** is a training-free reasoning paradigm that maintains revisable reasoning memory through delayed diagnosis and evidence-calibrated reflection, comprising ❶ **Predictive anchoring** caches the decoding state at high-entropy positions. ❷ **Delayed influence diagnosis** replays the same realized suffix from the cached anchor and compares its support under discrete CoT

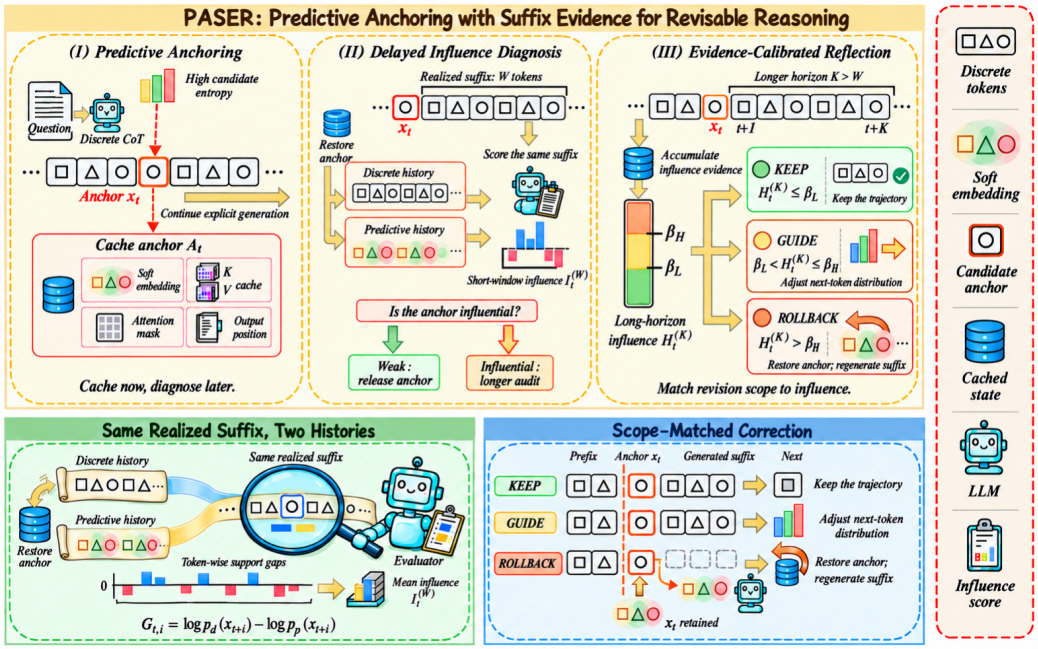

Figure 4: **An illustration of our proposed PASER.** (a) ***The top panel*** shows the main reasoning, consisting of Predictive Anchoring, Delayed Influence Diagnosis, and Evidence-Calibrated Reflection. (b) ***The bottom-left panel*** compares support for the same suffix under discrete and predictive histories to quantify influence. (c) ***The bottom-right panel*** maps influence evidence to scope-matched KEEP, GUIDE, and ROLLBACK actions.

and predictive soft histories. ❸ **Evidence-Calibrated reflection** accumulates downstream influence and selects KEEP, GUIDE, or ROLL BACK according to its scope.

4.2 PREDICTIVE ANCHORING

Candidate Anchor. Let p_t denote the predictive distribution at step t, and let $\mathcal{V}_t = \mathrm{TopK}(p_t, 3)$ contain its most probable tokens. We renormalize their probabilities $\bar{p}_t(v)$ and compute the normalized entropy $H_t = -1/\log 3 \cdot \sum_{v \in \mathcal{V}_t} \bar{p}_t(v) \log \bar{p}_t(v)$. A large H_t indicates competition among plausible candidates. Let $\mathcal{A}$ denote the set of anchor. A candidate at position t is instantiated only when:

$$a_t = \mathbb{I}[H_t \geq 0.5] \cdot \mathbb{I}[|\mathcal{A}_t| = 0], \tag{2}$$

where $\mathbb{I}[\cdot]$ is the indicator function and $a_t = 1$ denotes anchor instantiation. Importantly, this step serves solely to determine which reasoning should retain revisability, and does not entail any immediate latent reasoning mode to the current CoT trajectory.

Recoverable Decoding State Cache. For instantiated anchor, PASER caches decoding state:

$$\mathcal{A}_\mathbf{t} = \left(\tilde{e}_t, \mathcal{K}_{<\mathbf{t}}, \mathcal{M}_{<\mathbf{t}}, \mathcal{O}_\mathbf{t}\right), \tag{3}$$

where $\tilde{e}_t$ is the continuous embedding, $\mathcal{K}_{<\mathbf{t}}$ is the key/value cache before consuming the anchor token, $\mathbf{M}_{<\mathbf{t}}$ is the corresponding attention mask, and $\mathcal{O}_\mathbf{t}$ records the current output position. The main path then consumes the discrete $\mathbf{E}_{x_t}$ and continues standard explicit generation. Anchoring therefore makes the reasoning recoverable without changing any generated tokens. If subsequent evidence reveals no material influence, $\mathcal{A}_t$ is released and the CoT trajectory remains intact.

4.3 DELAYED INFLUENCE DIAGNOSIS

Replayed Suffix Contrasting. After establishing a candidate anchor, the main discrete CoT path generates a suffix of length W, $\mathcal{S}_\mathbf{t} = x_{t+1:t+W}$. PASER then restores the pre-anchor state stored in $\mathcal{A}_\mathbf{t}$ and re-evaluates the same suffix under two histories. The discrete CoT history directly uses the realized tokens $x_t, x_{t+1}, \ldots$, whereas the predictive soft history uses the soft embeddings $\tilde{e}_t, \tilde{e}_{t+1}, \ldots$ For suffix position i, we define the token-level support gap as:

$$\mathcal{G}_{t,i} = \log p_\theta\left(x_{t+i} \mid q, x_{<t}, x_{t:t+i-1}\right) - \log p_\theta\left(x_{t+i} \mid q, x_{<t}, \tilde{e}_{t:t+i-1}\right). \tag{4}$$

Table 1: **Comparison of PASER** with baseline methods on **math** reasoning and **STEM** reasoning tasks. Cell background colors indicate relative performance: darker colors denote better results within each model group.

Model	Method	Math Domain								STEM Domain	Avg.
		GSM8K	Math500	AIME24	AIME25	AIME26	AMC23	M-MMLU	GSM-Hard	GPQA	
Qwen3-8B (Yang et al., 2025)	Soft Thinking	95.60	95.80	64.17	62.50	52.50	90.00	92.00	63.76	55.05	74.60
	CoT Sampling	95.75	97.20	67.50	65.00	64.17	97.50	96.00	62.93	55.56	77.96
	CoT Greedy	95.07	96.00	65.83	58.33	57.50	92.50	91.00	62.47	51.52	74.47
	CopT	95.75	95.60	76.67	66.67	56.67	90.00	82.00	66.49	54.04	75.99
	SwiR	95.60	97.60	74.17	73.33	65.83	90.00	95.00	62.32	60.10	79.33
	SeLaR	95.83	96.80	75.83	74.17	65.83	95.00	87.00	63.61	61.11	79.46
	PASER	95.98	97.60	80.83	77.50	70.00	100.00	96.00	68.92	62.63	**83.27**
DS-R1-Qwen-7B (Guo et al., 2025)	Soft Thinking	89.61	91.00	54.17	41.67	36.67	82.50	95.00	56.33	42.42	65.49
	CoT Sampling	90.98	94.20	53.33	37.50	40.00	92.50	93.00	54.36	41.92	66.42
	CoT Greedy	90.37	89.20	43.33	40.83	44.17	82.50	80.00	52.92	29.80	61.46
	CopT	88.63	85.60	42.50	25.84	37.50	60.00	59.00	50.64	33.84	53.73
	SwiR	92.80	90.60	50.83	35.83	41.67	80.00	93.00	47.76	39.39	63.54
	SeLaR	92.04	92.80	46.67	31.67	40.83	80.00	88.00	46.63	36.36	61.67
	PASER	92.65	94.40	57.50	44.17	47.50	92.50	95.00	59.82	43.43	**69.66**
DS-R1-Llama-8B (Guo et al., 2025)	Soft Thinking	85.37	82.20	33.33	21.67	17.50	80.00	74.00	-	37.37	53.93
	CoT Sampling	88.70	88.20	40.00	27.50	20.83	90.00	77.00	-	47.47	59.96
	CoT Greedy	85.14	83.60	30.83	24.17	27.50	72.50	73.00	-	28.79	53.19
	CopT	72.10	84.00	35.00	25.83	37.50	85.00	71.00	-	14.65	53.14
	SwiR	89.01	91.20	44.17	25.83	31.67	87.50	74.00	-	45.96	61.17
	SeLaR	89.31	86.40	44.17	30.83	29.17	82.50	68.00	-	43.94	59.29
	PASER	90.98	91.20	50.83	34.17	38.34	92.50	78.00	-	49.49	**65.69**
DSR-1.5B-Preview (Luo et al., 2025)	Soft Thinking	77.48	80.00	37.50	31.67	17.50	80.00	71.00	-	-	56.45
	CoT Sampling	86.50	83.80	38.33	28.33	24.17	82.50	79.00	-	-	60.38
	CoT Greedy	79.83	75.60	30.83	25.00	19.17	65.00	77.00	-	-	53.20
	SwiR	82.34	73.00	35.00	25.00	24.17	85.00	85.00	-	-	58.50
	SeLaR	86.35	88.20	34.17	31.67	25.00	80.00	78.00	-	-	60.48
	PASER	87.11	89.20	41.67	39.17	31.67	85.00	85.00	-	-	**65.55**
Llama3.1-NN-8B (Bercovich et al., 2025)	Soft Thinking	89.00	80.00	41.67	32.50	35.00	82.50	73.00	38.29	30.30	55.81
	CoT Sampling	91.89	94.00	56.67	43.33	58.33	92.50	86.00	64.75	48.99	70.72
	CoT Greedy	91.36	94.60	54.17	41.67	44.17	92.50	85.00	61.26	48.99	68.19
	CopT	89.00	89.40	54.17	41.67	50.83	77.50	72.00	56.41	46.97	64.22
	SeLaR	91.21	94.40	57.50	48.33	60.00	95.00	83.00	65.12	40.91	70.61
	PASER	92.19	95.60	61.67	51.67	61.67	97.50	89.00	67.40	51.01	**74.19**

Then, the short-window influence score is calculated as $\mathcal{I}_t^{(W)} = \frac{1}{W}\sum_{i=1}^{W} \mathcal{G}_{t,i}$. A small $\mathcal{I}$ indicates that the discrete and predictive histories provide comparable support for the suffix, suggesting that latent reasoning at a high-entropy anchor has little downstream influence. In contrast, a large score indicates that the suffix depends strongly on the discrete reasoning mode at position t, *i.e.*, uncertainty may affect subsequent output. In practice, we set an influence threshold δ. Anchor $\mathcal{A}_t$ with insufficient influence, *i.e.*, $\mathcal{I} \leq \delta$, are released, whereas influential ones proceed to long-horizon auditing.

4.4 Evidence-Calibrated Reflection

Long-Horizon Influence. For each confirmed anchor, the main CoT path continues until a longer suffix $x_{t+1:t+K}$ has been generated, where $K > W$. **PASER** restores the same pre-anchor state, replays the predictive soft history, and computes:

$$\mathcal{H}_t^{(K)} = \frac{1}{K}\sum_{i=1}^{K}\left[\log p_\theta\left(x_{t+i} \mid q, x_{<t}, x_{t:t+i-1}\right) - \log p_\theta\left(x_{t+i} \mid q, x_{<t}; \tilde{e}_{t:t+i-1}\right)\right]. \tag{5}$$

The short-window score $\mathcal{I}_t^{(W)}$ determines whether activating latent reasoning at a high-entropy uncertain token affects the subsequent reasoning, whereas the long-horizon score $\mathcal{H}_t^{(K)}$ measures whether this influence persists over a broader portion of the trajectory.

Hierarchical Reflection. Given two influence thresholds $\beta_L < \beta_H$, **PASER** maps the long-horizon evidence to a hierarchical reflective action:

❶ **Keep.** When the accumulated influence is weak ($\mathcal{H}_t^{(K)} \leq \beta_L$), **PASER** releases the anchor $\mathcal{A}_t$ and continues along the original discrete CoT trajectory without intervention.

❷ **Guide.** When the influence within $\beta_L < \mathcal{H}_t^{(K)} \leq \beta_H$, **PASER** removes the anchor $\mathcal{A}_t$ and modifies next-token soft embedding. We introduce a influence-calibrated contrastive guidance that steers the next-token distribution toward preferences supported by the predictive history. Let p_t^{d} and p_t^{p} denote the next-token distributions under discrete and predictive histories. We compute:

$$p_t^{\mathrm{g}}(v) = \frac{p_t^{\mathrm{p}}(v)\exp\left[g_t \tanh\left(\log \frac{p_t^{\mathrm{p}}(v)}{p_t^{\mathrm{d}}(v)}\right)\right]}{\sum_{u\in\mathcal{V}} p_t^{\mathrm{p}}(u)\exp\left[g_t \tanh\left(\log \frac{p_t^{\mathrm{p}}(u)}{p_t^{\mathrm{d}}(u)}\right)\right]}, \quad g_t = \frac{\mathcal{H}_t^{(K)} - \beta_{\mathrm{L}}}{\beta_{\mathrm{H}} - \beta_{\mathrm{L}}} \tag{6}$$

Table 2: **Comparison of PASER** with baseline methods on **coding** reasoning and **general** reasoning tasks. Cell background colors indicate relative performance: darker colors denote better results within each model group.

Model	Method	Coding Domain						General Domain					Avg.
		HumanE	HumanE+	MBPP	MBPP+	LeetCode-C	LiveCode-B	2WM-HQA	ComS-QA	HotpotQA	MuSiQ	StrgyQA	
Qwen3-1.7B (Yang et al., 2025)	Soft Thinking	81.10	69.51	69.26	70.63	61.67	33.69	65.00	71.26	69.76	41.70	81.83	65.04
	CoT Sampling	81.10	78.66	77.82	70.11	61.67	35.13	65.50	73.79	70.53	40.38	82.45	67.01
	CoT Greedy	84.15	75.00	78.99	69.31	59.44	34.05	66.50	74.28	69.71	40.38	81.40	66.66
	CopT	49.39	47.56	61.48	53.70	42.78	24.01	41.00	69.86	57.89	31.03	74.32	50.27
	SwiR	84.76	79.88	81.32	72.49	66.11	35.84	66.00	74.53	69.47	41.70	81.75	68.53
	SeLaR	75.00	68.90	73.54	73.81	51.11	26.52	64.50	72.81	68.76	41.00	82.23	63.47
	PASER	86.59	79.27	82.10	81.22	67.22	36.56	68.00	74.37	72.40	42.37	85.02	**70.47**
Qwen3-8B (Yang et al., 2025)	Soft Thinking	92.07	80.49	93.78	76.19	56.11	56.27	79.50	83.54	77.83	62.39	85.90	76.73
	CoT Sampling	95.12	87.20	94.55	80.69	59.44	61.29	80.50	85.09	77.92	63.09	87.86	79.34
	CoT Greedy	93.90	84.15	91.44	79.37	55.56	49.82	79.00	83.95	77.80	62.18	88.17	76.85
	CopT	93.90	79.88	86.77	72.75	45.00	51.97	77.00	84.85	77.88	63.09	86.94	74.55
	SwiR	95.12	85.36	94.94	78.83	62.22	62.72	78.50	85.18	77.97	62.10	88.34	79.21
	SeLaR	89.63	82.93	93.00	78.57	51.67	42.29	78.00	84.19	77.11	61.48	89.21	75.28
	PASER	96.95	88.41	95.33	81.22	63.33	63.80	82.00	85.75	80.80	64.50	90.13	**81.11**
DS-R1-Qwen-7B (Guo et al., 2025)	Soft Thinking	85.98	78.66	82.88	65.34	-	29.75	29.50	55.94	46.60	22.09	31.57	52.83
	CoT Sampling	88.41	81.10	82.10	71.96	-	39.43	29.00	63.47	44.02	21.43	33.19	55.41
	CoT Greedy	87.80	73.78	80.54	69.05	-	29.39	30.50	62.24	44.51	22.05	30.09	53.00
	SwiR	90.24	78.66	80.16	70.37	-	41.58	31.50	61.83	43.98	22.42	34.19	55.49
	SeLaR	71.95	62.20	77.04	67.46	-	25.81	27.00	54.46	42.35	21.18	37.47	48.69
	PASER	89.63	82.93	84.82	73.28	-	42.29	33.50	66.26	58.65	22.88	37.34	**59.16**
DS-R1-Llama-8B (Guo et al., 2025)	Soft Thinking	82.32	69.51	76.65	70.37	-	30.82	48.50	60.52	41.51	30.74	-	56.77
	CoT Sampling	84.15	81.10	82.88	69.58	-	36.92	49.00	63.72	45.74	34.34	-	60.83
	CoT Greedy	81.71	72.56	77.04	68.78	-	26.88	48.50	56.92	43.96	31.44	-	56.42
	CopT	82.93	78.05	82.49	66.13	-	34.41	44.00	52.66	37.25	28.84	-	56.31
	SwiR	83.54	83.54	83.66	70.90	-	40.14	46.00	62.49	45.08	34.46	-	61.09
	SeLaR	81.71	72.56	78.21	65.87	-	24.73	48.00	58.07	43.67	33.93	-	56.31
	PASER	89.02	82.93	83.66	72.75	-	42.29	53.00	65.60	47.45	35.91	-	**63.62**
Llama3.1-NN-8B (Bercovich et al., 2025)	Soft Thinking	88.41	82.32	84.82	71.16	63.33	46.24	-	69.45	28.18	-	81.18	68.34
	CoT Sampling	87.80	84.15	87.94	74.07	65.56	49.46	-	71.91	29.87	-	83.71	70.50
	CoT Greedy	87.20	84.15	86.77	73.28	61.11	50.54	-	70.43	30.93	-	83.14	69.73
	CopT	71.34	60.37	65.37	60.05	40.00	33.69	-	68.88	32.67	-	84.72	57.45
	SeLaR	80.49	73.17	78.21	72.22	48.33	37.28	-	69.53	28.45	-	82.79	63.39
	PASER	89.63	84.76	89.88	75.40	66.67	51.97	-	71.25	37.47	-	84.89	**72.44**

The next-token soft embedding is $\tilde{e}_t = \sum_{v \in \mathcal{V}} p_t^{\mathrm{g}}(v)\mathbf{E}_v$. As the influence $\mathcal{H}_t^{(K)}$ increases within the guide regime, **PASER** progressively amplifies token preferences that are more strongly supported by the predictive history than by the discrete history.

❸ **ROLLBACK.** When the influence persists throughout the longer suffix ($\mathcal{H}_t^{(K)} > \beta_H$), **PASER** removes tokens generated after the anchor, restores $\mathcal{K}_{<t}$ and $\mathcal{M}_{<t}$, and enters latent reasoning mode to regenerates the subsequent trajectory using the soft embedding $\tilde{e}_t$. The anchor token x_t is retained, but its internal representation for subsequent computation is changed from E_{x_t} to $\tilde{e}_t$. After completing the backtracking, it releases the anchor $\mathcal{A}_t$. Rollback therefore revises only the suffix whose states depend on the influential anchor token, while preserving the earlier reasoning prefix for which no corrective evidence exists.

5 EXPERIMENT

We comprehensively evaluate our proposed method, **PASER**, through four axes: **Q1** (Superiority), **Q2** (Sensitivity), **Q3** (Effectiveness), and **Q4** (Cost-effectiveness).

5.1 EXPERIMENTAL SETUP

Models and Benchmarks. We evaluate **PASER** on eight reasoning models from diverse families and parameter scales: `Qwen3-1.7B` (Yang et al., 2025), `Qwen3-8B` (Yang et al., 2025), `Qwen3-32B` (Yang et al., 2025), `DS-R1-Qwen-32B` (Guo et al., 2025), `DS-R1-Qwen-7B` (Guo et al., 2025), `DS-R1-Llama-8B` (Guo et al., 2025), `Llama-3.1-NN-8B` (Bercovich et al., 2025), and `DSR-1.5B-Preview` (Luo et al., 2025). The benchmarks cover mathematical reasoning (GSM8K (Cobbe et al., 2021), MATH500 (Hendrycks et al., 2021), AIME 2024-2026 (HuggingFaceH4, 2024; Yentinglin, 2025; MathArena, 2026), AMC 2023 (Project Numina, 2024), GSM-Hard (Gao et al., 2023), and Math MMLU (Hendrycks et al., 2020)), STEM reasoning (GPQA Diamond (Rein et al., 2023)), coding reasoning (HumanEval (Chen et al., 2021), HumanEval+ (Liu et al., 2023), MBPP (Austin et al., 2021), MBPP+ (Liu et al., 2023), LiveCodeBench (Jain et al., 2025), and LeetCode Contest (Guo et al., 2024)), and general reasoning (2WikiMultiHopQA (Ho et al., 2020), CommonsenseQA (Talmor et al., 2019), StrategyQA (Geva et al., 2021), MuSiQue (Trivedi et al., 2022) and HotpotQA (Yang et al., 2018)). Additional details are deferred to Appendix A.

Baselines and Implementation Details. We compare **PASER** against multiple baselines, including chain-of-thought reasoning with sampling (CoT Sampling) and greedy decoding (CoT Greedy), Soft-Thinking(Zhang et al., 2026), SeLaR (Fu & Luo, 2026), SwiR (Shi et al., 2026a), and CopT (Shi et al.,

Table 3: **Comparison of PASER** with other baseline methods on **larger** reasoning models. Cell background colors indicate relative performance: darker colors denote better results within each model group.

Model	Method	Math Domain					Coding Domain		Avg.
		GSM8K	Math500	AIME2024	AIME2025	GPQA Diamond	HumanEval	MBPP	
Qwen3-32B (Yang et al., 2025)	Soft Thinking	95.60	97.20	75.00	66.67	67.17	96.95	96.89	85.07
	CoT Sampling	95.91	97.40	77.50	71.67	66.16	97.56	97.28	86.21
	SwiR	95.98	97.80	80.83	74.17	70.20	95.73	98.05	87.54
	SeLaR	96.06	97.60	80.00	78.33	67.17	92.07	91.44	86.10
	PASER	96.82	98.20	81.67	79.17	69.70	98.17	98.44	**88.88**
DS-R1-Qwen-32B (Guo et al., 2025)	Soft Thinking	95.53	94.40	75.00	59.17	64.64	96.34	94.55	82.80
	CoT Sampling	95.38	94.80	74.17	54.17	63.13	96.95	96.11	82.10
	SwiR	95.15	96.20	70.83	64.17	62.63	97.56	93.77	82.90
	SeLaR	94.62	94.40	67.50	57.50	54.04	83.54	90.66	77.47
	PASER	96.13	95.80	78.33	64.17	65.66	98.78	96.89	**85.11**

Table 4: **Ablation study of reasoning strategy and revision policy.**

Method	MATH500	GSM8K	AIME24	AIME25	HumanEval	MBPP	2WH-HQA	ComS-QA	Avg.
PASER (Full)	**97.60**	**95.98**	**80.83**	**77.50**	**96.95**	**95.33**	**82.00**	**85.75**	**88.99**
- w/o Predictive anchor	96.20	94.09	74.17	71.67	94.51	94.55	81.00	84.03	86.28
- Delayed diagnosis w/ **16** tokens	96.60	94.47	77.50	74.17	95.73	94.55	81.00	84.85	87.36
- Delayed diagnosis w/ **32** tokens	97.60	95.98	80.83	77.50	96.95	95.33	82.00	85.75	88.99
- Delayed diagnosis w/ **64** tokens	97.40	95.98	80.83	76.67	96.34	95.72	81.50	85.91	88.79
- Delayed diagnosis w/ **96** tokens	97.80	96.06	80.83	77.50	96.95	95.72	82.00	85.99	89.11
- w/o Long-horizon audit	95.80	94.16	75.00	72.50	94.51	94.94	80.50	84.68	86.51
- w/o Hierarchical reflection	95.20	93.93	73.33	70.00	93.29	94.16	81.00	84.19	85.64

2026b). All experiments are conducted on NVIDIA A100 GPUs, using their official implementations and released hyperparameters. All methods use the same decoding settings: temperature 0.6, top-$p = 0.95$, top-$k = 20$, and min-$p = 0.0$. More details are provided in Appendix A.5.

5.2 Superiority across Multiple Models and Domains

To answer **Q1**, we evaluate on different reasoning language models and broad reasoning domains.

Performance ❶ Math and STEM Reasoning. Tab. 1 reports results across math and STEM benchmarks, and model families. **PASER** achieves state-of-the-art pass@1 performance, outperforming all baselines. **PASER** improves upon SeLaR by +3.81% on Qwen3-8B, CoT Sampling by +3.24% on DS-R1-Qwen-7B, SwiR by +4.52% on DS-R1-Llama-8B, CoT Sampling by +3.47% on Llama-3.1-NN-8B and SeLaR by +5.07% on DSR-1.5B-Preview, respectively. The gains suggest that delayed, evidence-calibrated reflection is especially beneficial for long mathematical derivations.

Performance ❷ Coding Reasoning. Tab. 2 reports results across coding benchmarks and model families. **PASER** achieves the strongest aggregate performance, improving upon SwiR by +2.09% on Qwen3-1.7B, SwiR by +2.39% on DS-R1-Qwen-7B, CoT Sampling by +3.20% on DS-R1-Llama-8B. On Llama-3.1-NN-8B, it improves HumanEval from 88.41% to 89.63% and MBPP from 87.94% to 89.88%. A further gain of 7.41% on MBPPPlus is obtained with Qwen3-1.7B.

Performance ❸ General Reasoning. As shown in Tab. 2, **PASER** attains aggregate performance over baselines across multi-hop, commonsense, and knowledge reasoning benchmarks. Built upon Qwen3-8B, **PASER** yields consistent performance gains on all evaluated tasks. For DS-R1-Llama-8B, **PASER** obtains substantial advancements of 4.0% on 2WikiMultiHopQA and 1.45% on MuSiQue. Furthermore, when equipped with Llama3.1-NN-8B, it boosts the HotpotQA performance from 32.67% to 37.47%, demonstrating its strong generalization ability on general reasoning problems.

Performance ❹ Larger Models. We further conduct evaluation on larger reasoning models in Tab. 3. **PASER** achieves better pass@1 accuracy on math and coding reasoning tasks over the latest training-free baselines. Our method improves upon SwiR by +1.34% on Qwen3-32B, and SwiR by +2.21% on DS-R1-Qwen-32B, respectively. These results indicate **PASER** scales to larger models.

5.3 Effectiveness on Reasoning Designs

Hierarchical Reflection Threshold. To address **Q2**, we examine the sensitivity of **PASER** to the reflection thresholds β_{L} and β_{H}, which map the accumulated influence $\mathcal{H}_t^{(K)}$ to Keep, Guide, or Roll Back. We sweep each threshold independently and report Pass@1 accuracy in Fig. 6. A small

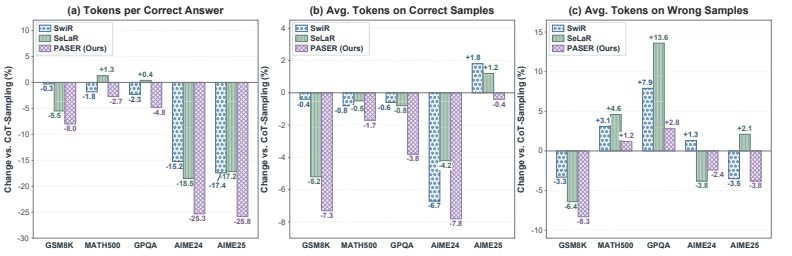

Figure 5: **Computational overhead** on `Qwen3-8B` relative to CoT: (a) tokens per correct answer and average tokens for (b) correct and (c) incorrect samples.

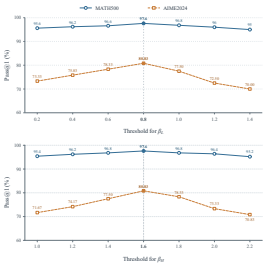

Figure 6: Ablation on reflection threshold β_L, β_H.

β_{L} over-activates guidance for weak influence, whereas a large β_{H} prevents the persistent drift from being adequately reflected. Intermediate values achieve the best balance. Therefore, we set $\beta_{\mathrm{L}} = 0.8$ and $\beta_{\mathrm{H}} = 1.6$ by default. More results are provided in Appendix C.3.

Ablations on Reasoning Strategy. To answer **Q3**, Tab. 4 examines two design choices. **Obs.❶** Removing predictive anchoring, *i.e.,* triggering reflection at global steps, yields an average performance drop of 2.71 points. This indicates that indiscriminate reflection can corrupt originally reasoning trajectories. **Obs.❷** The delayed diagnosis window presents a performance trade-off. Too short a window with fewer tokens cannot capture downstream effects and slightly hurts performance. Increasing window size exposes impacts on subsequent trajectories and improves results, yet an over-long window yields diminishing returns. Default window size of 32 tokens strikes trade-off.

Ablations on Revision Policy. We investigate the impacts of narrowing the evidence scope and reducing the reflection granularity on overall performance in Tab. 4. **Obs.❸** Removing long-term auditing and relying solely on short-term implication degrades Pass@1 from 88.99 to 86.51. This result suggests that short-window influence can preliminarily filter high-entropy tokens with no substantive impact, whereas long-range influence is critical for determining whether their impacts persist. **Obs.❹** Replacing the hierarchical reflection strategy drops Pass@1 to 85.64. Combined with the analysis results in Fig. 3, this indicates that a single reflective action approach, *i.e.,* latent reasoning switching, is not the best option.

5.4 Computational Overhead

Inference cost-effectiveness depends on both accuracy and computation. To address **Q4**, we follow SeLaR and compare **PASER** with CoT Sampling, Soft-Thinking, SeLaR, and SwiR via TPCA and average processed tokens for correct (T_c) and incorrect (T_w) samples. **PASER** achieves improved efficiency. As shown in Fig. 5, it outperforms SeLaR in TPCA by 2.5, 4.0, 5.2, 6.8 and 8.6 percentage points on GSM8K, MATH500, GPQA Diamond, AIME 2024 and AIME 2025, respectively. This advantage stems from our strategy of delayed diagnosis and graded reflection, which filters out low-impact high-entropy uncertain tokens and thereby reduces latent reasoning. These results demonstrate that **PASER** concentrates computation on critical reasoning while retaining superior overall cost-effectiveness. For an in-depth analysis, please refer to Appendix C.1.

6 Conclusion

This work revisits training-free latent reasoning from a fundamental question: whether activating the latent reasoning mode is required for uncertain high-entropy tokens? Our analyses show that high entropy identifies plausible alternatives but not necessarily immediate latent reasoning, while heterogeneous influences require different reflection scopes. We therefore propose **PASER**, which combines predictive anchoring, delayed influence diagnosis, and evidence-calibrated reflection. It preserves uncertain high-entropy tokens as recoverable anchors, evaluates the effect of activating latent reasoning at high-entropy tokens through matched suffixes, and selects reflective action among Keep, Guide, and Roll Back. Experiments across diverse tasks and model families, together with ablation studies, validate the effectiveness of delayed evidence and hierarchical revision. Overall, **PASER** offers an expert-inspired paradigm that preserves alternatives, defers verification until downstream consequences emerge, and revises only when warranted by the influence.

THE USE OF LARGE LANGUAGE MODEL

In accordance with the ICLR 2027 policy on the use of Large Language Models (LLMs), we state the following. During the preparation of this paper, we used LLM-based tools exclusively to aid in language polishing and stylistic refinement of the manuscript (*e.g.,* improving clarity, grammar, and readability). No parts of the research process itself-including problem formulation, theoretical derivation, algorithm design, experimental setup, implementation, or analysis of results-were conducted by or delegated to LLMs. Furthermore, LLMs were not used for literature retrieval, ideation, or to generate any substantive scientific content.

All technical contributions, proofs, models, and experiments presented in this work originate entirely from the authors.

ETHICS STATEMENT

We confirm adherence to the ICLR Code of Ethics. This work focuses on improving the token efficiency and reasoning performance of latent reasoning paradigms for large language models, presenting a novel evidence-based delayed reasoning framework (PASER). All experiments are conducted exclusively on publicly available open-source LLMs and standard academic benchmark datasets, all of which are used in accordance with their official licenses and ethical usage guidelines. This study did not involve human participants or the use of personal or sensitive data, thereby negating the need for an institutional ethics review.

REPRODUCIBILITY STATEMENT

To ensure that our findings can be accurately and transparently replicated, we have provided a comprehensive account of our experimental methodology. Exhaustive details concerning the models and evaluation protocols are documented in the main text and further elaborated in Appendix.

REFERENCES

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. GPT-OSS-120B and GPT-OSS-20B model card, 2025. URL https://arxiv.org/abs/2508.10925.

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

Akhiad Bercovich, Itay Levy, Izik Golan, Mohammad Dabbah, Ran El-Yaniv, Omri Puny, Ido Galil, Zach Moshe, Tomer Ronen, Najeeb Nabwani, et al. Llama-Nemotron: Efficient reasoning models, 2025. URL https://arxiv.org/abs/2505.00949.

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

Runjin Chen, Zhenyu Zhang, Junyuan Hong, Souvik Kundu, and Zhangyang Wang. SEAL: Steerable reasoning calibration of large language models for free, 2025. URL https://arxiv.org/abs/2504.07986.

Jeffrey Cheng and Benjamin Van Durme. Compressed chain of thought: Efficient reasoning through dense representations, 2024. URL https://arxiv.org/abs/2412.13171.

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, 2021. URL https://arxiv.org/abs/2110.14168.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. GLM: General language model pretraining with autoregressive blank infilling. In *ACL*, pp. 320–335, 2022.

Renyu Fu and Guibo Luo. SeLaR: Selective latent reasoning in large language models. In *ACL*, pp. 7073–7087, 2026.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. PAL: Program-aided language models. In *ICML*, pp. 10764–10799. PMLR, 2023.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. In *NeurIPS*, volume 38, pp. 41340–41391, 2026.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361, 2021.

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. Think before you speak: Training language models with pause tokens. In *ICLR*, volume 2024, pp. 27896–27923, 2024.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Yifan Wu, YK Li, et al. DeepSeek-Coder: When the large language model meets programming—the rise of code intelligence, 2024. URL https://arxiv.org/abs/2401.14196.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

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, 2024. URL https://arxiv.org/abs/2412.06769.

Alex Havrilla, Yuqing Du, Sharath Chandra Raparthy, Christoforos Nalmpantis, Jane Dwivedi-Yu, Maksym Zhuravinskyi, Eric Hambro, Sainbayar Sukhbaatar, and Roberta Raileanu. Teaching large language models to reason with reinforcement learning, 2024. URL https://arxiv.org/abs/2403.04642.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2020. URL https://arxiv.org/abs/2009.03300.

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, 2021. URL https://arxiv.org/abs/2103.03874.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *COLING*, pp. 6609–6625, 2020.

HuggingFaceH4. AIME 2024: American Invitational Mathematics Examination. Hugging Face Dataset, 2024. URL https://huggingface.co/datasets/HuggingFaceH4/aime_2024.

Naman Jain, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination-free evaluation of large language models for code. In *ICLR*, volume 2025, pp. 58791–58831, 2025.

Nan Jiang, Ziming Wu, De-Chuan Zhan, Fuming Lai, and Shaobing Lian. DART: Distilling autoregressive reasoning to silent thought. In *EMNLP*, pp. 5100–5108, 2025.

Mingyu Jin, Weidi Luo, Sitao Cheng, Xinyi Wang, Wenyue Hua, Ruixiang Tang, William Yang Wang, and Yongfeng Zhang. Disentangling memory and reasoning ability in large language models. In *ACL*, pp. 1681–1701, 2025.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *NeurIPS*, volume 35, pp. 22199–22213, 2022.

Kuang-Huei Lee, Ian Fischer, Yueh-Hua Wu, Dave Marwood, Shumeet Baluja, Dale Schuurmans, and Xinyun Chen. Evolving deeper LLM thinking, 2025. URL https://arxiv.org/abs/2501.09891.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? rigorous evaluation of large language models for code generation. In *NeurIPS*, volume 36, pp. 21558–21572, 2023.

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. https://huggingface.co/agentica-org/DeepScaleR-1.5B-Preview, 2025. Notion Blog.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-Refine: Iterative refinement with self-feedback. In *NeurIPS*, volume 36, pp. 46534–46594, 2023.

MathArena. MathArena. 2026. Hugging Face Dataset, 2026. URL https://huggingface.co/datasets/MathArena/aime_2026.

Jacob Pfau, William Merrill, and Samuel R. Bowman. Let's think dot by dot: Hidden computation in transformer language models, 2024. URL https://arxiv.org/abs/2404.15758.

Project Numina. AI-MO validation AMC dataset. https://huggingface.co/datasets/AI-MO/aimo-validation-amc, 2024.

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, 2023. URL https://arxiv.org/abs/2311.12022.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Yang Wu, et al. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. CODI: Compressing chain-of-thought into continuous space via self-distillation. In *EMNLP*, pp. 677–693, 2025.

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. In *ICLR*, volume 2026, pp. 137060–137093, 2026a.

Dachuan Shi, Hanlin Zhu, Xiangchi Yuan, Wanjia Zhao, Kejing Xia, Wen Xiao, and Wenke Lee. CopT: Contrastive on-policy thinking with continuous spaces for general and agentic reasoning, 2026b. URL https://arxiv.org/abs/2605.20075.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. OpenAI GPT-5 system card, 2025. URL https://arxiv.org/abs/2601.03267.

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, 2025. URL https://arxiv.org/abs/2502.03275.

Jihoon Tack, Jack Lanchantin, Jane Dwivedi-Yu, Andrew Cohen, Ilia Kulikov, Janice Lan, Shibo Hao, Yuandong Tian, Jason E. Weston, and Xian Li. LLM pretraining with continuous concepts. In *ICLR*, volume 2026, pp. 54214–54231, 2026.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *NAACL*, pp. 4149–4158, 2019.

Wenhui Tan, Jiaze Li, Jianzhong Ju, Zhenbo Luo, Ruihua Song, and Jian Luan. Think silently, think fast: Dynamic latent compression of LLM reasoning chains. In *NeurIPS*, volume 38, pp. 4646–4668, 2026.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. LLaMA: Open and efficient foundation language models, 2023. URL https://arxiv.org/abs/2302.13971.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain-of-thought reasoning in language models, 2022. URL https://arxiv.org/abs/2203.11171.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V. Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, volume 35, pp. 24824–24837, 2022.

Ting-Ruen Wei, Haowei Liu, Xuyang Wu, and Yi Fang. A survey on feedback-based multi-step reasoning for large language models on mathematics, 2025. URL https://arxiv.org/abs/2502.14333.

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. SoftCoT: Soft chain-of-thought for efficient reasoning with LLMs. In *ACL*, pp. 23336–23351, 2025.

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, 2025. URL https://arxiv.org/abs/2505.09388.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*, pp. 2369–2380, 2018.

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. In *NeurIPS*, volume 36, pp. 11809–11822, 2023.

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

Fangxu Yu, Lai Jiang, Haoqiang Kang, Shibo Hao, and Lianhui Qin. Flow of reasoning: Training LLMs for divergent reasoning with minimal examples, 2024. URL https://arxiv.org/abs/2406.05673.

Boyi Zeng, Shixiang Song, Siyuan Huang, Yixuan Wang, He Li, Ziwei He, Xinbing Wang, Zhouhan Lin, et al. PonderLM: Pretraining language models to ponder in continuous space. In *ICLR*, pp. 91874–91893, 2026.

Zhen Zhang, Xuehai He, Weixiang Yan, Ao Shen, Chenyang Zhao, and Xin Wang. Soft thinking: Unlocking the reasoning potential of LLMs in continuous concept space. In *NeurIPS*, volume 38, pp. 168990–169012, 2026.

Reason Forward, Reflect When Needed: Predictive Anchoring with Suffix-Evidence For Revisable LLM Reasoning — Supplementary Materials —

A Experimental Details **15**

A.1 PASER Implementation 15

A.2 Model Coverage 15

A.3 Benchmark Domains 16

A.3.1 Mathematical Reasoning 17

A.3.2 STEM Reasoning 17

A.3.3 Coding Reasoning 18

A.3.4 General Reasoning 18

A.4 Evaluation metrics 19

A.5 Experimental Configuration 19

B Latent Reasoning Intervention Analysis **19**

B.1 Do All High-Entropy Tokens Require Latent Reasoning Mode Intervention? 19

B.2 When to Trigger Latent Reasoning Mode Intervention? 24

B.3 Is the Prevailing Single-Mode Latent Reasoning Intervention Optimal? 24

B.4 Summary of Analysis 25

C Additional Experiments **25**

C.1 Efficiency Analysis 25

C.2 Semantic Analysis of Reasoning Traces 26

C.3 Ablation Analysis 28

D Discussion **29**

D.1 Different from Existing Methods 29

D.2 Future Work 30

E Case Study **30**

Algorithm 1 **PASER**: Predictive Anchoring with Suffix Evidence for Revisable Reasoning

Require: Question $x_{1:n}$, model $\mathcal{M}$, max steps $S_{\max}$, top-k, W, K, influence threshold δ, reflection thresholds $\beta_L < \beta_H$
Ensure: Answer $y_{1:m}$

Init: Embeddings $\mathbf{E}$, anchors $\mathcal{A} \leftarrow \varnothing$
for $t = 1$ to $S_{\max}$ **do**
  $p_t \leftarrow \mathrm{softmax}(\mathcal{M}(x_{1:t-1}))$ ▷ Forward pass
  $V_t \leftarrow \mathrm{TopK}(p_t, 3)$ ▷ Normalize top-3 probabilities
  $H_t \leftarrow -\frac{1}{\log 3} \sum_{v \in V_t} \bar{p}_t(v) \log \bar{p}_t(v)$ ▷ Normalized entropy
  $\widetilde{e}_t \leftarrow \sum_{v \in \mathcal{V}} p_t(v) \mathbf{E}[v]$
  if $H_t \geq 0.5$ and $|\mathcal{A}| < 1$ **then**
    Anchor instantiation; Save $\mathcal{A}_t$ ▷ **Predictive anchoring**
  end if
  Continue feeding $\mathbf{E}[x_t]$ to $\mathcal{M}$ ▷ Continue discrete reasoning
  if a candidate anchor reaches W suffix tokens **then**
    $\mathcal{I}_t^{(W)} \leftarrow \frac{1}{W} \sum_{i=1}^{W} \mathcal{G}_{t,i}$ ▷ **Short-window diagnosis**
    Discard $\mathcal{A}_t$ if $\mathcal{I}_t^{(W)} \leq \delta$; otherwise anchor as confirmed
  end if
  if a confirmed anchor reaches K suffix tokens **then**
    $\mathcal{H}_t^{(K)} \leftarrow \frac{1}{K} \sum_{i=1}^{K} \log \frac{p_\theta(x_{t+i} | q, x_{<t}, x_{t:t+i-1})}{p_\theta(x_{t+i} | q, x_{<t}; \widetilde{e}_{t:t+i-1})}$ ▷ **Long-horizon audit**
    if $\mathcal{H}_t^{(K)} \leq \beta_L$ **then**
      Keep the current trajectory; Remove $\mathcal{A}_t$ ▷ **KEEP**
    else if $\mathcal{H}_t^{(K)} \leq \beta_H$ **then**
      Redirect the next-token embedding; Release $\mathcal{A}_t$ ▷ **GUIDE**
    else
      Restore $\mathcal{A}_t$ and discard tokens after x_t ▷ **ROLLBACK**
      Feed $\widetilde{e}_t$ in place of $\mathbf{E}[x_t]$ and regenerate the suffix in latent reasoning mode
      After completing backtracking, then remove $\mathcal{A}_t$
    end if
  end if
  if $x_t =$ <EOS> **then**
    break
  end if
end for
Extract answer y from $x_{n+1:t}$
return y

A Experimental Details

A.1 PASER Implementation

Alg.1 provides the detailed implementation of **PASER**. High entropy triggers anchor storage while explicit decoding continues. Short-window diagnosis filters high-entropy tokens that have no substantive impact, and long-horizon audit selects KEEP, GUIDE, or ROLLBACK.

A.2 Model Coverage

Table 5 reports that the evaluation described in Section 5 spans eight backbones from different model families and parameter scales. For ease of exposition, we use shorthand notations for several evaluated models throughout this paper. DS-R1-Qwen-7B indicates DeepSeek-R1-Distill-Qwen-7B; DS-R1-Llama-8B indicates DeepSeek-R1-Distill-Llama-8B; DS-R1-Qwen-32B indicates DeepSeek-R1-Distill-Qwen-32B; Llama-3.1-NN-8B indicates Llama-3.1-Nemotron-Nano-8B; DSR-1.5B-Preview indicates DeepScaleR-1.5B-Preview.

Table 5: Multiple backbones across model families and scales.

Family or Grouping	Backbones
QWEN	Qwen3-1.7B (Yang et al., 2025); Qwen3-8B (Yang et al., 2025); Qwen3-32B (Yang et al., 2025)
DeepSeek-R1	DS-R1-Qwen-7B (Guo et al., 2025); DS-R1-Llama-8B (Guo et al., 2025); DS-R1-Qwen-32B (Guo et al., 2025)
Llama-Nemotron	Llama-3.1-NN-8B (Bercovich et al., 2025)
DeepScaleR	DSR-1.5B-Preview (Luo et al., 2025)

Qwen3-1.7B: A lightweight dense large language model from Alibaba's Qwen3 family with 1.7 billion parameters. It supports seamless switching between reasoning and non-reasoning modes, delivers competent multilingual and tool-use capabilities, and is tailored for edge deployment and low-compute scenarios. 🤗: https://huggingface.co/Qwen/Qwen3-1.7B.

Qwen3-8B: A medium-scale dense model in the Qwen3 series with 8 billion parameters. Its balanced architecture achieves strong performance in mathematical reasoning, code generation, and instruction following, serving as a widely adopted baseline for general LLM research. 🤗: https://huggingface.co/Qwen/Qwen3-8B.

Qwen3-32B: A high-capacity dense model of the Qwen3 family with 32 billion parameters. It exhibits state-of-the-art open-source performance in complex multi-step reasoning, long-context understanding, and agentic tasks, with a native 32K context window. 🤗: https://huggingface.co/Qwen/Qwen3-32B.

DeepSeek-R1-Distill-Qwen-32B: A reasoning-distilled language model fine-tuned on Qwen-32B using high-quality chain-of-thought data sourced from DeepSeek-R1. It delivers strong mathematical and logical reasoning performance at the 32B parameter scale, achieving a favorable balance between reasoning ability and computational overhead. 🤗: https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-32B.

DeepSeek-R1-Distill-Qwen-7B: A reasoning-distilled language model fine-tuned on Qwen-7B with high-quality chain-of-thought data from DeepSeek-R1. It substantially boosts mathematical and logical reasoning capabilities at the 7B parameter scale, offering an excellent performance-to-cost ratio. 🤗: https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-7B.

DeepSeek-R1-Distill-Llama-8B: A reasoning-optimized distilled model built upon Llama-3.1-8B, trained with DeepSeek-R1's reasoning corpus. It excels at code generation and formal reasoning tasks while maintaining full compatibility with the Llama ecosystem and toolchains. 🤗: https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Llama-8B.

Llama-3.1-Nemotron-Nano-8B: A reasoning-enhanced derivative of Meta Llama-3.1-8B-Instruct developed by NVIDIA. It undergoes multi-phase post-training including supervised fine-tuning and reinforcement learning, delivering strong performance in mathematical reasoning, tool calling, and RAG tasks with a native 128K context window. 🤗: https://huggingface.co/nvidia/Llama-3.1-Nemotron-Nano-8B-v1.

DeepScaleR-1.5B-Preview: A lightweight reasoning model fine-tuned from DeepSeek-R1-Distill-Qwen-1.5B via distributed reinforcement learning (GRPO). It surpasses O1-Preview at the 1.5B parameter scale with extended long-context reasoning capability. 🤗: https://huggingface.co/agentica-org/DeepScaleR-1.5B-Preview.

A.3 Benchmark Domains

To verify the superiority of our **PASER**, we conduct evaluations on four reasoning domains with multiple reasoning benchmarks, including mathematical reasoning (GSM8K (Cobbe et al., 2021), MATH500 (Hendrycks et al., 2021), AIME 2024-2026 (HuggingFaceH4, 2024; Yentinglin, 2025; MathArena, 2026), AMC 2023 (Project Numina, 2024), GSM-Hard (Gao et al., 2023), and Math MMLU (Hendrycks et al., 2020)); STEM reasoning (GPQA Diamond (Rein et al., 2023)); coding reasoning (HumanEval (Chen et al., 2021), HumanEvalplus (Liu et al., 2023), MBPP Austin et al.

(2021), MBPPPlus (Liu et al., 2023), LiveCodeBench (Jain et al., 2025), and LeetCode Contest (Guo et al., 2024)); and general reasoning (2WikiMultiHopQA (Ho et al., 2020), CommonsenseQA (Talmor et al., 2019), StrategyQA (Geva et al., 2021), MuSiQue (Trivedi et al., 2022) and HotpotQA (Yang et al., 2018)). For ease of exposition, we use shorthand notations for several evaluated benchmarks throughout this paper. HumanE+ indicates HumanEvalplus; MBPP+ indicates MBPPPlus; LeetCode-C indicates LeetCode Contest; LiveCode-B indicates LiveCodeBench; 2WM-HQA indicates 2WikiMultiHopQA; ComS-QA indicates CommonsenseQA; StrgyQA indicates StrategyQA.

A.3.1 Mathematical Reasoning

- **GSM8K:** A benchmark of grade-school mathematical word problems requiring several arithmetic steps expressed in natural language. Its standard release contains 7,473 training problems and 1,319 test problems, with worked solutions and final numerical answers. It assesses the ability to translate everyday descriptions into a consistent sequence of calculations. 🤗: https://huggingface.co/datasets/openai/gsm8k.
- **MATH500:** A 500-problem test subset of the MATH benchmark, used to assess competition-level mathematical problem solving. Each problem is accompanied by a worked solution, a final answer, a subject label, and a difficulty level, enabling analysis across mathematical topics and levels of challenge. 🤗: https://huggingface.co/datasets/HuggingFaceH4/MATH-500.
- **AIME 2024:** Contains 30 problems from the 2024 American Invitational Mathematics Examination, combining AIME I and AIME II. The problems require substantial mathematical derivations but have concise integer answers. 🤗: https://huggingface.co/datasets/HuggingFaceH4/aime_2024.
- **AIME 2025:** Contains 30 problems from the 2025 American Invitational Mathematics Examination. It continues the integer-answer competition format with a distinct annual problem set involving algebraic manipulation, geometric reasoning, counting, and number-theoretic arguments. 🤗: https://huggingface.co/datasets/yentinglin/aime_2025.
- **AIME 2026:** The benchmark released by MathArena consists of 30 problems from the 2026 American Invitational Mathematics Examination (AIME). It further expands the evaluation scope, covering a collection of competition problems requiring multi-step mathematical reasoning, including geometric configurations, combinatorial counting, and algebraic constraints. 🤗: https://huggingface.co/datasets/MathArena/aime_2026.
- **AMC 2023:** Covers selected problems from the 2023 AMC 12 examinations, used to test the reasoning ability of the model. 🤗: https://huggingface.co/datasets/math-ai/amc23.
- **GSM-Hard:** A numerically more demanding variant of GSM8K that replaces numbers in the original questions with larger, less common values. The public release contains 1,319 problems, each with a question, an executable reference solution, and a target answer. It probes arithmetic robustness under changed numerical values. 🤗: https://huggingface.co/datasets/reasoning-machines/gsm-hard.
- **Math MMLU:**Mathematical multiple-choice questions assess domain knowledge and problem-solving abilities across MMLU's broader collection of 57 subjects. Relevant subjects include elementary mathematics, high-school mathematics, college mathematics, abstract algebra, and statistics. Our evaluation focuses on a subset of the MMLU math partition, which incorporates 100 abstract-algebra problems into the test set. 🤗: https://huggingface.co/datasets/cais/mmlu.

.

A.3.2 STEM Reasoning

- **GPQA Diamond:** A curated subset of 198 graduate-level multiple-choice questions in biology, chemistry, and physics. The questions are written and validated by domain experts; the Diamond selection emphasizes agreement among expert validators while remaining difficult for non-experts. It evaluates scientific reasoning requiring specialized knowledge and careful discrimination among plausible alternatives. 🤗: https://huggingface.co/datasets/Idavidrein/gpqa

.

A.3.3 CODING REASONING

- **HumanEval:** Evaluates functional correctness in Python code generation through 164 hand-written programming tasks in a single test split. Each task supplies a function signature and natural-language specification, with reference code and unit tests used to check whether a generated implementation satisfies the required behavior. 🤗: https://huggingface.co/datasets/openai/openai_humaneval.
- **HumanEvalplus:** Retains the 164 programming tasks of HumanEval while strengthening correctness assessment with substantially expanded tests from EvalPlus. It probes whether generated Python functions remain correct beyond the original test cases, exposing brittle implementations that can pass a smaller suite despite failing on additional inputs. 🤗: https://huggingface.co/datasets/evalplus/humanevalplus.
- **MBPP:** Measures Python program synthesis from short natural-language descriptions of basic programming tasks. Problems include reference solutions and automated tests covering programming fundamentals and standard-library functionality. The public release contains 974 tasks overall; its full and sanitized configurations provide test splits of 500 and 257 problems, respectively. We use the test set of 257 Python programming problems. 🤗: https://huggingface.co/datasets/google-research-datasets/mbpp.
- **MBPPPlus:** Extends curated MBPP tasks with a larger collection of executable tests to assess the reliability of generated Python programs. The released v0.2.0 benchmark contains 378 tasks after removing problematic items. Its expanded tests examine functional correctness more thoroughly than the original small test suites. 🤗: https://huggingface.co/datasets/evalplus/mbppplus.
- **LiveCodeBench:** This dataset includes 279 programming problems, where the prompt requires the model to generate Python programs complying with the problem specifications and passing all test cases. 🤗: https://huggingface.co/datasets/PrimeIntellect/LiveCodeBench-v5.
- **LeetCode Contest:** This dataset evaluates models' ability to comprehend algorithmic problems and generate executable code. It consists of 180 competitive programming problems collected between July 2023 and January 2024. The evaluation is based on whether the generated solution passes the provided tests. 🤗: https://huggingface.co/datasets/TechxGenus/LeetCode-Contest.

A.3.4 GENERAL REASONING

- **2WikiMultiHopQA:** Evaluates reasoning across multiple Wikipedia passages using questions constructed with structured Wikidata relations. It includes comparison, inference, compositional, and bridge-comparison questions. We use the 2WikiMultihopQA set with 200 problems from LongBench. 🤗: https://huggingface.co/datasets/xanhho/2WikiMultihopQA.
- **CommonsenseQA:** An English multiple-choice benchmark requiring everyday knowledge to distinguish one correct answer from four distractors. Its questions are associated with ConceptNet concepts. We use the validation set of 1,221 multiple-choice QA questions. 🤗: https://huggingface.co/datasets/tau/commonsense_qa.
- **HotpotQA:** Tests question answering over multiple Wikipedia documents, including bridge questions and comparisons between entities. Sentence-level supporting facts accompany the answers, enabling evaluation of evidence selection. We used 7,405 validation samples for the test. 🤗: https://huggingface.co/datasets/hotpotqa/hotpot_qa.
- **MuSiQue:** Constructs connected reasoning problems by composing single-hop questions so that later steps depend on earlier answers. We used 2,417 validation instances for the test. 🤗: https://huggingface.co/datasets/dgslibisey/MuSiQue.
- **StrategyQA:** Evaluates yes-or-no questions whose intermediate reasoning steps must be inferred rather than read directly from the question. We used 2,290 samples with answers and decomposition structures for evaluation. 🤗: https://huggingface.co/datasets/tasksource/strategy-qa.

Table 6: Notations and definitions.

Notation	Definition	Value
$\mathbf{E}$	Token-embedding matrix	-
$\tilde{e}$	Soft embedding	-
H	Token entropy	-
$\mathcal{S}^d$	Discrete CoT trajectory	-
$\mathcal{S}^p$	Predictive soft trajectory	-
$\mathcal{A}$	Decoding state	-
$\mathcal{I}$	Short-window influence score	-
$\mathcal{H}$	Long-Horizon influence score	-
Top-p	Sampling hyperparameters	0.95
Top-k	Sampling hyperparameters	20
Min-p	Sampling hyperparameters	0.0
δ	Influence threshold	0.8
W	Short window for delayed influence diagnosis	32 tokens
K	Long horizon for longer auditing	256 tokens
β_L, β_H	Thresholds for Evidence-Calibrated Reflection	0.8, 1.6

To afford LLMs adequate reasoning space, we follow the experimental setup of `Qwen3` (Yang et al., 2025) and set the maximum generation length to $32,768$ tokens for the GSM8K, Math500, GPQA Diamond, AMC 2023, GSM-Hard, Math MMLU, HumanEval, HumanEvalplus, MBPP, MBPPPlus, LiveCodeBench, LeetCode-Contest, 2WikiMultihopQA, CommonsenseQA, HotpotQA, MuSiQue and StrategyQA benchmarks. For AIME 2024, AIME 2025 and AIME 2026, we use a maximum generation length of $38,912$ tokens. On AIME 2024, AIME 2025 and AIME 2026, we conduct the experiments with four repeated trials and report the average accuracy of all baseline methods.

A.4 Evaluation Metrics

We evaluate all baselines using **Pass@1** (↑). Unless otherwise specified, results are reported with the Transformers implementation.

A.5 Experimental Configuration

We compare **PASER** with 1) explicit baselines, including CoT Sampling and CoT Greedy, and 2) training-free latent-reasoning baselines, including Soft-Thinking, SeLaR, SwiReasoning, and CopT. For fair comparison, all baselines are reproduced with their official implementations and published hyperparameters under identical hardware conditions, *i.e.,* NVIDIA A100 GPUs. All methods share identical decoding hyperparameters: temperature of 0.6, top-p of 0.95, top-k of 20, and min-p of 0.0. For **PASER**, we set $\delta = 0.8$, $W = 32$, $K = 256$, $\beta_L = 0.8$, $\beta_H = 1.6$ as default. Specifically, we present a comprehensive review of the commonly used notations, their definitions, and values, as shown in Tab. 6.

B Latent Reasoning Intervention Analysis

B.1 Do All High-Entropy Tokens Require Latent Reasoning Mode Intervention?

To quantitatively measure the reasoning intensity of latent reasoning methods at high-entropy tokens, we conduct intervention experiments across three additional models and four types of reasoning tasks. First, to verify *whether latent reasoning intervention is required for uncertain high-entropy tokens* as adopted by existing latent reasoning paradigms, we collect high-entropy uncertain tokens using normalized top-3 entropy, *i.e.,* those with entropy exceeding 0.5, throughout reasoning trajectories and perform trajectory comparisons by applying latent reasoning intervention at each corresponding token position t. Specifically, we quantify the distribution discrepancy using the Jensen–Shannon divergence between discrete CoT trajectories $\mathcal{S}^d_{t:t+256}$ and latent reasoning trajectories $\mathcal{S}^p_{t:t+256}$ over

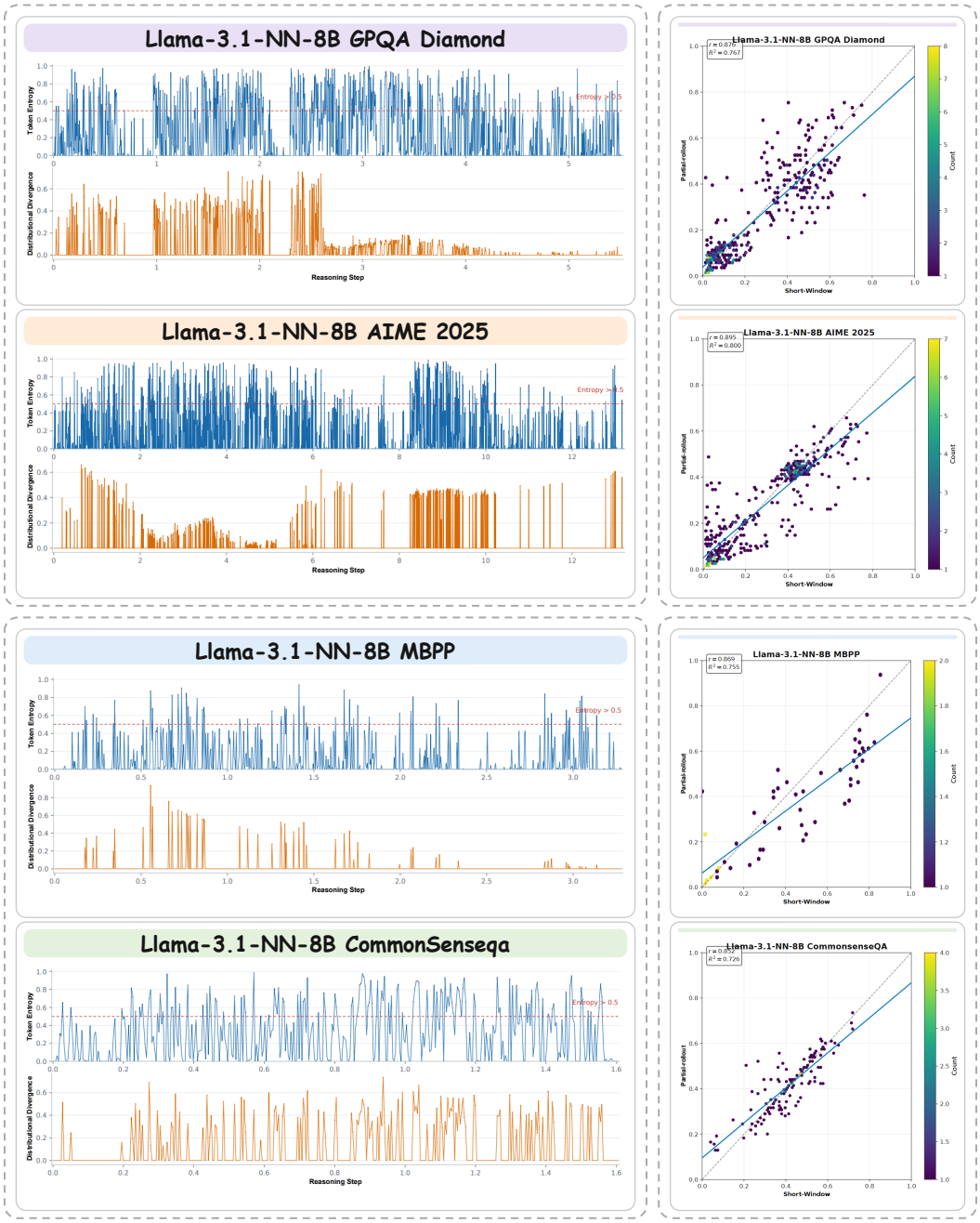

Figure 7: More preliminary diagnostic analysis on `Llama-3.1-Nemotron-Nano-8B` . ***Left:*** Token entropy and trajectory divergence. ***Right:*** Correlation between short-window influence and partial-rollout estimates across tasks in terms of r and $\mathcal{R}^2$.

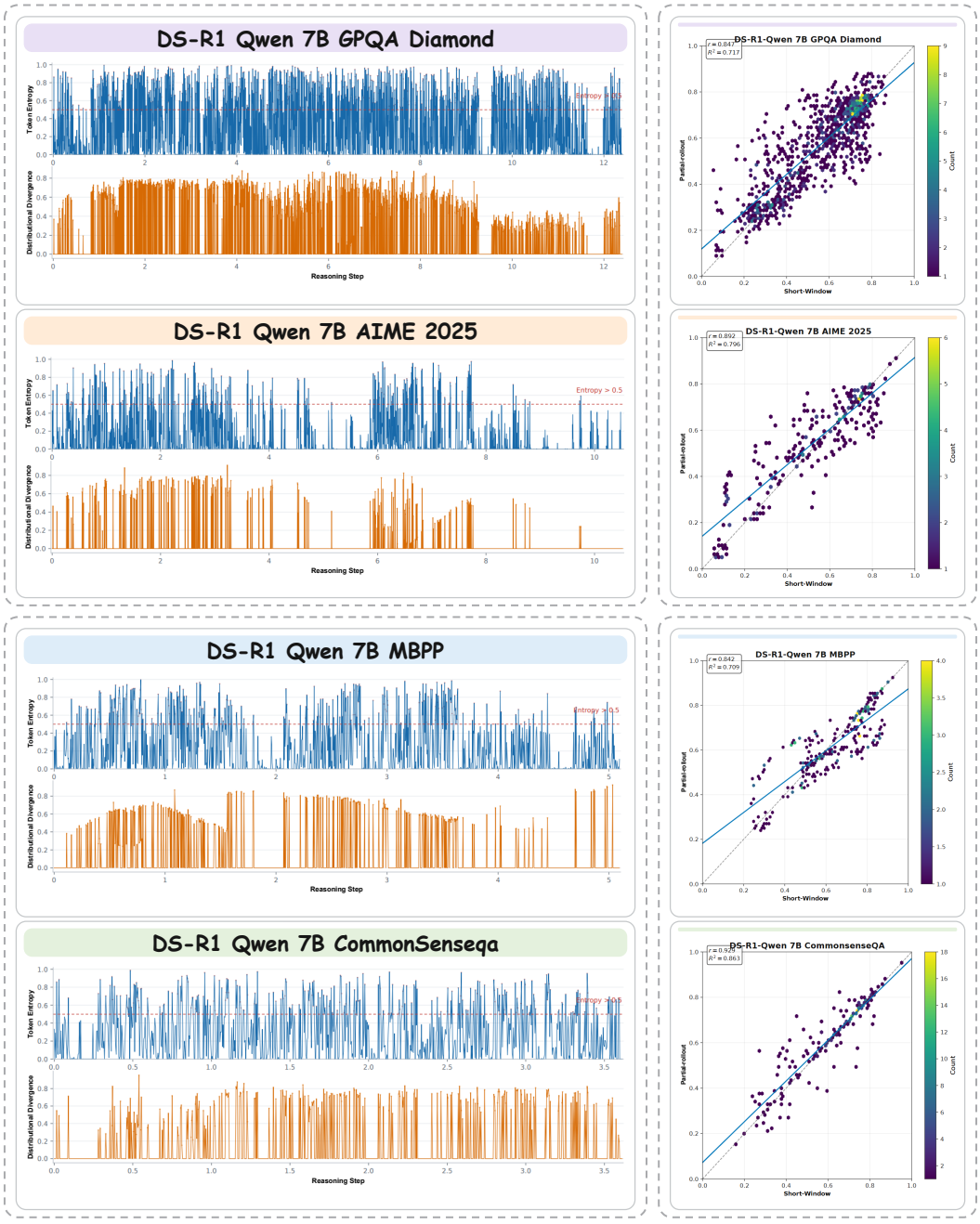

Figure 8: More preliminary diagnostic analysis on `DeepSeek-R1-Qwen-7B`. ***Left:*** Token entropy and trajectory divergence. ***Right:*** Correlation between short-window influence and partial-rollout estimates across tasks in terms of r and $\mathcal{R}^2$.

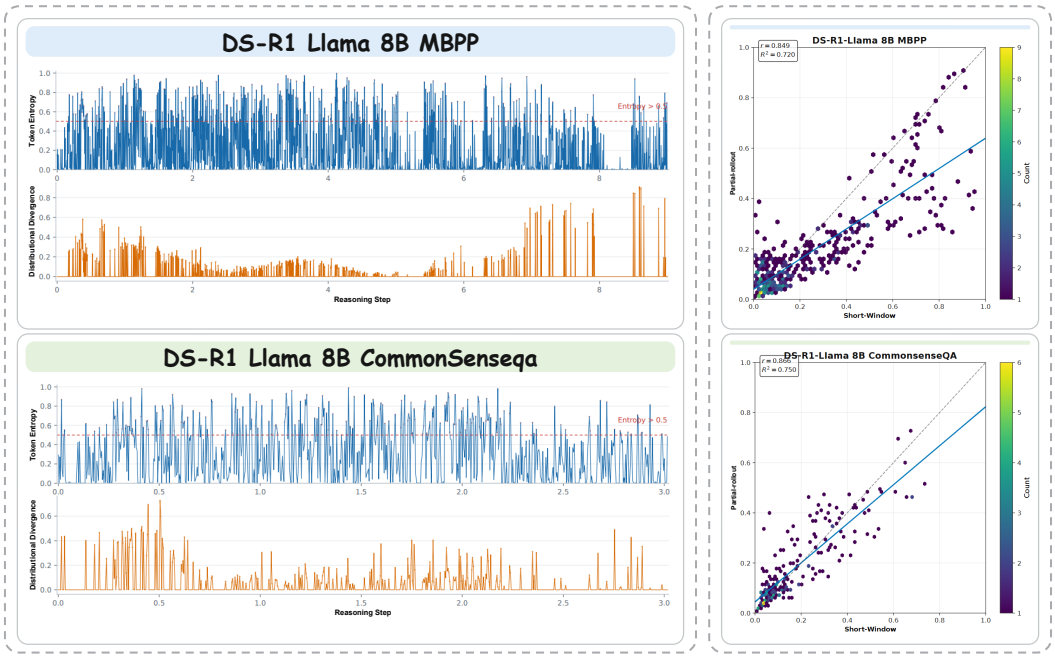

Figure 9: More preliminary diagnostic analysis on `DeepSeek-R1-Llama-8B`. ***Left:*** Token entropy and trajectory divergence. ***Right:*** Correlation between short-window influence and partial-rollout estimates across tasks in terms of r and $\mathcal{R}^2$.

the subsequent 256 tokens after high-entropy token.

$$\begin{aligned}\mathcal{D}_{t,d,p} &:= \mathrm{JSD}\left(\mathcal{S}^{d}_{t:t+256} \,\|\, \mathcal{S}^{p}_{t:t+256}\right) \\ &= H\left(\frac{\mathcal{S}^{d}_{t:t+256} + \mathcal{S}^{p}_{t:t+256}}{2}\right) - \frac{1}{2}H(\mathcal{S}^{d}_{t:t+256}) - \frac{1}{2}H(\mathcal{S}^{p}_{t:t+256}),\end{aligned} \tag{7}$$

where $H(\cdot)$ denotes Shannon entropy. As illustrated in Fig. 7, 8, and 9(a) latent reasoning intervention at certain high-entropy tokens yields limited changes to subsequent model trajectories. This indicates that high-entropy tokens can indicate potential reasoning candidates but do not necessarily require immediate latent reasoning mode.

Theoretical connection to the suffix-support gap. Fix an anchor position t and a suffix length W. Let $\mathcal{F}_{i-1}$ denote the information available before generating the i-th suffix token X_{t+i}, including the prompt, the decoding state, and the previously generated tokens. Define

$$P_i(v) = p_\theta(v \mid h^d_i), \qquad Q_i(v) = p_\theta(v \mid h^p_i), \tag{8}$$

where h^d_i and h^p_i are the discrete and predictive-soft histories, respectively. Both distributions are $\mathcal{F}_{i-1}$-measurable.

We assume that P_i and Q_i are strictly positive on the finite vocabulary and that the expectations below are finite. All logarithms are natural. All expectations are over the rollout, conditional on the fixed initial anchor state.

For each suffix position, define

$$G_{t,i} = \log \frac{P_i(X_{t+i})}{Q_i(X_{t+i})}, \qquad I^{(W)}_t = \frac{1}{W}\sum_{i=1}^{W} G_{t,i}. \tag{9}$$

The corresponding average tokenwise JSD is

$$J^{(W)}_t = \frac{1}{W}\sum_{i=1}^{W} \mathrm{JSD}(P_i, Q_i). \tag{10}$$

Here, the distributions are compared at aligned suffix positions; $J_t^{(W)}$ is the average of these tokenwise divergences. cleveref

Lemma: KL–JSD Decomposition. For any two strictly positive distributions P and Q, let $M = (P + Q)/2$. Then

$$D_{\mathrm{KL}}(P\|Q) = 2\,\mathrm{JSD}(P, Q) + 2D_{\mathrm{KL}}(M\|Q). \tag{11}$$

In particular,

$$D_{\mathrm{KL}}(P\|Q) \geq 2\,\mathrm{JSD}(P, Q). \tag{12}$$

Proof. By definition,

$$2\,\mathrm{JSD}(P, Q) = D_{\mathrm{KL}}(P\|M) + D_{\mathrm{KL}}(Q\|M). \tag{13}$$

Therefore,

$$\begin{aligned} &D_{\mathrm{KL}}(P\|Q) - 2\,\mathrm{JSD}(P, Q) \\ &= D_{\mathrm{KL}}(P\|Q) - D_{\mathrm{KL}}(P\|M) - D_{\mathrm{KL}}(Q\|M) \\ &= \sum_{v\in\mathcal{V}} P(v) \log \frac{M(v)}{Q(v)} + \sum_{v\in\mathcal{V}} Q(v) \log \frac{M(v)}{Q(v)} \\ &= \sum_{v\in\mathcal{V}} \big(P(v) + Q(v)\big) \log \frac{M(v)}{Q(v)} \\ &= 2D_{\mathrm{KL}}(M\|Q). \end{aligned} \tag{14}$$

Rearranging gives Eq.11. Eq.12 follows from the non-negativity of KL divergence.

Proposition: Suffix-support Discrepancy. Suppose the discrete rollout samples from the scored distribution, i.e.,

$$X_{t+i} \mid \mathcal{F}_{i-1} \sim P_i. \tag{15}$$

Then

$$\mathbb{E}[G_{t,i} \mid \mathcal{F}_{i-1}] = D_{\mathrm{KL}}(P_i\|Q_i). \tag{16}$$

Consequently,

$$\mathbb{E}[I_t^{(W)}] = \frac{1}{W} \sum_{i=1}^{W} \mathbb{E}[D_{\mathrm{KL}}(P_i\|Q_i)] \geq 2\mathbb{E}[J_t^{(W)}]. \tag{17}$$

Proof. Conditioning on $\mathcal{F}_{i-1}$ fixes the two next-token distributions.

$$\begin{aligned} \mathbb{E}[G_{t,i} \mid \mathcal{F}_{i-1}] &= \sum_{v\in\mathcal{V}} P_i(v) \log \frac{P_i(v)}{Q_i(v)} \\ &= D_{\mathrm{KL}}(P_i\|Q_i). \end{aligned} \tag{18}$$

Applying the law of total expectation gives

$$\begin{aligned} \mathbb{E}[I_t^{(W)}] &= \frac{1}{W} \sum_{i=1}^{W} \mathbb{E}[\mathbb{E}[G_{t,i} \mid \mathcal{F}_{i-1}]] \\ &= \frac{1}{W} \sum_{i=1}^{W} \mathbb{E}[D_{\mathrm{KL}}(P_i\|Q_i)]. \end{aligned} \tag{19}$$

Letting $M_i = (P_i + Q_i)/2$ and applying Eq. equation 11, we obtain

$$\begin{aligned} \mathbb{E}[I_t^{(W)}] &= 2\mathbb{E}[J_t^{(W)}] + \frac{2}{W} \sum_{i=1}^{W} \mathbb{E}[D_{\mathrm{KL}}(M_i\|Q_i)] \\ &\geq 2\mathbb{E}[J_t^{(W)}]. \end{aligned} \tag{20}$$

This proves the proposition. Independence between suffix tokens is not required.

Interpretation. JSD provides a symmetric diagnostic of the separation between the two predictive distributions. The realized support gap measures their relative support for the observed suffix.

B.2 When to Trigger Latent Reasoning Mode Intervention?

This observation further motivates a critical question: *when should latent reasoning intervention be triggered?* We analyze both short-window and partial-rollout trajectory distribution discrepancies induced by activating latent reasoning mode at high-entropy token and observe a general correlation between short-window trajectory variations and partial-rollout trajectory changes. Specifically, for each generated latent reasoning rollout $\mathcal{S}^p_{t:t+256}$, we compute a normalized JS divergence trace $\{J_k\}_{k=t}^{t+255}$, where $J_k \in [0,1]$ measures the distributional revision at token k. We define the remaining partial-rollout (224 tokens) mean normalized JS and the short-window (32 tokens) mean normalized JS as:

$$\bar{J}_{\text{par}}(\mathcal{S}^p_{t+32:t+256}) = \frac{1}{224}\sum_{k=t+32}^{t+255} J_k, \quad \bar{J}_{\text{short}}(\mathcal{S}^p_{t:t+32}) = \frac{1}{32}\sum_{k=t}^{t+31} J_k. \tag{21}$$

To assess whether the short window is predictive of the remaining 224-token latent reasoning rollout, we collect N valid pairs $\{(\bar{J}_{\text{par},i}, \bar{J}_{\text{short},i})\}_{i=1}^N$. We compute the Pearson correlation coefficient:

$$r = \frac{\sum_{i=1}^N (\bar{J}_{\text{short},i} - \mu_{\text{short}})(\bar{J}_{\text{par},i} - \mu_{\text{par}})}{\sqrt{\sum_{i=1}^N (\bar{J}_{\text{short},i} - \mu_{\text{short}})^2}\sqrt{\sum_{i=1}^N (\bar{J}_{\text{par},i} - \mu_{\text{par}})^2}}, \tag{22}$$

and fit a simple linear regression $\bar{J}_{\text{par},i} = a \cdot \bar{J}_{\text{short},i} + b + \varepsilon_i$. The coefficient of determination is:

$$\mathcal{R}^2 = 1 - \frac{\sum_{i=1}^N (\bar{J}_{\text{par},i} - \hat{\bar{J}}_{\text{par},i})^2}{\sum_{i=1}^N (\bar{J}_{\text{par},i} - \mu_{\text{par}})^2}, \tag{23}$$

where $\hat{\bar{J}}_{\text{par},i} = \hat{a} \cdot \bar{J}_{\text{short},i} + \hat{b}$. As shown in Fig. 7, 8, and 9(b), the corresponding Pearson correlation coefficient r stably ranges from 0.84 to 0.93, and the coefficient of determination $\mathcal{R}^2$ ranges from 0.70 to 0.86. This demonstrates that trajectory variations are locally observable and can serve as a reliable delayed indicator to verify whether activating latent reasoning at high-entropy token substantially affects subsequent reasoning dynamics.

B.3 Is the Prevailing Single-Mode Latent Reasoning Intervention Optimal?

Building on the observed impact of high-entropy tokens on subsequent reasoning trajectories, we further examine *whether the prevailing single-mode latent reasoning intervention strategy is optimal*. To answer this, we conduct grouped experiments with three distinct intervention actions across all high-entropy tokens on AIME 2025, GPQA Diamond under Qwen3-8B and MBPP, HumanEval under Qwen3-1.7B , which are stratified by the magnitude of their downstream trajectory impact. We then compare the performance of each intervention against the original baseline without intervention. Specifically, we categorize high-entropy tokens into three tiers by the impact of activating latent reasoning on downstream reasoning: weak ($\mathcal{D} \leq 0.3$), moderate ($0.3 < \mathcal{D} \leq 0.7$), and strong ($\mathcal{D} > 0.7$). The same three intervention actions are evaluated for every tier: preserving the original discrete reasoning trajectory (keep), steering the encoding of the next token to induce reflective reasoning (guide), and backtracking the full trajectory segment for reflection (roll back). Therefore, we conduct the following group experiments:

$$\text{Weak}\begin{cases}\textsc{Keep},\\ \textsc{Guide},\\ \textsc{Roll back}\end{cases}, \quad \text{Moderate}\begin{cases}\textsc{Keep},\\ \textsc{Guide},\\ \textsc{Roll back}\end{cases}, \quad \text{Strong}\begin{cases}\textsc{Keep},\\ \textsc{Guide},\\ \textsc{Roll back}\end{cases}$$

As shown in Fig. 10 and 11, no single intervention delivers universally optimal performance. For weak downstream impact, retaining the discrete original trajectory performs best. For moderate impact, guiding the next-token encoding to enable reflection yields superior performance. For strong impact, backtracking-based reflection achieves the highest performance.

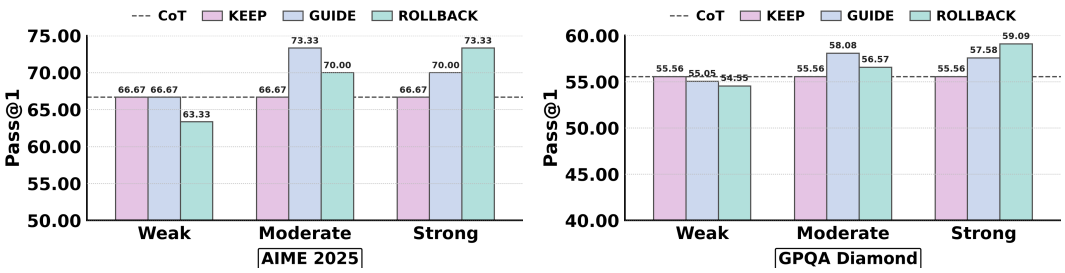

Figure 10: Intervention effectiveness of three reflective actions at high-entropy anchors under Qwen3-8B, showing the overall performance of each action across specified influence ranges.

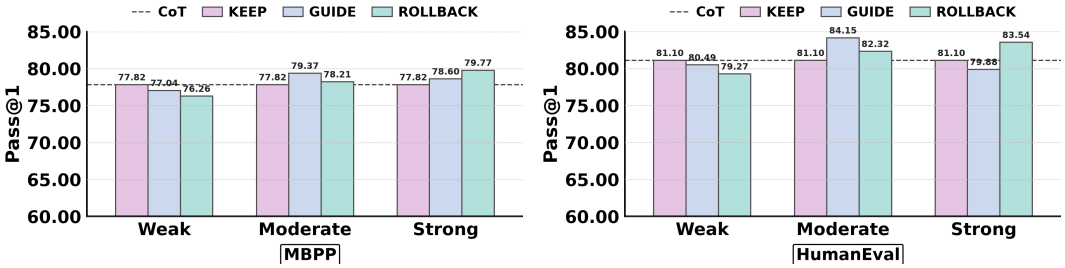

Figure 11: Intervention effectiveness of three reflective actions at high-entropy anchors under Qwen3-1.7B, showing the overall performance of each action across specified influence ranges.

B.4 Summary of Analysis

Overall, our experimental analyses yield three key findings: **Finding ❶** Uncertainty is immediately observable but not intervention-sufficient. **Finding ❷** Intervention impacts are delayed but locally observable. **Finding ❸** Revision scope should match influence scope. Based on these findings, it prompts us to consider a new reasoning paradigm, shifting the potential reasoning process from immediate latent intervention to evidence-based delayed revision reasoning.

C Additional Experiments

C.1 Efficiency Analysis

We comprehensively evaluate the overall efficiency of **PASER** by comparing its inference performance against other baseline methods. Specifically, we report three key metrics: tokens per second (TPS), time per request (TPR) and GPU memory usage. We conduct our efficiency evaluation on the AIME24 benchmark, as summarized in Fig. 12.

- **Token generation efficiency:** In terms of TPS, **PASER** achieves superior per-token generation efficiency compared with existing training-free latent reasoning models.
- **Inference latency acceleration:** In terms of TPR, **PASER** reduces inference time relative to prior training-free latent reasoning methods, *e.g.,* SwiR and SeLaR.
- **GPU memory usage:** As shown in Tab. 7 and 8, our method introduces no additional GPU-memory overhead. On the contrary, it reduces GPU memory consumption compared with SwiR and SeLaR.

In summary, **PASER** achieves strong performance in TPS, TPR and GPU memory usage, while maintaining better Pass@1 accuracy. This benefit stems from our delayed verification and hierarchical revision strategy, which filters low-impact yet high-entropy uncertain tokens and eliminates unnecessary latent reasoning steps.

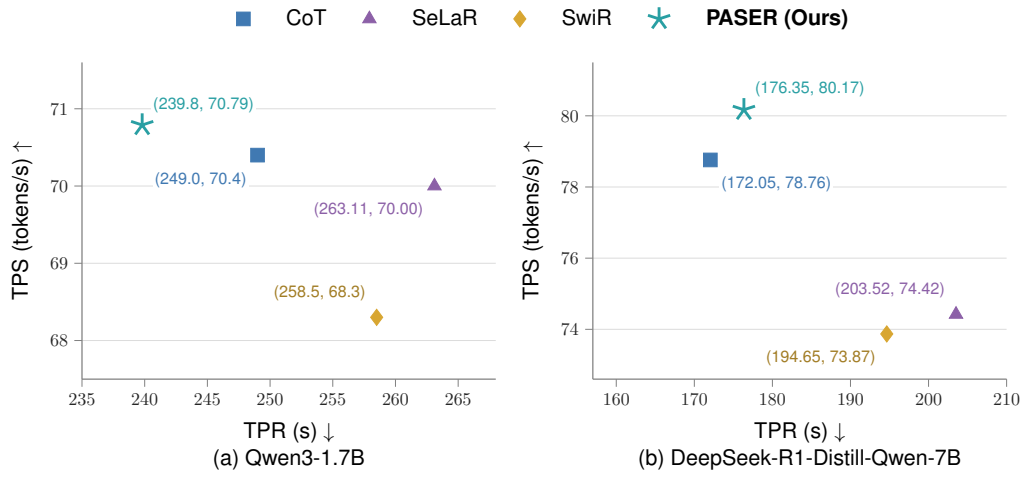

Figure 12: Efficiency analysis on TPS and TPR under Qwen3-1.7B and DeepSeek-R1-Distill-Qwen-7B.

Table 7: GPU Memory with batch size 1 on Qwen3-1.7B.

Method	GPU Memory (GB)	Pass@1
SeLaR	21.59	46.67
SwiR	22.88	49.17
PASER	20.88	52.50

Table 8: GPU Memory with batch size 1 on DeepSeek-R1-Distill-Qwen-7B.

Method	GPU Memory (GB)	Pass@1
SeLaR	18.56	46.67
SwiR	19.64	50.83
PASER	18.37	57.50

C.2 Semantic Analysis of Reasoning Traces

Furthermore, we utilized the *Transition* and *Reflection* vocabulary list introduced by SEAL (Chen et al., 2025) to conduct a systematic semantic analysis of the reasoning trajectory of Qwen3-8B.

$\mathcal{V}_{\mathrm{R}}$ = { `wait`, `verify`, `make sure`, `hold on`, `think again`, `'s correct`, `'s incorrect`, `let me check`, `seems right` }
$\mathcal{V}_{\mathrm{T}}$ = { `alternatively`, `think differently`, `another way`, `another approach`, `another method`, `another solution`, `another strategy`, `another technique` }

For each method, let d_i denote the reasoning trace for question i, with N questions in total. Given a marker set $\mathcal{V} \in \{\mathcal{V}_{\mathrm{R}}, \mathcal{V}_{\mathrm{T}}\}$, let $c(w, d_i)$ be the number of occurrences of marker w in d_i, and define $C_{i,\mathcal{V}} = \sum_{w \in \mathcal{V}} c(w, d_i)$. Matching is case-insensitive, and each occurrence of a multiword phrase counts as one marker. Let L_i be the number of English words in d_i. We compute:

$$\mathrm{WC}_{\mathcal{V}} = \frac{1}{N} \sum_{i=1}^{N} C_{i,\mathcal{V}}, \tag{24}$$

$$\mathrm{TF}_{\mathcal{V}} = \frac{1000}{N} \sum_{i=1}^{N} \frac{C_{i,\mathcal{V}}}{L_i}, \tag{25}$$

$$\text{TF-IDF}_{\mathcal{V}} = \frac{1000}{N} \sum_{i=1}^{N} \sum_{w \in \mathcal{V}} \frac{c(w, d_i)}{L_i} \mathrm{idf}(w). \tag{26}$$

To obtain shared IDF weights, we pool the reasoning traces from all compared methods within the same model–dataset pair, treating each trace as a separate document. For a corpus containing D



Figure 13: Performance comparison across models and benchmarks. Each sector compares one baseline method with **PASER**. Scores are normalized by **PASER**'s score on the same benchmark under the same model.

Table 9: Semantic statistics of *Transition* and *Reflection* vocabularies across different methods and models, and corresponding Pass@1. For each method on each model, we report the word count, term frequency (TF), and TF-IDF score for both vocabularies.

Method	Reflection			Transition			Pass@1
	Word Count	TF	TF–IDF	Word Count	TF	TF–IDF	
`Qwen3-8B`							
SeLaR	34.4	16.5	18.9	11.8	6.1	7.6	96.80
SwiR	35.0	16.8	19.3	11.8	6.2	7.8	97.60
PASER	29.9	15.1	17.2	9.9	5.5	6.7	97.60
`DeepSeek-R1-Distill-Qwen-7B`							
SeLaR	26.5	12.5	16.4	6.0	3.2	5.3	92.80
SwiR	22.7	12.2	15.4	9.2	3.8	6.1	90.60
PASER	21.0	10.6	13.6	5.4	3.2	5.0	94.40

Table 10: Ablation study of next-token steering.

Method	MATH500	GSM8K	AIME24	AIME25	HumanEval	MBPP	Avg.
PASER (Full)	97.60	95.98	80.83	77.50	96.95	95.33	90.70
- w/o Influence-Calibrated Contrastive Guidance	96.80	94.84	78.33	74.17	95.73	94.16	89.01
Influence-Calibrated Contrastive Guidance	97.60	95.98	80.83	77.50	96.95	95.33	90.70
Contrastive Regularization	97.20	95.22	77.50	75.83	95.12	94.55	89.24

documents, we use the smoothed inverse document frequency:

$$\mathrm{idf}(w) = \ln\left(\frac{D+1}{\mathrm{df}(w)+1}\right) + 1, \tag{27}$$

where $\mathrm{df}(w)$ is the number of documents containing at least one occurrence of w. Word Count measures the average number of marker occurrences per question. TF additionally accounts for trace length, while TF–IDF also weights markers by their prevalence across documents.

As shown in Tab. 9, we quantify the changes in model semantics under SeLaR, SwiR, and our method. For ease of reading, both TF and TF-IDF scores are normalized by multiplying by 1000. Observations reveal that compared to SeLaR and SwiR, our method reduce some reflection and switching markers while maintaining a higher accuracy rate. This indicates that our strategy of delayed verification and graded reflection can filter out those low-impact high-entropy uncertain tokens, thereby reducing latent reasoning to reduce redundant thinking and retain necessary reasoning behaviors.

C.3 Ablation Analysis

Ablation study of next-token steering. To investigate the effectiveness of the proposed next-token steering strategy, we compare the full **PASER** with variants that remove the Influence-Calibrated Contrastive Guidance or replace it with Contrastive Regularization in SeLaR. As shown in Tab. 10, removing our steering strategy decreases the average performance from 90.70 to 89.01. This suggests that directly preserving the original next-token prediction without influence-aware guidance is insufficient for handling moderately influential reasoning states. Moreover, replacing our strategy with Contrastive Regularization used in SeLaR reduces the average score to 89.24. These results indicate that our strategy explicitly calibrates the steering strength according to the estimated downstream impact and selectively amplifies token preferences supported by the predictive history, thereby achieving better steering.

Ablation study on influence threshold δ. To analyze how the threshold affects the selection of influential anchors and subsequent reasoning trajectories, we conduct ablation experiments on Qwen3-8B across five standard benchmarks: GSM8K, Math500, AIME2024, AIME2025, and HumanEval. As illustrated in Tab. 11, an excessively small threshold fails to effectively identify anchors with substantial downstream impacts, resulting in degraded performance. Conversely, an overly large threshold retains very few valid influential anchors, which also leads to suboptimal results. A

Table 11: Ablation study on δ.

δ	GSM8K	MATH 500	AIME 2024	AIME 2025	HumanEval	**Average**
0.2	94.16	95.60	71.67	71.67	95.12	85.64
0.4	94.77	97.20	75.00	71.67	95.12	86.75
0.6	95.53	97.20	79.17	75.00	95.73	88.53
0.8	95.98	97.60	80.83	77.50	96.95	89.77
1.0	95.38	96.60	78.33	72.50	96.34	87.83
1.2	94.69	96.40	74.17	71.67	95.12	86.41
1.4	94.01	95.80	72.50	70.00	94.51	85.36

Table 12: Ablation study on K.

K	GSM8K	MATH 500	AIME 2024	AIME 2025	HumanEval	**Average**
64	94.39	95.80	70.83	70.00	94.51	85.11
128	95.14	96.40	75.00	71.67	95.12	86.67
192	95.75	97.20	79.17	75.83	95.73	88.74
256	95.98	97.60	80.83	77.50	96.95	89.77
320	95.00	96.60	80.83	75.00	95.12	88.51
384	94.47	96.20	75.00	71.67	93.90	86.25
448	94.31	95.60	72.50	70.00	94.51	85.38

Table 13: Results are reported as mean accuracy over four independent random seeds on Qwen3-8B.

Method	STEM Domain	Coding Domain		Avg.
	GPQA Diamond	HumanEval	MBPP	
SwiR	59.97	94.97	94.36	83.10
SeLaR	60.60	89.33	92.32	80.75
PASER	62.25	96.19	94.94	84.46

Table 14: Results are reported as mean accuracy over four independent random seeds on DeepSeek-R1-Distill-Qwen-7B.

Method	STEM Domain	Coding Domain		Avg.
	GPQA Diamond	HumanEval	MBPP	
SwiR	38.76	89.17	79.77	69.23
SeLaR	36.49	72.10	76.75	61.78
PASER	42.42	88.56	84.05	71.68

moderate threshold achieves the best trade-off across all tasks. Accordingly, we set the default δ threshold to 0.8.

Ablation study on long-horizon K. To further investigate the impact of the long-horizon size K on reflection, we conduct ablation experiments using Qwen3-8B on five benchmarks. As shown in Tab. 12, a small K limits the number of tokens available for reflection and fails to effectively trigger delayed verification of reasoning anchors, leading to inferior performance. Meanwhile, an excessively large K also yields suboptimal results. A moderate K achieves the optimal trade-off across all tasks. Consequently, we set the default long-horizon size K to 256 tokens.

Multi-seed performance. We repeat the evaluation over four independent random seeds on GPQA Diamond, HumanEval, and MBPP. As shown in Tabs. 13 and 14, PASER maintains the better average performance on both Qwen3-8B and DeepSeek-R1-Distill-Qwen-7B, achieving average accuracies of 84.46 and 71.68, respectively.

D Discussion

D.1 Different from Existing Methods

Existing training-free latent reasoning methods typically adopt a fixed intervention rule: latent reasoning is immediately activated at high-entropy uncertain tokens during generation. This paradigm treats all high-entropy positions equally and applies identical intervention logic, without differentiating how activating latent reasoning at each anchor token affects downstream reasoning paths. Such a uniform intervention strategy overlooks the heterogeneous effects of high-entropy tokens on subsequent reasoning trajectories, inevitably introducing unnecessary computational overhead and limiting improvements in reasoning performance.

In contrast, our study reveals three key insights that fundamentally depart from this conventional design. ❶ High entropy alone is insufficient as a triggering condition: not all high-entropy tokens require latent reasoning intervention. Activating latent reasoning at some high-entropy tokens exerts only limited influence over subsequent reasoning paths. ❷ Although the impact of activating latent reasoning at high-entropy tokens may only emerge after several subsequent reasoning steps, its downstream influence can be effectively estimated from a local trajectory window, enabling us to predict the influence of each anchor without costly rollouts. ❸ There exists no universally optimal reflective strategy. A single fixed latent reasoning switching strategy cannot deliver optimal performance.

Building on these findings, we propose the **PASER** framework, a revisable reasoning paradigm. Unlike the prevalent single-mode switching adopted in prior latent reasoning methods, **PASER** first

quantifies the impact of activating latent reasoning at high-entropy token on the downstream trajectory and then dynamically selects appropriate reflective actions. This design applies backtracking reflection only to tokens with strong downstream impacts, while retaining lightweight discrete generation for weakly influential anchors. These distinguish our method from existing training-free latent reasoning baselines including Soft-Thinking, SeLaR, and SwiR.

D.2 Future Work

We identify two promising directions for future research.

(I) We will explore training-based extensions of **PASER**, particularly through reinforcement learning, to learn policies for when and how to revise reasoning. Such extensions could jointly optimize reasoning quality and computational efficiency while preserving the principle of evidence-guided revision.

(II) We will investigate **PASER** in broader reasoning settings, including interactive and tool-augmented tasks. Incorporating more diverse sources of feedback could extend predictive anchoring and delayed revision to more complex reasoning processes.

E Case Study

To further demonstrate the advantages of our method, we present qualitative comparisons on two representative problems in abstract algebra and coding. For example, in the abstract algebra case, SwiR repeatedly revisits the relationship between homomorphic images and quotient groups but retains the incorrect conclusion that both statements are false. In contrast, the **PASER** output contains an erroneous intermediate assumption: that S_3 admits a surjective homomorphism onto C_3. However, subsequent reasoning examines the kernel required by this assumption. Such a homomorphism would require a normal subgroup of order two, which S_3 does not possess. This contradiction leads to the rejection of the proposed counterexample. The output ultimately invokes the first isomorphism theorem and the natural projection onto quotient groups to obtain the correct answer. This case shows how later reasoning can provide diagnostic evidence that overturns an earlier judgment.

Together, these cases highlight key differences in the reasoning processes of the two methods: assessing candidate solutions before committing and revising assumptions when subsequent evidence contradicts them. These observations are consistent with **PASER**'s design of preserving the revisability of earlier reasoning and using subsequent evidence to guide reflection. In particular, the algebra example demonstrates a substantive update to an earlier judgment, beyond repeated reconsideration that leaves the original error unresolved.

Case1: DeepSeek-R1-Distill-Llama-8B with SwiR **Failure, not passed**

►Problem:
Statement 1: Every homomorphic image of a group G is isomorphic to a factor group of G.
Statement 2: The homomorphic images of a group G are the same (up to isomorphism) as the factor groups of G.

A. True, True B. False, False C. True, False D. False, True

►Output:
Step 1: Evaluating Statement 1
Starting with Statement 1: "Every homomorphic image of a group G is isomorphic to a factor group of G."
Wait, no, actually image is a group, but not necessarily a factor group. So, in example, take G and the group, H is any group, then image is a subgroup of H, which not H is a quotient group, G, which image isn't necessarily a factor group of G. So I 1 is false.

[...]

Step 2: Reconsidering the Correspondence
Wait, the 2 is: that the homomorphic images of the same up the groups, to isomorphism. So, that each homomorphic image H of G, H is isomorphic to some factor group of G, and conversely, every factor group of G is a homomorphic image of G.
Wait, is that the case?
For, first the homomorphic image H of G, H is a factor group of G? But, because we thought before, H is a subgroup of some group, but not necessarily a factor group.

[...]

But, for G factor group G/N, is G a homomorphic image of So? Because is, is there a surjective homomorphism from G to G/N?Yes, the natural projection is: $G \to G/N$ is ajective. and its image is G/N. So, every factor group is a homomorphic image.But is converse is not true. as a homomorphic images can be any subgroup that G maps map onto, which necessarily factor groups.
So, the hom of homomorphic images includes all factor groups, but also more groups. So, the hom of homomorphic images is larger than the set of factor groups.
So, Statement 2 is false.
Therefore, Statement 1 is false, Statement 2 is false. So, the are false. So, the is B.

Step 3: Rechecking the Non-Abelian Case
[...]
Wait, so if the, for non-abelian groups, the every image a factor group? For example, take $G = S_3$, [...] Then the image is S_3,, which is not a factor group of S_3, S_3 is not abelian, and its only factor subgroups are A_3 and trivial.

[...]

So, so abelian groups, every subgroup is normal, so any hom is a factor group. So, in that case, Statement 1 would true for abelian groups.

[...]

Wait for abelian case, it's true. but since non-abelian case it's false. So, since general, Statement's false.
So, Statement 1 is false, Statement 2 is false false.
Wait, the are false, so the answer is B.

►Final Answer:

$\boxed{B}$ ✗

Correct Answer: A (True, True)

Case1: DeepSeek-R1-Distill-Llama-8B with PASER Success, passed

►Problem:
Statement 1: Every homomorphic image of a group G is isomorphic to a factor group of G.
Statement 2: The homomorphic images of a group G are the same (up to isomorphism) as the factor groups of G.

A. True, True B. False, False C. True, False D. False, True

►Output:
Step 1: Reconsidering the Quotient-Group Interpretation
Wait, factor groups are defined as G/N for some normal subgroup N. So if the image is G/N, it's a factor group. But if you have another homomorphism to a different group H, the image is a subgroup of H, but unless H is a quotient group, it doesn't have to be a factor group of G.
Wait, but isn't the image always a quotient group? Or is it that the image is always a quotient group of G, but not necessarily a factor group of G?
Wait, no. Wait, the image is always isomorphic to a quotient group of G. Because for any homomorphism $\phi : G \to H$, the image is isomorphic to $G/\ker\phi$. So, the image is a quotient group of G. But that quotient group is a factor group only if $\ker\phi$ is normal in G.

[...]

Step 2: Checking a Proposed Counterexample
Wait, can S_3 map onto C_3?
Yes, because S_3 has elements of order 3, so you can have a surjective homomorphism from S_3 to C_3. So the image is C_3, which is a cyclic group of order 3.

[...]

Wait, so that contradicts my earlier conclusion.

[...]

Wait, so if $\phi : S_3 \to C_3$ is a surjective homomorphism, then the image is C_3, which is isomorphic to $S_3/\ker\phi$. So, what is $\ker\phi$? Since ϕ is surjective, $\ker\phi$ is a normal subgroup of S_3. So, $S_3/\ker\phi \cong C_3$. So, what is $\ker\phi$?
Since S_3 has order 6, and C_3 has order 3, the kernel has order 2. So, $\ker\phi$ is a subgroup of order 2 in S_3. But S_3 has normal subgroups only of order 1, 3, and 6. There are no normal subgroups of order 2 in S_3. Therefore, $\ker\phi$ cannot be a normal subgroup. So, is this possible?
Wait, if ϕ is a surjective homomorphism from S_3 to C_3, then $\ker\phi$ must be a normal subgroup of S_3. But S_3 has no normal subgroup of order 2, so such a ϕ cannot exist. Therefore, S_3 cannot have a surjective homomorphism to C_3. Therefore, the only homomorphic images of S_3 are S_3, C_2, and the trivial group.

[...]

Wait, so in that case, the homomorphic images of S_3 are only S_3, C_2, and the trivial group, which are exactly the factor groups of S_3. So, maybe my initial confusion was wrong.

Step 3: Applying the First Isomorphism Theorem
[...]
Wait, perhaps I should refer to the first isomorphism theorem, which states that the image of a group under a homomorphism is isomorphic to a quotient group, which is a factor group. So, every homomorphic image is a factor group.
Therefore, Statement 1 is true.
And since for every factor group, we can define a homomorphism whose image is that factor group, the sets are the same, so Statement 2 is also true.
Therefore, the answer is A.

►Final Answer:

$\boxed{A}$ ✓

Correct Answer: A (True, True)

Case2: Qwen3-32B with SwiR Failure, not passed

►Problem:

```
def Strongest_Extension(class_name, extensions):
    """
    You will be given the name of a class (a string) and a list of extensions. The extensions are to
      be used to load additional classes to the class. The strength of the extension is as follows:
      Let CAP be the number of the uppercase letters in the extension's name, and let SM be the
      number of lowercase letters in the extension's name, the strength is given by the fraction CAP -
       SM.
    You should find the strongest extension and return a string in this format: ClassName.
      StrongestExtensionName.
```

```
    If there are two or more extensions with the same strength, you should choose the one that comes
      first in the list.
    For example, if you are given "Slices" as the class and a list of the extensions: ['SErviNGSliCes
      ', 'Cheese', 'StuFfed'] then you should return 'Slices.SErviNGSliCes' since 'SErviNGSliCes' is
      the strongest extension (its strength is -1).
    Example:
    for Strongest_Extension('my_class', ['AA', 'Be', 'CC']) == 'my_class.AA'
    """
```

►Output:
Let me think about the steps:

1. Iterate over each extension in the extensions list.
2. For each extension, count the number of uppercase letters (CAP) and lowercase letters (SM).
3. Calculate strength as CAP - SM.
4. Keep track of the extension with the highest strength. If a new extension has a higher strength than the current maximum, update. If equal strength, since we pick the first, we don't replace if it's equal but later in the list.

[...]

Potential issues:
What about characters that are not letters? The problem examples don't have that, but the problem statement says 'extension's name' but doesn't specify. The problem says that CAP and SM are the number of uppercase and lowercase letters. So code counts only letters? Well, the code as written counts any character that is uppercase or lowercase. But according to the problem's examples, perhaps the extensions are only letters.
But according to the code, any character in the extension name that is not uppercase or lowercase (like numbers, symbols) would not contribute to CAP or SM. But according to the problem's description, perhaps all characters are letters. But since the problem didn't specify, the code should handle any string. However, the problem's examples use letters only, so perhaps it's safe to assume that.

[...]

Now, code-wise, the helper function for calculating CAP and SM is correct?
Yes. For each character in the extension name, check if uppercase. If yes, add to CAP. Else, add to SM. Because a lowercase letter is considered as SM. But what if there are other characters (like numbers, underscores, etc)? Well, the problem didn't consider them, but according to the code as per problem's examples, perhaps the code is correct.

►Final Answer:

```python
def Strongest_Extension(class_name, extensions):
    max_strength = None
    max_ext = None
    for ext in extensions:
        cap = 0
        sm = 0
        for c in ext:
            if c.isupper():
                cap += 1
            else:
                sm += 1
        strength = cap - sm
        if max_ext is None or strength > max_strength:
            max_strength = strength
            max_ext = ext
    return f"{class_name}.{max_ext}"
```

►Verification:
Input: `Strongest_Extension('Sp', ['671235', 'Bb'])`

Returned: `Sp.Bb` ✗
Expected: `Sp.671235`

Case2: Qwen3-32B with PASER Success, passed

►Problem:

```
def Strongest_Extension(class_name, extensions):
    """
    You will be given the name of a class (a string) and a list of extensions. The extensions are to
      be used to load additional classes to the class. The strength of the extension is as follows:
      Let CAP be the number of the uppercase letters in the extension's name, and let SM be the
      number of lowercase letters in the extension's name, the strength is given by the fraction CAP -
       SM.
    You should find the strongest extension and return a string in this format: ClassName.
      StrongestExtensionName.
    If there are two or more extensions with the same strength, you should choose the one that comes
      first in the list.
    For example, if you are given "Slices" as the class and a list of the extensions: ['SErviNGSliCes
      ', 'Cheese', 'StuFfed'] then you should return 'Slices.SErviNGSliCes' since 'SErviNGSliCes' is
      the strongest extension (its strength is -1).
    Example:
    for Strongest_Extension('my_class', ['AA', 'Be', 'CC']) == 'my_class.AA'
    """
```

►Output:
So the steps are:

1. Iterate over each extension in extensions, in order.
2. For each extension, calculate its strength.

3. Keep track of the maximum strength found so far and the corresponding extension name.

But since we need to pick the first one in case of a tie, when a new extension has the same strength as the current max, we don't replace it.

[...]

So the key part is to compute the strength correctly for each extension. Now, how to compute the CAP and SM for an extension name. For a given string (extension name), loop through each character, check if it's uppercase or lowercase. In Python, for a character c, c.isupper() returns True if it's uppercase, c.islower() for lowercase.
So for each character in the extension name:

```
CAP = sum(1 for c in extension if c.isupper())
SM = sum(1 for c in extension if c.islower())
strength = CAP - SM
```

So that's how to compute the strength for each extension.

[...]

Another sample input: `Strongest_Extension('my_class', ['AA', 'Be', 'CC'])` → `'my_class.AA'`.
Let's compute strengths. `'AA'` → CAP=2, SM=0 → 2-0=2. `'Be'` → CAP=1 (B), SM=1 (e) → 0. `'CC'` → CAP=2, SM=0 → 2. So the first one is chosen since it comes first. So the result is `my_class.AA`. So the code would select the first occurrence of the maximum strength.

►Final Answer:

```
def Strongest_Extension(class_name, extensions):
    max_strength = float('-inf')
    best_extension = None
    for ext in extensions:
        cap = 0
        sm = 0
        for c in ext:
            if c.isupper():
                cap += 1
            elif c.islower():
                sm += 1
        strength = cap - sm
        if strength > max_strength:
            max_strength = strength
            best_extension = ext
    return f"{class_name}.{best_extension}"
```

►Verification:
Input: `Strongest_Extension('Sp', ['671235', 'Bb'])`

Returned: `Sp.671235` ✓
Expected: `Sp.671235`