CalConf: Calibrated Confidence for Efficient Test-Time Reasoning

Tangyu Jiang[1] **Haodi Wang**[2] **Yuanbing Zhu**[1]
Xiaojiang Du[3] **Xiaohua Jia**[2] **Bo Han**[1†]
[1]TMLR Group, Department of Computer Science, Hong Kong Baptist University
[2]Department of Computer Science, City University of Hong Kong
[3]Department of Electrical and Computer Engineering, Stevens Institute of Technology

Abstract

Test-time scaling boosts the reasoning performance of large language models (LLMs) on challenging tasks but incurs substantial computational overhead. The existing training-free approaches adopt token-level confidence as a proxy and terminate trajectories once the confidence falls below a threshold. However, these methods treat all low-confidence tokens uniformly, overlooking the fact that some of them actually reflect benign generations. We introduce Calibrated Confidence (CalConf), a training-free proxy for test-time scaling in LLM reasoning. The design rests on a simple empirical observation: *flawed trajectories tend to be long*. CalConf therefore learns a length-calibrated token-weight map and uses it to selectively terminate degrading trajectories during generation. We theoretically prove that CalConf attains the desired early-stop rate in practice without per-model tuning. Across five benchmarks (on mathematical reasoning, scientific reasoning, and code generation tasks) and three open-source LLMs, CalConf consistently matches or exceeds offline majority-vote accuracy while reducing generated tokens by 20–44%. Token-level analyses trace these gains directly to CalConf's ability to isolate harmful patterns that uniform-confidence baselines cannot distinguish. Our code is available at `https://github.com/tmlr-group/calconf`.

1 Introduction

Recent advances in large reasoning models have demonstrated that scaling inference-time computation can substantially improve performance on challenging tasks such as mathematical problem solving, code generation, and scientific reasoning. These approaches exploit the test-time scaling paradigm by allocating additional compute during inference [23, 37, 29, 28, 34], which has become a cornerstone of state-of-the-art reasoning systems.

Despite its effectiveness, test-time scaling often incurs substantial computational overhead because it replaces single-pass inference with multi-trajectory generation and aggregation [28]. In practice, much of the overhead is wasted on low-quality trajectories or trajectories likely to fail. This results in excessive token consumption, leading to unnecessary inference latency and increased serving costs.

Prior efforts to mitigate this issue can be broadly categorized into training-based and training-free approaches. Training-based methods aim to produce more concise reasoning by imposing explicit token budgets or by learning compressed reasoning trajectories through reinforcement learning [6, 16, 20, 35]. While effective, these approaches require costly retraining or fine-tuning and are not readily applicable to off-the-shelf models. In contrast, training-free methods adopt dynamic early-exit strategies that terminate generation once sufficient evidence for the final answer has been

†Correspondence to Bo Han (bhanml@comp.hkbu.edu.hk).

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

Case 1: Incorrect-associated bigram

Problem 14: Let N denote the number of ordered triples of positive integers (a, b, c) with $a, b, c \leq 3^6$ such that $a^3 + b^3 + c^3$ is a multiple of 3^7; find $N \bmod 1000$.
Observation: The bigram `modulo 3` appears at low-confidence positions 127 times, all exclusively within incorrect trajectories.
Trajectory excerpt: "cubes `modulo` powers of 3. Specifically, `modulo 3` ^k for some k..."
Analysis: The model retreats to the coarser modulus instead of directly working modulo 3^7, losing the fine structure needed. This early commitment to an oversimplified framework propagates through the trajectory and yields an incorrect answer.

Case 2: Correct-associated bigram

Problem 16: Find the sum of all positive integers n such that $n + 2$ divides $3(n+3)(n^2+9)$.
Observation: The bigram `n+` appears at low-confidence positions 288 times, of which 286 occur in correct trajectories.
Trajectory excerpt: "...is there a possibility that for some n, `n+` 2 divides the expression... What if `n+` 2=39, we did, works. What if `n+` 2=13, worked. `n+` 2=3, worked..."
Analysis: The low confidence reflects branching uncertainty over which candidate to verify next, while the underlying strategy of exhaustive case analysis remains sound and ultimately yields a correct answer. This benign exploration does not indicate a flawed reasoning trajectory.

Figure 1: Two low-confidence bigrams with opposite trajectory outcomes. **Left**: An incorrect-associated bigram where low confidence reflects a harmful algebraic framework. **Right**: A correct-associated bigram where low confidence reflects benign exploration within a valid strategy.

accumulated. Generally speaking, those methods exploit the model's own internal confidence as a proxy to evaluate trajectory quality [8, 4, 10]. At each decoding step, they compute a token-level confidence score and terminate a trajectory once its averaged confidence falls below a threshold. This technique relies on model-internal confidence signals, thus removing the need for auxiliary training.

Though effective, existing training-free methods share a structural oversight: they aggregate token-level confidence uniformly along a trajectory, implicitly assuming every low-confidence token is equally harmful and leads to incorrect answers. This assumption conflates two distinct sources of low confidence. Some low-confidence tokens are genuinely harmful: they mark the moment a trajectory commits to a flawed reasoning framework from which it rarely recovers. However, other low-confidence tokens are *benign*: they reflect expected branching uncertainty within a valid strategy that ultimately succeeds. Because uniform aggregation cannot separate the two, existing methods may prematurely terminate trajectories that would otherwise reach a correct answer.

Two examples using Qwen3-30B-A3B on AIME 2025 illustrate this dichotomy in Figure 1. The bigram `modulo 3` appears at low-confidence positions 127 times, exclusively within incorrect trajectories, indicating negative effects on the results. Here, the low-confidence tokens mark the moment the model retreats to a coarser modulus instead of working directly modulo 3^7, committing to an oversimplified framework that propagates to a wrong answer (i.e., harmful low-confidence tokens). Conversely, the bigram `n+` appears at low-confidence positions 288 times, 286 of which occur in correct trajectories, showing an association with successful reasoning. Here, the low confidence reflects branching uncertainty over which candidate to verify next during an exhaustive case-analysis strategy that ultimately succeeds (i.e., benign low-confidence tokens). Previous confidence-based methods are *biased* and cannot tell these cases apart: they penalize benign low-confidence tokens the same as harmful ones, terminating valid reasoning paths that would otherwise reach a correct answer.

We therefore ask:

Can we design an alternative to the confidence-based proxy that can distinguish benign generations from reasoning failure, enabling test-time scaling that is both efficient and accurate?

We propose **Calibrated Confidence (CalConf)**, a training-free proxy used for test-time scaling in LLM reasoning. CalConf calibrates the naive confidence-based proxies by learning a token-level weight map, which reweights the token-level confidence during generation so that benign generation and reasoning failure receive different scores rather than being conflated. The design rests on a simple empirical observation: *flawed trajectories tend to be long.* We exploit this by upweighting tokens that appear predominantly in shorter rollouts, preserving benign exploration while terminating protracted,

failing reasoning. We theoretically analyze CalConf using a split-conformal calibration framework and bound the deviation of its early-stop rate from the target under conditional i.i.d. sampling assumptions. We fully implement CalConf and conduct comprehensive experiments. Across five benchmarks (on mathematical reasoning, scientific reasoning, and code generation tasks) and three open-source LLMs, CalConf exceeds offline majority-vote accuracy in most settings while reducing generated tokens by 20–44%. In all, the contributions can be summarized as follows.

- We propose CalConf, a training-free proxy for test-time scaling in LLM reasoning grounded in a new observation: some low-confidence tokens are harmful, while others are benign, reflecting explorations within successful trajectories. CalConf learns a length-calibrated token-level weight map to disentangle the two, enabling selective early termination without sacrificing benign branching.
- We provide a theoretical analysis using a split-conformal calibration framework; we derive a non-asymptotic PAC bound on CalConf's early-stop rate. Under conditional i.i.d. sampling assumptions, the result bounds the deviation of the realized early-stop rate from the user-specified target.
- We demonstrate the effectiveness of CalConf across the AIME 2024/2025, Brumo 2025, GPQA Diamond, and LiveCodeBench v5 benchmarks using three open-source LLMs. Our method improves accuracy and exceeds offline majority-vote accuracy in most settings while substantially reducing generated tokens.

2 Related Work

Test-Time Scaling for LLM Reasoning. Test-time scaling improves LLM reasoning by allocating additional inference-time computation. Prompting-based approaches elicit structured intermediate steps [29, 37] or aggregate multiple sampled trajectories [28], while subsequent methods introduce more structured exploration via search, action interleaving, or self-correction [34, 33, 22]. A complementary paradigm relies on Best-of-N sampling guided by trained verifiers [6]. Recent work further formalizes test-time scaling as a distinct dimension that can rival model scaling [23, 5, 31]. Though effective, these methods incur an order-of-magnitude increase in generated tokens, which is resource-consuming and economically expensive.

Efficient Test-Time Scaling. A growing body of work aims to reduce the total tokens generated during test-time scaling, broadly split into training-based and training-free approaches. Training-based methods rely on additional training, either of the base LLM itself or of an auxiliary model. One line fine-tunes the base LLM so that each trajectory consumes fewer tokens, via budget-aware fine-tuning [9], reinforcement learning with length-aware rewards [1, 18, 32], difficulty-adaptive reasoning [3, 21], or fine-tuning to verbalize calibrated uncertainty [17]. Another line trains auxiliary process- or outcome-reward models that score and prune trajectories from a Best-of-N pool, including speculative rejection that exploits partial reward signals to terminate low-scoring candidates early [6, 16, 26, 25]. These approaches, however, all require costly retraining and are not directly applicable to off-the-shelf models. Training-free methods instead introduce a proxy for evaluating partial trajectories at inference time, so that low-quality ones can be terminated early without any extra training. A common proxy is the LLM's own intrinsic signals, such as token probabilities, entropy, or self-certainty [13, 14, 12]. Built on this idea, confidence-based early-exit methods terminate trajectories whose aggregated token-level confidence falls below a threshold [8, 4, 10]. Despite their effectiveness, existing confidence-based proxies treat all low-confidence tokens uniformly, conflating harmful errors with benign exploration and potentially terminating valid trajectories.

3 Method

3.1 Preliminaries

Notation. Let $\mathcal{V} = \{v_1, v_2, \cdots, v_{|\mathcal{V}|}\}$ denote the vocabulary. Let a trajectory t be a sequence of tokens: $t = \{x_1, x_2, \ldots, x_L\}$, $x_i \in \mathcal{V}$. We use $\mathcal{D}$ as the dataset and $\mathcal{E}$ to represent the set of trajectories generated during inference.

Confidence proxy. Consider an autoregressive language model that generates a trajectory $\{x_1, x_2, \ldots, x_L\}$ from an input prompt. At each step l, the model outputs a distribution $p(\cdot \mid x_{<l})$ over the vocabulary and samples x_l accordingly. Following prior work [8], we quantify step-l confidence using the negative average log-probability of the top-K candidates, such that

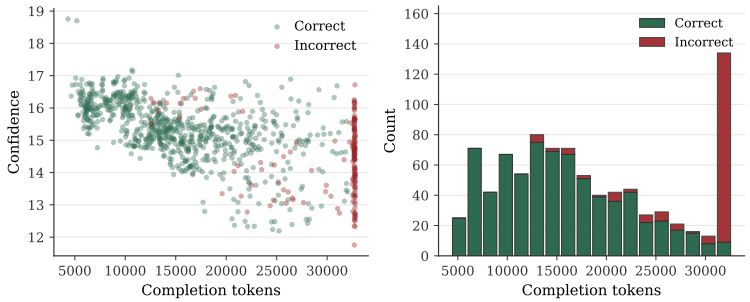
$$c_l \;\coloneqq\; -\frac{1}{K}\sum_{j=1}^{K} \log p_j(\cdot|x_{<l}), \tag{1}$$

where $p_j(\cdot|x_{<l})$ is the probability of the j-th most likely token at step l. Previous confidence-based methods aggregate c_l via uniform averaging as a proxy for trajectory quality, such that

$$\mathrm{Conf} \;\coloneqq\; \frac{1}{L}\sum_{l=1}^{L} c_l, \tag{2}$$

Thus, Conf is used to decide whether to early terminate a generation. While simple, Equation (2) implicitly assumes that all low-confidence tokens are harmful to the trajectories, and thus might terminate the exploration containing low-confidence tokens that will lead to correct answers.

Observations. We point out that using confidence alone to terminate explorations early is biased. Thus, in this paper, we propose a calibrated proxy as an alternative. Our key insight is that the length of a reasoning trajectory is a strong indicator of its correctness: flawed trajectories tend to be long. Intuitively, a long trajectory typically suggests that the model is struggling with the problem or stuck in a flawed reasoning loop, producing more tokens without reaching a valid answer. In such cases, the model is also less certain about its own output. Similar length–accuracy correlations have been observed in recent work [4, 24, 32]. As justification, we sampled 30 trajectories per problem from Qwen3-30B-A3B on AIME 2025 [36] and examined the relationship among trajectory length, correctness, and aggregated confidence. As shown in Figure 2, the number of correct trajectories decreases as length grows, while incorrect trajectories cluster in the long tail. The same trend appears in confidence: Figure 2a shows a strong correlation between the trajectory length and confidence (Spearman's $|\rho| = 0.61$), with short trajectories concentrated in the high-confidence region. In all, longer trajectories exhibit both lower confidence and lower correctness. Therefore, the length of the trajectories is a good indicator of generation quality.



(a) #Token vs. confidence ($|\rho| = 0.61$). (b) Token count vs. correctness.

Figure 2: Offline reasoning trajectories from Qwen3-30B-A3B on AIME 2025 (30 samples / problem).

3.2 CalConf: Length-Calibrated Proxy

Based on the observation, we use the trajectories' length as a calibration of the existing confidence-based proxy. Our insight is to analyze how each token in the vocabulary is distributed across trajectories of different lengths. Tokens that appear more frequently in shorter trajectories are assigned higher weights, while those predominantly in longer trajectories receive lower weights. This calibration mitigates the bias introduced by relying solely on confidence, particularly the systematic undervaluation of low-confidence tokens. To this end, we propose CalConf, a length-calibrated proxy that evaluates the quality of the trajectories without training.

Specifically, suppose $\mathcal{S} = \{t_1, \cdots, t_N\}$ is a set containing N trajectories, where each trajectory $t_i = \{x_1^i, x_2^i, ..., x_{L_i}^i\}$ contains L_i tokens. For each trajectory, we design a length-aware weight as

$$w_i \;=\; 1 + \frac{1}{\log(1 + L_i)}, \tag{3}$$

which is motivated by three desiderata: (a) *monotone decay in length*, so that short trajectories receive strictly larger weights than longer ones; (b) *bounded in practice*, so that the weight never vanishes and stays inside $(1, 2]$ for any non-trivial trajectory, preventing a few very short trajectories from dominating the estimate; and (c) *sub-linear sensitivity*, achieved by the $\log$ term, so that weights respond smoothly to typical variation in reasoning length rather than collapsing onto extreme values. For token $v \in \mathcal{V}$, let $n_i(v) = \sum_{j=1}^{L_i} \mathbf{1}[x_j^i = v]$ count its occurrences in trajectory t_i. We compute the occurrence-weighted mean

$$W_v \;=\; \frac{\sum_{i=1}^{N} n_i(v)\, w_i}{\sum_{i=1}^{N} n_i(v)}, \tag{4}$$

where $\mathbf{1}[\cdot]$ is the indicator. Each occurrence contributes w_i, including repetitions within a trajectory. We set $W_v = 1$ when $\sum_i n_i(v) = 0$. Therefore, we define the length-calibrated proxy for the i-th trajectory t_i as

$$\mathrm{CalConf}(t_i) \;=\; \frac{1}{L_i} \sum_{j=1}^{L_i} W_{v=x_j^i}\, c_j^i. \tag{5}$$

where c_j^i is the confidence of token x_j^i in Equation (1). Intuitively, tokens that appear consistently in shorter trajectories receive higher importance than those that appear in longer trajectories.

3.3 CalConf in Test-Time Reasoning

Using $\mathrm{CalConf}$ as a proxy, we can efficiently evaluate rollouts during generation and early-stop rollouts that are inferior to reduce tokens. To achieve this, we embed $\mathrm{CalConf}$ into a three-stage framework, which consists of (i) a warm-up stage that estimates W_v and a stopping threshold T from a small set, (ii) an online inference stage that uses $\mathrm{CalConf}$ and T for early stopping to generate candidate trajectories, and (iii) a standard majority voting stage that determines the final prediction. Figure 3 depicts the overall pipeline. We describe each stage in turn.

Stage 1: Warm-Up. Suppose $\mathcal{D}$ is the dataset for the inference task. Before inference, we sample a small answer set $\mathcal{S}_{\mathrm{A}}$ on $\mathcal{D}$ that has the same joint distribution as $\mathcal{E}$ via LLM inference. $\mathcal{S}_{\mathrm{A}}$ is then split into two equal-sized subsets $\mathcal{S}_{\mathrm{wgt}}$ and $\mathcal{S}_{\mathrm{thr}}$, which are used to compute W_v and T, respectively.

Specifically, we first estimate the token weights W_v from $\mathcal{S}_{\mathrm{wgt}}$ using Equations (3) and (4). We then slide a window of size m over the tokens in the trajectory $t_i \in \mathcal{S}_{\mathrm{thr}}$, and compute $C(t_i)$ as the minimum $\mathrm{CalConf}$ across all sliding windows:

$$C(t_i) \;\coloneqq\; \min_{k}\, \mathrm{CalConf}(x_{k:k+m}^i), \tag{6}$$

where $x_{k:k+m}^i$ is the token sequence in the k-th sliding window. For all $t_i \in \mathcal{S}_{\mathrm{thr}}$, we set

$$T \;=\; \mathrm{Quantile}_{\tau}\Big(\big\{C(t_i) : t_i \in \mathcal{S}_{\mathrm{thr}}\big\}\Big), \tag{7}$$

for a user-specified level $\tau \in (0, 1)$. T is the τ-quantile of the minimum sliding-window scores $\{C(t) : t \in \mathcal{S}_{\mathrm{thr}}\}$ defined in (6). Since $\mathcal{S}_{\mathcal{A}}$ has the same distribution as $\mathcal{E}$, T serves as a threshold for determining whether a rollout satisfies the criterion, and only those that fail to meet the threshold are early-stopped. Here, using a quantile rather than a fixed threshold makes T robust to scale differences across models and benchmarks.

Stage 2: Online Inference. Having W_v and T, we can adopt them during inference to dynamically terminate the rollouts that have a low $\mathrm{CalConf}$ score. More concretely, we monitor the rollout in generation by again maintaining a sliding window of size m over the rollout and compute $\mathrm{CalConf}(x_{k:k+m}^i)$ in the k-th sliding window. For any k, if $\mathrm{CalConf}(x_{k:k+m}^i) < T$, we regard trajectory t_i as low-quality and early-stop its generation. The inference will generate P candidate trajectories $\{S^{(1)}, \ldots, S^{(P)}\}$ with final answers $\{a^{(1)}, \ldots, a^{(P)}\}$.

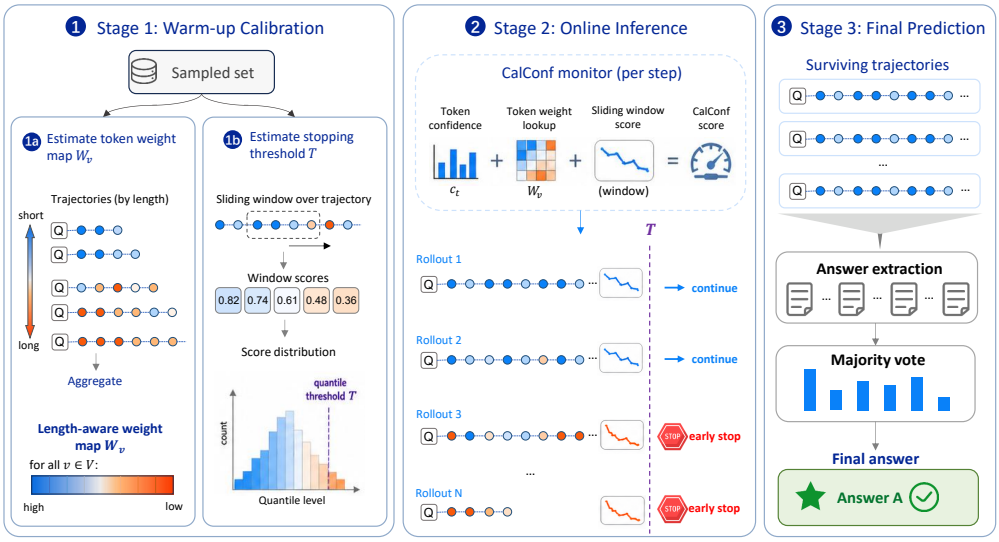


Figure 3: Overview of the CalConf framework. CalConf operates in three stages: (1) Warm-up Calibration estimates a length-aware token-weight map W_v and a stopping threshold T from a small sampled answer set; (2) Online Inference monitors each rollout's per-step weighted confidence over a sliding window and early-stops those whose window score falls below T; (3) Final Prediction aggregates the surviving trajectories' answers via majority voting.

Stage 3: Final Prediction. The final prediction $\hat{a}$ is obtained by standard majority voting,

$$\hat{a} \;=\; \arg\max_{a} \sum_{n=1}^{P} \mathbf{1}\left[a^{(n)} = a\right], \tag{8}$$

The complete algorithm of our method is summarized as Algorithm 1 in Appendix A.3.

3.4 Theoretical Analysis

Our method computes a threshold T in the warm-up stage using a small answer set $\mathcal{S}_{\mathcal{A}}$ that has the same joint distribution as $\mathcal{E}$ over (q, x). T is then adopted in the inference stage to determine the early-stopped rollouts. In this subsection, we analyze the relationship between the realized early-stop rate $\hat{r}_M$ and the desired rate τ. Let $N := |\mathcal{S}_{\text{thr}}|$ and $M := |\mathcal{E}|$. For this analysis, each evaluation rollout t_i denotes the complete trajectory that would be generated without early stopping. The online rule stops this rollout if and only if its minimum sliding-window score $C(t_i)$ in (6) falls below T. We present the following proposition.

Proposition 1 (Realized early-stop rate of CalConf)**.** *Conditional on $\mathcal{S}_{\text{wgt}}$, assume that the N calibration scores $\{C(t) : t \in \mathcal{S}_{\text{thr}}\}$ and the M evaluation scores $\mu_i := C(t_i)$ are mutually independent and identically distributed with a continuous CDF F. Let $\hat{\mu}_\tau := T$ be the empirical τ-quantile in* (7)*, with $\hat{F}_N(z) := N^{-1}\sum_{t\in\mathcal{S}_{\text{thr}}} \mathbf{1}[C(t) \le z]$, and assume $|\hat{F}_N(\hat{\mu}_\tau) - \tau| \le 2/N$. This rank-error condition holds, for example, for the inverse empirical CDF. Then the realized early-stop rate $\hat{r}_M := M^{-1}\sum_{i=1}^{M} \mathbf{1}[\mu_i < \hat{\mu}_\tau]$ satisfies, for any $\epsilon_N, \epsilon_M > 0$,*

$$\Pr\Big(\big|\hat{r}_M - \tau\big| \;\le\; \tfrac{2}{N} + \epsilon_N + \epsilon_M\Big) \;\ge\; 1 - 2\exp(-2N\epsilon_N^2) - 2\exp(-2M\epsilon_M^2). \tag{9}$$

where $\frac{2}{N}$ is a rank-rounding offset with order $\mathcal{O}(1/N)$, ϵ_N is the DKW term [7] with order $\mathcal{O}(1/\sqrt{N})$, and ϵ_M is the Hoeffding term with order $\mathcal{O}(1/\sqrt{M})$. Under the stated assumptions, Proposition 1 gives a distribution-free, finite-sample bound on the deviation of the realized early-stop rate from τ. The calibration and evaluation terms decrease as N and M increase, respectively. We provide the full proof in Appendix A.10.

Table 1: Main results across three benchmarks and three open-source LLMs. We compare Full inference, the Confidence baseline [8], and CalConf (ours, highlighted).

Model	Benchmark	Accuracy ↑			Voting ↑			Avg Tokens ↓			ES Rate		
		Full	Conf.	CalConf	Full	Conf.	CalConf	Full	Conf.	CalConf	Full	Conf.	CalConf
Qwen3-4B-Thinking	AIME 2024	74.89	78.98	**88.20**	86.67	85.19	**91.67**	19,345	13,383	12,930 (+1,901.6)	-	38.22	42.56
	AIME 2025	72.22	**89.85**	88.65	86.67	95.65	**96.15**	21,170	13,886	14,152 (+2,155.1)	-	42.00	42.22
	Brumo 2025	50.33	56.55	**63.14**	63.33	64.00	**66.67**	20,197	14,098	11,325 (+2,099.1)	-	41.44	56.89
Qwen3-30B-A3B	AIME 2024	87.22	90.51	**96.32**	**93.33**	92.31	92.59	15,852	10,725	11,283 (+1,589.0)	-	45.78	39.56
	AIME 2025	81.33	88.17	**89.05**	90.00	93.10	**96.15**	18,000	13,412	12,745 (+1,797.0)	-	32.33	38.11
	Brumo 2025	61.56	**63.93**	63.49	**73.33**	64.29	72.41	16,841	12,588	13,449 (+1,684.5)	-	40.56	32.44
openPangu-7B	AIME 2024	71.89	75.35	**89.63**	90.00	86.67	**93.10**	18,826	11,725	12,596 (+1,854.5)	-	49.78	43.22
	AIME 2025	63.22	71.41	**73.88**	80.00	82.14	**86.20**	21,137	16,879	16,512 (+2,106.4)	-	31.22	32.78
	Brumo 2025	45.00	48.64	**49.15**	63.33	51.85	**64.30**	19,765	11,980	13,749 (+1,928.9)	-	50.89	41.44

Table 2: Results on GPQA Diamond and LiveCodeBench v5 across three open-source LLMs. Each cell compares Full inference, the Confidence baseline [8], and CalConf (ours, highlighted).

Model	Dataset	Accuracy ↑			Voting ↑			Avg Tokens ↓			ES Rate		
		Full	Conf.	CalConf	Full	Conf.	CalConf	Full	Conf.	CalConf	Full	Conf.	CalConf
Qwen3-4B-Thinking	GPQA Diamond	65.22	**65.43**	62.39	**73.33**	66.67	70.37	8,661	6,689	4,987	-	35.78	57.00
	LiveCodeBench v5	42.11	44.49	**46.75**	43.33	**48.28**	46.15	10,536	6,702	5,010	-	40.56	56.00
Qwen3-30B-A3B	GPQA Diamond	70.78	71.16	**86.05**	70.00	72.41	**86.67**	7,391	4,615	1,859	-	40.89	64.89
	LiveCodeBench v5	46.67	57.61	**62.96**	46.67	58.33	**65.22**	9,052	4,715	4,181	-	45.67	50.33
openPangu-7B	GPQA Diamond	49.56	51.02	**52.57**	**83.33**	76.67	73.33	7,665	5,482	3,755	-	24.44	47.44
	LiveCodeBench v5	37.33	49.01	**66.74**	40.00	56.00	**68.75**	15,409	9,873	4,837	-	43.00	70.89

4 Experiments

4.1 Experimental Setup

Models. We evaluate our method on three open-source LLMs covering a range of scales and architectures (dense / MoE): openPangu-Embedded-7B-v1.1 [3], Qwen3-4B-Thinking, and the MoE model Qwen3-30B-A3B-Thinking-2507 [30].

Benchmarks. We use three challenging mathematical reasoning benchmarks: AIME 2024/2025, each consisting of 30 problems; and Brumo 2025, a competition-style Olympiad benchmark. We further use GPQA Diamond [19] for graduate-level scientific reasoning and LiveCodeBench v5 [11] for contamination-controlled code generation.

Evaluation metrics. We follow the settings in standard test-time scaling methods. For each problem q we sample 30 independent evaluation responses $\{y_q^{(n)}\}_{n=1}^{30}$ at temperature T_{dec}=0.8 with a maximum generation length of 32,768 tokens. We use Accuracy, Voting, average tokens (Avg Tokens), and early-stop rate (ES Rate) to evaluate the performance of our method. More details and statistical analysis can be found in Appendix A.4.

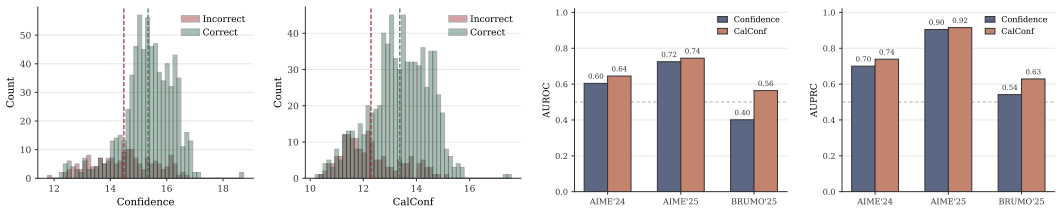
Counterpart comparisons. We compare our method against four groups of baselines: (i) Full inference (generation to completion or the 32k-token cap); (ii) the Confidence method of Fu et al. [8], an online early-stopping rule based on confidence; (iii) SC-Sel and SC-Brd [14], which leverage self-certainty for scalable best-of-N selection; and (iv) T-Gap [10], which utilizes token-level gap signals to guide LLM reasoning. All baselines share CalConf's sampling budget and voting aggregation, isolating the empirical contribution of our proxy-based early stopping.

Configurations. In Section 3, we assume $\mathcal{S}_{\mathcal{A}}$ and $\mathcal{E}$ are drawn from the same distribution. (More analysis can be found in Appendix A.6.) In practice, we use the same dataset $\mathcal{D}$ to generate $\mathcal{S}_A$ via a few offline inferences. Specifically, for each evaluation problem, we draw 33 i.i.d. rollouts. Among the rollouts, 30 are used as $\mathcal{E}$, following the standard setting. The remaining three (randomly selected) are assigned to $\mathcal{S}_{\mathcal{A}}$ (see justifications in Appendix A.9). Thus, $|\mathcal{Q}|$=30 problems of each benchmark yield $\mathcal{S}_{\mathcal{A}}$ of size 90 and $\mathcal{E}$ of size M=900 per (model, benchmark) cell. After splitting into two parts, $|\mathcal{S}_{\mathrm{wgt}}| = |\mathcal{S}_{\mathrm{thr}}| = 45$. We use a sliding window of $m = 256$ tokens and select the top-$K = 20$ values for c_l in Equation (1). The stopping criterion is based on the CalConf score defined in Equation (5).

Table 3: Voting accuracy (%) comparison on AIME 2024/2025 and Brumo 2025. All methods use 30 rollouts per problem at T_{dec}=0.8. **Bold**: best in each row. ΔTokens: percentage reduction in generated tokens relative to Full inference.

Model	Benchmark	Seq-LP	Min-Pr	SC-Sel[14]	SC-Brd[14]	T-Gap[10]	Conf.[8]	**CalConf**	**ΔTokens**
Qwen3-4B-Thinking	AIME 2024	66.67%	80.00%	66.67%	86.67%	86.67%	85.19%	**91.67%**	33.16%
	AIME 2025	66.67%	73.33%	60.00%	86.67%	86.67%	95.65%	**96.15%**	33.15%
	Brumo 2025	60.00%	50.00%	46.67%	63.33%	63.33%	64.00%	**66.67%**	43.93%
Qwen3-30B-A3B	AIME 2024	86.67%	**93.33%**	73.33%	90.00%	**93.33%**	92.31%	92.59%	28.82%
	AIME 2025	76.67%	83.33%	70.00%	86.67%	90.00%	93.10%	**96.15%**	29.19%
	Brumo 2025	56.67%	66.67%	46.67%	**73.33%**	**73.33%**	64.29%	72.41%	20.14%
openPangu-7B	AIME 2024	83.33%	86.67%	76.67%	90.00%	90.00%	86.67%	**93.10%**	33.09%
	AIME 2025	70.00%	73.33%	66.67%	83.33%	80.00%	82.14%	**86.20%**	21.88%
	Brumo 2025	33.33%	43.33%	33.33%	**66.67%**	63.33%	51.85%	64.30%	30.44%
Average		66.67%	72.22%	60.00%	80.74%	80.74%	79.47%	**84.36%**	30.42%

We set $\tau = 0.35$ across all (model, benchmark) cells. We conduct inference using eight Ascend 910B accelerators.



(a) Confidence (b) CalConf score (c) AUROC (d) AUPRC

Figure 4: Empirical separation. (a,b) Histograms of the Confidence vs. CalConf on correct (green) vs. incorrect (red) trajectories; (c,d) AUROC and AUPRC comparison.

4.2 Main Results

Table 1 reports Accuracy, voting accuracy, average generated tokens, and early-stop (ES) rate. CalConf outperforms or matches full inference and the Confidence baseline Fu et al. [8] in most settings. The improvement on Accuracy indicates that CalConf early terminates a substantial fraction of low-quality trajectories. The improvement in voting accuracy shows that this filtering also produces a cleaner aggregate prediction. At the same time, CalConf is efficient, which reduces the average number of generated tokens by 20% to 44% relative to full inference. The two settings where Confidence slightly outperforms CalConf on Accuracy come at the cost of voting accuracy (95.65% vs. 96.15% and 64.29% vs. 72.41%), indicating that the baseline is biased, which might stop correct trajectories early and thus degrade the performance. Note that in Table 1, both CalConf and the Confidence baseline report average token consumption during online inference only. For completeness, the second line of each CalConf Avg Tokens entry additionally reports the one-time warm-up overhead amortized over the evaluation rollouts. Even after charging the entire warm-up phase once, CalConf reduces average generated-token consumption by 10.14%–33.53% relative to Full inference across all settings. Beyond the math-reasoning benchmarks, Table 2 further evaluates CalConf on GPQA Diamond and LiveCodeBench v5. CalConf consistently improves Accuracy on Qwen3-30B-A3B and openPangu-7B. In particular, it raises Accuracy from 71.16% to 86.05% on GPQA Diamond with Qwen3-30B-A3B, and from 49.01% to 66.74% on LiveCodeBench v5 with openPangu-7B, while substantially reducing the average number of generated tokens. These results suggest that CalConf generalizes beyond math reasoning to scientific question answering and code-generation tasks.

Table 3 compares CalConf against a broader set of recent test-time scaling methods. All of the baselines in Table 3 require full generation, and "Conf" represents the offline method in [8]. Across all three models and benchmarks, CalConf attains the highest average voting accuracy (84.36%), exceeding the strongest competitors SC-Brd and T-Gap by 3.62 points, with the largest gains on AIME 2024/2025 across every model scale. Crucially, CalConf attains these voting gains while

Table 4: Warm-up subset sensitivity. Results are mean ± standard deviation over five seeds.

Warm-up problems	Accuracy	Voting	Avg Tokens
3 (10%)	93.61±2.20%	93.74±1.63%	8,904.75±3,049.92
6 (20%)	91.05±2.66%	91.52±2.83%	10,952.33±2,467.16
15 (50%)	92.38±3.18%	92.31±1.06%	11,437.84±1,495.51

Table 5: Five-seed warm-up stability with fixed evaluation trajectories.

Setting	Threshold CV ↓	Weighted Spearman ↑	Voting ↑	Avg Tokens ↓
Qwen3-30B-A3B / AIME 2025	1.43%	0.79±0.08	93.93±2.22%	12,622.59±863.63
openPangu-7B / GPQA Diamond	2.47%	0.87±0.03	80.57±3.46%	4,556.04±542.31

shortening generation by an average of 30.42% per problem, whereas the other selection methods operate on the full generation. This shows that calibrated early stopping not only saves compute but also yields a cleaner trajectory pool for majority voting than the baselines.

4.3 Further Analysis

Warm-up coverage. CalConf does not require an external calibration benchmark or correctness labels. For each target benchmark, it samples unlabeled warm-up rollouts from that benchmark and splits them to estimate the token weights W_v and stopping threshold T. To examine reduced warm-up coverage, Table 4 restricts the warm-up set on Qwen3-30B-A3B / AIME 2025 to problem-disjoint subsets, averaged over five seeds. All subset variants fall below the 96.15% Voting accuracy of the standard protocol, indicating that incomplete benchmark coverage yields less reliable estimates of both W_v and T. The token-cost variance decreases with coverage, and the 50% subset has the lowest Voting variance; thus, subset warm-up is a viable amortized variant, while broad target-benchmark coverage remains preferable.

Warm-up sampling stability. With evaluation trajectories fixed, we independently rebuild W_v and T from five warm-up seeds in one math and one non-math setting. As shown in Table 5, the low threshold CVs (1.43%–2.47%), high weight-map rank correlations (0.79–0.87), and small Voting deviations (2.22–3.46 points) show stable calibrated components and downstream accuracy across warm-up samples, although exact token cost varies more.

Fixed-denominator voting. In Table 1, Voting is computed separately for each method over problems with at least one retained rollout yielding a parseable answer. Early-stopped and unparseable trajectories are excluded. We increase sampling from 30 to 50 rollouts per problem across six model–benchmark settings and report conditional and fixed-denominator Voting over all 30 problems, counting no-vote problems as incorrect. Under the fixed denominator, CalConf matches or exceeds Full inference in four of the six settings and trails by one problem in the other two, while reducing inference tokens by 16.13%–20.45%. Of the seven no-vote cases, all 50 trajectories are stopped in three, while four retain only unparseable outputs. Full voting is incorrect in five and correct in two.

Distributional separation. To show that CalConf can effectively separate correct and incorrect trajectories, we visualize the distribution of the raw Conf and CalConf scores on correct vs. incorrect trajectories in Figure 4(a,b). The figure shows that Conf overlaps heavily across classes, whereas CalConf concentrates correct trajectories in a high-value tail and shifts incorrect ones to a lower value, producing a cleaner separation. This translates into higher AUROC and AUPRC on the same cell (Figure 4(c,d)), and provides a downstream explanation for the accuracy–token gains in Table 1.

Token-weight selectivity. We examine the token weights W_v learned by Equation (4) at low-confidence positions. Figure 5 (left) shows that the majority of low-confidence tokens cluster around a central region where the weight distributions on correct and incorrect trajectories largely overlap. These tokens are effectively neutral, carrying little discriminative information about trajectory quality. In contrast, a smaller subset of tokens populates the two tails, exhibiting clear directional tendencies that correlate strongly with whether a trajectory is correct or incorrect. Figure 5 (right) further illustrates this asymmetry at the bigram level. These observations indicate that W_v can concentrate discriminative weight on informative tokens, which allows CalConf to filter harmful low-confidence signals from benign ones that uniform aggregation cannot distinguish.

Table 6: Further analysis of fixed-denominator Voting with 50 rollouts per problem.

Model	Benchmark	Voting	CalConf All-30	Full All-30	Δ Tokens
Qwen3-4B-Thinking	AIME 2024	26/27 (96.30%)	26/30 (86.67%)	27/30 (90.00%)	19.32%
	AIME 2025	27/29 (93.10%)	27/30 (90.00%)	27/30 (90.00%)	17.51%
	Brumo 2025	20/30 (66.67%)	20/30 (66.67%)	19/30 (63.33%)	19.63%
Qwen3-30B-A3B	AIME 2024	27/29 (93.10%)	27/30 (90.00%)	28/30 (93.33%)	20.45%
	AIME 2025	28/29 (96.55%)	28/30 (93.33%)	28/30 (93.33%)	19.35%
	Brumo 2025	22/29 (75.86%)	22/30 (73.33%)	22/30 (73.33%)	16.13%

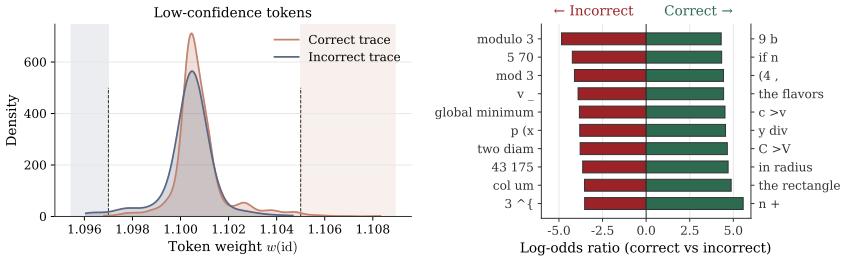


Figure 5: Distribution of W_v for low-confidence tokens (Qwen3-30B-A3B on AIME 2025). **Left**: KDE of W_v. **Right**: trajectory-direction divergence for representative bigrams.

Ablation Study. We isolate the contribution of the length-calibrated token weights by setting $w_i \equiv 1$, which makes W_v uniform in Equation (4), while keeping the sliding-window rule and all other components unchanged. We evaluate this no-weight variant across four model–benchmark settings. As shown in Table 7, restoring W_v consistently improves Voting by 2.67–12.45 percentage points in every setting, demonstrating that the weight map provides a direct and consistent contribution across two models and four model–benchmark settings.

Table 7: W/O the length-calibrated token weights.

Model	Benchmark	w/o Weights	CalConf	Gain
Qwen3-4B-Thinking	AIME 2024	85.19%	91.67%	+6.48 pp
	Brumo 2025	64.00%	66.67%	+2.67 pp
openPangu-7B	AIME 2024	86.67%	93.10%	+6.43 pp
	Brumo 2025	51.85%	64.30%	+12.45 pp

Hyperparameter Sensitivity. Figure 6 shows the performance of CalConf under different τ. As τ increases, Avg Tokens consistently decreases on both AIME 2024 and AIME 2025, indicating that more aggressive early stopping effectively reduces inference cost. Meanwhile, Accuracy generally declines, suggesting that overly large τ may terminate useful reasoning trajectories. Overall, τ controls the accuracy–efficiency trade-off, and a small-to-moderate quantile level provides a better balance. We set $\tau = 0.35$. More results can be found in Appendix A.9.

5 Conclusion

We present CalConf, a training-free proxy for efficient test-time scaling that disentangles benign from harmful low-confidence tokens via a length-calibrated token-weight map. Within the split-conformal framework, we further establish a non-asymptotic PAC bound on CalConf's realized early-stop rate under conditional i.i.d. sampling assumptions. Across five benchmarks and three open-source LLMs, CalConf matches or exceeds offline majority-vote accuracy in most settings while reducing generated tokens by 20–44%, demonstrating that calibrated confidence is a more reliable proxy than uniform aggregation for efficient LLM reasoning.

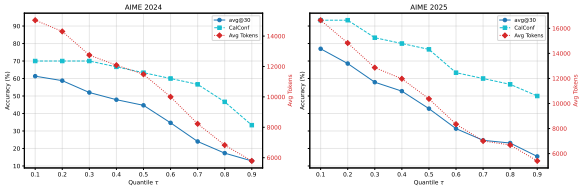


Figure 6: CalConf under different τ.

Acknowledgments and Disclosure of Funding

TYJ, YBZ, and BH were supported by RGC General Research Fund No. 12202026, RGC Research Fellow Scheme No. RFS2627-2S02, RGC Young Collaborative Research Grant No. C2005-24Y, RGC General Research Fund No. 12200725, NSFC Major Research Plan No. 92570109, NSFC General Program No. 62376235, Guangdong and Hong Kong Universities "1+1+1" Joint Research Collaboration Scheme, HKBU Faculty Niche Research Areas No. RC-FNRA-IG/25-26/SCI/05, and HKBU CSD Departmental Incentive Scheme. XHJ and HDW were supported by HK RGC research grant No.: SRFS2425-1S01.

References

[1] Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=4jdIxXBNve.

[2] Kazuoki Azuma. Weighted sums of certain dependent random variables. *Tohoku Mathematical Journal, Second Series*, 19(3):357–367, 1967.

[3] Hanting Chen, Yasheng Wang, Kai Han, Dong Li, Lin Li, Zhenni Bi, Jinpeng Li, Haoyu Wang, Fei Mi, Mingjian Zhu, et al. Pangu embedded: An efficient dual-system llm reasoner with metacognition. *arXiv preprint arXiv:2505.22375*, 2025.

[4] Wei-Lin Chen, Liqian Peng, Tian Tan, Chao Zhao, Blake JianHang Chen, Ziqian Lin, Alec Go, and Yu Meng. Think deep, not just long: Measuring llm reasoning effort via deep-thinking tokens. *arXiv preprint arXiv:2602.13517*, 2026.

[5] Yanxi Chen, Xuchen Pan, Yaliang Li, Bolin Ding, and Jingren Zhou. A simple and provable scaling law for the test-time compute of large language models. *arXiv e-prints*, pages arXiv–2411, 2024.

[6] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[7] Aryeh Dvoretzky, Jack Kiefer, and Jacob Wolfowitz. Asymptotic minimax character of the sample distribution function and of the classical multinomial estimator. *The Annals of Mathematical Statistics*, pages 642–669, 1956.

[8] Yichao Fu, Xuewei Wang, Yuandong Tian, and Jiawei Zhao. Deep think with confidence. In *NeurIPS 2025 Workshop on Efficient Reasoning*, 2025.

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

[10] Jaehui Hwang, Dongyoon Han, Sangdoo Yun, and Byeongho Heo. Oops, wait: Token-level signals as a lens into llm reasoning. *arXiv preprint arXiv:2601.17421*, 2026.

[11] Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *The Thirteenth International Conference on Learning Representations*, 2025.

[12] Tangyu Jiang, Haodi Wang, and Rongfang Bie. Meco: Zero-shot nas with one data and single forward pass via minimum eigenvalue of correlation. *Advances in neural information processing systems*, 36: 61020–61047, 2023.

[13] Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

[14] Zhewei Kang, Xuandong Zhao, and Dawn Song. Scalable best-of-n selection for large language models via self-certainty. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

[15] Jing Lei, Max G'Sell, Alessandro Rinaldo, Ryan J Tibshirani, and Larry Wasserman. Distribution-free predictive inference for regression. *Journal of the American Statistical Association*, 113(523):1094–1111, 2018.

[16] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The twelfth international conference on learning representations*, 2023.

[17] Stephanie Lin, Jacob Hilton, and Owain Evans. Teaching models to express their uncertainty in words. *arXiv preprint arXiv:2205.14334*, 2022.

[18] Haotian Luo, Li Shen, Haiying He, Yibo Wang, Shiwei Liu, Wei Li, Naiqiang Tan, Xiaochun Cao, and Dacheng Tao. O1-pruner: Length-harmonizing fine-tuning for o1-like reasoning pruning. *arXiv preprint arXiv:2501.12570*, 2025.

[19] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

[20] Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding progress: Scaling automated process verifiers for LLM reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=A6Y7AqlzLW.

[21] Yi Shen, Jian Zhang, Jieyun Huang, Shuming Shi, Wenjing Zhang, Jiangze Yan, Ning Wang, Kai Wang, Zhaoxiang Liu, and Shiguo Lian. Dast: Difficulty-adaptive slow-thinking for large reasoning models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 2322–2331, 2025.

[22] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

[23] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

[24] Jinyan Su, Jennifer Healey, Preslav Nakov, and Claire Cardie. Between underthinking and overthinking: An empirical study of reasoning length and correctness in llms. *arXiv preprint arXiv:2505.00127*, 2025.

[25] Hanshi Sun, Momin Haider, Ruiqi Zhang, Huitao Yang, Jiahao Qiu, Ming Yin, Mengdi Wang, Peter Bartlett, and Andrea Zanette. Fast best-of-n decoding via speculative rejection. *Advances in Neural Information Processing Systems*, 37:32630–32652, 2024.

[26] Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

[27] Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic learning in a random world*. Springer, 2005.

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

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

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

[31] Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Minghui Chen, Zheng Lin, and Weiping Wang. Dynamic early exit in reasoning models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=NpU7ZXafRi.

[32] Wenkai Yang, Shuming Ma, Yankai Lin, and Furu Wei. Towards thinking-optimal scaling of test-time compute for LLM reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=6ICFqmixlS.

[33] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*, 2022.

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

[35] Kaiwen Zha, Zhengqi Gao, Maohao Shen, Zhang-Wei Hong, Duane S Boning, and Dina Katabi. RL tango: Reinforcing generator and verifier together for language reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=JRkFZl0TJ2.

[36] Zhang, Yifan and Math-AI, Team. American invitational mathematics examination (aime) 2025, 2025.

[37] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, et al. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*, 2023.

A Technical Appendices and Supplementary Material

This appendix contains (i) limitations of the approach (Section A.1), (ii) broader impact of this work (Section A.2), (iii) the full CalConf algorithm pseudocode (Section A.3), (iv) extended empirical analysis deferred from the main text (Sections A.4 to A.9), and (v) proofs of all theoretical results together with two auxiliary concentration bounds (Section A.10).

A.1 Limitations

We analyze the limitations of CalConf as follows. First, CalConf rests on the observation that flawed trajectories tend to be long. This makes CalConf more suitable for thinking models and difficult tasks. We analyze more in Appendix A.5. Second, our theoretical analysis focuses on the early-stop rate matching the target τ, not on accuracy preservation. The tendency of CalConf to terminate trajectories that would yield incorrect answers is supported empirically. Third, CalConf needs the top-K token log-probabilities at every decoding step. This is fine for open-weight models, but most closed-source APIs (GPT, Claude, Gemini) do not expose per-step top-K logprobs, so CalConf cannot be applied to them as-is.

A.2 Broader Impacts

CalConf improves inference efficiency by reducing token generation and associated compute, which can lower energy consumption and operational cost of large language models. This can make advanced AI systems more accessible and environmentally sustainable. At the same time, increased efficiency may also lower the barrier to large-scale deployment, which could amplify existing risks of LLMs such as misuse for generating misleading content, automated spam, or other unintended applications. Since CalConf does not modify model behavior or training data, it does not introduce new model-specific risks, but rather inherits those of the underlying systems. Potential mitigations include standard safeguards such as usage monitoring, rate limiting, and alignment techniques applied at the system level.

A.3 Full CalConf Algorithm

For completeness, Algorithm 1 gives the end-to-end CalConf procedure used in all main-text experiments, combining the calibration-stage estimation of token weights and stopping threshold with the online weighted sliding-window early-stopping rule and final majority-vote aggregation.

A.4 Statistical Reliability

The Accuracy in Section 4 is computed as

$$\text{Accuracy} = \frac{\sum_{q\in\mathcal{Q}} \sum_{n=1}^{30} r_q^{(n)} \mathbf{1}\left[\hat{a}_q^{(n)} = a_q^\star\right]}{\sum_{q\in\mathcal{Q}} \sum_{n=1}^{30} r_q^{(n)}}. \tag{10}$$

Here, $r_q^{(n)} \in \{0,1\}$ indicates whether the n-th rollout for problem q is retained for accuracy evaluation, $\hat{a}_q^{(n)}$ denotes its parsed final answer, and $a_q^\star$ is the reference answer to problem q. Rollouts

Algorithm 1 Calibrated Confidence-Guided Reasoning (CalConf)

Require: calibration rollouts $\mathcal{S}_{\mathcal{A}}$, target questions $\mathcal{Q}$, model $\mathcal{M}$, rollouts per question α, window size m, top-K, quantile level τ

Sample calibration trajectories on $\mathcal{D}$ and store token log-probabilities
Split $\mathcal{S}_{\mathcal{A}}$ into $\mathcal{S}_{\mathrm{wgt}}$ and $\mathcal{S}_{\mathrm{thr}}$
Estimate token weights W_v from $\mathcal{S}_{\mathrm{wgt}}$ via Equation (3)–Equation (4)
Compute trajectory statistics on $\mathcal{S}_{\mathrm{thr}}$ and set threshold T via Equation (7)
for each question $q \in \mathcal{Q}$ **do**
    for $n = 1$ to α **do**
        Initialize sliding window $\mathcal{B} \leftarrow \emptyset$
        while generation not finished **do**
            Sample next token from $\mathcal{M}$ and obtain top-K log-probabilities
            Compute the top-K confidence c_l (Section 3.1) and weighted score $\tilde{c}_l = W_{x_l} c_l$
            Update $\mathcal{B}$ with $\tilde{c}_l$
            if $|\mathcal{B}| = m$ **and** $\mathrm{mean}(\mathcal{B}) < T$ **then**
                Stop generation (force EOS)
            end if
        end while
    end for
    Return $\hat{a} = \arg\max_a \sum_{n=1}^{P} \mathbf{1}[a^{(n)} = a]$
end for

with $r_q^{(n)} = 0$ contribute to neither the numerator nor the denominator. As analyzed before, when computing Accuracy, we consider the benchmark problems for which at least one retained rollout yields a parseable answer only.

To justify the number of samples used in CalConf, i.e., 30 samples per problem, we quantify how much the offline Avg@30 itself can fluctuate due to sampling variation, and check whether CalConf's gains lie outside this range. We use a non-parametric bootstrap to quantify this uncertainty. For each model and benchmark, we resample the 30 problems with replacement 10^4 times, recompute the offline Avg@30 on each resample, and report the resulting 95% confidence interval. Table 8 summarizes the intervals for the three thinking models on AIME 2024, AIME 2025, and Brumo 2025; the half-widths range from ± 9.9 to ± 16.3 percentage points, reflecting the limited sample size rather than any property of CalConf. CalConf's improvements lie clearly outside this noise floor on most settings. On Qwen3-4B-Thinking with AIME 2025, the offline mean is 72.2% with a confidence interval of $[58.9, 84.7]$, while CalConf reaches 88.65%—a $+16.4$-point improvement that exceeds the upper bound of the interval. On openPangu-7B with AIME 2024, CalConf improves Avg@30 by $+17.7$ points, again beyond the ± 12.7 half-width of the offline interval. Even on Qwen3-30B-A3B, where the intervals are narrowest (± 9.9 on AIME 2024 and ± 11.4 on AIME 2025), CalConf still delivers improvements of $+9.1$ and $+7.7$ points. Across all three models and three benchmarks, CalConf's Avg@30 is higher than the offline mean in every setting, supporting the claim that the gains in Table 1 are not artifacts of small-sample variation. The non-thinking model setting, where confidence signals are weaker, is analyzed separately in Section A.5.

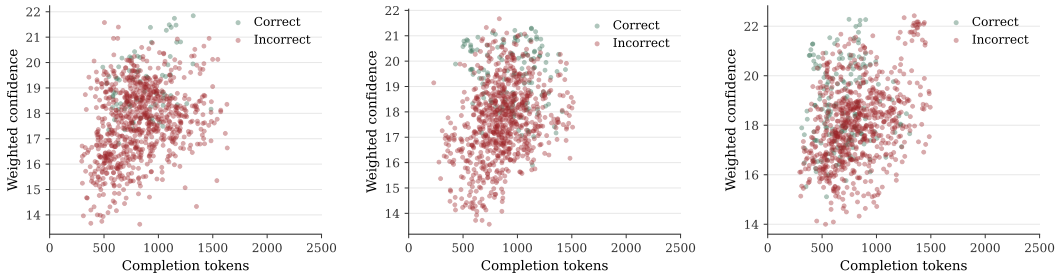
Table 8: 95% bootstrap confidence intervals for the *offline* baseline Avg@30.

Model	AIME 2024	AIME 2025	Brumo 2025
Qwen3-30B-A3B	$87.2_{\,[76.1,\,95.9]}$	$81.3_{\,[69.2,\,92.0]}$	$61.6_{\,[46.8,\,76.0]}$
Qwen3-4B-Thinking	$74.9_{\,[61.7,\,86.4]}$	$72.2_{\,[58.9,\,84.7]}$	$50.3_{\,[34.3,\,66.8]}$
openPangu-7B	$71.9_{\,[58.6,\,83.9]}$	$63.2_{\,[49.8,\,76.4]}$	$45.0_{\,[30.6,\,59.8]}$

A.5 Scope of CalConf: Thinking vs. Non-Thinking Models

CalConf is established on the observation that the flawed trajectories tend to be long. We point out that this length–accuracy correlation is only prominent in thinking models. On non-thinking models, the length of the trajectories has low correlation with the confidence. For instance, we

conduct case studies on Qwen2.5-32B-Instruct, a dense instruction-tuned model that produces short, non-chain-of-thought responses, on the same three benchmarks. Figure 7 plots weighted confidence against completion length on AIME 2024, AIME 2025, and Brumo 2025. In contrast to the strong length–confidence correlation observed on thinking models, Qwen2.5-32B-Instruct exhibits only a weak correlation (i.e., Spearman coefficients of 0.279, 0.294, and 0.259, respectively), showing that the length-induced confidence collapse exploited by CalConf is specific to long-horizon reasoning and does not transfer to short instruction-tuned generations. The scatter plots also reveal substantial within-length dispersion: trajectories of similar length span wide confidence ranges, and correct and incorrect responses overlap heavily, so confidence carries much less discriminative information in this regime. We therefore position CalConf as a method for challenging tasks using thinking models, rather than a remedy for models whose signals are uninformative to begin with.



(a) AIME 2024 (b) AIME 2025 (c) Brumo 2025

Figure 7: Weighted confidence vs. completion length for *Qwen2.5-32B-Instruct* (a non-thinking model) on the three benchmarks.

A.6 Distributions of $\mathcal{S}_{\mathcal{A}}$ and $\mathcal{E}$

Another important assumption of CalConf is that $\mathcal{S}_{\mathcal{A}}$ and $\mathcal{E}$ come from the same distribution. In practice, we guarantee this by sampling $\mathcal{S}_{\mathcal{A}}$ offline from the dataset $\mathcal{D}$ during the warm-up stage. For comprehensiveness, we conduct experiments to show the performance of CalConf when $\mathcal{S}_{\mathcal{A}}$ and $\mathcal{E}$ come from *different* distributions. Specifically, we fix the evaluation target to AIME 2025 and change the source of the calibration set used to fit the token weights W_v and the stopping threshold T in the warm-up stage. Three $\mathcal{S}_{\mathcal{A}}$ sources are compared: (a) AIME 2025 itself via a per-question random split (in-domain reference; identical to the main experiments of Section 4.1), (b) AIME 2023 (same problem style, different year), and (c) MATH500 [16] (same task family, but a substantially different sub-distribution of competition math). Table 9 reports pass@1 (per-rollout correctness), majority-vote over the 30 AIME 2025 evaluation rollouts (Maj@30), average generated tokens, and the early-stop (ES) rate.

Table 9: Calibration-source transfer with evaluation fixed to AIME 2025. Each row uses a different calibration source to fit W_v and T; the evaluation set is always AIME 2025.

Model	Calibration source	pass@1	Voting	Avg Tokens	ES Rate
Qwen3-30B-A3B	AIME 2025 (in-domain)	55.06%	83.33%	12,751	38.2%
	AIME 2023 (cross-year)	52.96%	80.95%	10,118	45.2%
	MATH500 (cross-domain)	17.78%	50.00%	5,970	81.2%
openPangu-7B	AIME 2025 (in-domain)	49.67%	83.33%	16,512	32.8%
	MATH500 (cross-domain)	14.78%	40.00%	7,752	83.6%

It can be seen from the table: (i) Cross-year transfer is benign. $\mathcal{S}_{\mathcal{A}}$ on AIME 2023 (different year but essentially the same problem style and difficulty distribution) and $\mathcal{E}$ on AIME 2025 costs Qwen3-30B-A3B only 2.10 pass@1 points and 2.38 Maj@30 points relative to in-domain AIME 2025 calibration; the realized ES rate rises modestly from 38.2% to 45.2%. Although strict exchangeability is broken across years, the observed early-stop rate remains closer to the target than under cross-domain calibration, and (ii) Cross-domain transfer degrades substantially. When $\mathcal{S}_{\mathcal{A}}$ source is MATH500, a markedly different sub-distribution of competition math, and the learned W_v, T are then applied

to AIME 2025 rollouts, Qwen3-30B-A3B drops to 17.78% pass@1 / 50.00% Maj@30 with the ES rate more than doubling to 81.2%; openPangu-7B exhibits the same pattern (14.78% pass@1, 83.6% ES). The MATH500-fitted threshold T is miscalibrated for AIME 2025's confidence profile, causing premature termination of AIME trajectories that would otherwise have completed successfully.

A.7 Long-but-Correct Trajectories

To examine whether CalConf disproportionately terminates naturally long but correct reasoning, we retrospectively audit the full, unstopped trajectories from Qwen3-4B-Thinking and openPangu-7B on AIME 2024 and AIME 2025. For each model–benchmark setting, we select the longest 25% of trajectories, replay CalConf's prefix-only stopping rule, and inspect the full trajectory only to determine whether each stopped trajectory would ultimately yield a correct final answer. Table 10 reports the fraction of CalConf-stopped trajectories within this longest quartile whose eventual answers are incorrect.

Table 10: Retrospective audit of the longest 25% of trajectories.

Model	Benchmark	Incorrect among stopped trajectories
Qwen3-4B-Thinking	AIME 2024	83.12%
Qwen3-4B-Thinking	AIME 2025	97.18%
openPangu-7B	AIME 2024	94.90%
openPangu-7B	AIME 2025	100.00%

Across the four settings, 83.12%–100% of the long trajectories stopped by CalConf would eventually produce incorrect answers. Thus, although CalConf can stop some long-but-correct trajectories, the large majority of the long trajectories it removes are unsuccessful attempts. This pattern is relative to problem difficulty: a difficult problem may require a long correct solution, while its failed attempts tend to be even longer. Since CalConf stops trajectories according to their local weighted-confidence scores rather than length alone, these results indicate a favorable trade-off between retaining correct reasoning and avoiding wasted computation.

A.8 Trajectory-Level Ranking Quality

We further compare CalConf with every baseline that defines a continuous trajectory-level score. We compute AUROC on the 30 evaluation trajectories per problem in each of six settings—Qwen3-4B-Thinking and Qwen3-30B-A3B on AIME 2024, AIME 2025, and Brumo 2025—and report the equal-weight macro-average across settings in Table 11.

Table 11: Macro-average trajectory-level AUROC across six model–benchmark settings.

Method	AUROC ↑
Seq-LP	0.564
Min-Pr	0.699
Self-Certainty	0.305
T-Gap	0.702
Confidence	0.723
CalConf	$\mathbf{0.724}$

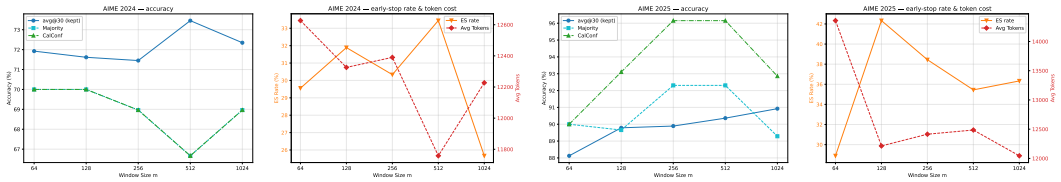
CalConf achieves the highest macro-average AUROC (0.724), essentially matching the strong Confidence baseline (0.723) while outperforming the remaining continuous-score baselines. This supplementary ranking analysis is consistent with CalConf's ability to separate correct and incorrect trajectories; the primary evaluation remains its online accuracy–efficiency trade-off.

A.9 Hyperparameter Sensitivity: m and $|\mathcal{S}_{\mathcal{A}}|$

Except for τ, CalConf exposes two other hyperparameters that we sweep here on the Qwen3-30B-A3B / AIME 2025 setup, again replaying early stopping on cached offline log-probabilities: the sliding

window size m and the number of calibration rollouts per question (denoted as $\alpha_{cal} := |\mathcal{S}_{\mathcal{A}}|/|\mathcal{Q}|$). The quantile level τ is fixed at 0.35 throughout, as described in Section 4.1.

Window size m. Figure 8 sweeps $m \in \{64, 128, 256, 512, 1024\}$. The default m=256 sits in the middle of the plateau and is the natural choice; smaller windows (e.g., m=64) react faster but are noisier, while larger windows smooth out transient drops at the cost of slightly delayed stopping.



(a) AIME 2024 (b) AIME 2024 (c) AIME 2025 (d) AIME 2025

Figure 8: Sensitivity to the sliding window size m on AIME 2024 and 2025 (Qwen3-30B-A3B).

Size of $\mathcal{S}_{\mathcal{A}}$. Figure 9 shows the number of rollouts per question $\alpha_{cal} \in \{1, 2, 3\}$. It can be seen that the calibration saturates at α_{cal}=2: the Maj@30 gap between α_{cal}=2 and α_{cal}=3 is below 1 point on every metric we tracked, so larger rollout budgets (e.g., $\alpha_{cal} \in \{5, 8\}$) are unnecessary. The default configuration (α_{cal}=3, thus $|\mathcal{S}_{\mathcal{A}}| = 90$) therefore lies comfortably on the saturation plateau.

A.10 Proofs and Additional Theoretical Results

This section provides (i) the full proof of Proposition 1 from the main text and (ii) two auxiliary Hoeffding-type concentration bounds on the estimated token weights $\hat{W}_v$ and the sliding-window statistic. For Proposition 1, we use the conditional i.i.d. score assumption stated in the proposition, following the split-calibration setup [27, 15]. Throughout its proof, we condition on $\mathcal{S}_{\text{wgt}}$, fixing the token-weight map, and let F denote the common conditional score CDF. For the auxiliary results, $\bar{P}(x) := \mathbb{E}_{q\sim\mathcal{Q}}[P(x \mid q)]$ denotes the marginal rollout law for a problem distribution $\mathcal{Q}$; their assumptions are stated separately.

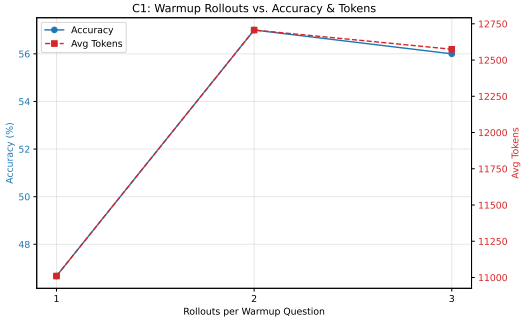


Figure 9: Rollouts per calibration question on AIME 2025 (Qwen3-30B-A3B, m=256, τ=0.35).

Proof of Proposition 1. We bound $|\hat{r}_M - \tau|$ via the triangle decomposition

$$|\hat{r}_M - \tau| \;\le\; |\hat{r}_M - F(\hat{\mu}_\tau)| \;+\; |F(\hat{\mu}_\tau) - \tau|,$$

where $F(z) := \Pr(C(t) \le z \mid \mathcal{S}_{\text{wgt}})$ is the common conditional CDF of the minimum sliding-window score in (6). All probability statements in Steps 1–3 below are conditional on $\mathcal{S}_{\text{wgt}}$. Throughout this proof, $\mu_i = C(t_i)$ and $\hat{\mu}_\tau = T$, so $\hat{r}_M$ counts precisely the rollouts stopped by the online rule. We control the two terms using separate concentration arguments. As assumed in Proposition 1, F is continuous; under the independent sampling assumption used below, an evaluation score equals the calibrated threshold with probability zero, so the strict stopping event has probability $F(\hat{\mu}_\tau)$.

Step 1 (Concentration of $\hat{r}_M$ around $F(\hat{\mu}_\tau)$). Under the assumptions of Proposition 1, conditional on $\mathcal{S}_{\text{wgt}}$, the calibration scores in $\mathcal{S}_{\text{thr}}$ and the M evaluation scores are mutually independent draws from F. The threshold $\hat{\mu}_\tau$ is a deterministic function of $\mathcal{S}_{\text{thr}}$ and the fixed token-weight map. Conditional additionally on $\hat{\mu}_\tau$, the indicators $\{\mathbf{1}[C(t_i) < \hat{\mu}_\tau]\}_{i=1}^{M}$ are therefore i.i.d. Bernoulli variables with mean $F(\hat{\mu}_\tau)$. Hoeffding's inequality applied conditionally and then marginalised over $\hat{\mu}_\tau$ gives, for any $\epsilon_M > 0$,

$$\Pr\big(|\hat{r}_M - F(\hat{\mu}_\tau)| \ge \epsilon_M\big) \;\le\; 2\exp(-2M\epsilon_M^2). \tag{11}$$

Step 2 (Concentration of $F(\hat{\mu}_\tau)$ around τ). For calibration rollouts $t^{(j)} \in \mathcal{S}_{\text{thr}}$, let $\mu^{(j)} := C(t^{(j)})$ and $\hat{F}_N(x) := \frac{1}{N}\sum_{j=1}^{N} \mathbf{1}[\mu^{(j)} \le x]$ denote the empirical cumulative distribution function of their

minimum sliding-window scores. By the rank-error condition in Proposition 1,

$$\left|\hat{F}_N(\hat{\mu}_\tau) - \tau\right| \;\le\; \tfrac{2}{N} \tag{12}$$

For example, if $\hat{\mu}_\tau$ is the $\lceil N\tau \rceil$-th smallest calibration score, continuity gives $\hat{F}_N(\hat{\mu}_\tau) = \lceil N\tau \rceil / N$ almost surely, so the rank error is at most $1/N$. The Dvoretzky–Kiefer–Wolfowitz inequality [7] applied to the i.i.d. calibration scores gives, for any $\epsilon_N > 0$,

$$\Pr\Big(\sup_{x \in \mathbb{R}} |\hat{F}_N(x) - F(x)| \ge \epsilon_N \Big) \;\le\; 2\exp(-2N\epsilon_N^2). \tag{13}$$

On the complementary event, $|F(\hat{\mu}_\tau) - \hat{F}_N(\hat{\mu}_\tau)| \le \epsilon_N$, and combining with (12) via the triangle inequality yields

$$|F(\hat{\mu}_\tau) - \tau| \;\le\; |F(\hat{\mu}_\tau) - \hat{F}_N(\hat{\mu}_\tau)| + |\hat{F}_N(\hat{\mu}_\tau) - \tau| \;\le\; \epsilon_N + \tfrac{2}{N}. \tag{14}$$

Step 3 (Combine). A union bound over the failure events of (11) and (13), together with the triangle inequality

$$|\hat{r}_M - \tau| \;\le\; |\hat{r}_M - F(\hat{\mu}_\tau)| + |F(\hat{\mu}_\tau) - \tau| \;\le\; \epsilon_M + \epsilon_N + \tfrac{2}{N},$$

yields the stated bound with conditional probability at least $1 - 2\exp(-2M\epsilon_M^2) - 2\exp(-2N\epsilon_N^2)$. Since this lower bound does not depend on $\mathcal{S}_{\mathrm{wgt}}$, averaging over $\mathcal{S}_{\mathrm{wgt}}$ gives the unconditional statement in Proposition 1. □

Auxiliary bound 1: concentration of $\hat{W}_v$. We assume throughout that all calibration rollouts in $\mathcal{S}_{\mathrm{wgt}}$ have length $L_i \ge 2$, which holds for any non-trivial reasoning trajectory; under this assumption $\log(1 + L_i) \ge \log 3$, so the length-aware sample weight $w_i = 1 + 1/\log(1 + L_i)$ is bounded in the interval $[1,\ 1 + 1/\log 3]$ with range $R \coloneqq 1 + 1/\log 3 - 1$.

Proposition 2 (Token-weight concentration)**.** *Fix a token v and let $N_{\mathrm{w}} := |\mathcal{S}_{\mathrm{wgt}}|$. Suppose the calibration rollouts are i.i.d. draws from $\bar{P}$. Let $n_i(v)$ count all occurrences of v in rollout i, and assume $0 \le n_i(v) \le B_v$ almost surely for a deterministic bound $B_v > 0$, such as the generation-length cap. Define the occurrence-weighted estimator and its population target by*

$$\hat{W}_v := \begin{cases} \dfrac{\sum_{i=1}^{N_{\mathrm{w}}} n_i(v) w_i}{\sum_{i=1}^{N_{\mathrm{w}}} n_i(v)}, & \sum_i n_i(v) > 0, \\ 1, & \sum_i n_i(v) = 0, \end{cases} \qquad W_v^\star := \frac{\mathbb{E}[n(v)w]}{\mathbb{E}[n(v)]},$$

where the expectations are under $\bar{P}$ and $\mu_v := \mathbb{E}[n(v)] > 0$. Then, for any $\epsilon > 0$,

$$\Pr\big(|\hat{W}_v - W_v^\star| \ge \epsilon\big) \;\le\; 2\exp\left(-\frac{2N_{\mathrm{w}}\mu_v^2\epsilon^2}{B_v^2 R^2}\right).$$

Proof. Both $\hat{W}_v$ and $W_v^\star$ lie in $[1, 1 + R]$. For a threshold t in this interval, define $Z_i(t) := n_i(v)(w_i - t)$. The variables $Z_i(t)$ are independent across rollouts and lie in $[B_v(1-t),\ B_v(1+R-t)]$, an interval of width $B_v R$. For $t = W_v^\star + \epsilon$, their mean is $-\mu_v\epsilon$, and the event $\hat{W}_v \ge t$ implies $N_{\mathrm{w}}^{-1}\sum_i Z_i(t) \ge 0$. Hoeffding's inequality therefore gives

$$\Pr(\hat{W}_v - W_v^\star \ge \epsilon) \;\le\; \exp\left(-\frac{2N_{\mathrm{w}}\mu_v^2\epsilon^2}{B_v^2 R^2}\right).$$

Likewise, for $t = W_v^\star - \epsilon$, the mean is $\mu_v\epsilon$, and $\hat{W}_v \le t$ implies $N_{\mathrm{w}}^{-1}\sum_i Z_i(t) \le 0$, including the zero-count case under the stated fallback. The corresponding lower-tail bound is identical. If either threshold lies outside $[1, 1 + R]$, its tail event is impossible. A union bound yields the result. □

This bound applies to the occurrence-weighted estimator while allowing arbitrary dependence between a token's occurrence count and its trajectory weight. Concentration depends on the number of independent rollouts N_{w} and the ratio μ_v / B_v; repeated occurrences within a rollout are not treated as independent samples. The bound can be conservative for rare tokens and does not by itself establish the stability of the stopping threshold or accuracy preservation.

Auxiliary bound 2: concentration of the sliding-window mean. Let $\tilde{c}_t = W_{x_t}\, c_t$ denote the weighted per-token confidence score at decoding step t, which is $\sigma(x_{1:t})$-measurable and, in practice, bounded in some interval $[0, B]$ with $B > 0$ given the model's numerical log-probability floor. Define the running average of per-step conditional means

$$\bar{M}_m \;\coloneqq\; \tfrac{1}{m} \sum_{t=1}^{m} \mathbb{E}[\tilde{c}_t \mid x_{1:t-1}],$$

which is the (random) target around which the windowed mean concentrates under the autoregressive law of $\bar{P}$.

Proposition 3 (Sliding-window mean concentration)**.** *For any window of m consecutive decoding steps, the windowed mean $\bar{C}_m = \frac{1}{m}\sum_{t=1}^{m} \tilde{c}_t$ satisfies, for any $u > 0$,*

$$\Pr\big(|\bar{C}_m - \bar{M}_m| \geq u\big) \;\leq\; 2\exp\big(-2mu^2/B^2\big).$$

Proof. Define the centred sequence $d_t \coloneqq \tilde{c}_t - \mathbb{E}[\tilde{c}_t \mid x_{1:t-1}]$, which is $\sigma(x_{1:t})$-measurable and has zero conditional mean by construction. Conditional on the history $x_{1:t-1}$, $\tilde{c}_t$ takes values in $[0, B]$, so d_t takes values in $[-\mu_t, B - \mu_t]$ with $\mu_t \coloneqq \mathbb{E}[\tilde{c}_t \mid x_{1:t-1}]$; in particular, the conditional support of d_t has width at most B. Hence $\{d_t\}$ is a martingale difference sequence with conditional support widths bounded by B, and the Hoeffding–Azuma inequality [2] in its width-based form ($\Pr(|S_m| \geq t) \leq 2\exp(-2t^2/\sum_t (b_t - a_t)^2)$) gives

$$\Pr\big(|S_m| \geq mu\big) \;\leq\; 2\exp\big(-2(mu)^2/(mB^2)\big) \;=\; 2\exp\big(-2mu^2/B^2\big).$$

Dividing through by m and noting that $S_m/m = \bar{C}_m - \bar{M}_m$ yields the stated bound. □

Proposition 3 bounds the deviation of the observed windowed mean from the per-step *conditional*-mean process $\bar{M}_m$ that CalConf actually monitors, rather than from any static unconditional expectation; this is the right notion of local stability for an autoregressive sequence whose tokens are not i.i.d. The bound implies a confidence-$(1-\delta)$ noise floor of $u_\delta = B\sqrt{\log(2/\delta)/(2m)}$ on the local windowed mean, which shrinks as $\mathcal{O}(1/\sqrt{m})$ in the window size. For the typical weighted-confidence scale of our experiments and our default $m = 256$, this floor is small relative to the magnitude of the calibrated threshold T, so the early-stopping decision is dominated by $\bar{M}_m$ rather than by local-stability noise.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The empirical and theoretical claims in the abstract and Section 3 are directly supported by our experimental results in Section 4 and Section 4.3.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section A.1 explicitly enumerates three limitations of CalConf.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: Proposition 1 states the assumptions explicitly and its full proof, together with two auxiliary concentration bounds, appears in Section A.10.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 4.1 together with the pseudocode in Algorithm 1 and the code release fully specify the experimental protocol.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: A code repository is linked from the abstract (`https://github.com/tmlr-group/calconf`), and all base models and benchmarks are publicly available.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: All hyperparameters, splits, and decoding details are specified in Section 4.1.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Non-parametric bootstrap 95% confidence intervals (10^4 resamples over the 30 test problems) are reported in Section A.4.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Section 4.1 specifies the inference settings and hardware configuration, including eight Ascend 910B accelerators. CalConf requires no additional model training.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The work uses only public open-source models and benchmarks, involves no human subjects or sensitive data, and conforms to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We analyze the broader impacts of our work in Appendix A.2.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper releases no pre-trained models or scraped datasets and poses no high-risk safeguards concerns.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All base LLMs and five benchmarks used in this work are properly cited and used within the terms of their public releases.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The only new asset is the CalConf code repository released at `https://github.com/tmlr-group/calconf` with a documented README.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or human subjects.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve human subjects, so IRB approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: This paper only adopts LLMs for writing polish.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.