

ThinkLeap: Reasoning Optimization Through Confidence-Guided Cognitive Leap Extraction

Yijia Zhu^{1,2*}, Zhenning Shi^{1,2*}, Junhan Shi¹, Xun Zhang³, Yong Jiang^{1,2†}, Qing Li²

¹Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen, China

²Peng Cheng Laboratory, Shenzhen, China ³Fuzhou University, Fuzhou, China

zyg26@mails.tsinghua.edu.cn, {shizn23, shijh23}@tsinghua.org.cn,

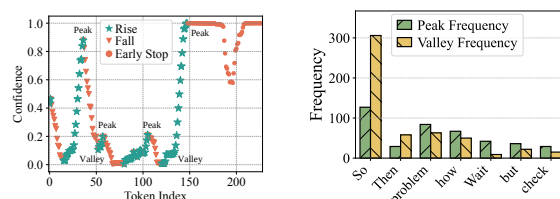
102304229@fzu.edu.cn, jiangy@sz.tsinghua.edu.cn, liq@pcl.ac.cn

Abstract

Large Reasoning Models (LRMs) show remarkable capabilities in complex reasoning tasks through Chain-of-Thought (CoT) mechanisms. However, their reasoning processes frequently suffer from overthinking, generating inefficient patterns and substantial overhead without improving quality. Existing pruning approaches typically rely on computationally expensive external teacher models or aggregated sequence-level metrics, failing to capture the fine-grained, intrinsic cognitive state of individual reasoning steps. In this paper, we propose ThinkLeap, a novel confidence-guided optimization framework that identifies and retains only critical cognitive leaps within reasoning chains. Our approach is grounded in the observation that a model’s internal confidence dynamics serve as a fine-grained structural signal: rising confidence often reflects evidence accumulation, while declining confidence signals doubt or redundancy. By leveraging these dynamics, ThinkLeap autonomously replaces inefficient steps with ellipsis markers, preserving the logical flow while signaling omitted deliberation. Extensive experiments across three challenging reasoning benchmarks demonstrate that ThinkLeap establishes a new paradigm for efficient reasoning, achieving an average 19.4% reduction in reasoning length while maintaining comparable accuracy¹.

1 Introduction

Large Reasoning Models (LRMs), such as OpenAI’s o1 (Jaech et al., 2024) and DeepSeek-R1 (Guo et al., 2025), mark a pivotal milestone in artificial intelligence through CoT mechanisms (Wei et al., 2022). However, this success incurs substantial computational overhead. A key challenge, termed overthinking in recent literature



(a) Confidence dynamics during reasoning. (b) Token frequency at peak and valley points.

Figure 1: Confidence dynamics reveal structured cognitive patterns in LRM reasoning processes. (a) A confidence peak signals the transition from justification to doubt, while a valley marks the beginning of a new justification phase. (b) This interpretation is validated by token analysis: doubt-initiating tokens (e.g., *Wait*, *but*) are over-represented at peaks, whereas justification-initiating tokens (e.g., *So*, *Then*) are prevalent in valleys.

(Chen et al., 2024; Sui et al., 2025), is the propensity of these models to generate verbose reasoning paths containing redundant phrasing and superfluous steps. This inefficiency not only consumes considerable computational resources but can also degrade performance by leading the model to incorrect conclusions, posing a critical bottleneck to its practical deployment.

To mitigate this issue, the research community has explored various compression methods. One line of work relies on external signals or static rules, such as post-hoc pruning based on perplexity (Cui et al., 2025; Yuan et al., 2025) or semantic importance assigned by an auxiliary evaluation model (Xia et al., 2025). Another prominent direction employs reinforcement learning (RL) to learn an optimal reasoning length, for instance by using the shortest correct response as a reward signal (Yi et al., 2026), imposing a strict token limit (Hou et al., 2025), predicting an explicit token budget (Li et al., 2026), or adaptively choosing whether to engage in elaborate reasoning at all (Fang et al., 2026). While these methods have achieved a de-

*Equal contribution.

†Corresponding author.

¹<https://github.com/YiJia-Zhu/ThinkLeap>

gree of success, they largely remain semantically agnostic. They rely on aggregated sequence-level metrics or extrinsic heuristics, such as overall path length or predefined static rules, which may not accurately reflect the model’s dynamic internal certainty at each token. Consequently, they cannot reliably distinguish between essential deliberation and inefficient logical meandering.

The central insight of this paper is that the dynamic fluctuation of a model’s confidence during its CoT process is not unstructured noise, but a rich signal reflecting its underlying cognitive state. As illustrated in Figure 1, when the model explores alternatives, encounters uncertainty, or performs backtracking, its confidence naturally declines, a phase we term the Doubt-Reflection phase. Conversely, when it appears to identify crucial evidence, rule out distractors, or construct more robust logical arguments, we observe that its confidence tends to rise significantly, a phase which we term the Reasoning-Justification phase.

Our analysis reveals a clear cognitive pattern at confidence turning points. Tokens indicating doubt (e.g., *Wait, but*) tend to occur near confidence peaks, while justification-related tokens (e.g., *So, Then*) are more frequent near valleys. This alignment suggests that confidence dynamics capture meaningful cognitive transitions, where peaks correspond to the start of doubt phases, and valleys mark the onset of justification.

This perspective is supported by recent studies (Yoon et al., 2026) that correlate a model’s nonlinear thought processes with its confidence dynamics. To obtain a precise and reliable metric, we move beyond verbalized confidence, which is often prone to overconfidence (Guo et al., 2017; Xu et al., 2024), and instead adopt probabilistic confidence, derived directly from the model’s token generation probabilities, as our core indicator (Ni et al., 2024; Kadavath et al., 2022; Geng et al., 2024; Shi et al., 2025b).

Building upon this insight, we introduce a novel self-distillation framework that teaches a model to emulate human-like cognitive leaps, thereby generating more concise and efficient reasoning paths without sacrificing accuracy (Shen et al., 2025). Our framework operates by employing our Doubt-Justification interpretation model to automatically identify and extract reasoning segments that achieve an effective cognitive leap from the model’s original, verbose CoT. These are core steps where the confidence gained in the justifi-

cation phase sufficiently outweighs the loss from the preceding doubt phase. These extracted, efficient reasoning trajectories are then leveraged as high-quality exemplars in a Supervised Fine-Tuning (SFT) stage, distilling this refined and efficient reasoning capability back into the model.

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

- We propose a novel framework for interpreting the reasoning process based on probabilistic confidence. We map the intrinsic fluctuations of model confidence into two cognitive stages: Doubt-Reflection and Reasoning-Justification, providing a new paradigm for semantically understanding and optimizing the thought processes of LLMs.
- We design an efficient reasoning-compression algorithm based on effective cognitive leaps. By quantifying confidence changes across the Doubt-Reflection and Reasoning-Justification phases, this algorithm heuristically preserves core reasoning steps while replacing redundant segments with ellipsis markers. This approach encourages the generation of efficient, non-linear reasoning paths that maintain structural coherence, moving beyond simple text truncation.
- We construct an end-to-end self-distillation pipeline to optimize the model’s reasoning efficiency. This pipeline automatically generates high-quality, condensed reasoning samples from the model’s verbose outputs and effectively distills this refined reasoning capability back into the model, enhancing its reasoning efficiency while preserving high accuracy.

2 Related Work

2.1 External Supervision for Reasoning Optimization

The verbosity of CoT reasoning has been widely recognized as the overthinking problem (Chen et al., 2024; Sui et al., 2025). Methods relying on external supervision employ teacher models or auxiliary systems to guide reasoning path compression. Distilled Reasoning Pruning (DRP) (Jiang et al., 2026) utilizes stronger teacher models to prune verbose reasoning through skill-aware decomposition, while TokenSkip (Xia et al., 2025)

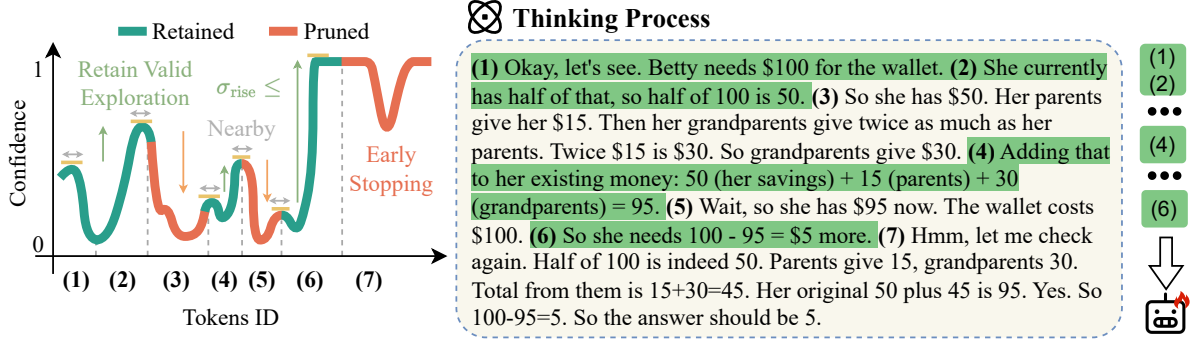


Figure 2: An example of ThinkLeap for refining LLM reasoning chains. The core principle is to retain segments identified as Valid Explorations. A path is considered valid only if the magnitude of a confidence rise surpasses the preceding drop by a predefined threshold, σ_{rise} . In the figure, the minor recovery at (3) and (5) fails to meet this criterion and is replaced with ellipsis ("..."), while the sharp ascent at (1), (2), (4) and (6) qualifies as a valid exploration. Crucially, the process is sentence-aware to preserve semantic integrity. Instead of cutting at the exact peak or valley tokens, the algorithm searches the vicinity for the closest sentence boundary to make the split. Finally, these refined sequences are used as training targets for SFT.

employs external models to score the semantic importance of individual tokens during inference. While these approaches demonstrate effectiveness, the reliance on external supervision limits their applicability in resource-constrained scenarios where self-contained optimization is preferred.

2.2 Self-Supervised Reasoning Optimization

Self-improvement approaches have gained prominence for their ability to optimize reasoning without external dependencies. Stochastic methods include FS-BoN (Munkhbat et al., 2025) using best-of-N sampling, while SpecCoT (Shi et al., 2025a) accelerates CoT reasoning through speculative exploration. Another contemporary approach, ConCISE (Qiao et al., 2025), also leverages model confidence, but its philosophy contrasts sharply with ours. ConCISE frames confidence drops as a confidence deficit that should be actively suppressed, for instance by injecting phrases to prevent reflection and immediately restore confidence. This strategy treats doubt as an inefficiency to be eliminated. Similarly, Certainindex (Fu et al., 2024) monitors answer stabilization for early stopping. However, its iterative scheduling of frequent generation-probing cycles increases KV cache management overhead and hinders the throughput optimizations of frameworks like vLLM. In contrast, ThinkLeap maintains a streamlined, one-shot execution pipeline with zero inference-time overhead. It’s built on the premise that these moments of doubt are not merely noise but can be productive (Yoon et al., 2026), triggering crucial backtracking and deliberation that

are hallmarks of strong reasoning. Reinforcement learning approaches have been widely adopted, including THINKPRUNE (Hou et al., 2025) with iterative length constraints, ShorterBetter (Yi et al., 2026) using shortest correct response lengths as rewards, SelfBudgeter (Li et al., 2026) with token budget prediction, and Thinkless (Fang et al., 2026) for adaptive reasoning engagement. However, these methods rely on aggregated sequence-level metrics like overall length or binary choices rather than analyzing the semantic role of each reasoning step. Our approach fundamentally differs by analyzing the microdynamics of the reasoning process itself. We dissect single, verbose paths to identify effective cognitive leaps guided by internal confidence. This focus on logical flow is supported by (Li et al., 2025), which shows that reasoning structure (including reflection and backtracking) is more critical than the factual correctness of each step.

3 Methodology

Our methodology refines LRM reasoning through a sequential compression pipeline, as illustrated in Figure 2. We first generate a confidence curve from token-wise scores of the reasoning process. This curve is then analyzed using peak detection to identify significant rising segments, which we define as valid reasoning. Segments that fail to meet our validity criterion are replaced with ellipsis markers ("..."), signaling omitted deliberation while preserving the overall reasoning structure.

3.1 Confidence Score Calculation

We quantify the model’s confidence at each step of its reasoning process. At each generated token, the confidence score is defined as the model’s probability of outputting the correct answer when prompted with a specific probe. After the LRM generates its original thought process up to a token k , we append a probe prompt to elicit the final answer. For example, on GSM8K, if the reasoning prefix is S_k and the ground-truth answer is “18”, we append the literal suffix “The correct answer is” and evaluate the probability of the answer sequence “18” after this prefix. For a multi-token answer, each target token is evaluated autoregressively, conditioned on the preceding ground-truth answer tokens.

Let the reasoning prefix at token k be S_k , the probe prompt be $\mathcal{P}_{\text{probe}}$, and the correct answer token sequence be $A = (a_1, a_2, \dots, a_m)$. The raw confidence c_k is the joint probability of the answer sequence, conditioned on the reasoning prefix and the probe prompt:

$$c_k = \prod_{i=1}^m P(a_i | S_k, \mathcal{P}_{\text{probe}}, a_{<i}) \quad (1)$$

where the context $(S_k, \mathcal{P}_{\text{probe}}, a_{<i})$ is formed by concatenating the reasoning prefix, the probe prompt, and the preceding answer tokens. The product follows the chain rule and gives the sequence-level probability of the answer string. Importantly, ThinkLeap compares c_k across different prefixes S_k for the same answer sequence A and the same probe template; it therefore relies on relative confidence changes along a reasoning trace rather than absolute confidence comparisons across different answers. Multiplying the token-level probabilities preserves this fixed-answer comparison while accounting for all tokens in the answer. For numerical stability, we accumulate the corresponding log-probabilities and exponentiate their sum.

To mitigate high-frequency fluctuations, we apply a centered moving average filter. A window of size W is centered around each token k ; for even values of W , the window is biased to the left. The smoothed score C'_k is the mean of the raw scores within this window, which is truncated at the sequence boundaries:

$$c'_k = \frac{1}{|\mathcal{W}_k|} \sum_{j \in \mathcal{W}_k} c_j \quad (2)$$

Here, $\mathcal{W}_k$ denotes the set of valid indices in the window centered at position k . Specifically, we

first define the left and right boundaries of the ideal window as

$$l_k = k - \lfloor \frac{W}{2} \rfloor, \quad r_k = k + \lfloor \frac{W-1}{2} \rfloor \quad (3)$$

The final truncated window:

$$\mathcal{W}_k = \{i \in \mathbb{Z} \mid l_k \leq i \leq r_k\} \cap \{1, \dots, N\} \quad (4)$$

Here, N is the total length of the sequence, and $|\mathcal{W}_k|$ is the actual number of elements in the truncated window.

Notably, the calculation of c_k for each token k in the reasoning trace is an independent forward pass that involves substantial input overlap. This structure allows confidence estimations for all tokens in a sequence to be computed efficiently in parallel, particularly when using engines like vLLM (Kwon et al., 2023), which significantly improves the method’s efficiency. In our practical test case, when processing the MATH-500 dataset with the Qwen3-14B model, a total of 806,230 probes were required to calculate the confidence scores. The total time to complete all these calculations was 65 minutes and 34 seconds using a single NVIDIA H100 GPU (80GB), leveraging the vLLM inference engine. This one-time offline pre-processing cost is acceptable compared to standard fine-tuning durations; a broader runtime comparison is reported in Section 4.3.7.

3.2 Peak and Valley Detection for Reasoning Phase Analysis

To robustly analyze the token-by-token confidence curve, which often exhibits significant fluctuations and noise, we must identify moments of increasing and decreasing certainty. These moments correspond to hypothesized “proof” and “doubt” phases in the model’s reasoning. Direct calculation of the curve’s extrema via numerical derivatives is often brittle and unreliable on such discrete, noisy data.

Therefore, we adopt a more robust, principle-based approach. The sets of significant peak indices ($\mathcal{P}$) and valley indices ($\mathcal{V}$) are obtained by applying our `find_peak` function (detailed in Algorithm 1) (Virtanen et al., 2020) to the smoothed confidence curve C'_{smooth} and its inversion $-C'_{\text{smooth}}$, respectively.

$$\mathcal{P} = \text{find_peak}(C'_{\text{smooth}}, \tau) \quad (5)$$

$$\mathcal{V} = \text{find_peak}(-C'_{\text{smooth}}, \tau) \quad (6)$$

The `find_peak` algorithm operates in two stages. First, it identifies all local maxima as candidate

peaks, where a candidate at index i must satisfy $c'_i > c'_{i-1} \wedge c'_i > c'_{i+1}$. Second, these candidates are filtered based on a prominence metric to retain only the most significant turning points. The prominence of a peak measures how much it stands out from its surrounding signal. For a peak p at index i , its prominence is its height relative to the higher of its two local bases. To find these bases, we search left and right from the peak. The left search extends from index $i - 1$ downwards, stopping at the first point higher than the peak itself, and the lowest value in this range is the left base, $c'_{\text{base},L}$. The right base, $c'_{\text{base},R}$, is found symmetrically. The prominence is then formally calculated as:

$$\text{prominence}(p) = c'_i - \max(c'_{\text{base},L}, c'_{\text{base},R}) \quad (7)$$

Finally, these significant peaks ($\mathcal{P}$) and valleys ($\mathcal{V}$) indices are used to partition the reasoning trace. They are merged with the curve's start (index 1) and end (index T) points to form a complete, ordered set of boundaries. By definition, any segment between a valley and a subsequent peak represents a rising “proof” phase, characterized by increasing model certainty. Conversely, a segment between a peak and a subsequent valley signifies a falling “doubt” phase, where certainty decreases. This segmentation transforms the raw confidence curve into a meaningful, alternating sequence of proof and doubt periods.

3.3 Retaining Valid Reasoning

To make the compression process more robust, we refine the dynamic replacement logic to handle all cases rigorously. The validity of a proof phase (a valley-to-peak segment) is evaluated based on the doubt phase (a peak-to-valley segment) that precedes it. To preserve semantic integrity, replacement points are determined at the nearest sentence boundary, identified using simple punctuation within a local window of the peak or valley, rather than at the exact token. Invalid segments are replaced with “...” rather than being completely removed, maintaining the trace of cognitive transitions.

The pruning process analyzes the alternating sequence of doubt and proof phases defined by the ordered sets of peaks $\mathcal{P}$ and valleys $\mathcal{V}$.

If the reasoning starts with a proof phase, it is retained by default.

3.3.1 General Pruning Rule

For any other proof phase that is preceded by a doubt phase, we apply our core pruning criterion. This fall-rise pair is defined by three consecutive turning points: a peak $p_i \in \mathcal{P}$, a subsequent valley $v_i \in \mathcal{V}$, and the next peak $p_{i+1} \in \mathcal{P}$.

- The “doubt” phase is the segment from index p_i to v_i .
- The “proof” phase is the segment from index v_i to p_{i+1} .

The confidence gain for the proof phase is:

$$\Delta C_{\text{proof}} = c'_{p_{i+1}} - c'_{v_i} \quad (8)$$

The magnitude of the confidence drop for the preceding doubt phase is:

$$|\Delta C_{\text{doubt}}| = c'_{p_i} - c'_{v_i} \quad (9)$$

The proof phase is retained only if its confidence gain sufficiently overcomes the preceding drop. To make this criterion tunable, we introduce a hyperparameter, σ_{rise} , which specifies a minimum required confidence increase. The proof phase is retained if its gain exceeds the preceding drop by this margin:

$$\Delta C_{\text{proof}} > |\Delta C_{\text{doubt}}| + \sigma_{\text{rise}} \quad (10)$$

Here, ΔC_{proof} represents the magnitude of the confidence increase during the proof phase (i.e., the rise from a confidence valley to a new peak), while $|\Delta C_{\text{doubt}}|$ is the magnitude of the confidence drop during the preceding doubt phase.

This condition intuitively ensures that a new line of reasoning is validated only if it represents a decisive recovery from uncertainty. The confidence gain must not only compensate for the previous drop but also overcome it by a margin of at least σ_{rise} . This hyperparameter acts as a strictness threshold, preventing the retention of trivial or hesitant fluctuations. In essence, this requires the new confidence peak to be significantly higher than the previous one.

If this condition is not met, the recovery from doubt is insufficient. The entire doubt-proof cycle, represented by the turning points p_i and p_{i+1} , is replaced with an ellipsis marker (“...”) in the reasoning trace. This replacement strategy preserves the position of cognitive transitions while removing verbose content.

3.3.2 Early Stopping

During offline trajectory construction, we also observe that many models, after reaching a stable and high level of confidence, enter redundant verification loops (e.g., outputting “Wait, let me double check...”). We therefore apply an early stopping rule to the generated training trajectories

Specifically, the trajectory is truncated at the first token index k where the smoothed confidence remains above a high threshold θ_{stop} (e.g., 0.99) for a continuous window of W_{stable} tokens. All subsequent tokens are discarded. This removes redundant verification loops from the training targets while avoiding any additional confidence computation during inference.

4 Experiments

4.1 Experimental Setup

Our experiments are conducted on three challenging benchmark datasets: ARC-Challenge (Clark et al., 2018), GSM8K (Cobbe et al., 2021), and a competition-level mathematics dataset, MATH-500 (Hendrycks et al., 2021) (60/40 train-test split). To evaluate broad applicability, we tested various LLMs with varying architectures and sizes: DeepSeek-R1-Distill-Qwen-1.5B (DS-R1-1.5B) (Guo et al., 2025), QwQ-32B (Qwen Team, 2025), and Qwen3-14B (Yang et al., 2025).

We compare our proposed approach (Ours) against the original model performance (Original) and four competitive baseline methods:

- **TokenSkip** (Xia et al., 2025): A method that employs a static rule-based strategy to skip tokens based on importance scores during inference. Following the original implementation, we set the threshold to 0.8 in our experiments.
- **FS-BoN** (Munkhbat et al., 2025): A sampling-based method that utilizes few-shot prompting to generate multiple candidate reasoning paths through Best-of-N sampling. We generate 16 candidate outputs and select the best one for training.
- **Certainindex** (Fu et al., 2024): An adaptive-compute inference strategy that interleaves chunked reasoning generation with periodic certainty probes, and performs early stopping once consecutive probe answers are consistent and confident. Following the original setup, we use a probe interval of 32 tokens and a certainty window of 2.
- **CoD** (Xu et al., 2025): A prompt-based Chain-of-Draft baseline that encourages concise intermediate reasoning while preserving step-by-step structure. We follow the standard CoD prompting protocol with a few-shot exemplars and require the final answer to be returned in a strict target format.

For the final parameters of our ThinkLeap framework used in all experiments, we set the prominence threshold τ to 0.1, the rise magnitude σ_{rise} to 0.01, and the smoothing window size W to 10. For the early stopping mechanism, the confidence threshold θ_{stop} was set to 0.99 with a stable window of $W_{\text{stable}} = 10$ tokens. These values were determined based on the performance on a validation set, as informed by our parameter study. To account for the inherent randomness of the fine-tuning process, all experiments were repeated three times, and we report the average results in the tables.

Models were trained on 8 NVIDIA H100 GPUs using the AdamW optimizer ($LR = 2 \times 10^{-5}$, batch size 2). During inference, we employ a greedy decoding strategy, setting the temperature at 0, to ensure deterministic and reproducible results in the inference process. The maximum sequence length was set to 16000 tokens.

4.2 Main Results

Table 1 shows that our method achieves the best overall trade-off between accuracy and token cost. Averaged across all model-dataset pairs, **Ours** reaches the highest accuracy (**87.3%**) while using 1615.3 tokens on average, with a mean token-usage ratio of 80.6% relative to the corresponding Original model. Compared with **CoT** (86.5%, 1958.0 tokens), our method is both more accurate and more efficient. Compared with **FS-BoN** (87.0%, 1701.4 tokens), our method remains slightly better in accuracy and uses fewer tokens. In addition, we include **Non-CoT** as a practical lower-bound baseline.

The baseline behaviors also reveal important limitations. **CoD** is unstable in our setting: although its prompt enforces concise drafts, instruction-following is often incomplete for reasoning-tuned models, likely due to the model’s internal <think> training behavior; representative input and output failures are provided in the appendix B. **Certainindex** performs relatively well on ARC-Challenge, where shorter and more direct reasoning may be sufficient, but its gains do not consistently transfer to GSM8K and MATH500, indicating that addi-

Dataset	Model	Non-CoT	CoT		TokenSkip			CoD			Certaindex			FS-BoN			ThinkLeap (Ours)		
		ACC (%)	ACC (%)	Avg Tok	ACC (%)	Avg Tok	Rate (%)	ACC (%)	Avg Tok	Rate (%)	ACC (%)	Avg Tok	Rate (%)	ACC (%)	Avg Tok	Rate (%)	ACC (%)	Avg Tok	Rate (%)
ARC-C	DS-R1-1.5B	31.7	58.7	1867.0	55.3	<u>1086.4</u>	58.2	30.3	1181.3	63.3	56.7	896.5	48.0	58.3	1201.5	64.4	57.7	1495.9	80.1
	QwQ-32B	92.7	<u>96.7</u>	565.1	96.0	556.7	98.5	95.0	<u>498.9</u>	88.3	95.3	402.3	71.2	97.3	499.2	88.3	<u>96.7</u>	517.6	91.6
	Qwen3-14B	82.0	<u>96.7</u>	469.9	93.7	97.1	20.7	92.0	492.5	104.8	92.7	<u>249.9</u>	53.2	97.3	445.9	94.9	96.3	426.0	90.7
GSM8K	DS-R1-1.5B	13.7	<u>79.9</u>	599.5	79.8	543.7	90.7	61.2	1289.4	215.1	55.1	422.8	70.5	78.3	457.1	76.3	82.9	420.7	70.2
	QwQ-32B	52.5	96.6	574.1	97.7	810.4	141.2	<u>97.0</u>	541.9	94.4	75.3	<u>431.8</u>	75.2	<u>97.0</u>	592.5	103.2	96.6	398.2	69.4
	Qwen3-14B	39.5	<u>95.8</u>	521.5	95.4	597.3	114.5	91.6	588.4	112.8	86.7	660.2	126.6	96.2	<u>480.0</u>	92.0	95.2	378.6	72.6
MATH500	DS-R1-1.5B	11.0	68.0	5069.3	68.0	4218.1	83.2	41.0	5355.2	105.6	<u>70.5</u>	3645.1	71.9	72.5	4145.0	81.8	72.5	4267.4	84.2
	QwQ-32B	45.0	94.5	3869.4	89.5	3613.9	93.4	88.0	3400.0	87.8	73.2	4250.6	109.9	<u>94.0</u>	<u>3588.8</u>	92.7	93.5	3148.2	81.4
	Qwen3-14B	19.5	92.0	4086.5	90.0	3824.9	93.6	76.0	2330.2	57.0	85.5	4480.3	109.6	<u>92.5</u>	3902.8	95.5	94.5	<u>3485.0</u>	85.3
Average		43.1	86.5	1958.0	85.0	1705.4	88.2	74.7	1742.0	103.2	76.8	1715.5	<u>81.8</u>	<u>87.0</u>	<u>1701.4</u>	87.7	87.3	1615.3	80.6

Table 1: Performance comparison. Highlights (bold/underline) are applied among all methods except Non-CoT. Rate (%) represents the token usage ratio relative to the corresponding Original model.

tional hyperparameter tuning is needed.

Method	ACC (%)	Avg Tokens
Original	95.82	521.50
Human Labeled	85.55	133.07
Verbalized Confidence	95.82	507.97
w/o Smoothing	35.36	10.38
w/o Early Stopping	95.06	480.08
w/o Doubt	87.07	310.65
only Early Stopping	95.82	412.71
only Last Proof	47.53	33.22
Full	95.16	378.64

Table 2: Ablation study on Qwen3-14B using GSM8K. We compare our final method (Full) against several variants, reporting accuracy (ACC) and the average number of tokens used per inference.

4.3 Analysis

4.3.1 Analysis of Cognitive Patterns

To quantitatively validate our hypothesis that confidence fluctuations mirror cognitive states, we performed a statistical analysis on the linguistic patterns at confidence peaks and valleys (Light et al., 2004; Vincze et al., 2008). We aimed to demonstrate a significant association between two defined token categories: doubt-initiating tokens (e.g., *how*, *Wait*, *but*), which introduce uncertainty, and justification-initiating tokens (e.g., *So*, *Then*, *Therefore*), which signal logical progression.

Figure 1(b) visualizes the frequency of these tokens, revealing a distinct pattern. A Pearson’s chi-squared (χ^2) test confirmed a highly significant association between the token categories and the confidence states ($\chi^2(1) = 76.91, p < .001$). To further dissect this relationship, we analyzed the standardized residuals (SR), where an absolute value $|\text{SR}| > 1.96$ indicates a statistically significant deviation from the expected frequency. The analysis

revealed that doubt-initiating tokens appeared significantly more often than expected at confidence peaks ($\text{SR} = 5.89$) and significantly less often at valleys ($\text{SR} = -4.76$). Conversely, justification-initiating tokens were significantly overrepresented at confidence valleys ($\text{SR} = 2.88$) and underrepresented at peaks ($\text{SR} = -3.56$).

This statistically significant association provides empirical support for our interpretation. It suggests that confidence fluctuations are linguistically associated with transitions in the reasoning process (Kemper et al., 2001). A peak, rich with doubt-initiating tokens, signals the onset of a “Doubt-Reflection” phase, leading to a subsequent drop in confidence. A valley, marked by a high frequency of justification-initiating tokens, indicates the initiation of a “Reasoning-Justification” phase, where the model establishes a clear path forward, causing confidence to rise.

4.3.2 Ablation Study

As shown in Table 2, the analysis reveals several indispensable components. Removing confidence smoothing (w/o Smoothing) or the doubt analysis phase (w/o Doubt) leads to a catastrophic collapse in accuracy, confirming they are essential for robust performance. Likewise, an overly aggressive strategy that retains only Last Proof proves ineffective, as it discards vital reasoning steps.

Other variants highlight the nuanced contributions to efficiency and provide useful benchmarks. For instance, using only Early Stopping is less efficient than the complete model, while its removal (w/o Early Stopping) negates most token savings, confirming its key role. The Human Labeled data offers a crucial insight: while it is the most concise, its lower accuracy suggests that human thought processes can be highly abbreviated. Hu-

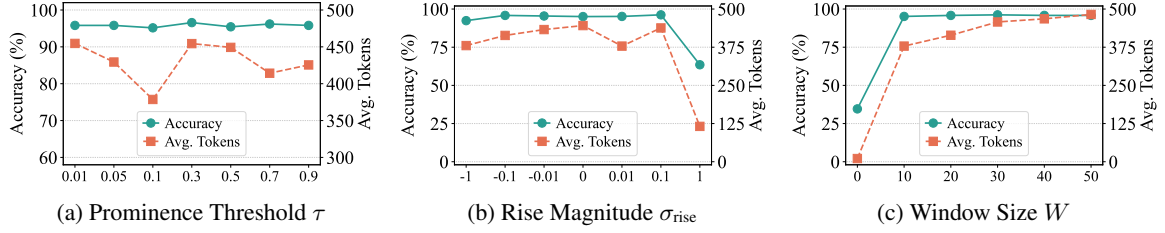


Figure 3: Sensitivity analysis of ThinkLeap parameters. We evaluate the effect of varying the three parameters on accuracy and average token usage.

Dataset	Model	CoT		TokenSkip			FS-BoN			Ours		
		ACC	Token	ACC	Token	Rate	ACC	Token	Rate	ACC	Token	Rate
AIME24	DS-R1-1.5B	23.3	9642.9	5.0	6346.6	65.8	20.0	9194.4	95.3	14.4	8936.3	92.7
	QwQ-32B	60.0	9311.9	56.7	9564.6	102.7	56.7	8762.4	94.1	64.4	7599.8	81.6
	Qwen3-14B	62.2	8375.2	70.0	8002.6	95.6	56.7	8159.2	97.4	64.4	7514.5	89.7
AMC23	DS-R1-1.5B	49.2	6549.1	55.0	4158.3	63.5	62.5	4814.2	73.5	59.2	4816.6	73.5
	QwQ-32B	85.0	5702.8	75.0	6815.7	119.5	85.0	5716.8	100.2	86.7	4433.1	77.7
	Qwen3-14B	93.3	4657.3	80.0	4928.5	105.8	95.0	4592.6	98.6	90.8	3839.0	82.4
Average		62.2	7373.2	56.9	6636.0	92.2	62.6	6873.3	93.2	63.3	6189.9	83.0

Table 3: Comparison of different methods on out-of-distribution datasets (AIME24, AMC23) after training on GSM8K.

mans often skip intermediate steps that, while intuitive to them, are vital for an LLM to follow the logical chain, thus hindering its performance. Finally, the Verbalized Confidence approach proves inefficient with negligible token reduction.

In contrast, our full method demonstrates a clear synergy among all parts. It achieves the best overall balance, maintaining high accuracy while significantly reducing the average token count. This result validates that our complete, integrated framework is the most effective configuration.

4.3.3 Parameter Study

We analyzed the sensitivity of our core hyperparameters, with results depicted in Figure 3. The analysis shows that our method is robust to the window size (W) and rise magnitude (σ_{rise}), maintaining high accuracy across a wide range. And the Prominence Threshold (τ) is the most critical factor that we need to adjust. However, all results presented in Table 1 and Table 3 were obtained using a fixed hyperparameter configuration ($W = 10, \sigma_{rise} = 0.01, \tau = 0.1$). This consistency across diverse architectures (from 1.5B to 32B) and domains indicates that our method does not rely on dataset-specific tuning. The prominence threshold τ specifically serves as a principled filter for the model’s inherent hesitation level, capturing fundamental cognitive dynamics rather than

noise-driven artifacts.

4.3.4 Generalization

To assess whether our method learns a generalizable reasoning paradigm, we trained models on GSM8K and evaluated them on two out-of-distribution (OOD) datasets: AIME24 and AMC23. This setup tests the ability to transfer learned efficiencies to unseen tasks.

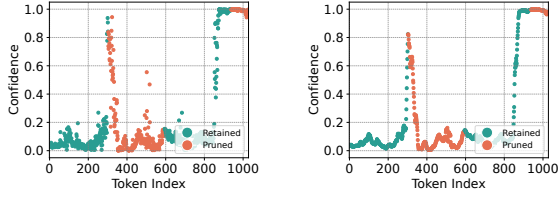
As presented in Table 3, ThinkLeap achieves an average token reduction of 17.0% relative to CoT while achieving slightly higher average accuracy (63.3% vs. 62.2%). The trend is not uniform across all model–dataset pairs: accuracy improves in several settings but decreases modestly in others, while token usage is consistently reduced.

At the model level, QwQ-32B improves both accuracy and efficiency on both OOD datasets. Qwen3-14B improves on AIME24 but trades a minor accuracy drop on AMC23 (from 93.3% to 90.8%) for a 17.6% token reduction; DS-R1-1.5B similarly reduces tokens on both datasets, with accuracy dropping on AIME24 but rising on AMC23.

These results support the transfer of efficiency improvements beyond GSM8K, while also highlighting task- and model-specific accuracy trade-offs rather than universal gains.

Dataset	Model	Subset	Samples	Original		Ours		Token Change
				ACC (%)	Avg Tok	ACC (%)	Avg Tok	
MATH500	DS-R1-1.5B	≥ 5000 tokens	68	19.12	11328.1	39.71	7440.2	-34.3%
	DS-R1-1.5B	longest 25%	50	2.00	12288.0	40.00	6998.1	-43.0%
	Qwen3-14B	≥ 5000 tokens	51	78.43	8918.0	78.43	6804.5	-23.7%
	Qwen3-14B	longest 25%	50	78.00	8996.2	78.00	6861.0	-23.7%

Table 4: Long-horizon analysis on MATH500 using two subsets selected by original reasoning length: the longest 25% and examples with at least 5,000 tokens. The subsets are complementary diagnostics and may overlap.



(a) Raw Confidence.

(b) Smoothed Confidence.

Figure 4: Comparison of confidence dynamics before and after smoothing. The raw data (a) contains substantial noise, while the smoothed curve (b) reveals the underlying cognitive patterns used by ThinkLeap to identify critical reasoning steps.

4.3.5 Visualization of Confidence Dynamics and Smoothing

Probabilistic confidence serves as the core signal for ThinkLeap due to its token-level granularity, which captures internal cognitive fluctuations. As illustrated in Figure 4, raw confidence scores frequently exhibit high-frequency noise inherent to the discrete token generation process. This volatility makes direct identification of cognitive transitions unreliable through numerical derivatives alone. By applying the centered moving average filter (Eq. 2), the signal is transformed into a structured trajectory that reveals distinct Reasoning-Justification and Doubt-Reflection phases.

4.3.6 Long-Horizon Robustness

To assess the scalability of confidence-guided pruning, we evaluate two complementary subsets selected by the original reasoning length: the longest 25% of examples and examples exceeding 5,000 tokens. ThinkLeap reduces the average token count by 34.3% for DS-R1-1.5B and 23.7% for Qwen3-14B on the latter subset; for DS-R1-1.5B, accuracy rises from 19.12% to 39.71%, while for Qwen3-14B it remains at 78.43%. This stress test evaluates pruning stability on long reasoning traces rather than ranking against additional baselines; complete results are shown in Table 4.

Dataset	Model	FS-BoN	ThinkLeap
GSM8K	DS-R1-1.5B	23min51s	8min36s
GSM8K	Qwen3-14B	48min11s	18min19s
ARC-C	DS-R1-1.5B	26min13s	6min35s
ARC-C	Qwen3-14B	37min32s	18min38s
MATH500	DS-R1-1.5B	43min06s	36min17s
MATH500	Qwen3-14B	160min50s	106min15s

Table 5: One-time preprocessing wall-clock time for FS-BoN and ThinkLeap, measured on four NVIDIA RTX 4090 GPUs.

4.3.7 Runtime Cost

To quantify practical overhead, we report the one-time preprocessing cost measured on four NVIDIA RTX 4090 GPUs. Preprocessing is dominated by confidence estimation and is performed offline, so it does not affect inference. Prefix sharing and batched execution (e.g., via vLLM) make these computations practical even for long traces. Table 5 reports preprocessing time only; training and evaluation times are omitted because they are comparable between methods.

The measured gap is primarily due to preprocessing. FS-BoN samples 16 full trajectories per example and selects among them, whereas ThinkLeap computes confidence within a single trajectory, making overall data construction more efficient in practice.

5 Conclusion

In this work, we introduced ThinkLeap, a confidence-guided framework that leverages internal confidence dynamics to identify and retain critical cognitive leaps in LRM reasoning. By modeling reasoning as alternating doubt-reflection and reasoning-justification phases, ThinkLeap achieves 19.4% token reduction while maintaining accuracy. Strong out-of-distribution performance confirms it learns generalizable efficiency patterns rather than task-specific heuristics, offering a practical approach to efficient reasoning optimization.

Limitations

While our ThinkLeap framework demonstrates strong performance on mathematical reasoning benchmarks, several limitations warrant discussion. First, our evaluation primarily focuses on mathematical reasoning tasks, and different reasoning domains may exhibit distinct confidence dynamics. Second, free-form answers require conversion to multiple-choice format using an auxiliary LLM, introducing potential errors and preprocessing overhead. Future work will explore direct confidence estimation for free-form answers and evaluate ThinkLeap across broader reasoning domains.

Acknowledgments

This work is supported by Open Project of National Engineering Laboratory for Technology of Internet Domain Name (No. KF202512), Guangdong S&T Programme 2024A0101010001, and Shenzhen Key Laboratory of Software Defined Networking under Grant ZDSYS20140509172959989.

References

- Elaf Alhazmi, Quan Z Sheng, Wei Emma Zhang, Munazza Zaib, and Ahoud Alhazmi. 2024. Distractor generation in multiple-choice tasks: A survey of methods, datasets, and evaluation. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pages 14437–14458.
- Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, et al. 2024. Do not think that much for $2+3=?$ on the overthinking of o1-like llms. *arXiv preprint arXiv:2412.21187*.
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.
- Yingqian Cui, Pengfei He, Jingying Zeng, Hui Liu, Xianfeng Tang, Zhenwei Dai, Yan Han, Chen Luo, Jing Huang, Zhen Li, Suhang Wang, Yue Xing, Jiliang Tang, and Qi He. 2025. [Stepwise perplexity-guided refinement for efficient chain-of-thought reasoning in large language models](#). In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 18581–18597, Vienna, Austria. Association for Computational Linguistics.
- Gongfan Fang, Xinyin Ma, and Xinchao Wang. 2026. Thinkless: Llm learns when to think. *Advances in neural information processing systems*, 38:151268–151295.
- Yichao Fu, Junda Chen, Siqi Zhu, Zheyu Fu, Zhongdongming Dai, Yonghao Zhuang, Yian Ma, Aurick Qiao, Tajana Rosing, Ion Stoica, et al. 2024. Efficiently scaling llm reasoning with certainindex. *arXiv preprint arXiv:2412.20993*.
- Jiahui Geng, Fengyu Cai, Yuxia Wang, Heinz Koepl, Preslav Nakov, and Iryna Gurevych. 2024. A survey of confidence estimation and calibration in large language models. In *Proceedings of the 2024 conference of the north American chapter of the association for computational linguistics: human language technologies (volume 1: long papers)*, pages 6577–6595.
- Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR.
- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.
- Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. 2025. Thinkprune: Pruning long chain-of-thought of llms via reinforcement learning. *arXiv preprint arXiv:2504.01296*.
- Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.
- Yuxuan Jiang, Dawei Li, and Francis Ferraro. 2026. Drp: Distilled reasoning pruning with mathematical skill-aware step decomposition for efficient large reasoning models. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 4020–4039.
- Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. 2022. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*.
- Susan Kemper, Marilyn Thompson, and Janet Marquis. 2001. Longitudinal change in language production: effects of aging and dementia on grammatical complexity and propositional content. *Psychology and aging*, 16(4):600.

- Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626.
- Dacheng Li, Shiyi Cao, Tyler Griggs, Shu Liu, Xiangxi Mo, Eric Tang, Sumanth Hegde, Kourosh Hakhmaneshi, Shishir G Patil, Matei Zaharia, et al. 2025. Llms can easily learn to reason from demonstrations structure, not content, is what matters! *arXiv preprint arXiv:2502.07374*.
- Zheng Li, Qingxiu Dong, Jingyuan Ma, Di Zhang, Kai Jia, and Zhifang Sui. 2026. Selfbudgeter: Adaptive token allocation for efficient llm reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 21135–21156.
- Marc Light, Xin Ying Qiu, and Padmini Srinivasan. 2004. The language of bioscience: Facts, speculations, and statements in between. In *HLT-NAACL 2004 workshop: linking biological literature, ontologies and databases*, pages 17–24.
- Tergel Munkhbat, Namgyu Ho, Seo Hyun Kim, Yongjin Yang, Yujin Kim, and Se-Young Yun. 2025. Self-training elicits concise reasoning in large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 25127–25152.
- Shiyu Ni, Keping Bi, Lulu Yu, and Jiafeng Guo. 2024. Are large language models more honest in their probabilistic or verbalized confidence? In *China Conference on Information Retrieval*, pages 124–135. Springer.
- Nisarg Parikh, Alexander Scarlatos, Nigel Fernandez, Simon Woodhead, and Andrew Lan. 2025. Lookalike: Consistent distractor generation in math mcqs. In *Proceedings of the 20th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2025)*, pages 294–311.
- Ziqing Qiao, Yongheng Deng, Jiali Zeng, Dong Wang, Lai Wei, Guanbo Wang, Fandong Meng, Jie Zhou, Ju Ren, and Yaoyue Zhang. 2025. Concise: Confidence-guided compression in step-by-step efficient reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 8021–8040.
- Qwen Team. 2025. [Qwq-32b: Embracing the power of reinforcement learning](#).
- Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. 2025. Codi: Compressing chain-of-thought into continuous space via self-distillation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 677–693.
- Junhan Shi, Yijia Zhu, Zhenning Shi, Dan Zhao, Qing Li, and Yong Jiang. 2025a. Speccot: Accelerating chain-of-thought reasoning through speculative exploration. In *EMNLP (Findings)*, pages 24405–24415.
- Zhenning Shi, Yijia Zhu, Yi Xie, Junhan Shi, Guorui Xie, Haotian Zhang, Yong Jiang, Congcong Miao, and Qing Li. 2025b. Reasoning under uncertainty: Efficient llm inference via unsupervised confidence dilution and convergent adaptive sampling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 32192–32206.
- Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Hanjie Chen, et al. 2025. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*.
- Veronika Vincze, György Szarvas, Richárd Farkas, György Móra, and János Csirik. 2008. The bioscope corpus: biomedical texts annotated for uncertainty, negation and their scopes. *BMC bioinformatics*, 9(Suppl 11):S9.
- Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. 2020. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17(3):261–272.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.
- Heming Xia, Chak Tou Leong, Wenjie Wang, Yongqi Li, and Wenjie Li. 2025. Tokenskip: Controllable chain-of-thought compression in llms. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 3351–3363.
- Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. 2025. Chain of draft: Thinking faster by writing less. *arXiv preprint arXiv:2502.18600*.
- Tianyang Xu, Shujin Wu, Shizhe Diao, Xiaozhe Liu, Xingyao Wang, Yangyi Chen, and Jing Gao. 2024. Sayself: Teaching llms to express confidence with self-reflective rationales. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5985–5998.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.
- Jingyang Yi, Jiazheng Wang, and Sida Li. 2026. Shorterbetter: Guiding reasoning models to find optimal inference length for efficient reasoning. *Advances in*

Neural Information Processing Systems, 38:39011–39043.

Dongkeun Yoon, Seungone Kim, Sohee Yang, Sunkyoung Kim, Soyeon Kim, Yongil Kim, Eunbi Choi, Yireun Kim, and Minjoon Seo. 2026. Reasoning models better express their confidence. *Advances in Neural Information Processing Systems*, 38:103869–103896.

Hang Yuan, Bin Yu, Haotian Li, Shijun Yang, Christina Dan Wang, Zhou Yu, Xueyin Xu, Weizhen Qi, and Kai Chen. 2025. Not all tokens are what you need in thinking. *arXiv preprint arXiv:2505.17827*.

Yuhui Zhang, Yuchang Su, Yiming Liu, Xiaohan Wang, James Burgess, Elaine Sui, Chenyu Wang, Josiah Aklilu, Alejandro Lozano, Anjiang Wei, et al. 2025. Automated generation of challenging multiple-choice questions for vision language model evaluation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 29580–29590.

A Handling Answer Ambiguity

The calculation of confidence scores requires matching the model’s output against the ground-truth answer. For datasets with short, standardized answers (e.g., GSM8K), we can directly compute the probability of the correct answer token sequence. However, free-form mathematical answers (e.g., MATH500) present a challenge, as semantically identical answers can have varied token representations (e.g., 0.09 vs. $\frac{9}{100}$ vs. $9/100$).

Beam Search with Answer Matching. We initially explored a beam search strategy: after inserting the probe prompt, we generate the top- k most probable answer candidates using beam search, then match each against the ground-truth to identify correct responses. The confidence score is calculated as the sum of probabilities for all correct candidates:

$$c_k = \sum_{i=1}^k P(A_i | S_k, P_{\text{probe}}) \cdot \mathbb{I}[A_i = A_{\text{correct}}] \quad (11)$$

where $\mathbb{I}[\cdot]$ is the indicator function. While this approach effectively handles answer variance, it incurs a k -fold increase in inference cost, as generating k candidates for each token position becomes computationally prohibitive for long reasoning chains.

Multiple-Choice Conversion. To achieve comparable effectiveness with significantly improved efficiency, we adopt a multiple-choice conversion strategy. This approach is inspired by recent work in automated question conversion (Zhang et al., 2025; Alhazmi et al., 2024; Parikh et al., 2025), which uses an auxiliary LLM to generate plausible distractors. For a given problem with ground-truth answer A_{correct} , we prompt an auxiliary LLM to generate $k - 1$ plausible but incorrect distractors. When the LLM fails to produce sufficient unique distractors, we supplement them with incorrect answers from other problems in the dataset. The final option set $\mathcal{O}$ is formed by shuffling to prevent positional bias:

$$\mathcal{O} = \text{shuffle}(\{A_{\text{correct}}\} \cup \{d_1, d_2, \dots, d_{k-1}\}) \quad (12)$$

This standardization creates an unambiguous target sequence for probability calculation, ensuring a reliable and well-calibrated confidence metric (Ni et al., 2024).

Algorithm 1 find_peak

Input: Curve $C' = \{c'_1, c'_2, \dots, c'_T\}$, Prominence τ

Output: Peak indices $\mathcal{P}$

- 1: Identify all candidate peaks $\mathcal{P}_{\text{cand}}$ where $c'_i > c'_{i-1} \wedge c'_i > c'_{i+1}$ for $i \in [2, T - 1]$.
 - 2: $\mathcal{P} \leftarrow \emptyset$ {Initialize set of significant peaks}
 - 3: **for** each peak p at index i in $\mathcal{P}_{\text{cand}}$ **do**
 - 4: Find left search boundary L_{bound} : the first index $j < i$ where $c'_j \geq c'_i$, or 1 if none exists.
 - 5: Find right search boundary R_{bound} : the first index $k > i$ where $c'_k \geq c'_i$, or T if none exists.
 - 6: $c'_{\text{base},L} \leftarrow \min(C'[L_{\text{bound}} \dots i])$
 - 7: $c'_{\text{base},R} \leftarrow \min(C'[i \dots R_{\text{bound}}])$
 - 8: $P_p \leftarrow c'_i - \max(c'_{\text{base},L}, c'_{\text{base},R})$ {Calculate prominence per Eq. 7}
 - 9: **if** $P_p > \tau$ **then**
 - 10: Add i to $\mathcal{P}$
 - 11: **end if**
 - 12: **end for**
 - 13: **return** $\mathcal{P}$
-