Logit–Entropy Adaptive Stopping Heuristic for Efficient Chain-of-Thought Reasoning

Mohammad Atif Quamar
Independent Researcher
atif7102@gmail.com

Mohammad Areeb
Purdue University
mareeb@purdue.edu

Abstract

Chain-of-Thought (CoT) prompting is a key technique for enabling complex reasoning in large language models. However, generating full, fixed-length rationales is computationally wasteful, inflating both token usage and latency. We introduce **LEASH**: **L**ogit-**E**ntropy **A**daptive **S**topping **H**euristic, a training-free decoding algorithm that adaptively halts rationale generation. **LEASH** monitors two intrinsic signals: the slope of token-level entropy and the improvement in the top-logit margin. It terminates the generation once both signals plateau, indicating the model has reached a stable reasoning state. Across four instruction-tuned models on the GSM8K and AQuA-RAT benchmarks, **LEASH** reduces average token generation by $\approx$ 30–35% and latency by $\approx$ 27%, while incurring a $\approx$ 10 p.p. accuracy drop relative to CoT. **LEASH** is model-agnostic and requires no additional training or supervision, offering a simple and efficient alternative to CoT decoding.

1 Introduction

Large language models solve many reasoning problems more reliably when prompted to "think out loud" using chain-of-thought (CoT) decoding [1]. Yet those rationales are costly: Vanilla CoT and vote-heavy schemes inflate token usage and tail latency, which limits deployment under tight budgets and interactive constraints. The core challenge is to decide, online and per question, when enough reasoning has been generated, early enough to save tokens, but not so early that accuracy suffers.

Existing approaches offer unsatisfying trade-offs. Fixed budgets ignore instance difficulty and routinely over-generate [2]. Heuristic triggers (e.g., stopping on "Therefore," or punctuation patterns) are brittle and prompt-dependent [3]. Multi-sample reranking improves quality but wastes compute on full sequences that were already off-trajectory [4]. What is missing is a training-free, model-agnostic criterion that uses signals already available at decoding time to adaptively halt reasoning without extra models, supervision, or architectural changes.

We introduce **Logit–Entropy Adaptive Stopping Heuristic (LEASH)**, a simple decoding-time algorithm that monitors two intrinsic indicators of reasoning convergence: the local slope of token-level entropy and the improvement in the top-logit margin. **LEASH** generates a brief rationale and halts when both signals plateau within a short sliding window after a small minimum length, then elicits a concise final answer. Because **LEASH** relies only on logits already produced by the base model, it is gradient-free, drop-in, and compatible with greedy or sampled decoding, quantized or full-precision inference, and common toolchains.

Prior efforts to curb over-generation in CoT either truncate at a fixed depth, add auxiliary heads or verifiers, or stop on a *level* signal such as an absolute entropy threshold over an explicit answer set [5]. In contrast, we use intrinsic token-level signals available under standard top-p sampling: the windowed slope of next-token entropy and the trend in the top-logit margin, together with a $p_{\max}$ saturation guard, to detect convergence. As our approach operates on logits already computed during decoding, it integrates with existing APIs, supports quantized inference, and extends to free-form

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Efficient Reasoning.

numeric reasoning without additional scaffolding. Empirically, this instance-wise token-granular rule yields large reductions in generated tokens and latency while maintaining competitive accuracy, offering a training-free alternative to answer-entropy halting.

Our empirical study focuses on grade-school math, where CoT is standard. On GSM8K [6] with four instruction-tuned models, **LEASH** retains $\approx 85\%$ of vanilla CoT accuracy, while using $\sim 30-35\%$ fewer tokens and cutting end-to-end latency by $\sim 25-30\%$.

To probe the generality of our findings, we report results on the test split of the AQuA-RAT dataset [7]. On this benchmark, **LEASH** closely tracks the accuracy of vanilla-CoT while reducing compute. We also examine robustness across sampling temperatures and decoding settings, finding that **LEASH** maintains performance without per-task retuning. Taken together, these results position **LEASH** as a practical, training-free alternative to vanilla CoT: it adapts rationale length to item difficulty, preserves accuracy under tight budgets, and delivers immediate token and latency savings for real-world deployments.

2 LEASH: Training-Free Stopping for Chain of Thought

We consider an instruction-tuned language model that first generates a chain-of-thought (CoT) rationale and then a short final answer. For a prompt x, let $y_{1:t}$ denote the partial rationale at step t, and let $z_t \in \mathbb{R}^V$ be the next-token logits over a vocabulary of size V.

During rationale generation, end-of-sequence termination is disabled and halting is governed by the rule in (7). After halting, a second prompt requests a short final answer.

For Numerical stability, we upcast logits to `fp32`, replace non-finite entries with zero, and clip componentwise logits to a fixed band $[-B, B]$, i.e., $\tilde{z}_t = \mathrm{clip}(\mathrm{finite}(z_t), -B, B)$.

Let $p_t = \mathrm{softmax}(\tilde{z}_t)$ denote the token probabilities and $\ell_t = \mathrm{logsoftmax}(\tilde{z}_t)$ be the log-probabilities at step t. We monitor two primary intrinsic signals: the token-level entropy H_t and the top-two log-probability margin M_t. Entropy measures the models uncertainty, while the margin measures its confidence in the top choice.

$$H_t = -\sum_{v=1}^{V} p_t(v) \log p_t(v) \tag{1}$$

$$M_t = \ell_t^{(1)} - \ell_t^{(2)} \tag{2}$$

where $\ell_t^{(1)}$ and $\ell_t^{(2)}$ are the log-probabilities of the top-two tokens. While M_t is algebraically equivalent to the logit-space margin $z_t^{(1)} - z_t^{(2)}$, we compute it using log-probabilities for numerical stability. Highly confident steps, where the peak probability $p_{\max}(t) = \max_v p_t(v)$ exceeds a threshold τ_p, are treated as "saturated" and are excluded from our trend analysis. We use an indicator Σ_t to mark these steps:

$$\Sigma_t = \mathbb{I}[p_{\max}(t) \geq \tau_p] \tag{3}$$

Windowed trends. Given a window size $k \geq 1$, we compute the k-step entropy slope s_H and the k-step margin improvement ΔM for all non-saturated steps.

$$s_H(t;k) = \frac{H_t - H_{t-k}}{k} \tag{4}$$

$$\Delta M(t;k) = M_t - M_{t-k} \tag{5}$$

Adaptive Stopping Criterion. Our stopping rule is determined by a per-step plateau test, Π_t, which is active only for non-saturated steps ($\Sigma_t = 0$). The test passes if the entropy slope has flattened and the margin improvement has stalled, given small tolerances $\varepsilon_H > 0$ and $\delta_M > 0$:

$$\Pi_t = \mathbb{I}\big[\, s_H(t;k) \geq -\varepsilon_H \,\big] \cdot \mathbb{I}\big[\, \Delta M(t;k) \leq \delta_M \,\big] \cdot \mathbb{I}\big[\, \Sigma_t = 0 \,\big] \tag{6}$$

The final stopping time τ is the first step t that satisfies three conditions: (i) It is past a minimum warm-up period, $t_{\min} = \max(m + w, k + L)$. (ii) A majority of the last L non-saturated steps

Algorithm 1 Logit–Entropy Adaptive Stopping Heuristic (**LEASH**)

Inputs: window k, vote span L, slacks $(\varepsilon_H, \delta_M)$, min/max (m, M), warmup w, saturation threshold τ_p, entropy drop γ
Initialize state and histories; disable EOS during rationale generation
for $t = 1$ **to** M **do**
  Decode next token to get the logits z_t; compute ℓ_t and p_t
  Compute entropy H_t by Eq.(1), margin M_t by Eq.(2), and the peak probability $p_{\max}(t)$
  If first k steps completed, compute the reference entropy H_{ref}
  If $t \geq t_{\min}$ **and** the $H_{\text{ref}} - H_t \geq \gamma$:
    Compute plateau votes: $votes \leftarrow \sum_{j \in \mathcal{J}_L(t)} \Pi_j$
    If $votes \geq \lceil |\mathcal{J}_L(t)|/2 \rceil$, **break**
end for
Query the model for the short final answer conditioned on the generated rationale

(indexed by $\mathcal{J}_L(t)$) have passed the plateau test. (iii) An entropy-drop gate, $H_{\text{ref}} - H_t \geq \gamma$, is satisfied, preventing premature stops. $H_{\text{ref}} = \text{median}(H_1, \ldots, H_k)$ is a reference entropy computed over the first k steps. The full stopping criterion, capped at a maximum length M, is:

$$\tau \;=\; \min\left\{ t \;\geq\; t_{\min} \;:\; \sum_{j \in \mathcal{J}_L(t)} \Pi_j \;\geq\; \left\lceil \frac{|\mathcal{J}_L(t)|}{2} \right\rceil \;\wedge\; \left(H_{\text{ref}} - H_t \geq \gamma\right) \right\} \;\wedge\; M \tag{7}$$

The rationale stage disables EOS, so halting is governed by (7). After stopping at τ, the model is prompted again to emit only the short final answer. Our complete method is given in Algorithm 1.

Implementation Details. The signals in (1)–(2) reuse logits already computed by the base model. We maintain ring buffers for the last k values of H_t and M_t, which yields $O(1)$ overhead per token in time and memory; runtime remains dominated by forward passes. The method exposes several hyperparameters (e.g., $k, L, \varepsilon_H, \delta_M, \gamma$), and concrete settings are reported in the experiments section.

Relation to Baselines. Vanilla CoT omits the adaptive stopping logic and sets $\tau = M$. Multi-sample reranking expands full-length sequences per sample. **LEASH** instead makes a per-instance sequential decision from local convergence signals, adapting the rationale length to the problem.

3 Experimental Setup

Language models. We evaluate **LEASH** on four instruction-tuned LLMs spanning different families and sizes: `Llama-3.1-8B-Instruct`[8], `Mistral-7B-v0.1`[9], `Phi-3-Mini-128k-Instruct`[10], and `Qwen2.5-7B-Instruct`[11]. All experiments use HuggingFace `transformers` with the models native tokenizers.

Tasks and datasets. We focus on math reasoning with short numeric answers. Our primary benchmark is GSM8K; we evaluate on a randomly sampled subset of n=300 test problems with a fixed seed. We also report results on the TEST SPLIT of AQuA-RAT, an algebraic word problem dataset.

Baselines. We compare **LEASH** against two decoding schemes: (i) **Vanilla-CoT**, which generates its answer using the chain-of-thought reasoning process, then a short final answer (ii) **No-CoT**, which directly predicts the final numeric answer with no rationale. All methods have prompts according to the task they need to perform.

Decoding settings. For the rationale phase we use nucleus sampling with p=0.95 and temperature 0.7 (`do_sample=True`); for the final answer, we decode with temperature 0.0. **LEASH** hyperparameters are held fixed across models unless otherwise noted: window k=8, consistency L=5, entropy slack ε_H=0.005, margin slack δ_M=0.05, minimum/maximum rationale lengths m=64, M=320.

Table 1: **Accuracy Results (↑).** We report accuracy (%) on the **GSM8K** and **AQuA-RAT** datasets for our method (**LEASH**), standard Chain-of-Thought (CoT), and vanilla decoding (No-CoT).

Model	GSM8K			AQuA-RAT		
	LEASH	CoT	No-CoT	**LEASH**	CoT	No-CoT
Llama-3.1-8B-Instruct	62.32	74.33	14.00	54.68	63.20	27.56
Mistral-7B	38.67	47.20	6.33	19.25	26.38	13.78
Phi-3-Mini-128k-Instruct	69.87	82.67	8.00	50.24	61.67	23.23
Qwen2.5-7B-Instruct	54.85	65.33	21.33	68.15	77.35	37.80

Table 2: **Efficiency Savings of LEASH vs. CoT (↑).** We report the percent reduction in generated tokens and end-to-end latency for **LEASH** relative to standard CoT on **GSM8K** and **AQuA-RAT**.

Model	GSM8K		AQuA-RAT	
	Token Red.	Latency Red.	Token Red.	Latency Red.
Llama-3.1-8B-Instruct	30.97	29.74	28.60	26.10
Mistral-7B-v0.1	35.12	27.80	34.20	27.50
Phi-3-Mini-128k-Instruct	41.50	25.15	28.30	28.75
Qwen2.5-7B-Instruct	33.45	24.90	28.15	28.10

Metrics. We evaluate all methods on three primary metrics. (i) **Accuracy** is the exact-match percentage on the final numeric answer after normalization. (ii) **Token Reduction (%)** and (iii) **Latency Reduction (%)** measure the efficiency gains of **LEASH** relative to the standard CoT baseline. Token reduction is based on the count of all generated tokens (rationale + answer), and latency reduction is based on the end-to-end time (s) per example.

4 Experimental Results

We present our main results in Table 1 (Accuracy) and Table 2 (Efficiency). We analyze the accuracy trade-offs of our method, followed by its significant efficiency gains.

Accuracy and Trade-offs. We first report the accuracy of **LEASH**, standard Chain-of-Thought (CoT), and direct-answer (No-CoT) in Table 1. As an early-stopping method, **LEASH** introduces an accuracy trade-off compared to CoT. We observe a manageable cost, with an average accuracy drop of ≈ 10.9 percentage points on GSM8K and ≈ 9.1 percentage points on AQuA-RAT. However, **LEASH** substantially outperforms No-CoT in all cases. This demonstrates that it successfully preserves the core reasoning structure of the CoT process. For instance, on GSM8K, **LEASH** accuracy on Llama-3.1-8B (62.32%) and Mistral-7B (38.67%) is **4.4**× and **6.1**× higher, respectively, than their No-CoT counterparts (14.00% and 6.33%).

Efficiency Gains. The benefits of this trade-off are the significant efficiency gains detailed in Table 2. **LEASH** achieves substantial reductions in both compute and latency across all models. On average, **LEASH** reduces the number of generated tokens by 35.3% and end-to-end latency by 26.9% on GSM8K. The savings are similarly strong on AQuA-RAT, with average reductions of 29.8% in tokens and 27.6% in latency. We observe that token savings are most pronounced on GSM8K, with Phi-3-Mini showing the largest reduction at 41.5%. In contrast, latency savings are highly consistent, particularly on AQuA-RAT, where all models cluster in a 26–29% reduction range. For example, **LEASH** reduced the latency for Llama-3.1-8B-Instruct from 4.04s (CoT) to 2.84s, achieving a **29.7%** speed-up while generating **31.0%** fewer tokens. These results confirm that **LEASH** is highly effective at reducing the computational and latency costs of CoT reasoning.

5 Conclusion

We presented the *Logit–Entropy Adaptive Stopping Heuristic* (**LEASH**), a training-free decoding-time criterion for adaptively halting chain-of-thought generation using only intrinsic signals produced

by the language model. **LEASH** monitors the windowed slope of token entropy together with the improvement in the top-logit margin and stops when both trends plateau. Across models and datasets, **LEASH** consistently reduces generated tokens and lowers end-to-end latency, with a modest reduction in accuracy. The method is model-agnostic, requires no additional training or reward models, and integrates seamlessly with standard decoding APIs, including quantized inference.

Limitations and future work. **LEASH** assumes access to token-level logits and is evaluated on short-answer math tasks; extending to long-form, non-numeric targets and tool-augmented settings is a promising direction. Analyzing theoretical stopping guarantees in Chain-of-Thought reasoning is also a crucial research direction.

References

[1] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL https://arxiv.org/abs/2201.11903.

[2] Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. Token-budget-aware llm reasoning. *arXiv preprint arXiv:2412.18547*, 2024.

[3] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *arXiv:2205.11916*, 2022. URL https://arxiv.org/abs/2205.11916.

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

[5] Yassir Laaouach. Halt-cot: Model-agnostic early stopping for chain-of-thought reasoning via answer entropy. *ICML 2025*, 2025. URL https://openreview.net/pdf?id=CX5c7C1CZa.

[6] Karl Cobbe and et al. Training verifiers to solve math word problems. In *NeurIPS*, 2021.

[7] Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation. In *ACL*, 2017.

[8] AI @ Meta. The llama 3 herd of models, 2024.

[9] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[10] Microsoft. Phi-3: Redefining what's possible with slms, 2024. URL https://arxiv.org/abs/2404.14219.

[11] An Yang et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.