Think Earlier, Not Longer: Prompt Optimization via Reducing Unhealthy Exploration

Ling-I Wu[1], Minyu Chen[1], Jingyang Li[1], Xi Chang[2], Guoqiang Li[1]*
[1]Shanghai Jiao Tong University, Shanghai 200240, China
[2]Shanghai Polytechnic University, Shanghai 201209, China
{edithwuly, minkow, lijjjjj, li.g}@sjtu.edu.cn, changxi@sspu.edu.cn

Abstract

While large language models exhibit strong reasoning capabilities, prior work shows that their performance can be further enhanced by encouraging greater exploration. However, existing approaches overlook the presence of unhealthy exploration that increases exploration-related token usage without contributing to effective problem-solving. In this work, we show that prompt ambiguity can artificially prolong early-stage exploration, manifested as an elevated and delayed early-stage entropy peak. Although this uncertainty may be gradually resolved as reasoning progresses, reflected in the eventual convergence of the late-stage entropy peak, it does not meaningfully improve accuracy or self-consistency and instead substantially reduces reasoning efficiency. Motivated by these observations, we propose an entropy-dynamics-aware prompt optimization framework that trains a lightweight optimizer to generate concise clarifications. These clarifications aim to reduce ambiguity-induced early-stage uncertainty while preserving the model's reasoning capabilities. Extensive experiments across multiple models, reasoning budgets, and benchmarks demonstrate that our approach consistently improves reasoning efficiency by up to 52%, by reducing unhealthy exploration without sacrificing accuracy.

1 Introduction

Recent work on large language model (LLM) reasoning has proposed a variety of techniques to improve reasoning performance by encouraging greater exploration during generation, such as test-time scaling (Muennighoff et al., 2025; Snell et al., 2025) and parallel reasoning (Wang et al., 2022; Yao et al., 2023; Besta et al., 2024). However, these approaches implicitly assume that increased exploration is uniformly beneficial, overlooking the presence of unhealthy exploration that substantially increases token usage without contributing to effective downstream problem-solving.

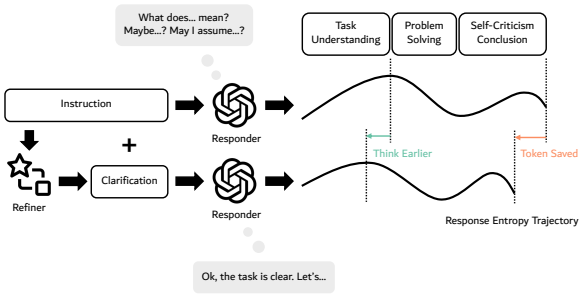


Figure 1: Early-stage exploration induced by prompt ambiguity can be effectively mitigated through concise clarification, thereby improving reasoning efficiency.

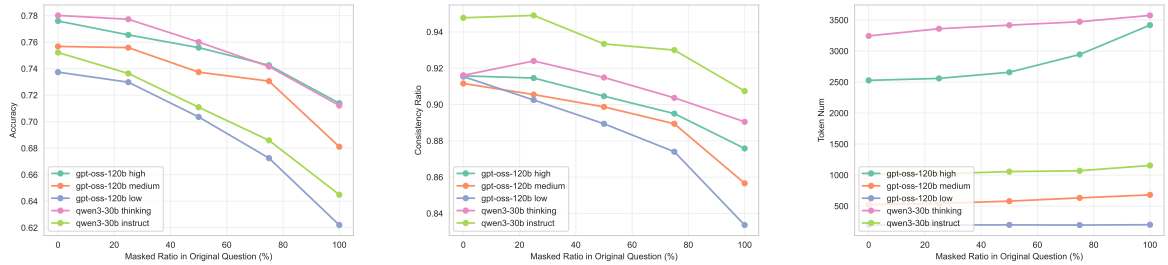
To expose this phenomenon, we conduct controlled prompt ambiguity experiments that isolate the effect of task underspecification while keeping the underlying reasoning task unchanged. Specifically, we systematically vary prompt ambiguity by masking technical terms with their abbreviations. Across multiple models and reasoning budgets, increased prompt ambiguity leads to a substantial inflation in token usage, while no improvement in either accuracy or self-consistency.

By decomposing token-level entropy trajectories into three stages: task understanding (early), problem solving (middle), and self-criticism (late), we observe that the early-stage entropy peak becomes both elevated and delayed. In contrast, the late-stage entropy peak remains highly convergent. This observation indicates that exploration induced by prompt ambiguity is progressively resolved rather than directly advancing problem-solving.

Motivated by these observations, we propose a prompt optimization framework that trains a lightweight optimizer via multi-turn reinforcement learning to generate concise clarifications for the original prompt. We design an entropy-peak–based reward that encourages the generated clarifications

*Corresponding Author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 16577–16592



(a) Accuracy (b) Self-Consistency (c) Token Usage

Figure 2: Accuracy, consistency, and token usage of GPT-oss-120B under three levels of reasoning effort, Qwen3-30B-Instruct and Qwen3-30B-Thinking as the mask ratio increases.

to both reduce and advance the early-stage entropy peak, while preserving the late-stage entropy peak. In other words, our approach enables the model to understand the task more efficiently, without harming its intrinsic multi-step reasoning capability.

Our experimental results show that the proposed prompt optimization method substantially improves reasoning efficiency, achieving up to 52.9% gains on in-domain benchmarks. Moreover, our method demonstrates strong generalization ability, improving reasoning efficiency by up to 38.54% on out-of-domain benchmarks.

In summary, the main contributions of our work are as follows:

- We conduct controlled prompt ambiguity experiments, revealing that ambiguity-driven unhealthy exploration artificially extends the early reasoning stage without improving problem-solving performance.
- We propose an entropy-dynamics-aware prompt optimization framework that trains a lightweight optimizer to generate concise clarifications, selectively reducing unhealthy early-stage uncertainty while preserving healthy reasoning.
- Extensive experiments across multiple models, reasoning budgets, and both in-domain and out-of-domain benchmarks demonstrate that our method consistently improves reasoning efficiency without sacrificing accuracy.

2 Entropy Dynamics Observation

2.1 Controlled Prompt Ambiguity

To elicit unhealthy exploration, we manipulate prompt ambiguity rather than reasoning depth. When key terms or constraints in a prompt are underspecified, the model expends additional tokens on task interpretation and linguistic clarification before committing to a concrete reasoning trajectory.

Specifically, we sample 1,000 instances from MMLU-Pro (Wang et al., 2024) and use GPT-5[1] to identify technical terms, which are then replaced by their initialisms (e.g., page fault → PF). By varying the proportion of masked terms, we systematically control the degree of prompt ambiguity while keeping the reasoning task itself unchanged.

We evaluate GPT-oss-120B (Agarwal et al., 2025a) under three levels of reasoning effort (high, medium, and low) across five mask ratios (0%, 25%, 50%, 75%, and 100%). The temperature is set to 0.7, and for each question we perform 16 rollouts. As shown in Figure 2, increasing prompt ambiguity consistently degrades accuracy and self-consistency, while increasing token usage.

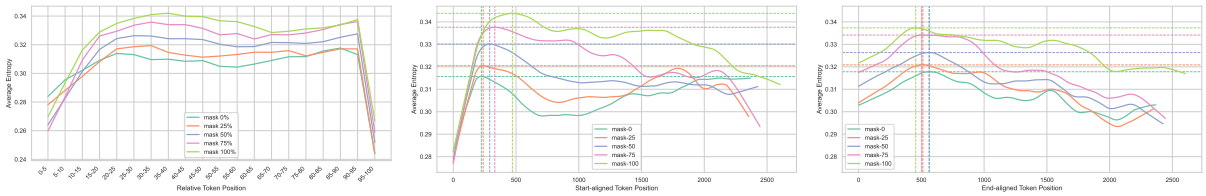
2.2 Token-Level Entropy Trajectories

To better understand how prompt ambiguity affects the generation process, we analyze token-level entropy trajectories under medium and high reasoning effort. Figures 3 and 4 report average token-level entropy curves over both relative positions and absolute positions.

Across settings, the entropy trajectory consistently exhibits a three-stage structure:

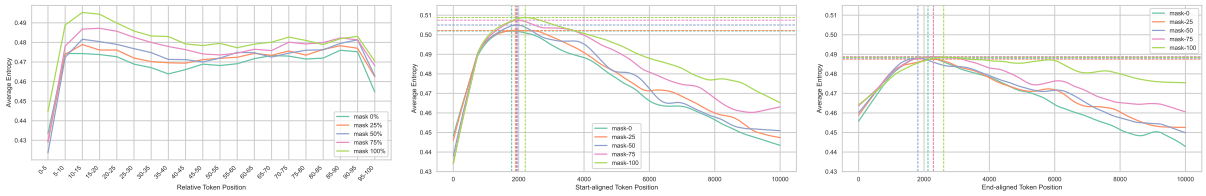
- Early Stage: Entropy increases sharply, reflecting the reasoning model's prompt interpretation and identification;
- Middle Stage: Entropy gradually decreases as the model commits to a specific reasoning trajectory and focuses on problem solving;

[1]https://cdn.openai.com/gpt-5-system-card.pdf



(a) Relative Position (b) Start-aligned Position (c) End-aligned Position

Figure 3: Token-level entropy of GPT-oss-120B (medium reasoning effort) across five mask ratios under three alignment schemes: relative position, which normalizes token indices by response length using 5% bins; start-aligned position, which uses absolute token indices aligned at the first generated token; and end-aligned position, which uses absolute token indices aligned at the final generated token (distance to termination).



(a) Relative Position (b) Start-aligned Position (c) End-aligned Position

Figure 4: Token-level entropy of GPT-oss-120B (high reasoning effort).

- Final Stage: Entropy shows a mild increase followed by a sharp drop, presenting reasoning verification, and answer finalization.

2.3 Early-stage Uncertainty

A salient feature of the entropy trajectory is the early-stage entropy peak, which marks the transition from prompt interpretation to committed problem solving. In this phase, the model is actively forming a task understanding and may explore multiple plausible interpretations and solution directions. When the prompt is underspecified, this early-stage exploration is augmented by interpretive uncertainty induced by prompt ambiguity, manifesting as an elevated and delayed entropy peak.

As shown in Figures 3b and 4b, increasing the mask ratio consistently elevates the early-stage entropy peak and shifts it to later token positions. Increasing reasoning effort partially compensates for this increased early-stage uncertainty, not by resolving it efficiently, but by doing so at the cost of substantial token expansion. Under medium reasoning effort, the difference in early-stage peak height between mask ratio 0 and 100 is 0.027, accompanied by an increase of 335 generated tokens. Under high reasoning effort, the peak height difference is reduced to 0.006, yet the corresponding token increase rises to 437.

2.4 Late-Stage Convergence

As shown in Figures 3c and 4c, despite substantial differences in early-stage uncertainty, entropy trajectories exhibit a contrasting pattern near generation termination. When responses are aligned by their termination points, entropy dynamics in the late stage become increasingly similar across different mask ratios.

Notably, higher reasoning effort leads to stronger late-stage convergence. Under medium reasoning effort, the difference in late-stage peak height across mask ratios is 0.019, representing a compression to approximately 70% of the corresponding early-stage difference. Under high reasoning effort, this difference further collapses to 5.1×10^{-5}, amounting to only 0.8% of the early-stage peak difference. This trend mirrors the observation that higher reasoning effort also exhibits greater robustness to the accuracy and self-consistency degradation introduced by prompt ambiguity.

Importantly, this convergence does not imply the disappearance of all early-stage exploration. As the model commits to a task interpretation, the ambiguity-induced unhealthy components component is progressively resolved, while structurally necessary healthy components persists and is carried forward into the late stage, where it supports answer verification and finalization.

3 Prompt Optimization

Building on the observations in Section 2, we design a framework that trains a lightweight prompt optimizer using entropy-dynamics-guided multi-turn reinforcement learning. The optimizer learns to generate a clarification that helps the reasoning model more accurately interpret the prompt. By reducing the prevalence of ambiguity-induced unhealthy exploration, this approach effectively mitigates unnecessary token expenditure and improves reasoning efficiency. An overview of the training process is shown in Figure 5.

3.1 Problem Formulation

To clarify our objective, we distinguish between two qualitatively different forms of exploration. Unhealthy exploration refers to uncertainty-driven token expenditure in the early stage of generation, where the model has not yet formed a clear understanding of the task. This form of exploration manifests as elevated and delayed early-stage entropy. Importantly, although this ambiguity-induced exploration is largely self-resolving as the model commits to a task interpretation, it is inefficient, contributing directly to response length inflation without improving reasoning quality.

In contrast, healthy exploration occurs after the task has been sufficiently specified, when the model actively explores multiple plausible reasoning paths. This form of exploration is a desirable component of complex reasoning and is not eliminated as task interpretation stabilizes. Instead, it also contributes to answer verification, and ultimately leading to more reliable conclusions.

Our goal is to improve reasoning efficiency by selectively reducing unhealthy exploration while preserving healthy exploration. To this end, we introduce a lightweight prompt optimizer P_θ that augments the original prompt x with a concise clarification:

$$x' = x \oplus P_\theta(x), \tag{1}$$

The clarification is restricted to clarifying ambiguous aspects of the task formulation and must not reveal solution steps, or answer choices.

3.2 Entropy Peak Reward

The prompt optimizer is guided solely by entropy dynamics, which reflect the internal reasoning behavior of the model, without any correctness-based supervision. This design ensures that the optimizer focuses on improving how the reasoning model commits to a solution, rather than what answer it produces.

Given an augmented prompt x', the frozen reasoning model generates a response of length T. At each generation step t, the model produces a predictive distribution $p_t(\cdot \mid x', y_{<t})$, whose token-level entropy is defined as

$$e_t = -\sum_v p_t(v) \log p_t(v). \tag{2}$$

Early Stage Peak Reward The primary objective of prompt optimization is to reduce early stage peak reward and encourage earlier commitment to a reasoning trajectory.

We focus on the early portion of the response, corresponding to the prompt interpretation phase. Let $T^{\text{early}} = \lfloor \alpha T \rfloor$, where $\alpha \in (0, 1)$ is a fixed ratio. We apply a smoothing operator $\mathcal{S}(\cdot)$ to the entropy sequence and define the early-stage entropy peak as

$$\begin{aligned} p^{\text{early}} &= \arg \max_{1 \le t \le T^{\text{early}}} \mathcal{S}(e_t), \\ h^{\text{early}} &= \max_{1 \le t \le T^{\text{early}}} \mathcal{S}(e_t), \end{aligned} \tag{3}$$

where p^{early} denotes the peak position and h^{early} the peak height. Intuitively, p^{early} measures how long the model remains uncertain before committing to a reasoning trajectory, while h^{early} quantifies the severity of uncertainty.

To obtain robust estimates, we perform K independent rollouts of the reasoning model. For each rollout k, we extract a peak pair $\left(p^{\text{early},(k)}, h^{\text{early},(k)}\right)$ and aggregate them as

$$\begin{aligned} \bar{p}^{\text{early}} &= \frac{1}{K} \sum_{k=1}^{K} p^{\text{early},(k)}, \\ \bar{h}^{\text{early}} &= \frac{1}{K} \sum_{k=1}^{K} h^{\text{early},(k)}. \end{aligned} \tag{4}$$

For each prompt, baseline statistics $\bar{p}_0^{\text{early}}$ and $\bar{h}_0^{\text{early}}$ are precomputed using the original prompt. The effect of optimization is then measured by

$$\begin{aligned} \Delta p^{\text{early}} &= \bar{p}_0^{\text{early}} - \bar{p}^{\text{early}}, \\ \Delta h^{\text{early}} &= \bar{h}_0^{\text{early}} - \bar{h}^{\text{early}}. \end{aligned} \tag{5}$$

Finally, we define the early-stage peak reward as

$$R^{\text{early}} = \mathbb{I}!\left(\Delta p^{\text{early}} > 0 \wedge \Delta h^{\text{early}} > 0\right), \tag{6}$$

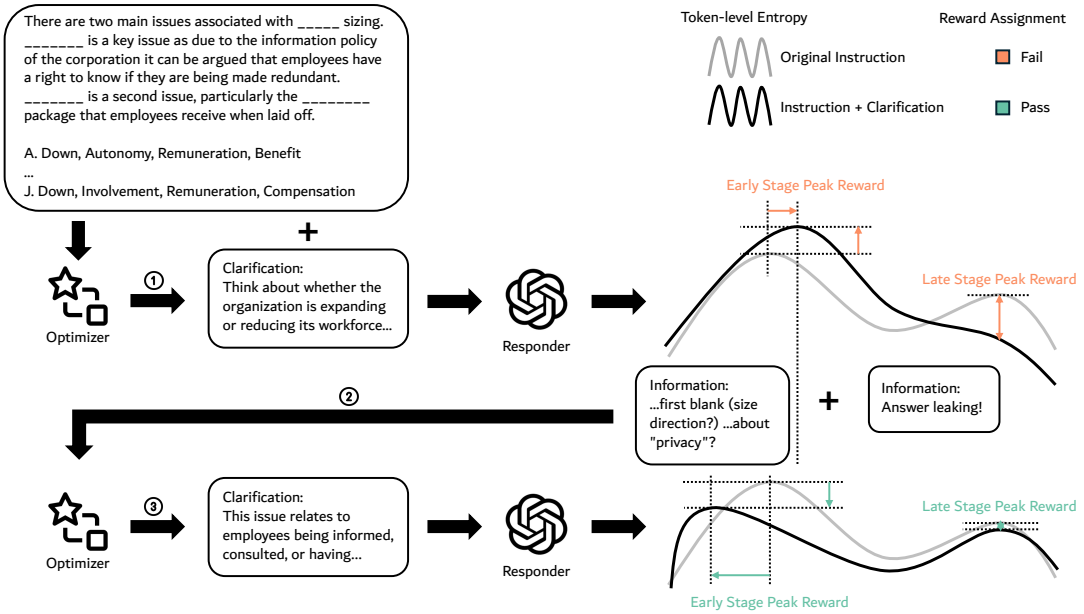


Figure 5: Overview of the proposed prompt optimizer training framework. The optimizer is first given the original prompt and generates a initial clarification (Step 1). This clarification is then appended to the original prompt and provided to the Responder. If the resulting response passes the entropy peak check, the clarification is considered qualified. Otherwise, feedback information is presented to the optimizer (Step 2), which then generates a refined clarification in the next turn (Step 3). This process repeats until a qualified clarification is produced or the maximum number of turns is reached. Finally, the last-turn clarification reward is used to update the optimizer.

which assigns a positive reward only when prompt optimization successfully reduces both the timing and magnitude of the early-stage entropy peak.

Late Stage Peak Reward The late-stage peak reward serves as a regularization signal rather than a primary optimization objective, ensuring that healthy exploration in later stages is preserved.

To enforce this constraint, we additionally monitor entropy behavior in the termination neighborhood. Let $T^{\text{late}} = \lceil (1-\gamma)T \rceil$, where $\gamma \in (0,1)$ specifies the fraction of tokens considered as the late stage. Using the same smoothing operator $\mathcal{S}(\cdot)$, we define the late-stage entropy peak height for a single rollout as

$$h^{\text{late}} = \max_{T^{\text{late}} \le t \le T} \mathcal{S}(e_t). \tag{7}$$

To obtain robust estimates, we again perform K independent rollouts. For each rollout k, we extract a late-stage peak height $h^{\text{late},(k)}$ and aggregate them as

$$\bar{h}^{\text{late}} = \frac{1}{K} \sum_{k=1}^{K} h^{\text{late},(k)}. \tag{8}$$

As with the early-stage statistics, we precompute a baseline late-stage peak height $\bar{h}_0^{\text{late}}$ using the original prompt. The effect of optimization is then measured by

$$\Delta h^{\text{late}} = \bar{h}_0^{\text{late}} - \bar{h}^{\text{late}}. \tag{9}$$

Finally, we define the late-stage peak reward as

$$R^{\text{late}} = \mathbb{I}\left(\Delta h^{\text{late}} < \Delta h^{\text{early}}\right), \tag{10}$$

which assigns a positive reward only when that the late-stage entropy peak remains closer to that of the original prompt than the early-stage peak.

3.3 Multi-Turn Reinforcement Learning

Directly optimizing early-stage uncertainty using single-step reinforcement learning is challenging, as a scalar reward provides limited information. To address this limitation, we introduce a multi-turn reinforcement-learning framework in which the optimizer iteratively refines its clarifications based on the feedback derived from the reasoning model until a qualified clarification is produced or the maximum number of turns is reached.

In each training episode, the optimizer first generates an initial clarification, which is appended to the original prompt and passed to the reasoning model. The resulting response is then analyzed to determine whether it satisfies the entropy peak reward criteria. If the early-stage entropy peak

Model	Effort	Method	In-domain Benchmarks						Out-of-domain Benchmarks					
			MMLU-Pro			SuperGPQA			BBH			MedQA		
			Acc.↑	Tok.↓	Eff.↑	Acc.↑	Tok.↓	Eff.↑	Acc.↑	Tok.↓	Eff.↑	Acc.↑	Tok.↓	Eff.↑
GPT-oss-120B	High	Pure	**80.60**	2383.35	3.38	49.40	4500.31	1.09	84.40	1496.53	5.63	**91.10**	1372.63	6.63
		Early-Stop	58.70	**1285.58**	4.56	28.30	**2069.67**	1.36	57.70	**865.16**	6.66	85.70	1164.39	7.36
		EvoPrompt	80.10	2943.89	2.72	**50.40**	5635.07	0.89	-	-	-	-	-	-
		Ours	80.30	1552.30	**5.17**	49.80	3432.54	**1.45**	85.40	1094.72	**7.80**	90.20	**1095.40**	**8.23**
	Medium	Pure	79.30	566.69	13.99	47.20	1055.71	4.47	**84.40**	421.18	20.03	88.00	292.65	30.07
		Early-Stop	74.50	508.60	14.64	39.30	772.76	5.08	65.70	**306.84**	21.41	88.00	291.64	30.17
		EvoPrompt	79.90	654.63	12.20	49.50	1020.56	4.85	-	-	-	-	-	-
		Ours	**80.60**	**472.89**	**17.04**	**48.60**	**725.61**	**6.69**	83.90	371.77	**22.56**	**88.30**	**261.31**	**33.79**
	Low	Pure	75.70	216.22	35.01	42.70	294.56	14.49	83.30	206.78	40.28	**85.40**	100.86	84.67
		Early-Stop	74.80	213.46	35.04	41.60	281.57	14.77	72.20	**181.01**	39.88	85.30	100.79	84.63
		EvoPrompt	**76.60**	245.61	31.18	44.90	315.00	14.25	-	-	-	-	-	-
		Ours	76.20	**186.15**	**40.93**	**45.40**	**252.01**	**18.01**	**83.50**	188.35	**44.33**	84.90	**100.04**	**84.86**
Qwen3-30B	Thinking	Pure	81.40	3316.56	2.45	**53.30**	4765.22	1.11	**85.90**	1798.92	4.77	**84.90**	1982.31	4.28
		Early-Stop	54.70	**2138.95**	2.55	33.30	**2669.79**	1.24	40.20	**1012.05**	3.97	83.40	1945.48	4.28
		EvoPrompt	79.60	3670.03	2.16	51.80	4111.81	1.25	-	-	-	-	-	-
		Ours	**81.70**	2491.31	**3.27**	53.10	3795.22	**1.39**	85.30	1406.23	**6.06**	84.80	**1795.78**	**4.72**
	Instruct	Pure	76.50	1147.30	6.66	52.50	1986.15	2.64	83.30	564.12	14.76	**73.40**	55.71	131.75
		Early-Stop	52.10	**464.20**	**11.22**	32.00	**858.16**	**3.72**	54.00	**339.47**	15.90	72.70	**51.44**	141.32
		EvoPrompt	78.60	1704.12	4.61	49.50	2474.67	2.00	-	-	-	-	-	-
		Ours	**80.10**	875.32	9.15	**52.60**	1497.04	3.51	**83.40**	463.99	**17.97**	73.20	51.68	**141.64**

Table 1: Accuracy (Acc.), token usage (Tok.) and reasoning efficiency (Eff.) of reasoning models evaluated on both in-domain and out-of-domain benchmarks. Bolded values indicate the best performance for each reasoning model.

conditions are not met, the optimizer receives the reasoning model's generation tokens preceding the early-stage entropy peak. These tokens indicate where the model begins to exhibit uncertainty. If the late-stage entropy peak conditions are not satisfied, the optimizer instead receives feedback signals that do not leak information that could directly influence the reasoning process of the model. The optimizer is encouraged to generate a more targeted clarification in the subsequent turn.

The multi-turn interaction is introduced solely as a training scaffold to stabilize optimization and improve credit assignment. At inference time, the prompt optimizer operates in a single-shot manner, generating at most one clarification per prompt and introducing no additional interaction overhead.

4 Experiments

4.1 Setup

Training Setting We adopt Qwen3-4B-Instruct as a lightweight prompt optimizer. Based on observations from our entropy dynamics experiments, we set the early-stage ratio to 0.5 and the late-stage ratio to 0.3. We reuse 1,000 samples from MMLU-Pro and additionally sample 1,000 instances from SuperGPQA (Du et al., 2025) to construct the training set. These datasets span a wide range of professional domains, enabling the prompt optimizer to learn robust clarification strategies.

The target reasoning models used during training include GPT-oss-120B under low, medium, and high reasoning effort settings, as well as both the thinking and instruct variants of Qwen3-30B (Yang et al., 2025). For entropy peak reward computation, we set the reasoning model temperature to 0.7 and perform 16 rollouts to estimate the average peak position and peak height. We approximate entropy using only the top-5 token log probabilities.

Evaluation Setting We set the reasoning temperature to 0.0 and perform a single rollout to ensure reproducibility. The prompt template used for evaluation is provided in Appendix C. To quantify reasoning efficiency, we compute:

$$\text{Reasoning Efficiency} = \frac{\text{Accuracy}}{\text{Token Usage}} \tag{11}$$

4.2 Empirical Results

In-domain Evaluation The main baselines we compare against are Pure, Early-Stop (Sharma and Chopra, 2025) and EvoPrompt (Tong et al., 2025). For the in-domain benchmarks, we follow the original implementation of the Early-Stop method and EvoPrompt method. Specifically, for the former, the entropy threshold is computed as the mean entropy of each reasoning model on the training sets of MMLU-Pro and SuperGPQA, and the patience parameter is set to 50. For the latter, we obtain optimized prompts by applying the evolutionary algorithm on the same training sets from MMLU-Pro and SuperGPQA.

TU Token	Enrichment	SC Token	Enrichment
interpret	4.01	double-check	2.54
what does	1.85	verify	1.58
means	1.53	mismatch	1.47
assume	1.38	recompute	1.24

Table 2: Task-understanding (TU) tokens enriched around early-stage entropy peaks and self-criticism (SC) tokens enriched around late-stage entropy peaks.

As shown in Table 1, our method achieves the highest reasoning efficiency on most reasoning models, without harming, and in some cases even slightly improving, accuracy. In contrast, the Early-Stop method also demonstrates effectiveness in improving reasoning efficiency, but it carries the risk of reducing accuracy, as it directly truncates subsequent tokens once the entropy remains below a threshold for a predefined number of consecutive steps. Meanwhile, EvoPrompt primarily focuses on improving accuracy. Although it successfully enhances accuracy for most reasoning models, it often increases token usage, which in turn leads to lower reasoning efficiency in most cases.

Out-of-domain Evaluation To assess generalization, we further evaluate our method on out-of-domain benchmarks, including BBH (Suzgun et al., 2023) and MedQA (Jin et al., 2021). For the Early-Stop method, we estimate the entropy threshold by averaging the thresholds obtained for each reasoning model on MMLU-Pro and SuperGPQA, while keeping the patience parameter fixed at 50. In contrast, EvoPrompt requires a development set for prompt optimization and is therefore not applicable in this out-of-domain setting.

As shown in Table 1, our method achieves a significant improvement in reasoning efficiency across different reasoning models, indicating that the learned clarification strategies are robust to substantial domain shifts. We further report the total token consumption and the corresponding reasoning efficiency in Appendix D.

5 Analysis

5.1 Tokens Associated with Entropy Peaks

To quantify the association between entropy peaks and specific token types, we measure token enrichment using a normalized occurrence ratio. Specifically, for each token w, we define

$$\mathrm{Enrich}(w) = \frac{C_{\mathrm{seg}}(w)/N_{\mathrm{seg}}}{C_{\mathrm{other}}(w)/N_{\mathrm{other}}}, \qquad (12)$$

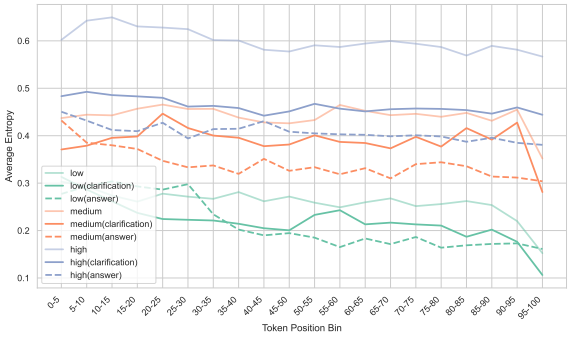


Figure 6: Relative-position entropy trajectories of GPT-oss-120B under high, medium and low reasoning effort, with clarification and answer conditioning.

where $C_{\mathrm{seg}}(w)$ and N_{seg} denote the number of occurrences of w and the total number of tokens within entropy-peak-adjacent segments, and $C_{\mathrm{other}}(w)$ and N_{other} are defined analogously for all remaining positions in the response.

As shown in Table 2, early-stage entropy peaks are selectively associated with tokens indicating task understanding and semantic clarification. In contrast, late-stage entropy peaks are characterized by the enrichment of self-criticism and verification-related tokens. This apparent enrichment of characteristic functional tokens provide empirical evidence that entropy peaks are not merely statistical artifacts, but are systematically aligned with distinct cognitive functions, further supporting our stage decomposition of the reasoning process.

5.2 Entropy Dynamics with Answer Conditioning

To investigate how answer leakage influences a model's reasoning capability, we compare the token-level entropy dynamics of GPT-oss-120B with clarification-given prompting and answer-conditioned prompting, in which the answer is explicitly provided. As shown in Figure 6, under medium and high reasoning effort, answer-conditioned prompting leads to a sharp drop in entropy at the early stage, indicating a loss of exploratory behavior. Moreover, the late stage entropy peak almost entirely disappears. In contrast, when clarifications are provided, the model's overall entropy trajectory remains much closer to that observed with the original prompting. This suggests that our prompt optimization method preserves the model's intrinsic reasoning capability while avoiding answer leakage. Human evaluation in Appendix E further support this conclusion.

Model	Effort	Method	Early	Middle	Late
GPT-oss-120B	**High**	Pure	482.36	1564.49	336.48
		Ours	270.51	1435.80	245.97
	Medium	Pure	143.63	334.04	89.02
		Ours	67.69	304.69	70.49
Qwen3-30B	**Thinking**	Pure	881.13	1815.56	618.86
		Ours	445.55	1661.84	462.64
	Instruct	Pure	298.67	670.41	178.21
		Ours	222.51	651.29	161.50

Table 3: Average token usage of GPT-oss-120B and Qwen3-30B across early, middle and late stages.

Under low reasoning effort, the behavior differs slightly. For both the original and clarification-given settings, entropy decreases in the early stage, reflecting the model's limited exploratory capacity under constrained reasoning budgets. However, with answer-conditioned prompting, entropy initially increases before sharply dropping. We hypothesize that this entropy increase reflects a transient conflict between the model's internal reasoning trajectory and the externally imposed answer-conditioned conclusion. This further highlights that answer conditioning fundamentally alters the model's reasoning dynamics.

5.3 Stage-wise Token Usage Ablation

To examine whether the improvement in reasoning efficiency arises from reducing unhealthy exploration in the early stage, we conduct a token-usage stage ablation analysis by measuring the average number of tokens generated during the early, middle, and late stages of generation.

As shown in Table 3, our method consistently reduces early-stage token usage across all reasoning models. This suggests that the proposed strategy enables the model to converge to a stable semantic interpretation more efficiently, avoiding redundant reformulations and excessive exploratory phrasing at the beginning of generation.

In contrast, only modest reductions are observed in the middle and late stages. This indicates that sufficient exploration capacity for multi-step reasoning is preserved. Such behavior aligns well with our design goal of avoiding over-regularization during the main reasoning phase.

5.4 Comparison with Outcome-Based Reward

To further examine whether the gains of our method arise from modeling entropy dynamics rather than directly optimizing task outcomes, we compare against two outcome-based alternatives: (1) a prompt optimizer trained with a joint accuracy and response length reward (**Opt$_{\text{Acc+Len}}$**), and (2) direct reinforcement learning with verifiable rewards applied to the reasoning model itself (**RLVR**).

Method	MMLU-Pro		MedQA	
	Acc.	Eff.	Acc.	Eff.
Pure	79.30	13.99	88.00	30.07
Opt$_{\text{Acc+Len}}$	82.40	21.26	79.80	23.75
Opt$_{\text{Ours}}$	**80.60**	**17.04**	**88.30**	**33.79**

Table 4: Comparison with Acc+Len-based prompt optimizer on GPT-oss-120B (medium effort).

Comparison with Acc+Len Optimizer. We first train a prompt optimizer for GPT-oss-120B under medium reasoning effort using a joint reward

$$R = \lambda \cdot \text{Acc} + (1-\lambda) \cdot \left(1 - \frac{\text{Len}}{\text{Len}_{\text{orig}}}\right), \quad (13)$$

which encourages both higher accuracy and shorter responses.

As shown in Table 4, **Opt$_{\text{Acc+Len}}$** improves both accuracy and reasoning efficiency on the in-domain benchmark MMLU-Pro. However, this gain does not transfer to the out-of-domain benchmark MedQA, where both accuracy and efficiency fall below the pure baseline. This pattern suggests that directly optimizing outcome-level signals may encourage the optimizer to exploit domain-specific regularities that are useful on the training distribution but less reliable under distribution shift. In addition, the resulting entropy trajectory exhibits a weaker late-stage peak, resembling the behavior observed under answer-conditioned prompting. In contrast, our entropy-guided objective yields consistent efficiency gains while preserving out-of-domain accuracy, indicating that it improves task understanding without substantially altering the model's downstream reasoning behavior.

Comparison with Direct RLVR. We further compare with a direct RLVR baseline on Qwen3-30B-Instruct using a scalarized outcome reward:

$$R = (1-\lambda) \cdot \text{Acc} + \lambda \cdot \frac{\text{Len}_{\text{orig}} - \text{Len}}{\text{Len}_{\text{orig}}}, \quad (14)$$

where λ controls the accuracy–efficiency trade-off. Results across three values of λ are shown in Table 5.

As shown in Table 5, direct RLVR exhibits a clear trade-off on the in-domain benchmark. When

Method	MMLU-Pro		MedQA	
	Acc.	Eff.	Acc.	Eff.
Pure	76.50	6.66	73.40	131.75
RLVR (λ=0.25)	**84.40**	8.26	73.40	133.11
RLVR (λ=0.50)	78.10	9.52	72.90	132.98
RLVR (λ=0.75)	69.80	11.28	73.00	133.50
Opt_{Ours}	80.10	**9.15**	**73.20**	**141.64**

Table 5: Comparison with direct RLVR on Qwen3-30B-Instruct.

λ is small, RLVR improves accuracy but yields only limited efficiency gains. As λ increases, efficiency improves further, but accuracy degrades substantially, indicating that outcome-level scalarization makes it difficult to balance correctness and compression in a stable manner. On the out-of-domain benchmark MedQA, performance remains close to the pure baseline across different λ values, suggesting limited generalization of the learned policy. By contrast, our method improves reasoning efficiency while maintaining competitive accuracy in both in-domain and out-of-domain settings.

Overall, these results suggest that entropy-guided prompt optimization provides a more stable and transferable objective than directly optimizing outcome-based rewards. Because it targets the structure of the reasoning process rather than the final answer alone, it is better aligned with our goal of reducing unhealthy exploration while preserving the model's intrinsic reasoning capability.

6 Related Work

6.1 Entropy Dynamics

Entropy is widely used to probe the internal behavior of large language models. In instruction-tuned models, prior work (Kuhn et al., 2023; Nikitin et al., 2024) primarily interprets entropy and related uncertainty measures as indicators of response reliability and model confidence. Farquhar et al. (Farquhar et al., 2024) further introduce semantic entropy to measure disagreement in the semantic space of generated outputs and apply it to hallucination detection. Subsequent work (Han et al., 2024; Nguyen et al., 2025) extends this line by using semantic entropy and related measures to identify hallucinations in LLM outputs.

With the emergence of large reasoning models, recent studies have begun to interpret entropy as a signal of reasoning convergence and to use it reactively to dynamically adjust reasoning depth (Zhang et al., 2025) or as a confidence signal for early stopping (Sharma and Chopra, 2025). In addition, several works (Wang et al., 2025; Agarwal et al., 2025b) incorporate entropy directly into training objectives, using it as a reward or regularization signal to prevent entropy collapse and thereby encourage exploration.

6.2 Prompt Optimization

Prompt optimization aims to improve task performance by rewriting, expanding, or searching over prompts, while keeping model parameters fixed. Early work in this direction treats prompts as discrete textual objects and relies on heuristic strategies (Schick and Schütze, 2020; Shin et al., 2020) or black-box search (Wallace et al., 2019) to explore prompt variants.

More recent approaches leverage LLMs themselves as prompt optimizers. In this paradigm, an LLM acts as a meta-optimizer (Yang et al., 2023), generating candidate prompts (Zhou et al., 2022), mutating them via evolutionary operators (Tong et al., 2025) with improved variants selected in a closed optimization loop.

Beyond using frozen LLMs as prompt optimizers, recent work has explored explicitly training or adapting LLMs to better function as optimizers. This line of research treats optimization itself as a learnable behavior, where models are trained to propose improved solutions based on feedback signals such as task rewards (Deng et al., 2022) or preference comparisons (Lin et al., 2024).

7 Conclusion

In this work, we show that a substantial portion of exploration in LLM reasoning is unhealthy: it is induced by prompt ambiguity, and inflates token usage without improving reasoning quality. We propose an entropy-dynamics-aware prompt optimization framework that generates concise clarifications to reduce early-stage uncertainty while preserving healthy exploration. Extensive experiments show that our method consistently improves reasoning efficiency without sacrificing accuracy.

Limitations

Despite the effectiveness of our proposed prompt optimization framework, our work has several limitations that merit discussion:

- Although we evaluate across multiple models, reasoning budgets, and in- and out-of-domain benchmarks, our experiments remain limited to a finite set of architectures and datasets. It remains unclear how well the proposed approach generalizes to other generation paradigms, such as interactive dialogue, or open-ended creative generation.

- Although entropy peaks empirically align with distinct reasoning stages, entropy remains an indirect proxy for internal uncertainty and exploration. Future work may incorporate complementary signals, such as semantic entropy or representation-level measures, to better distinguish unhealthy from healthy exploration.

- Our explicit restriction that clarifications remain concise and non-informative with respect to solution steps may limit the optimizer's ability to handle prompts that are deeply underspecified or structurally flawed. In such cases, more substantial prompt reformulation may be required, which falls beyond the scope of the current framework.

References

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, and 1 others. 2025a. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*.

Shivam Agarwal, Zimin Zhang, Lifan Yuan, Jiawei Han, and Hao Peng. 2025b. The unreasonable effectiveness of entropy minimization in llm reasoning. *arXiv preprint arXiv:2505.15134*.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and 1 others. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 17682–17690.

Mingkai Deng, Jianyu Wang, Cheng-Ping Hsieh, Yihan Wang, Han Guo, Tianmin Shu, Meng Song, Eric Xing, and Zhiting Hu. 2022. Rlprompt: Optimizing discrete text prompts with reinforcement learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3369–3391.

Xinrun Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, King Zhu, Minghao Liu, Yiming Liang, Xiaolong Jin, Zhenlin Wei, and 1 others. 2025. Supergpqa: Scaling llm evaluation across 285 graduate disciplines. *arXiv preprint arXiv:2502.14739*.

Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. 2024. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630.

Jiatong Han, Jannik Kossen, Muhammed Razzak, Lisa Schut, Shreshth A Malik, and Yarin Gal. 2024. Semantic entropy probes: Robust and cheap hallucination detection in llms. In *ICML 2024 Workshop on Foundation Models in the Wild*.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421.

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

Xiaoqiang Lin, Zhongxiang Dai, Arun Verma, See-Kiong Ng, Patrick Jaillet, and Bryan Kian Hsiang Low. 2024. Prompt optimization with human feedback. *arXiv preprint arXiv:2405.17346*.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. 2025. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 20286–20332.

Dang Nguyen, Ali Payani, and Baharan Mirzasoleiman. 2025. Beyond semantic entropy: Boosting llm uncertainty quantification with pairwise semantic similarity. *arXiv preprint arXiv:2506.00245*.

Alexander Nikitin, Jannik Kossen, Yarin Gal, and Pekka Marttinen. 2024. Kernel language entropy: Fine-grained uncertainty quantification for llms from semantic similarities. *Advances in Neural Information Processing Systems*, 37:8901–8929.

Timo Schick and Hinrich Schütze. 2020. Few-shot text generation with pattern-exploiting training. *arXiv preprint arXiv:2012.11926*.

Aman Sharma and Paras Chopra. 2025. Think just enough: Sequence-level entropy as a confidence signal for llm reasoning. *arXiv preprint arXiv:2510.08146*.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.

Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2025. Scaling llm test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations*.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and 1 others. 2023. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051.

Zeliang Tong, Zhuojun Ding, and Wei Wei. 2025. Evoprompt: Evolving prompts for enhanced zero-shot named entity recognition with large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 5136–5153.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing nlp. *arXiv preprint arXiv:1908.07125*.

Chen Wang, Zhaochun Li, Jionghao Bai, Yuzhi Zhang, Shisheng Cui, Zhou Zhao, and Yue Wang. 2025. Arbitrary entropy policy optimization: Entropy is controllable in reinforcement fine-tuning. *arXiv preprint arXiv:2510.08141*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, and 1 others. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. 2023. Large language models as optimizers. In *The Twelfth International Conference on Learning Representations*.

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

Jinghan Zhang, Xiting Wang, Fengran Mo, Yeyang Zhou, Wanfu Gao, and Kunpeng Liu. 2025. Entropy-based exploration conduction for multi-step reasoning. *arXiv preprint arXiv:2503.15848*.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022. Large language models are human-level prompt engineers. In *The eleventh international conference on learning representations*.

A Baseline

Early Stop (Sharma and Chopra, 2025) is an inference-time efficiency method that monitors sequence-level entropy during generation and terminates reasoning once the entropy remains below a predefined threshold for a fixed number of steps. By truncating later-stage generation, it aims to reduce unnecessary token usage while maintaining acceptable accuracy.

Evoprompt (Tong et al., 2025) is a prompt optimization framework that applies evolutionary algorithms to iteratively refine prompts based on task performance. It focuses on improving answer accuracy by exploring prompt variants through mutation and selection, and requires a development set to evaluate candidate prompts and guide the evolutionary search process.

B Benchmarks

MMLU-Pro (Wang et al., 2024) is a large-scale multi-task benchmark designed to evaluate professional-level language understanding across diverse academic and technical domains. Compared to the original MMLU benchmark, it features increased difficulty and reduced annotation noise, making it well-suited for evaluating advanced reasoning capabilities of large language models.

SuperGPQA (Du et al., 2025) is a comprehensive benchmark covering 285 graduate-level disciplines, with questions curated to require domain-specific knowledge and multi-step reasoning. Its breadth and difficulty make it a challenging testbed for assessing both reasoning robustness and generalization across professional domains.

Big-Bench Hard (BBH) (Suzgun et al., 2023) a subset of the BIG-Bench benchmark consisting of tasks that are empirically difficult for large language models. The benchmark emphasizes compositional and multi-step reasoning, and is commonly used to evaluate the effectiveness of advanced reasoning strategies.

MedQA (Jin et al., 2021) is a medical question answering benchmark derived from professional medical licensing examinations. The dataset requires

Model	Effort	Method	In-domain Benchmarks						Out-of-domain Benchmarks					
			MMLU-Pro			SuperGPQA			BBH			MedQA		
			Acc.↑	Tok.↓	Eff.↑	Acc.↑	Tok.↓	Eff.↑	Acc.↑	Tok.↓	Eff.↑	Acc.↑	Tok.↓	Eff.↑
GPT-oss-120B	High	Pure	**80.60**	2383.35	3.38	49.40	4500.31	1.09	84.40	1496.53	5.63	**91.10**	1372.63	6.63
		Ours	80.30	**1883.77**	**4.26**	**49.80**	**3469.76**	**1.43**	**85.40**	**1107.80**	**7.70**	90.20	**1216.28**	**7.41**
	Medium	Pure	79.30	566.69	13.99	47.20	1055.71	4.47	**84.40**	421.18	20.03	88.00	292.65	30.07
		Ours	**80.60**	**522.75**	**15.41**	**48.60**	**776.53**	**6.25**	83.90	**387.09**	**21.67**	**88.30**	**280.23**	**31.50**
	Low	Pure	75.70	216.22	35.01	42.70	294.56	14.49	83.30	206.78	40.28	**85.40**	**100.86**	**84.67**
		Ours	**76.20**	**209.88**	**36.30**	**45.40**	**263.75**	**17.25**	**83.50**	**200.52**	**41.64**	84.90	116.78	72.70
Qwen3-30B	Thinking	Pure	81.40	3316.56	2.45	**53.30**	4765.22	1.11	**85.90**	1798.92	4.77	**84.90**	1982.31	4.28
		Ours	**81.70**	**2553.35**	**3.19**	53.10	**3855.48**	**1.37**	85.30	**1452.83**	**5.87**	84.80	**1828.35**	**4.63**
	Instruct	Pure	76.50	1147.30	6.66	52.50	1986.15	2.64	83.30	564.12	14.76	**73.40**	**55.71**	**131.75**
		Ours	**80.10**	**1025.40**	**7.81**	**52.60**	**1951.21**	**2.69**	**83.40**	**484.39**	**17.21**	73.20	67.80	107.96

Table 6: Total token usage of reasoning models.

complex clinical reasoning and domain knowledge, making it a standard benchmark for evaluating reasoning performance in the medical domain.

C Prompt

Clarification Generation

You are a helpful assistant that generates a hint to help a reasoning model understand the question and avoid ambiguity.

Target Reasoning Model: {{Target Model}}

Rules:

1) Do NOT change the question text or options.

2) Conclude your response with label 'Clarification:', followed by one single your generated clarification.

3) No solution steps, no answer letter.

Instruction: {{Instruction}}

Clarification Refinement (Early Stage Reward Failure)

*Your previous hint did not sufficiently reduce early-stage uncertainty. Below are the tokens generated by the reasoning model *before* its main uncertainty peak, indicating where the model started to become confused or ambiguous.*

Early-stage Tokens: {{Tokens}}

Based on these tokens, identify what the reasoning model misunderstood or was uncertain about, and generate a NEW, more targeted hint to clarify the instruction, ensuring:

1) Do NOT change the question text or options.

2) Conclude your response with label 'Clarification:', followed by one single your generated clarification.

3) No solution steps, no answer letter.

Clarification Refinement (Late Stage Reward Failure)

Your previous hint destabilized the reasoning model's late-stage processing. This means the hint may have been too specific, leaked information, or disrupted the model's natural reasoning flow.

Please generate a NEW, more targeted hint to clarify the instruction, ensuring:

1) Do NOT change the question text or options.

2) Conclude your response with label 'Clarification:', followed by one single your generated clarification.

3) No solution steps, no answer letter.

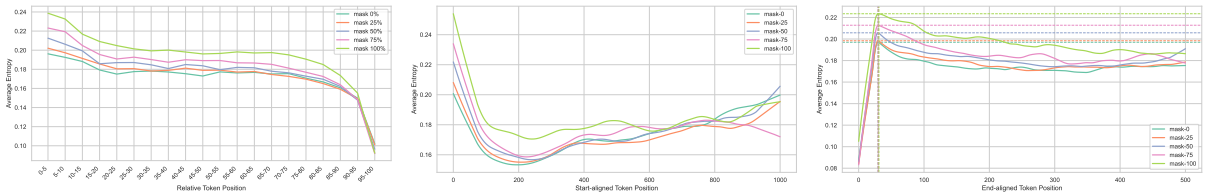
D Total Token Usage

To demonstrate that our prompt optimizer does not introduce excessive token overhead that would offset the reasoning efficiency gains of the reasoning model, we report both the total token usage and the corresponding reasoning efficiency.

We denote by T_o the number of tokens used by the prompt optimizer, P_o the parameter size of the optimizer, P_r the number of activated parameters of the reasoning model during inference, and T_r the number of tokens used by the reasoning model. The total token usage is computed as

$$T_{\text{total}} = T_o \cdot \frac{P_o}{P_r} + T_r. \tag{15}$$

Specifically, $T_o = 4$ for our prompt optimizer trained on Qwen3-4B-Instruct. For GPT-OSS-120B, the number of activated parameters during inference is $P_r = 5.1$, as reported in our analysis, while for the Qwen3-30B series, $P_r = 3$. As shown in Table 6, even after accounting for the token usage introduced by the prompt optimizer, our proposed prompt optimization framework still achieves a significant improvement in reasoning efficiency.



(a) Relative Position (b) Start-aligned Position (c) End-aligned Position

Figure 7: Token-level entropy of GPT-oss-120B under low reasoning effort.

E Human Evaluation

Although the entropy-dynamics analysis in Section 2 suggests that our prompt optimization framework preserves the reasoning behavior of the underlying model, we further conduct a manual evaluation to assess whether the generated clarifications exhibit answer leakage.

Specifically, we randomly sample 100 generated clarifications from the evaluation set and ask three independent annotators with backgrounds in computer science and machine learning to label each clarification according to the following three-level taxonomy:

- **No Leakage**: The clarification only disambiguates the task or constraints, without revealing the answer, intermediate solution steps, or strong hints toward a specific option.
- **Partial Leakage**: The clarification provides mild directional cues or narrowing signals that may reduce the reasoning space, but does not directly reveal the correct answer.
- **Full Leakage**: The clarification explicitly reveals the correct answer or provides decisive information that largely removes the need for reasoning.

All clarifications are annotated independently, and the final label for each sample is determined by majority vote. Among the 100 sampled clarifications, 66% receive unanimous three-way agreement and the remaining 34% receive two-of-three agreement; no instance exhibits complete three-way disagreement. Nearly all disagreements occur at the boundary between *No Leakage* and *Partial Leakage*, suggesting that the taxonomy is well-defined for the vast majority of cases while remaining sensitive to borderline directional cues.

In addition to human annotation, we also perform an automatic assessment using an LLM-as-a-judge. We use GPT-5 as an external evaluator and ask it to classify each clarification using the same three leakage categories. The judging prompt is shown below.

	No Leakage	Partial Leakage	Full Leakage
Human	83	17	0
GPT-5	89	11	0

Table 7: Human evaluation and LLM-as-a-judge results on answer leakage in generated clarifications.

LLM-as-a-Judge Prompt

You are an impartial evaluator assessing whether a clarification generated by a prompt optimizer exhibits answer leakage.

Original Question: {{Question}}

Clarification: {{Clarification}}

Your task is to classify the clarification into one of the following three categories:

1) No Leakage: The clarification purely disambiguates the task or constraints. It does NOT reveal the answer or strong hints toward a specific answer choice.

2) Partial Leakage: The clarification contains mild cues, narrowing signals, or directional hints that reduce the reasoning space, but does NOT explicitly reveal the answer.

3) Full Leakage: The clarification explicitly reveals the answer.

No need to consider correctness of the clarification.

Conclude your response with label "Judgement:", followed by No Leakage / Partial Leakage / Full Leakage.

To reduce stochastic effects, GPT-5 is queried with temperature set to zero, and each clarification is evaluated independently.

As shown in Table 7, the majority of generated clarifications are labeled as *No Leakage* by both human annotators and GPT-5, and neither evaluation identifies any *Full Leakage* case. This result provides direct evidence that the learned clarifications

	GPT-5: No Leakage	GPT-5: Partial Leakage
Human: No Leakage	75	8
Human: Partial Leakage	10	7

Table 8: Human–LLM confusion matrix on the No-Leakage / Partial-Leakage categories. Full Leakage is omitted because neither evaluator assigns any instance to this category.

		α		
		0.2	**0.5**	**0.8**
γ	**0.4**	77.80 / 14.65	79.20 / 16.75	82.10 / 17.21
	0.3	80.40 / 15.66	**80.60 / 17.04**	80.60 / 16.87
	0.2	79.20 / 14.35	83.60 / 16.48	79.90 / 17.43

Table 9: Accuracy / Efficiency (%) under varying early-stage ratio α and late-stage ratio γ on MMLU-Pro (GPT-oss-120B, medium effort).

are generally non-revealing and remain consistent with our design constraint of avoiding solution disclosure.

The confusion matrix in Table 8 further shows that GPT-5 agrees with the human majority label on 75 out of 83 human-labeled *No Leakage* cases (90.4%). Most disagreements are concentrated near the boundary between *No Leakage* and *Partial Leakage*, where GPT-5 tends to apply a slightly more conservative standard. We view this pattern as expected, since borderline directional hints are inherently more subjective than explicit answer revelation.

These findings are also consistent with our entropy-based analysis. Direct answer leakage would substantially collapse the model's reasoning dynamics, especially by weakening or eliminating the late-stage entropy peak associated with verification. By contrast, mild partial leakage is more likely to alter the model's early reasoning behavior by injecting directional cues, which can elevate or delay the early-stage entropy peak. Both behaviors deviate from the desired entropy dynamics and are therefore discouraged by our entropy-peak-based reward.

F Additional Entropy Dynamics

We also present the entropy dynamics of GPT-oss-120B under low reasoning effort, together with results from the Qwen3-30B series.

As shown in Figure 7, unlike the medium- and high-reasoning-effort settings, GPT-oss-120B under low reasoning effort does not exhibit a pronounced early-stage entropy increase, even when prompt ambiguity is introduced. Nevertheless, elevated entropy remains observable in the late stage of generation, indicating that uncertainty is not eliminated but manifests at a later phase. These observations reveal an asymmetric capacity allocation under constrained reasoning budgets: early-stage exploratory behaviors are sacrificed first, while late-stage verification and answer finalization are comparatively preserved. With more ample reasoning budgets, models are able to invest additional tokens in early-stage exploration, which explains why token usage increases more rapidly with higher mask ratios under higher reasoning effort.

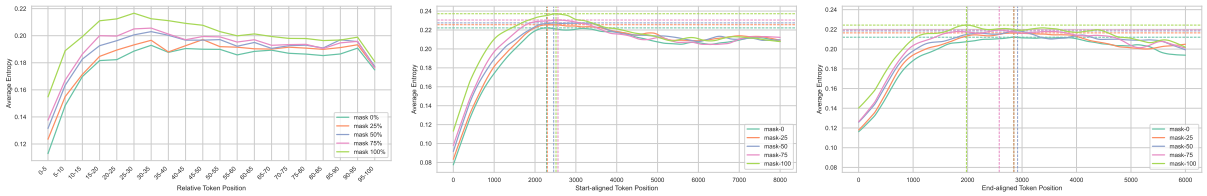
As shown in Figure 8 and Figure 9, the Qwen3-30B series exhibits trends similar to those observed for GPT-oss-120B. Specifically, the entropy trajectory follows a consistent pattern: it initially increases, then decreases, rises again, and finally drops sharply toward termination. Moreover, as prompt ambiguity increases, the early-stage entropy peak becomes both elevated and delayed, while the late-stage entropy peak remains highly convergent. These results further demonstrate the generality of the observed dual-peak entropy dynamics across different model families, supporting the robustness of our analysis.

G Hyperparameter Sensitivity

G.1 Stage Window Parameters

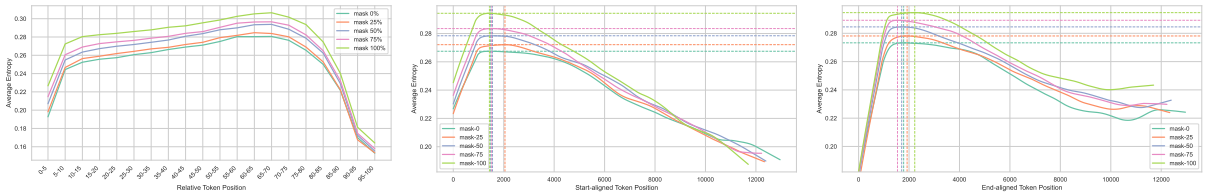
Our framework uses the early-stage ratio α to define the prompt-understanding window and the late-stage ratio γ to define the termination-neighborhood window. We study the sensitivity of these two parameters by training the prompt optimizer with different (α, γ) combinations, targeting GPT-oss-120B under medium reasoning effort on MMLU-Pro.

As shown in Table 9, performance is mainly affected when the early-stage window is too narrow. In particular, when $\alpha = 0.2$, the optimizer becomes less stable, suggesting that such a short window may not fully cover the early-stage entropy peak. Once α is increased to 0.5 or 0.8, performance becomes substantially more stable across different γ values. By contrast, the method is relatively insensitive to γ, indicating that the late-stage entropy signal remains informative under moderate shifts of the termination-neighborhood window. Overall, these results support the robustness of our default setting, $\alpha = 0.5$ and $\gamma = 0.3$.



(a) Relative Position (b) Start-aligned Position (c) End-aligned Position

Figure 8: Token-level entropy of Qwen3-30b-instruct.



(a) Relative Position (b) Start-aligned Position (c) End-aligned Position

Figure 9: Token-level entropy of Qwen3-30b-thinking.

G.2 Temperature Robustness

The entropy-peak reward is computed at training temperature $T = 0.7$. To examine whether the underlying entropy structure remains stable under higher sampling temperatures, we analyze the entropy dynamics of GPT-oss-120B under medium reasoning effort at $T \in \{1.0, 1.2, 1.5\}$.

At $T = 1.0$ and $T = 1.2$, the entropy trajectory preserves the same overall three-stage pattern observed in the main text: a sharp rise in the early stage, a gradual decline through the middle stage, and a final drop near termination. As temperature increases, entropy values shift upward overall, and the early-stage peak moves slightly later, from approximately 25% to 30% of the sequence, while the late-stage peak remains stably located around 85%.

At $T = 1.5$, the entropy curve becomes more monotonically increasing, indicating a qualitative change in the signal shape. Nevertheless, the middle-stage growth remains slower than that of the early stage, and the sharp entropy drop near termination is still preserved. These results suggest that the entropy-based reward remains structurally stable under moderate temperature increases ($T \leq 1.2$), which covers the range commonly used in reasoning generation. At more extreme temperatures, additional entropy normalization may be needed, which we leave to future work.

H Ambiguity and Complexity Analysis

To further examine whether our method suppresses beneficial exploration in genuinely difficult problems, we partition benchmark instances using a 2×2 design along two axes: prompt ambiguity and task complexity.

Prompt ambiguity is proxied by the early-stage entropy peak height under the original prompt. Instances in the top 50% are labeled *Ambiguous*, and the remainder are labeled *Clear*. Task complexity is proxied by the reasoning model's pass rate on the original prompt. Instances in the bottom 50% are labeled *Complex*, and the remainder are labeled *Simple*.

Table 10: Early-stage token saving ratio and accuracy change across the 2×2 Ambiguity $\times$ Complexity partition.

Ambiguity	Complexity	Token Saved (%)	Δ Acc (%)
Ambiguous	Simple	27.70	+1.56
	Complex	33.80	+3.65
Clear	Simple	4.26	−0.82
	Complex	6.11	+0.89

As shown in Table 10, early-stage token savings are concentrated primarily in the *Ambiguous* subsets, while the corresponding reductions in the *Clear* subsets are much smaller. This pattern suggests that the prompt optimizer mainly removes ambiguity-induced uncertainty, rather than uniformly compressing the reasoning process regardless of the input.

The largest gains are observed in the *Ambiguous* $\times$ *Complex* subset, where early clarification yields both substantial token savings and improved accuracy. This result indicates that structurally difficult problems benefit most when ambiguity in the task formulation is resolved before the model commits to a reasoning trajectory.

By contrast, performance remains largely stable in the *Clear* subsets. The small fluctuation in *Clear* $\times$ *Simple* is within normal variance and does not suggest that our method suppresses useful exploration, especially given that the baseline accuracy in this subset is already above 90%.

At the same time, a small fraction of cases (0.21%) exhibits the opposite behavior: the reasoning model answers correctly under the original prompt but fails after clarification. These failures are concentrated almost entirely in the *Ambiguous* $\times$ *Complex* subset. This suggests that, in a limited number of high-complexity cases, broad early-stage exploration may contain structurally useful hypothesis testing, and overly restrictive clarification can occasionally reduce robustness. We view this interaction between ambiguity and complexity as an important boundary condition of our framework.