PAC Reasoning: Controlling the Performance Loss for Efficient Reasoning

Hao Zeng[1*] **Jianguo Huang**[2*] **Bingyi Jing**[3,5]
Hongxin Wei[4†] **Bo An**[2]

[1]School of Mathematical Sciences, University of Electronic Science and Technology of China, China
[2]College of Computing and Data Science, Nanyang Technological University, Singapore
[3]School of Artificial Intelligence, The Chinese University of Hong Kong, Shenzhen, China
[4]Department of Statistics and Data Science, Southern University of Science and Technology, China
[5]Shenzhen Loop Area Institute, China

Abstract

Large reasoning models (LRMs) have achieved remarkable progress in complex problem-solving tasks. Despite this success, LRMs typically suffer from high computational costs during deployment, highlighting a need for efficient inference. A practical direction is to switch the LRM between thinking and non-thinking modes dynamically. However, such approaches often introduce additional reasoning errors and lack statistical guarantees for the performance loss, which are critical for high-stakes applications. In this work, we propose *Probably Approximately Correct* (PAC) *Reasoning* that controls the performance loss under the user-specified tolerance. Specifically, we construct an upper confidence bound on the performance loss and determine a threshold for switching to the non-thinking model. Theoretically, using the threshold to switch between the thinking and non-thinking modes ensures bounded performance loss in a distribution-free manner. Our experiments on reasoning benchmarks show that the proposed method can save computational budgets and control the user-specified performance loss.

1 Introduction

Large reasoning models (LRMs) show strong performance in tackling complex problems (DeepSeek-AI et al., 2025; Yang et al., 2025a). However, the strong performance largely depends on long reasoning chains, which substantially increase the computational cost during inference. This phenomenon, often referred to as *overthinking* (Yue et al., 2025), is evident in mathematical and logic-intensive tasks. In applications requiring real-time interaction or large-scale processing, such as text generation (Zhang et al., 2022) and chatbot (Roller et al., 2021), inference efficiency directly determines usability and user experience. Thus, it is important to improve the efficiency of LRMs.

To address this, some existing works proposed to switch the LRM into a non-thinking mode to avoid overthinking (Chung et al., 2025; Fang et al., 2025; Li et al., 2025b; Liang et al., 2025; Ma et al., 2025; Paliotta et al., 2025; Pan et al., 2025; Xiao and Gan, 2025; Yong et al., 2025). While effective in reducing computational cost, using a non-thinking model often degrades solution quality or introduces additional errors. For instance, in theorem-proving tasks, switching techniques may lead to invalid logical inference steps, and in mathematical reasoning, it can result in calculation mistakes or overlooked solution paths. This motivates studying performance-loss control together with efficient reasoning and the limitation raises a fundamental issue:

*Equal contribution.
†Corresponding author: weihx@sustech.edu.cn.

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

How to improve the efficiency of LRMs, controlling the performance loss provably?

In this work, we formalize this challenge by the concept of a PAC efficient model, where an LRM provides probabilistic guarantees that its performance loss relative to a reference thinking model remains within a user-specified tolerance in Definition 2.1. We address the problem of overthinking, where reasoning models spend unnecessary computation on easy queries. To achieve this goal, we propose **PAC Reasoning**, which constructs a composite model $\widehat{f}$ by adaptively routing each input between a high-cost thinking model f and a cheaper non-thinking model $\tilde{f}$. Specifically, PAC Reasoning determines a threshold via a calibration procedure (Algorithm 1). During deployment (Algorithm 2), the composite model uses $\tilde{f}$ for inputs whose uncertainty score falls below the calibrated threshold, and defers to f otherwise. By calibrating the switching rule with respect to an explicit loss tolerance and confidence level, PAC Reasoning provides rigorous statistical guarantees on the resulting performance loss. Furthermore, we provide a comprehensive discussion of related work in Appendix A.

Theoretically, we show that PAC Reasoning provides distribution-free control of performance loss with high-probability guarantees. We formalize this through a composite model $\widehat{f}$, for which the performance loss is non-decreasing with respect to the uncertainty threshold. We calibrate the threshold by testing candidate thresholds from a pre-specified grid in increasing order, stopping at the first one whose upper confidence bound exceeds a user-specified tolerance. Under some regular assumptions, we prove that PAC Reasoning controls the loss below the user-specified tolerance with high probability, thereby satisfying PAC efficiency.

We then present experimental results in Section 4 that evaluate PAC Reasoning across diverse reasoning benchmarks, including MATH-500, ZebraLogic, and Arena-Hard. The results demonstrate that our approach effectively controls the performance loss and significantly reduces inference cost. For example, on Arena-Hard with tolerance $\epsilon = 0.08$ for the logits uncertainty score, our method controls the average empirical performance loss at 0.06 (below the tolerance), and achieves token savings exceeding 40%. We also find that the logits-based uncertainty score provides more stable performance loss control compared to the verbalized-based score. [3]

Our contributions are as follows:

- We are the first to define **PAC efficient** models through high-probability control of their performance loss relative to a thinking model.
- We propose a method, **PAC Reasoning**, that combines a thinking-mode model with its non-thinking counterpart via an uncertainty-based mechanism to skip unnecessary thinking. The method provides *model-free* and *distribution-free* performance guarantees based on statistical theories.
- We provide experiments on mathematical reasoning, logical deduction, and text generation, validating that PAC Reasoning achieves significant cost savings in token and expert calls while satisfying the statistical validity of the PAC efficient guarantee.

Notations. Let f be an LRM with a thinking-mode, which is computationally expensive but delivers high performance on its answers. Given an input prompt x, f produces an output $y = f(x)$, which we regard as the "expert answer". The second is the non-thinking LRM $\tilde{f}$, which is computationally cheaper but less accurate, and $\tilde{y} = \tilde{f}(x)$. And for any input x, we use y^{gold} as its "gold reference". Let $\mathcal{I}_{cal} = \{1, \ldots, n\}$ and $\mathcal{I}_{test} = \{n+1, \ldots, n+N\}$ denote the indices of the calibration and test sets, respectively. We define the calibration dataset and the test dataset as:

$$\mathcal{D}_{cal} = \{(x_i, y_i)\}_{i \in \mathcal{I}_{cal}}, \quad \mathcal{D}_{test} = \{(x_i, y_i)\}_{i \in \mathcal{I}_{test}}.$$

It is worth noting that the y_i is not a ground-truth label, but the "expert answer" provided by the LRM f. Finally, let $y_i = (y_{i,1}, \ldots, y_{i,l_{y_i}})$ denote an answer consisting of l_{y_i} tokens.

[3] The reproducibility code is available at https://github.com/Jianguo99/PAC-Reasoning.

2 Probably approximately correct reasoning

2.1 PAC efficient model

We aim to build an efficient LRM, denoted by $\widehat{f}$, that provides probably approximately correct guarantees for its performance loss while improving efficiency. Specifically, given an error tolerance ϵ and a confidence level $1-\alpha$, $\widehat{f}$ ensures its performance loss *relative to the only thinking-mode* LRM f does not exceed ϵ with probability at least $1-\alpha$. We formulate the PAC guaranteed $\widehat{f}$ as follows:

Definition 2.1 ((ϵ, α)-PAC efficient)**.** An LRM $\widehat{f}$ is called an (ϵ, α)-probably approximately correct (PAC) efficient model, if for error tolerance $\epsilon > 0$ and confidence level $\alpha \in (0, 1)$, it satisfies

$$\mathbb{P}\left(R(\widehat{f}) \leq \epsilon\right) \geq 1-\alpha,$$

where $R(\widehat{f}) = \mathbb{E}_{x\sim P}[\ell(f(x), \widehat{f}(x))]$ is the risk function, with the expert answer as the first argument of ℓ. Here $\ell(\cdot,\cdot)$ is a loss function[4], and x is an input prompt drawn from the test distribution P.

***Remark* 1.** The loss function could be a 0-1 loss for verifiable tasks or a semantic loss for generative tasks. We sometimes term (ϵ, α)-PAC efficient simply as PAC efficient. Since ϵ controls the performance loss relative to the thinking model f, a meaningful range of ϵ is $(0, R(\tilde{f}))$, where $R(\tilde{f})$ is the risk of always using the non-thinking model.

2.2 PAC Reasoning

Constructing such a controllable LRM $\widehat{f}$ may be straightforward intuitively. Given an LRM with thinking mode f and a fast LRM without thinking $\tilde{f}$, we create a middle model that selectively uses either the LRM with thinking or not based on certain conditions. This condition acts like a "sliding rheostat" that allows us to tune the performance trade-off by adjusting the "position" of the intermediate. We can obtain a model $\widehat{f}$ that achieves the desired error tolerance heuristically. However, this approach lacks statistical guarantees on the underlying distribution of performance loss. We calibrate a threshold to control performance loss while reducing computation.

Motivated by this, we present **PAC Reasoning**, which constructs a composite LRM $\widehat{f}$ that improves the efficiency of the thinking model f with a PAC guarantee for the performance loss. The core idea is to use **uncertainty scores** to build an **upper confidence bound**(UCB) for the performance loss. We could use the upper confidence bound to measure the uncertainty of performance loss for each value of the uncertainty score. Then we **calibrate** an uncertainty threshold to switch between the thinking and non-thinking models. We use the non-thinking LRM $\tilde{f}$ on most inputs and strategically invoke the expensive LRM with thinking f only for inputs whose generation by $\tilde{f}$ has high uncertainty. Mathematically, we could represent the composite model $\widehat{f}$ for a threshold u as follows:

$$\widehat{f}(x) \equiv \widehat{f}_u(x) = \begin{cases} \tilde{f}(x), & U(x) < u, \\ f(x), & U(x) \geq u. \end{cases} \tag{1}$$

Next, we provide the details of the PAC Reasoning.

Uncertainty scores and empirical performance loss We assume that for each input prompt x_i, the non-thinking LRM $\tilde{f}$ produces an output $\tilde{y}_i$, and that there exists a score $U_i \in [0,1]$ to quantify its uncertainty. This score should ideally correlate with the likelihood of disagreement with the reference model f. The core idea is to use these uncertainty scores to selectively use the expensive model with thinking f. We aim to find a threshold, $\widehat{u}$, and accept the non-thinking LRM's output $\tilde{y}_i$ for the instances such that $U_i < \widehat{u}$, while querying the model with thinking f for the cases where $U_i \geq \widehat{u}$. To formalize, we define the empirical performance loss function as:

$$L(u) = \frac{1}{N}\sum_{i=n+1}^{n+N} \ell(y_i, \tilde{y}_i)\mathbf{1}\{U_i < u\}. \tag{2}$$

[4]For simplicity, we often omit the phrase *with respect to* ℓ, since most tasks have their conventional loss functions.

This function measures the average performance loss over the test set, with nonzero contributions only from points whose uncertainty scores are strictly below u. If we could compute $L(u)$ for all u, we would choose the largest threshold u^* such that $L(u^*) \leq \epsilon$. However, computing $L(u)$ requires access to all expert answers $y_i = f(x_i)$ in the test set, which is computationally expensive. We try an alternative way to build a bound $\widehat{L}_u(\alpha)$ for $\mathbb{E}L(u)$ satisfying the following inequality pointwise w.r.t. α:

$$\mathbb{P}(\widehat{L}_u(\alpha) \geq \mathbb{E}L(u)) \geq 1 - \alpha. \tag{3}$$

The bound is called the upper confidence bound in the literature. By construction, $L(u)$ is non-decreasing in u since ℓ is non-negative and $\mathbf{1}\{U_i < u\}$ only expands as u increases. This monotonicity lets us obtain the PAC guarantee in Definition 2.1 and apply fixed-sequence single-start without additional corrections; we summarize it in Theorem 3.2.

Algorithm 1 CLT-based UCB for $\widehat{L}_u(\alpha)$

Input: Calibration set $\mathcal{S}_{\text{cal}} = \{(x_i, y_i)\}_{i=1}^n$, models f and $\tilde{f}$, uncertainty scores $\{U_i\}_{i=1}^n$, sampling weights $\{\pi_i\}_{i=1}^n$, sample size m, threshold u, confidence level α

Output: Confidence upper bound $\widehat{L}_u(\alpha)$

Initialize an empty list of samples $\mathcal{Z} \leftarrow []$.
$\tilde{y}_i \leftarrow \tilde{f}(x_i)$ for all $i = 1, \ldots, n$.
for $j = 1, \ldots, m$ **do**
  Sample an index $i_j \sim \text{Unif}(\{1, \ldots, n\})$.
  Sample a fresh Bernoulli random variable $\xi_j \sim \text{Bern}(\pi_{i_j})$, independently across draws.
  if $\xi_j = 1$ **then**
    Query the true label y_{i_j} and compute the importance-weighted loss $Z_j \leftarrow \ell(y_{i_j}, \tilde{y}_{i_j})/\pi_{i_j}$.
  else
    $Z_j \leftarrow 0$.
  end if
  Append Z_j to $\mathcal{Z}$.
end for
For a threshold u, set $Z_j(u) \leftarrow Z_j \cdot \mathbf{1}\{U_{i_j} < u\}$ for $j = 1, \ldots, m$.
$\widehat{\mu}_Z(u) \leftarrow \frac{1}{m}\sum_{j=1}^m Z_j(u)$
$\widehat{\sigma}_Z(u) \leftarrow \sqrt{\frac{1}{m-1}\sum_{j=1}^m (Z_j(u) - \widehat{\mu}_Z(u))^2}$
$z_{1-\alpha} \leftarrow (1-\alpha)$-quantile of the standard normal distribution.
Return $\widehat{L}_u(\alpha) \leftarrow \widehat{\mu}_Z(u) + z_{1-\alpha}\frac{\widehat{\sigma}_Z(u)}{\sqrt{m}}$.

Constructing the upper confidence bound (UCB) Next we discuss how to construct a valid UCB, via importance sampling. Given a sampling size m, we first collect m indices $\{i_1, \ldots, i_m\}$ by sampling uniformly with replacement from $\{1, \ldots, n\}$. Then, for each selected index i_j, we draw a fresh Bernoulli variable $\xi_j \sim \text{Bern}(\pi_{i_j})$, independently across draws, to decide whether to query its expert answer y_{i_j}. The weights satisfy $0 < \pi_i \leq 1$ and are fixed conditional on the calibration set. Given this set, the procedure yields m i.i.d. random variables:

$$Z_j(u) = \ell(y_{i_j}, \tilde{y}_{i_j})\frac{\xi_j}{\pi_{i_j}}\mathbf{1}\{U_{i_j} < u\}. \tag{4}$$

Its conditional mean is the calibration-pool risk $\overline{L}_n(u) = n^{-1}\sum_{i=1}^n \ell(y_i, \tilde{y}_i)\mathbf{1}\{U_i < u\}$, since $\mathbb{E}[\xi_j/\pi_{i_j} \mid i_j, \mathcal{D}_{cal}] = 1$. Algorithm 1 uses a central limit theorem (CLT) approximation for this mean. The population risk is $R(u) = \mathbb{E}_{\mathcal{D}_{cal}}[\overline{L}_n(u)] = \mathbb{E}[L(u)]$ under i.i.d. sampling from the deployment law.

***Remark* 2.** For a fixed threshold u, importance sampling gives an unbiased estimate of $\overline{L}_n(u)$ conditional on the calibration pool and of $R(u)$ after averaging over the original calibration sample. The resampling draws are independent only conditional on the shared pool. Here n is the number of original calibration observations and m is the number of auditing draws; repeated indices do not add new observations from the deployment law. The CLT bound is asymptotic for the pool mean as m grows, while its use for the population risk in Assumption 3.1 must also account for the randomness of the n original observations. Appendix C gives the conditional CLT and separates

these two sources of randomness. Alternatively, if the importance-weighted losses are bounded, one could use concentration inequalities like Hoeffding's or Bernstein's inequality to construct a valid confidence bound (Bentkus, 2004; Hao et al., 2019; Hoeffding, 1994; Learned-Miller and Thomas, 2020; Ramdas et al., 2022; Waudby-Smith and Ramdas, 2021; 2024), which may provide better guarantees for smaller sample sizes. We give an example based on Hoeffding's inequality (Hoeffding, 1994) in Algorithm 3 in Appendix B.

Calibration Fix a deterministic finite grid $\Lambda = \{u_1 < \cdots < u_K\} \subset (0, 1]$, with $K \geq 1$, before observing the calibration data $\mathcal{D}_{cal}$. For example, $u_k = k/K$ gives a grid with a size K chosen in advance. Set $u_0 = 0$, which gives the expert-only rule in Equation (1) and has zero expert-relative loss since $U(x) \geq 0$ and $\ell(y, y) = 0$. We test $\widehat{L}_{u_k}(\alpha) \leq \epsilon$ in increasing order of k, stop at the first failure, and select the preceding threshold:

$$\tau = \min\Big(\{k \in \{1, \ldots, K\} : \widehat{L}_{u_k}(\alpha) > \epsilon\} \cup \{K + 1\}\Big), \widehat{u} \quad = u_{\tau-1}. \tag{5}$$

If the first candidate fails, $\widehat{u} = u_0 = 0$; if all K candidates pass, $\tau = K + 1$ and $\widehat{u} = u_K$. The fallback u_0 is safe by construction and does not require a UCB test. A UCB equal to ϵ passes, but no later candidate is tested after a failure, even if its UCB would pass. At deployment, the fast model is used only when $U(x) < \widehat{u}$; equality is assigned to the expert, including $U(x) = 1$ when $\widehat{u} = 1$. Algorithm 2 implements this fixed-sequence early-stopping rule.

Algorithm 2 PAC Reasoning

Input: Calibration set $\{(x_i, y_i)\}_{i=1}^n$, test prompts $x_i, i \in \mathcal{I}_{test}$, model without thinking $\tilde{f}$, model with thinking f, loss function ℓ, error tolerance ϵ, confidence level $1 - \alpha$, deterministic grid $\Lambda = \{0 < u_1 < \cdots < u_K \leq 1\}$ fixed before calibration

Output: Composite model $\widehat{f}$

Compute confidence bound function $\widehat{L}_u(\alpha)$ via Algorithm 1 on calibration data $\{(x_i, y_i)\}_{i=1}^n$.
Initialize $\widehat{u} \leftarrow u_0 = 0$ (expert-only fallback).
for $k = 1, \ldots, K$ **do**
  if $\widehat{L}_{u_k}(\alpha) > \epsilon$ **then**
    break
  end if
  $\widehat{u} \leftarrow u_k$.
end for
Define $\widehat{f} = \widehat{f}_{\widehat{u}}$ by Equation (1), using $\tilde{f}$ for $U(x) < \widehat{u}$ and f for $U(x) \geq \widehat{u}$.
Return $\widehat{f}$.

3 Theoretical analysis

We show how fixed-sequence selection turns pointwise risk bounds into a guarantee for the selected policy. Fix the models $f, \tilde{f}$, the score function $U : \mathcal{X} \to [0, 1]$, and a nonnegative loss ℓ with $\ell(y, y) = 0$ before calibration. Let $x \sim P$ and $y = f(x)$, and let $\mathcal{D}_{cal}$ consist of n i.i.d. draws from this law. Assume $\mathbb{E}_{x\sim P}[\ell(y, \tilde{f}(x))] < \infty$. For the policy $\widehat{f}_u$ in Equation (1), its population risk and independent test-sample risk are

$$R(u) = \mathbb{E}_{x\sim P}[\ell(y, \widehat{f}_u(x))], \qquad \widehat{R}(u) = \frac{1}{N} \sum_{i \in \mathcal{I}_{test}} \ell(y_i, \widehat{f}_u(x_i)).$$

In Assumption 3.1 and Theorem 3.2, probability is over $\mathcal{D}_{cal}$ and any additional randomness in the UCB procedure, not over the independent test sample.

Fix $\epsilon > 0$, $\alpha \in (0, 1)$, and the deterministic finite grid $\Lambda = \{u_1 < \cdots < u_K\} \subset (0, 1]$ before observing $\mathcal{D}_{cal}$. Set $u_0 = 0$, so $R(u_0) = 0$ because this policy always uses the expert. For $0 \leq u \leq v \leq 1$,

$$R(v) - R(u) = \mathbb{E}_{x\sim P}\Big[\ell(y, \tilde{f}(x))\mathbf{1}\{u \leq U(x) < v\}\Big] \geq 0.$$

Thus, the true risk is non-decreasing in the threshold; no monotonicity of the UCB is assumed.

Assumption 3.1 (Pointwise UCB validity)**.** For every fixed $u \in \Lambda$, the UCB computed from $\mathcal{D}_{cal}$ satisfies

$$\mathbb{P}\big(R(u) \le \widehat{L}_u(\alpha)\big) \ge 1 - \alpha.$$

This is a pointwise condition, not an event on which all grid points are covered at once.

This high-level condition concerns the population risk $R(u)$. A bound for a fixed empirical calibration pool alone does not establish it.

Theorem 3.2 (PAC guarantee)**.** *Under the setup above, let $\widehat{u}$ be selected on the fixed finite grid Λ by Equation (5) and Algorithm 2. Assume that $R(u)$ is non-decreasing, $R(u_0) = 0$, and Assumption 3.1 holds at every fixed grid point. If selection stops at the first failed test and returns the preceding threshold, with u_0 as the fallback, then*

$$\mathbb{P}\big(R(\widehat{f}_{\widehat{u}}) \le \epsilon\big) = \mathbb{P}\big(R(\widehat{u}) \le \epsilon\big) \ge 1 - \alpha.$$

Proof sketch. If all grid risks are at most ϵ, every possible output, including u_0, is safe. Otherwise, let $k^* = \min\{k : R(u_k) > \epsilon\}$ be the first unsafe index. It is fixed by P and the pre-specified grid, not by the calibration data. Selecting an unsafe threshold requires passing the test at u_{k^*} before stopping, so

$$\{R(\widehat{u}) > \epsilon\} \subseteq \{\widehat{L}_{u_{k^*}}(\alpha) \le \epsilon < R(u_{k^*})\}.$$

Pointwise validity at this single fixed candidate bounds the probability by α. Appendix D gives the full argument. □

The selection guarantee has the same finite-sample or asymptotic validity as the pointwise population-risk UCBs. The asymptotic result uses a fixed finite grid; see Appendix D. For loss bounded in $[a, b]$ with $a < b$, we also give an empirical test-risk bound.

Theorem 3.3 (Empirical risk PAC guarantee)**.** *Assume the setup and conditions of Theorem 3.2 hold, and the i.i.d. test batch $\mathcal{D}_{test}$ is independent of the calibration data and the UCB randomization. For $\epsilon > 0$, $\alpha \in (0, 1)$, $\ell \in [a, b]$ with $a < b$, and any $t > 0$,*

$$\mathbb{P}\big(\widehat{R}(\widehat{u}) \le \epsilon + t\big) \;\ge\; 1 - \alpha - \exp\Big(-\tfrac{2Nt^2}{(b-a)^2}\Big).$$

Here, the probability also includes the independent test sample.

Proof sketch. The result follows from the decomposition of the error event: $\{\widehat{R}(\widehat{u}) > \epsilon + t\} \subseteq \{R(\widehat{u}) > \epsilon\} \cup \{\widehat{R}(\widehat{u}) - R(\widehat{u}) > t\}$. The first term is bounded by α via Theorem 3.2. The second term is bounded by $\exp(-2Nt^2/(b-a)^2)$ using the conditional Hoeffding's inequality, as the test set is independent of $\widehat{u}$. A detailed proof is in Appendix E. □

A special case is a bounded loss $\ell \in [0, 1]$, e.g., 0-1 loss for binary verifiable answers. Then $b - a = 1$ and the bound simplifies to $\mathbb{P}\big(\widehat{R}(\widehat{u}) \le \epsilon + t\big) \ge 1 - \alpha - e^{-2Nt^2}$. The extra tail term accounts for test-sample error around the selected policy's population risk. Appendix F gives a separate guarantee for a fixed pool, and Appendix G extends fixed-sequence selection to multi-level policies.

4 Experiments

In this section, we conduct comprehensive experiments to answer the following research questions:

1. Does PAC Reasoning control the performance loss at the target tolerance with confidence?
2. What computational savings does PAC Reasoning achieve while bounding the error rate?

4.1 Setup

Large language models. In this study, we evaluate the PAC Reasoning based on the Qwen3 series models (Yang et al., 2025a) and Llama-3.1-8B–based models. Specifically, we employ the "Qwen3-4B-Thinking-2507" as the thinking model and "Qwen3-4B-Instruct-2507" as the non-thinking model.

We also use the "DeepSeek-R1-Distill-Llama-8B" as the thinking model and "Llama-3.1-8B-Instruct" as the non-thinking model, and show the results in Appendix I.1.

Uncertainty score. Our theoretical guarantees do not depend on the choice of uncertainty score. Here, we focus on two common uncertainty scores used in LLMs, the **logits-based score**, a white-box score from model logits, and the **verbalized score**, a black-box score from self-reported confidence. For the logits-based score, we use token-level probabilities from the prediction logits (Kwon et al., 2023; Zheng et al., 2024; Zhou et al., 2025). Specifically, we define the logits-based score of y_i as its average token probability (Hao et al., 2023; Huang et al., 2025):

$$U_{logits}(y_i) = 1 - \frac{1}{l_{y_i}} \sum_{j=1}^{l_{y_i}} \mathbb{P}(y_{i,j}|y_{i,1}, \ldots, y_{i,j-1}, x_i),$$

where $\mathbb{P}(y_{i,j}|y_{i,1}, \ldots, y_{i,j-1}, x_i)$ is the conditional probability of token $y_{i,j}$. For the verbalized uncertainty scores, the non-thinking model explicitly states its self-reported confidence (Xiong et al., 2023; Tian et al., 2023; Yang et al., 2024; Zhou et al., 2025). In this study, we report the average verbalized confidence over 10 trials, and the corresponding prompts are listed in Appendix H.

Datasets. We evaluate PAC Reasoning on datasets spanning reasoning and open-ended generation tasks. Specifically, our evaluation covers a high-level mathematics benchmark, **MATH-500** (Lightman et al., 2023), a text-based logical reasoning task, **ZebraLogic** (Lin et al., 2025), and an alignment-focused open-ended writing benchmark, **Arena-Hard** (Li et al., 2025a). For each dataset, we partition the original test set into a calibration subset and a held-out test subset randomly. Table 4 in Appendix H provides details of the splitting strategies.

Baselines. We compare PAC Reasoning with Naive threshold, empirical-mean Naive control, Router (Ong et al., 2025), Chain of Draft (CoD) (Xu et al., 2025), and NoThinking (Ma et al., 2025), with details in Appendix H.1. They cover fixed-threshold routing, empirical calibration, learned routing, and prompting-based efficiency methods.

Loss functions. PAC Reasoning controls the performance loss incurred by switching from the thinking model f to the non-thinking model $\tilde{f}$. Therefore, we define loss relative to the thinking model, not relative to the gold answer. We use two loss functions: (1) **Semantic cosine distance**. Given reference output $y_i = f(x_i)$ and PAC output $\widehat{y}_i = \widehat{f}(x_i)$, the semantic loss is given by:

$$\ell(y_i, \widehat{y}_i) = 1 - \frac{v_{y_i} \cdot v_{\widehat{y}_i}}{\|v_{y_i}\| \|v_{\widehat{y}_i}\|}, \tag{6}$$

where v_{y_i} and $v_{\widehat{y}_i}$ are the embeddings. We use "Qwen3-Embedding-4B" as the embedding model (Yang et al., 2025a). (2) **Binary 0–1 loss** for verifiable tasks. It measures the accuracy drop when the expert answer is correct:

$$\ell(y_i, \widehat{y}_i) = \ell(y_i, \widehat{y}_i \mid y_i^{gold}) = \mathbb{1}\{\widehat{y}_i \neq y_i^{gold}\}\mathbb{1}\{y_i = y_i^{gold}\}, \tag{7}$$

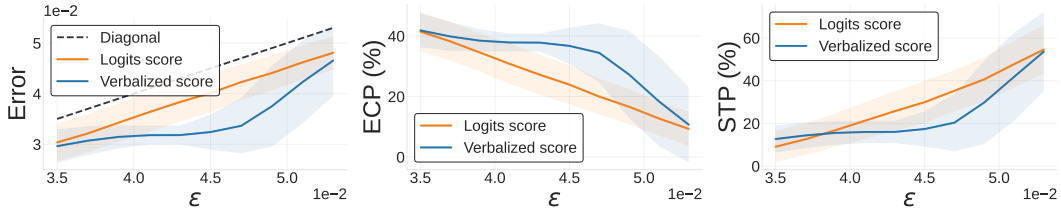
where y_i^{gold} is the gold answer for x_i. We discuss the choice of loss function in Appendix H.2.

Metrics. To evaluate the effectiveness of PAC Reasoning in budget usage, we define two metrics: **Expert Call Percentage** (ECP) and **Saved Token Percentage** (STP). These metrics are defined by:

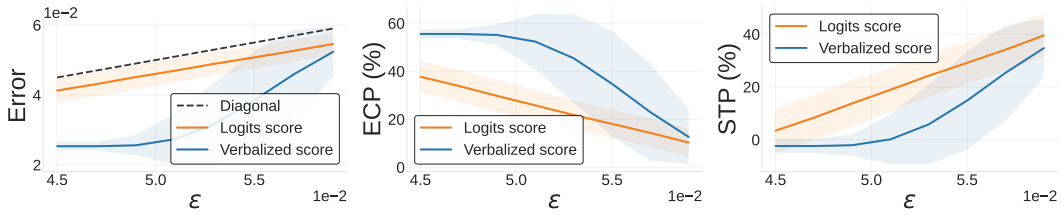
$$\text{ECP} := \frac{|\{i : U_i \geq \widehat{u},\ i \in \mathcal{I}_{\text{test}}\}|}{N} \times 100\%, \text{STP} := \frac{1}{N} \sum_{i \in \mathcal{I}_{\text{test}}} \left(1 - \frac{l_{\tilde{y}_i} + \mathbb{1}\{U_i \geq \widehat{u}\} l_{y_i}}{l_{y_i}}\right) \times 100\%,$$

where $l_{\tilde{y}_i}$ and l_{y_i} represent the number of tokens in the candidate answer $\tilde{y}_i$ and the reference answer y_i, respectively. The ECP measures the proportion of test cases requiring expert intervention. STP quantifies the token efficiency by comparing the token counts of candidate and reference answers, accounting for cases where expert calls are triggered. We also report the accuracy of our proposed method in Appendix I.3.

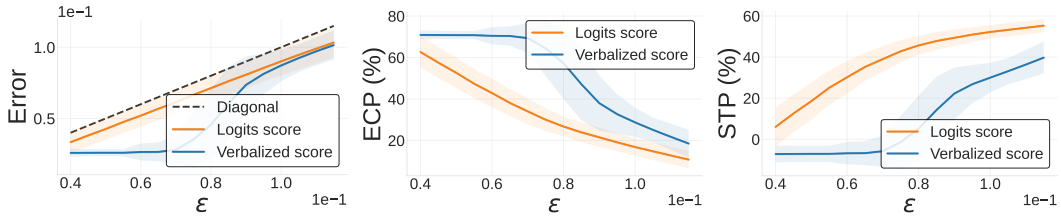
We repeat each experiment 100 times and report the mean and standard deviation of the budget savings. We fix $\alpha = 0.05$ throughout all experiments while varying ϵ, and set the sampling weight $\pi = \pi_i = 0.5$ for each $i \in \mathcal{I}_{cal}$ and the sample size $m = n/\pi$.



(a) MATH-500



(b) ZebraLogic



(c) Arena-Hard

Figure 1: **Error control, ECP and STP of PAC Reasoning** for semantic loss across three benchmarks at $\alpha = 0.05$. Uncertainty score includes the logits-based score and the verbalized score. All experiments are repeated 100 times, and the shaded areas represent standard deviations.

4.2 Results

PAC Reasoning improves efficiency while controlling performance loss. Figure 1 reports the results under semantic loss. Across all benchmarks, PAC Reasoning consistently keeps the empirical performance loss below the target tolerance ϵ while achieving substantial budget savings. For instance, on Arena-Hard with $\epsilon = 0.08$ and the logits-based score, it incurs only about 0.06 empirical performance loss, with ECP and STP of roughly 20% and 40%, respectively. These trends are consistent with those obtained using the Llama-based model in Appendix I.1, and also hold on additional benchmarks such as GPQA (Rein et al., 2024) and HumanEval (Chen et al., 2021) in Appendix I.2. Moreover, PAC Reasoning improves Pass@1 over the non-thinking baseline under semantic-loss control, as shown in Appendix I.3. Under the binary loss function, PAC Reasoning also maintains error rates within the target tolerance while saving computational budget (See Table 1). For example, on MATH-500, PAC Reasoning with the logits-based score calls the thinking model on 21.48% of test queries and saves 37.61% of tokens. We also report success counts over 100 runs and exact 95% binomial intervals for the empirical test-loss criterion in Appendix I.4. Beyond token savings, it also reduces wall-clock latency and dollar cost in real deployments (See Table 7 in Appendix I.4). Overall, PAC Reasoning bounds performance loss within the target tolerance and substantially improves inference efficiency on benchmarks.

PAC Reasoning saves inference cost without sacrificing error control. Table 1 compares PAC Reasoning with multiple baselines under binary loss, using $\epsilon = 0.03$ for MATH-500 and $\epsilon = 0.08$ for ZebraLogic. These tolerances are representative operating points for the two tasks. In an application, ϵ should reflect the acceptable loss relative to the thinking model; Appendix I.13 discusses this choice. PAC Reasoning satisfies the target error tolerance on both datasets while achieving substantial token savings, reflecting its explicit PAC-style control of performance loss. In contrast, heuristic methods may yield higher token savings but do not reliably control downstream prediction risk. For example, the router method attains the highest STP on MATH-500, but its empirical error exceeds the target tolerance. Similarly, CoD and NoThinking reduce reasoning cost by avoiding full reasoning, but can

Table 1: **PAC Reasoning enables safe, effective inference-cost reduction over heuristic baselines**. Green cells indicate valid loss control. Experiments use binary loss with $\epsilon = 0.03$ on MATH-500 and $\epsilon = 0.08$ on ZebraLogic.

Method	Score	MATH-500			ZebraLogic		
		Error ↓	ECP (%) ↓	STP (%) ↑	Error ↓	ECP (%) ↓	STP (%) ↑
Naive threshold	Logits-based	0.0179 ± 0.0068	14.44 ± 2.02	43.58 ± 4.78	0.0062 ± 0.0026	77.28 ± 1.36	-34.78 ± 1.26
	Verbalized	0.0346 ± 0.0095	2.83 ± 0.94	66.67 ± 4.91	0.0631 ± 0.0074	12.49 ± 1.07	32.70 ± 2.11
Naive control	Logits-based	0.0351 ± 0.0098	6.04 ± 2.75	61.52 ± 6.10	0.0959 ± 0.0099	5.47 ± 1.63	46.59 ± 2.43
	Verbalized	0.0351 ± 0.0098	6.04 ± 2.75	61.52 ± 6.10	0.0948 ± 0.0093	6.56 ± 0.92	42.75 ± 1.95
PAC Reasoning	Logits-based	0.0206 ± 0.0126	21.48 ± 17.85	37.61 ± 23.19	0.0618 ± 0.0181	22.50 ± 7.47	23.13 ± 9.09
	Verbalized	0.0209 ± 0.0141	24.59 ± 20.48	36.13 ± 26.44	0.0530 ± 0.0246	26.95 ± 20.68	21.21 ± 17.20
Router	–	0.0435 ± 0.0107	1.97 ± 0.74	74.73 ± 2.59	0.0781 ± 0.0082	23.77 ± 1.39	39.81 ± 1.06
CoD	–	0.3548 ± 0.0604	$--$	-1.33 ± 0.82	0.0492 ± 0.0071	$--$	0.03 ± 0.62
NoThinking	–	0.4445 ± 0.0583	$--$	-4.51 ± 1.00	0.0392 ± 0.0066	$--$	-0.52 ± 0.69

incur severe performance degradation, especially on MATH-500. Overall, PAC Reasoning provides a principled trade-off between efficiency and reliability, distinguishing it from heuristic approaches that lack explicit risk control.

Logits-based uncertainty score is more stable. From the results in Figure 1 and Table 1, the logits-based uncertainty score consistently shows lower variance in both ECP and STP. In contrast, the verbalized score exhibits larger fluctuations due to its sparse and clustered distribution. For instance, on ZebraLogic (see Table 1) with PAC reasoning, the standard deviations of ECP and STP under the verbalized score are 20.68% and 17.20%, considerably higher than the corresponding 7.47% and 9.09% values for the logits-based score. Similar patterns can also be observed in Figure 1, where the variance of the verbalized score is consistently higher than that of the logits-based score across different settings. Therefore, although the verbalized score occasionally achieves stronger risk control or higher savings, its calibration is less reliable, leading to less consistent performance.

Appendix I.9 compares accuracy and loss at four expert-call budgets, and Appendix I.11 reports additional operating points for the Llama pair.

5 Discussion

In this section, we study how robust PAC Reasoning is to practical design. We first consider the reliability of the uncertainty score using expected calibration error (ECE). Then we study the ablation on varying the uncertainty score and the calibration set size. Across these settings, PAC Reasoning keeps valid error control and shows stable efficiency.

Quality of uncertainty score. To assess the reliability of uncertainty scores, we compute the ECE, which measures the difference between predicted confidence and actual accuracy. As shown in Appendix I.5, logits-based scores achieve lower ECE than verbalized scores on most benchmarks. For example, on MATH-500, the logits-based score achieves an ECE of 0.0450, while the verbalized score has an ECE of 0.0634. Despite the differences in calibration quality, both approaches successfully maintain the PAC guarantee, demonstrating the robustness of our framework to imperfect uncertainty quantification. It means that perfect calibration is not necessary for PAC Reasoning. Appendix I.12 separates score compression from quantization and tests three UCBs under regular and sparse losses.

Calibration set size. Calibration size introduces a practical trade-off between calibration cost and guarantee tightness. Appendix I.6 studies calibration ratios from 10% to 90% and shows that error control remains stable, with average loss staying within the target tolerance even for small calibration sets. For instance, on MATH-500 with the logits-based score, the average loss changes only from 0.0331 to 0.0344 as the calibration ratio increases from 10% to 90%, while STP remains between 14.39% and 18.52%. These results describe the observed trade-off across calibration sizes. Appendix I.13 gives the cost decomposition and deployment break-even condition.

Alternative uncertainty scores. PAC Reasoning can use any scoring function correlated with the disagreement rate. One alternative is reward model-based scoring, where a trained reward model evaluates the non-thinking model's output. We conduct experiments in Appendix I.7 using reward

scores on MATH-500 and ZebraLogic, demonstrating that PAC Reasoning maintains valid error control even with this alternative scoring method. For example, on MATH-500 with $\epsilon = 0.045$, reward-based PAC Reasoning achieves an error of 0.0362 with 22.43% token savings. This flexibility is particularly valuable in scenarios where uncertainty scores are unreliable or unavailable.

Empirical behavior under distribution shift. We calibrate on 300 MATH-500 examples and evaluate on 200-example mixed sets containing 20, 40, or 60 AIME problems, repeated 100 times under binary loss. Across these mixtures, the mean test loss stays below the target tolerance, while the escalation rate increases and token savings decrease as the AIME proportion grows. Time savings are 61.73%–69.02% and monetary savings are 21.17%–37.24%. Appendix I.10 reports MATH-to-HumanEval and AIME-only evaluations.

6 Conclusion

We propose PAC Reasoning, a practical efficiency improvement method with statistical guarantees for reasoning models. Our approach constructs a composite model that selectively uses either an expert model or a candidate model based on a constructed confidence bound. PAC Reasoning contributes by delivering statistical assurances for model performance and demonstrating that it reliably achieves the specified error rate with probability. We validate our method through extensive experiments on real-world reasoning tasks, showing it can reduce computational costs while maintaining user-specified error tolerances with confidence. Future research will focus on developing better uncertainty estimation methods, exploring tighter theoretical bounds, and broadening the method's applicability to other large language model efficiency-improvement strategies.

6.1 Limitations

The population-risk guarantee assumes i.i.d. calibration data from the deployment law and valid pointwise population-risk UCBs, with the models, score, loss, and finite candidate sequence fixed before calibration. The conditional CLT in Appendix C controls resampling error within a fixed calibration pool; population-risk validity also requires control of calibration-sample error. Calibration requires extra expert queries and may need to be repeated when the models, score, loss, or deployment law change; reported ECP and STP exclude amortized calibration cost. Efficiency depends on the score and deployment law, and savings can disappear under distribution shift, which is outside the stated guarantee. The guarantee bounds loss relative to the chosen thinking model under the specified loss function; gold-answer accuracy is reported separately in Appendix I.3.

Acknowledgments and Disclosure of Funding

This research is supported by Guangdong Basic and Applied Basic Research Foundation (Grant No. 2026A1515011367). This project is also supported by the Jiangsu Provincial Key Discipline Construction Project (Statistics) and the Open Project of the Joint Lab for Statistics and Finance (Grant No. 2025JLSF101). We thank Shenzhen Guangming Science City Development and Construction Co., Ltd. and its Material Genome Big-Science Facilities Platform for providing technical support and assistance with data collection and analysis.

References

Anastasios N. Angelopoulos, Rina Foygel Barber, and Stephen Bates. Theoretical foundations of conformal prediction. *Preprint at arXiv:2411.11824*, 2025a.

Anastasios N. Angelopoulos, Stephen Bates, Emmanuel J. Candès, Michael I. Jordan, and Lihua Lei. Learn then test: Calibrating predictive algorithms to achieve risk control. *The Annals of Applied Statistics*, 19(2):1641–1662, 2025b. ISSN 1932-6157, 1941-7330.

Anastasios Nikolas Angelopoulos, Stephen Bates, Adam Fisch, Lihua Lei, and Tal Schuster. Conformal risk control. In *The Twelfth International Conference on Learning Representations*, 2023.

Stephen Bates, Anastasios Angelopoulos, Lihua Lei, Jitendra Malik, and Michael Jordan. Distribution-free, risk-controlling prediction sets. *Journal of the ACM*, 68(6):1–34, 2021. ISSN 0004-5411, 1557-735X.

Vidmantas Bentkus. On hoeffding's inequalities. *Annals of Probability*, 32(2):0–26, 2004. ISSN 0091-1798.

Vincent Blot, Anastasios Nikolas Angelopoulos, Michael Jordan, and Nicolas J.-B. Brunel. Automatically adaptive conformal risk control. In *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, pages 19–27. PMLR, 2025.

Chen, Tworek, Jun, Yuan, Pinto, Kaplan, and et al. Evaluating large language models trained on code. *Preprint at arXiv:2107.03374*, 2021.

Chen, Qin, Liu, Peng, Guan, Wang, and et al. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *Preprint at arXiv:2503.09567*, 2025.

Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance. *Preprint at arXiv:2305.05176*, 2023.

Xiaoxue Cheng, Junyi Li, Wayne Xin Zhao, and Ji-Rong Wen. Think more, hallucinate less: Mitigating hallucinations via dual process of fast and slow thinking. *Preprint at arXiv:2501.01306*, 2025.

Stephen Chung, Wenyu Du, and Jie Fu. Thinker: Learning to think fast and slow. *Preprint at arXiv:2505.21097*, 2025.

Clark, Cowhey, Etzioni, Khot, Sabharwal, Schoenick, and et al. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *Preprint at arXiv:1803.05457*, 2018.

Cobbe, Kosaraju, Bavarian, Chen, Jun, Kaiser, and et al. Training verifiers to solve math word problems. *Preprint at arXiv:2110.14168*, 2021.

DeepSeek-AI, Guo, Yang, Zhang, Song, Zhang, and et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *Preprint at arXiv:2501.12948*, 2025.

Gongfan Fang, Xinyin Ma, and Xinchao Wang. Thinkless: LLM learns when to think. *Preprint at arXiv:2505.13379*, 2025.

Tao Feng, Haozhen Zhang, Zijie Lei, Haodong Yue, Chongshan Lin, Ge Liu, and et al. LLMRouter: An Open-Source Library for LLM Routing, 2025.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017.

Hao, Gu, Ma, Hong, Wang, Wang, and et al. Reasoning with language model is planning with world model. *Preprint at arXiv:2305.14992*, 2023.

Botao Hao, Yasin Abbasi Yadkori, Zheng Wen, and Guang Cheng. Bootstrapping upper confidence bound. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Hendrycks, Burns, Basart, Zou, Mazeika, Song, and et al. Measuring massive multitask language understanding. *Preprint at arXiv:2009.03300*, 2021a.

Hendrycks, Burns, Kadavath, Arora, Basart, Tang, and et al. Measuring mathematical problem solving with the MATH dataset. *Preprint at arXiv:2103.03874*, 2021b.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. In N. I. Fisher and P. K. Sen, editors, *The Collected Works of Wassily Hoeffding*, pages 409–426. Springer, New York, NY, 1994. ISBN 978-1-4612-0865-5. doi: 10.1007/978-1-4612-0865-5_26.

Huang, Song, Wang, Zhao, Chen, Juefei-Xu, and et al. Look before you leap: An exploratory study of uncertainty measurement for large language models. *IEEE Transactions on Software Engineering*, 51(2):413–429, 2025. ISSN 0098-5589, 1939-3520, 2326-3881.

Guochao Jiang, Guofeng Quan, Zepeng Ding, Ziqin Luo, Dixuan Wang, and Zheng Hu. FlashThink: An early exit method for efficient reasoning. *Preprint at arXiv:2505.13949*, 2025.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada, 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1147.

Jaehun Jung, Faeze Brahman, and Yejin Choi. Trust or escalate: LLM judges with provable guarantees for human agreement. *Preprint at arXiv:2407.18370*, 2024.

Kwiatkowski, Palomaki, Redfield, Collins, Parikh, Alberti, and et al. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7: 452–466, 2019.

Kwon, Li, Zhuang, Sheng, Zheng, Yu, and et al. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, pages 611–626, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 979-8-4007-0229-7. doi: 10.1145/3600006.3613165.

Erik Learned-Miller and Philip S. Thomas. A new confidence interval for the mean of a bounded random variable. *Preprint at arXiv:1905.06208*, 2020.

Li, Chiang, Frick, Dunlap, Wu, Zhu, and et al. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. In *Forty-Second International Conference on Machine Learning, ICML 2025*, Vancouver, BC, Canada, 2025a.

Li, Wei, Huang, Yan, Chen, Kwok, and et al. DynamicMind: A tri-mode thinking system for large language models. *Preprint at arXiv:2506.05936*, 2025b.

Guosheng Liang, Longguang Zhong, Ziyi Yang, and Xiaojun Quan. ThinkSwitcher: When to think hard, when to think fast. *Preprint at arXiv:2505.14183*, 2025.

Lightman, Kosaraju, Burda, Edwards, Baker, Lee, and et al. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

Lin, Bras, Richardson, Sabharwal, Poovendran, Clark, and et al. ZebraLogic: On the scaling limits of LLMs for logical reasoning. In *Forty-Second International Conference on Machine Learning*, 2025.

Shuqi Liu, Jianguo Huang, and Luke Ong. Conformal prediction meets long-tail classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 23828–23836, 2026.

Wenjie Ma, Jingxuan He, Charlie Snell, Tyler Griggs, Sewon Min, and Matei Zaharia. Reasoning models can be effective without thinking. *Preprint at arXiv:2504.09858*, 2025.

Lunyiu Nie, Zhimin Ding, Erdong Hu, Christopher Jermaine, and Swarat Chaudhuri. Online cascade learning for efficient inference over streams. *Preprint at arXiv:2402.04513*, 2024.

Ong, Almahairi, Wu, Chiang, Wu, Gonzalez, and et al. RouteLLM: Learning to route LLMs with preference data. *Preprint at arXiv:2406.18665*, 2025.

Paliotta, Wang, Pagliardini, Li, Bick, Kolter, and et al. Thinking slow, fast: Scaling inference compute with distilled reasoners. *Preprint at arXiv:2502.20339*, 2025.

Pan, Zhang, Zhang, Liu, Wang, Li, and et al. DynaThink: Fast or slow? A dynamic decision-making framework for large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 14686–14695, Miami, Florida, USA, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.814.

Pan, Ji, Ding, Li, Chen, Wang, and et al. A survey of slow thinking-based reasoning LLMs using reinforced learning and inference-time scaling law. *Preprint at arXiv:2505.02665*, 2025.

Quach, Fisch, Schuster, Yala, Sohn, Jaakkola, and et al. Conformal language modeling. In *The Twelfth International Conference on Learning Representations*, Vienna, Austria, 2024.

Aaditya Ramdas, Johannes Ruf, Martin Larsson, and Wouter Koolen. Admissible anytime-valid sequential inference must rely on nonnegative martingales. *Preprint at arXiv:2009.03167*, 2022.

Rein, Hou, Stickland, Petty, Pang, Dirani, and et al. GPQA: A graduate-level google-proof Q&A benchmark. In *First Conference on Language Modeling*, 2024.

Roller, Dinan, Goyal, Ju, Williamson, Liu, and et al. Recipes for building an open-domain chatbot. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 300–325, Online, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.24.

Sui, Chuang, Wang, Zhang, Zhang, Yuan, and et al. Stop overthinking: A survey on efficient reasoning for large language models. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota, 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421.

Tian, Mitchell, Zhou, Sharma, Rafailov, Yao, and et al. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5433–5442, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.330.

Antonios Valkanas, Soumyasundar Pal, Pavel Rumiantsev, Yingxue Zhang, and Mark Coates. C3PO: Optimized large language model cascades with probabilistic cost constraints for reasoning. *Preprint at arXiv:2511.07396*, 2025.

Ian Waudby-Smith and Aaditya Ramdas. Confidence sequences for sampling without replacement. *Preprint at arXiv:2006.04347*, 2021.

Ian Waudby-Smith and Aaditya Ramdas. Estimating means of bounded random variables by betting. *Journal of the Royal Statistical Society Series B (Statistical Methodology)*, 86(1):1–27, 2024. ISSN 1369-7412.

Wenyi Xiao and Leilei Gan. Fast-slow thinking GRPO for large vision-language model reasoning. *Preprint at arXiv:2504.18458*, 2025.

Xiong, Hu, Lu, Li, Fu, He, and et al. Can LLMs express their uncertainty? An empirical evaluation of confidence elicitation in LLMs. In *The Twelfth International Conference on Learning Representations*, 2023.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. Chain of draft: Thinking faster by writing less. *Preprint at arXiv:2502.18600*, 2025.

Yang, Li, Yang, Zhang, Hui, Zheng, and et al. Qwen3 technical report. *Preprint at arXiv:2505.09388*, 2025a.

Yang, Si, Duan, Zhu, Zhu, Li, and et al. Dynamic early exit in reasoning models. *Preprint at arXiv:2504.15895*, 2025b.

Daniel Yang, Yao-Hung Hubert Tsai, and Makoto Yamada. On verbalized confidence scores for LLMs. *Preprint at arXiv:2412.14737*, 2024.

Wenlin Yao, Haitao Mi, and Dong Yu. HDFlow: Enhancing LLM complex problem-solving with hybrid thinking and dynamic workflows. *Preprint at arXiv:2409.17433*, 2024.

Xixian Yong, Xiao Zhou, Yingying Zhang, Jinlin Li, Yefeng Zheng, and Xian Wu. Think or not? Exploring thinking efficiency in large reasoning models via an information-theoretic lens. *Preprint at arXiv:2505.18237*, 2025.

Yue, Du, Wang, Gao, Yao, Wang, and et al. Don't overthink it: A survey of efficient R1-style large reasoning models. *Preprint at arXiv:2508.02120*, 2025.

Murong Yue, Jie Zhao, Min Zhang, Liang Du, and Ziyu Yao. Large language model cascades with mixture of thought representations for cost-efficient reasoning. In *The Twelfth International Conference on Learning Representations*, 2023.

Matteo Zecchin and Osvaldo Simeone. Localized adaptive risk control. In *Advances Neural Information Processing Systems*, volume 37, 2024.

Zhang, Roller, Goyal, Artetxe, Chen, Chen, and et al. OPT: Open pre-trained transformer language models. *Preprint at arXiv:2205.01068*, 2022.

Zhang, Li, Chen, Zhang, Ye, Bai, and et al. Beyond GPT-5: Making LLMs cheaper and better via performance-efficiency optimized routing. *Preprint at arXiv:2508.12631*, 2025a.

Zhang, Li, Wang, Chen, Zhang, Ye, and et al. The avengers: A simple recipe for uniting smaller language models to challenge proprietary giants. *Preprint at arXiv:2505.19797*, 2025b.

Zhang, Zheng, Wu, Zhang, Lin, Yu, and et al. The lessons of developing process reward models in mathematical reasoning. *Preprint at arXiv:2501.07301*, 2025c.

Xuechen Zhang, Zijian Huang, Ege Onur Taga, Carlee Joe-Wong, Samet Oymak, and Jiasi Chen. Efficient contextual LLM cascades through budget-constrained policy learning. *Preprint at arXiv:2404.13082*, 2024.

Zheng, Yin, Xie, Sun, Huang, Yu, and et al. SGLang: Efficient execution of structured language model programs. In *The Thirty-Eighth Annual Conference on Neural Information Processing Systems*, 2024.

Zhou, Tan, Li, Yao, Guo, Li, and et al. A theoretical study on bridging internal probability and self-consistency for LLM reasoning. *Preprint at arXiv:2510.15444*, 2025.

A Related work

Our work intersects efficiency improvement for reasoning models and the distribution-free inference for risk control of its performance loss with confidence.

Efficiency improvement for reasoning models. Large Reasoning Models (LRMs) have recently become a research hotspot due to their outstanding performance in handling complex tasks (Yue et al., 2025). However, the problem of overthinking has emerged (Sui et al., 2025; Chen et al., 2025), where LRMs tend to engage in unnecessarily long reasoning chains and redundant computational steps. This increases latency and cost, and may even cause error accumulation through extended reasoning paths. For example, in mathematical problems, LRMs may explore irrelevant solution branches or perform excessive intermediate calculations that do not contribute to the final answer. To alleviate this issue, recent studies propose efficient reasoning strategies such as early exit of thinking (Yang et al., 2025b; Jiang et al., 2025) and adaptive switching between "fast" and "slow" thinking modes to reduce reasoning tokens and avoid redundant steps (Cheng et al., 2025; Chung et al., 2025; Fang et al., 2025; Li et al., 2025b; Liang et al., 2025; Ma et al., 2025; Paliotta et al., 2025; Pan et al., 2024; Xiao and Gan, 2025; Yao et al., 2024; Yong et al., 2025; Zhang et al., 2025b;a).

LLM routing and cascades Table 2 compares the objectives and statistical analyses of six related methods. FrugalGPT learns a cascade and stopping thresholds under an average query-cost budget (Chen et al., 2023), while Mixture of Thoughts uses answer consistency across Chain-of-Thought and Program-of-Thought samples without training a separate router (Yue et al., 2023). TREACLE learns a budget-aware policy over models, prompts, and repeated queries (Zhang et al., 2024). Online cascade learning instead updates small models from LLM demonstrations as inputs arrive and gives a no-regret learning algorithm (Nie et al., 2024).

C3PO minimizes disagreement with the most powerful model subject to a bound on the probability of exceeding an inference-cost budget, and also gives regret generalization bounds (Valkanas et al., 2025). PAC Reasoning studies the complementary constraint: reducing thinking-model use while bounding expert-relative population loss by a user-chosen tolerance. Trust or Escalate calibrates selective LLM judges with exact binomial bounds and fixed-sequence testing to control disagreement with human preferences among accepted evaluations (Jung et al., 2024). Its cascade may abstain, whereas our loss averages over all inputs and the thinking model serves as the fallback.

Table 2: Objectives and statistical analyses of related LLM routing and cascade methods. The PAC Reasoning row states the conditions used by our selection theorem.

Method	Objective and controlled quantity	Calibration or analysis
FrugalGPT	Maximize task accuracy under an average inference-cost budget.	Supervised answer-correctness scores; empirical accuracy–cost evaluation.
Mixture of Thoughts	Reduce API cost while maintaining task accuracy.	Training-free answer consistency; empirical accuracy–cost evaluation.
TREACLE	Maximize answer accuracy under a total cost budget.	Learned reinforcement-learning policy; theory for oracle ordering and answer-consistency rules.
C3PO	Minimize strongest-model disagreement under a probabilistic cost constraint.	Conformal cost calibration under exchangeability; additional regret generalization bounds.
Trust or Escalate	Control human-disagreement risk among accepted LLM evaluations.	I.i.d. human-labelled calibration, exact binomial bounds, and fixed-sequence testing.
Online Cascade	Learn a cascade and deferral policy for input streams.	Online imitation learning from LLM demonstrations with a no-regret algorithm.
PAC Reasoning	Reduce thinking-model use subject to an expert-relative population-loss tolerance.	I.i.d. calibration, a fixed candidate sequence, and valid pointwise population-risk UCBs.

PAC-style error rate control. Conformal prediction provides finite-sample marginal coverage under exchangeability (Angelopoulos et al., 2025a; Liu et al., 2026). Conformal risk control extends this idea to the expected value of a monotone loss (Angelopoulos et al., 2023). Risk-controlling prediction sets provide distribution-free calibration of predictive risks (Bates et al., 2021), and Learn then Test

frames risk calibration as multiple hypothesis testing (Angelopoulos et al., 2025b). These methods have been applied to various domains, including conformal language modeling for text generation with guaranteed quality (Quach et al., 2024), and automatically adaptive conformal risk control that adjusts to the difficulty of test samples (Blot et al., 2025). Localized adaptive risk control enables online calibration with conditional guarantees (Zecchin and Simeone, 2024). PAC Reasoning uses this risk-control view for thinking/non-thinking acceleration. Its expert-relative loss makes the expert branch loss zero, and the fixed-sequence rule converts valid pointwise population-risk bounds into a guarantee for the selected threshold. Our Naive control baseline uses the empirical calibration loss directly, as specified in Appendix H.1.

B Finite-Sample Upper Confidence Bounds

This section presents an alternative algorithm for computing confidence bounds that provides strict finite-sample guarantees under bounded loss assumptions in Algorithm 3. This approach leverages concentration inequalities such as the Hoeffding inequality or betting-based confidence intervals (Bentkus, 2004; Hao et al., 2019; Hoeffding, 1994; Learned-Miller and Thomas, 2020; Ramdas et al., 2022; Waudby-Smith and Ramdas, 2021; 2024) which provides finite-sample validity, typically at the cost of looser bounds than the asymptotic normal approximation.

Algorithm 3 Compute Confidence Bound $\widehat{L}_u(\alpha)$ Based on Hoeffding Inequality

Input: Calibration set $\{(x_i, y_i)\}_{i=1}^n$, model with thinking f, model without thinking $\tilde{f}$, uncertainty scores $\{U_i\}_{i=1}^n$, sampling weights $\{\pi_i\}_{i=1}^n$, sampling size m, confidence level α, loss upper bound $B > 0$

Output: The finite-sample upper confidence bound $\widehat{L}_u(\alpha)$.

Initialize an empty list of samples $\mathcal{Z} = []$.
Let $\tilde{y}_i = \tilde{f}(x_i)$ for all $i = 1, \ldots, n$.
for $j = 1, \ldots, m$ **do**
  Sample an index $i_j \sim \text{Unif}(\{1, \ldots, n\})$.
  Sample a Bernoulli random variable $\xi_{i_j} \sim \text{Bern}(\pi_{i_j})$.
  if $\xi_{i_j} = 1$ **then**
    Query the true label y_{i_j} and compute $Z_j = \min\left(\frac{\ell(y_{i_j}, \tilde{y}_{i_j})}{\pi_{i_j}}, \frac{B}{\pi_{i_j}}\right)$.
  else
    $Z_j = 0$.
  end if
  Append Z_j to $\mathcal{Z}$.
end for
For a threshold u, define the variables $Z_j(u) = Z_j \cdot \mathbf{1}\{U_{i_j} < u\}$ for $j = 1, \ldots, m$.
$\widehat{\mu}_Z(u) \leftarrow \frac{1}{m}\sum_{j=1}^m Z_j(u)$
$R \leftarrow \frac{B}{\min_i \pi_i}$
$\delta_{\text{HB}}(\alpha) \leftarrow \sqrt{\frac{R^2 \log(2/\alpha)}{2m}}$
Return $\widehat{L}_u(\alpha) \leftarrow \widehat{\mu}_Z(u) + \delta_{\text{HB}}(\alpha)$.

C Validity of CLT-based upper confidence bound

We separate the n original calibration observations from the m auditing draws in Algorithm 1. For a fixed threshold u, write

$$g_i(u) = \ell(y_i, \tilde{y}_i)\mathbf{1}\{U_i < u\}, \qquad \overline{L}_n(u) = \frac{1}{n}\sum_{i=1}^n g_i(u).$$

Conditional on $\mathcal{D}_{cal}$ and its weights $0 < \pi_i \leq 1$, the independent index and fresh Bernoulli draws give

$$\mathbb{E}[Z_j(u) \mid \mathcal{D}_{cal}] = \overline{L}_n(u), \qquad \sigma^2_{Z,n}(u) := \text{Var}(Z_j(u) \mid \mathcal{D}_{cal}) = \frac{1}{n}\sum_{i=1}^n \frac{g_i(u)^2}{\pi_i} - \overline{L}_n(u)^2.$$

Proposition C.1 (Conditional asymptotic validity of the CLT UCB). *Fix $\alpha \in (0,1)$, the calibration pool, a threshold u, and weights for which $0 < \sigma^2_{Z,n}(u) < \infty$. For $m \geq 2$, let $\widehat{\mu}_Z(u) = m^{-1}\sum_{j=1}^m Z_j(u)$ and define the sample variance and sample standard deviation by*

$$\widehat{\sigma}^2_Z(u) = \frac{1}{m-1}\sum_{j=1}^{m}\big(Z_j(u) - \widehat{\mu}_Z(u)\big)^2, \qquad \widehat{\sigma}_Z(u) = \sqrt{\widehat{\sigma}^2_Z(u)}.$$

Define

$$\widehat{L}^{\mathrm{CLT}}_u(\alpha) := \widehat{\mu}_Z(u) + z_{1-\alpha}\frac{\widehat{\sigma}_Z(u)}{\sqrt{m}},$$

where $z_{1-\alpha}$ is the $(1-\alpha)$-quantile of the standard normal distribution. Then, with probability over the auditing draws,

$$\lim_{m\to\infty}\mathbb{P}\Big(\overline{L}_n(u) \leq \widehat{L}^{\mathrm{CLT}}_u(\alpha) \mid \mathcal{D}_{cal}\Big) = 1-\alpha.$$

Proof. Conditional on $\mathcal{D}_{cal}$, the variables $Z_j(u)$ are i.i.d. with finite positive variance, so the i.i.d. central limit theorem gives

$$\frac{\sqrt{m}\big(\widehat{\mu}_Z(u) - \overline{L}_n(u)\big)}{\sigma_{Z,n}(u)} \xrightarrow{d} \mathcal{N}(0,1).$$

The law of large numbers gives $\widehat{\sigma}^2_Z(u) \xrightarrow{p} \sigma^2_{Z,n}(u)$ and hence $\widehat{\sigma}_Z(u) \xrightarrow{p} \sigma_{Z,n}(u)$. All limits in this proof are conditional on the fixed pool. Slutsky's theorem and the symmetry of the standard normal law imply

$$\frac{\overline{L}_n(u) - \widehat{\mu}_Z(u)}{\widehat{\sigma}_Z(u)/\sqrt{m}} \xrightarrow{d} \mathcal{N}(0,1).$$

On the event $\widehat{\sigma}_Z(u) > 0$, this statistic is at most $z_{1-\alpha}$ exactly when $\overline{L}_n(u) \leq \widehat{L}^{\mathrm{CLT}}_u(\alpha)$. Since $\mathbb{P}(\widehat{\sigma}_Z(u) > 0 \mid \mathcal{D}_{cal}) \to 1$, the claim follows. □

Calibration and resampling randomness Under i.i.d. calibration sampling from P, the population risk is $R(u) = \mathbb{E}[\overline{L}_n(u)]$ and

$$\widehat{\mu}_Z(u) - R(u) = \big(\widehat{\mu}_Z(u) - \overline{L}_n(u)\big) + \big(\overline{L}_n(u) - R(u)\big).$$

The conditional CLT controls the first term, which comes from the m auditing draws. The second term comes from the n original calibration observations and remains random when m grows with n fixed. Thus, increasing m alone does not establish a population-risk UCB. To use the CLT UCB in Assumption 3.1, its studentized limit must hold for $R(u)$ under a joint sampling regime that includes both terms, or a separate bound must cover the calibration-sample error.

D Proof of Theorem 3.2

Proof. Write $\widehat{u} = u_{\widehat{k}}$, where $\widehat{k} \in \{0,\dots,K\}$ is the last index accepted by Algorithm 2. The fallback has risk $R(u_0) = 0$. If $R(u_k) \leq \epsilon$ for every $k = 1,\dots,K$, the output is safe for every realization of the calibration data and the UCB randomness.

Otherwise, define the first unsafe index

$$k^* = \min\{k \in \{1,\dots,K\} : R(u_k) > \epsilon\}.$$

The population law, models, score function, tolerance, and candidate grid are fixed before calibration. Hence k^* and u_{k^*} are fixed, even though they are unknown to the algorithm. By monotonicity, the safe grid indices are exactly $0,\dots,k^*-1$. In particular, the index k^* itself is unsafe.

If the selected policy is unsafe, then $\widehat{k} \geq k^*$. The algorithm accepts only a prefix of the fixed sequence and stops at the first failed test. It must therefore have accepted k^*, giving

$$\begin{aligned}\{R(\widehat{u}) > \epsilon\} &\subseteq \{\widehat{L}_{u_{k^*}}(\alpha) \leq \epsilon\} \\ &\subseteq \{\widehat{L}_{u_{k^*}}(\alpha) < R(u_{k^*})\}.\end{aligned}$$

Assumption 3.1, applied at this single fixed candidate, yields

$$\mathbb{P}\big(R(\widehat{u}) > \epsilon\big) \le \mathbb{P}\big(\widehat{L}_{u_{k^*}}(\alpha) < R(u_{k^*})\big) \le \alpha.$$

Taking complements proves the claim. This argument requires neither independence between tests nor a simultaneous confidence event over the grid. It also covers a failure at the first candidate and the case where all tests pass. □

Asymptotic pointwise bounds Keep P, the models, the score function, ϵ, α, and the finite grid fixed, and let $r \to \infty$ index a sampling regime for the UCB. Suppose that, for each fixed grid point,

$$\limsup_{r\to\infty} \mathbb{P}\big(\widehat{L}_u^{(r)}(\alpha) < R(u)\big) \le \alpha.$$

Apply the same selection rule to these bounds and denote its output by $\widehat{u}^{(r)}$. If an unsafe candidate exists, the same fixed k^* gives the event inclusion above for every r. If no unsafe candidate exists, every output is safe. Thus,

$$\liminf_{r\to\infty} \mathbb{P}\big(R(\widehat{u}^{(r)}) \le \epsilon\big) \ge 1 - \alpha.$$

This result requires pointwise population-risk validity under the full calibration and UCB sampling law.

E Proof of Theorem 3.3

E.1 Notation recalling and lemma

We present PAC guarantees for the empirical test risk under precise Hoeffding conditions, making explicit the roles of calibration and test randomness. For any threshold u, the deployment rule T_u predicts with the expert when $U(x) \ge u$ and otherwise uses the fast model. The population risk is

$$R(u) \;=\; \mathbb{E}_{(x,y)\sim P}\big[\ell\big(y,\, T_u(x)\big)\big].$$

Given an independent test set $\mathcal{D}_{\text{test}}$ drawn i.i.d. from P, the empirical test risk is

$$\widehat{R}(u) \;=\; \frac{1}{N}\sum_{j=n+1}^{n+N} \ell\big(y_j,\, T_u(x_j)\big).$$

Our guarantee for the empirical test risk uses the conditions of Theorem 3.2 and the following lemma.

Lemma E.1 (Conditional Hoeffding bound)**.** *Let $\widehat{u}$ be determined by $\mathcal{D}_{cal}$ and the UCB randomization, jointly independent of the i.i.d. test sample. Conditioned on $\widehat{u}$, the test losses $Z_j(\widehat{u}) := \ell(y_j, T_{\widehat{u}}(x_j))$ for $j \in \mathcal{I}_{test}$ are then i.i.d. with mean $R(\widehat{u})$. Assume they are bounded in $[a, b]$ with $a < b$. Then for any $t > 0$,*

$$\mathbb{P}\big(\widehat{R}(\widehat{u}) - R(\widehat{u}) > t \,\big|\, \widehat{u}\big) \;\le\; \exp\Big(-\tfrac{2Nt^2}{(b-a)^2}\Big).$$

Proof. Let $Z_j(\widehat{u}) = \ell(y_j, T_{\widehat{u}}(x_j))$ for $j \in \mathcal{I}_{test}$. The threshold $\widehat{u}$ is fixed when we condition on it. Since the i.i.d. test sample is independent of the calibration data and UCB randomization, the variables $\{Z_j(\widehat{u}) : j \in \mathcal{I}_{test}\}$ are conditionally independent and identically distributed.

By the boundedness of the loss function, each $Z_j(\widehat{u})$ is bounded in $[a, b]$. The conditional expectation of each $Z_j(\widehat{u})$ is $\mathbb{E}[Z_j(\widehat{u}) \,|\, \widehat{u}] = \mathbb{E}[\ell(y_j, T_{\widehat{u}}(x_j)) \,|\, \widehat{u}]$. Since (x_j, y_j) is independent of $\widehat{u}$, this is equal to the unconditional expectation over the data distribution, $\mathbb{E}_{(x,y)\sim P}[\ell(y, T_{\widehat{u}}(x))]$, which is the definition of the true risk $R(\widehat{u})$. The empirical risk is the sample mean:

$$\widehat{R}(\widehat{u}) = \frac{1}{N}\sum_{j\in\mathcal{I}_{test}} Z_j(\widehat{u}).$$

Its conditional expectation is

$$\mathbb{E}[\widehat{R}(\widehat{u}) \mid \widehat{u}] = \frac{1}{N}\sum_{j\in\mathcal{I}_{test}} \mathbb{E}[Z_j(\widehat{u}) \mid \widehat{u}] = R(\widehat{u}).$$

We can now apply Hoeffding's inequality (Hoeffding, 1994) to the conditional i.i.d. bounded variables $Z_j(\widehat{u})$. For any $t > 0$, the one-sided version states that

$$\mathbb{P}\left(\frac{1}{N}\sum_{j=n+1}^{n+N} Z_j(\widehat{u}) - \mathbb{E}\left[\frac{1}{N}\sum_{j=n+1}^{n+N} Z_j(\widehat{u}) \,\middle|\, \widehat{u}\right] > t \,\middle|\, \widehat{u}\right) \le \exp\left(-\frac{2Nt^2}{(b-a)^2}\right).$$

Substituting the empirical risk and true risk, we get

$$\mathbb{P}\left(\widehat{R}(\widehat{u}) - R(\widehat{u}) > t \,\middle|\, \widehat{u}\right) \le \exp\left(-\frac{2Nt^2}{(b-a)^2}\right).$$

This completes the proof. □

E.2 Proof of Theorem 3.3

Proof. The i.i.d. test sample is independent of the calibration data and UCB randomization, so Lemma E.1 applies. Use the inclusion

$$\{\widehat{R}(\widehat{u}) > \epsilon + t\} \subseteq \{R(\widehat{u}) > \epsilon\} \cup \{\widehat{R}(\widehat{u}) - R(\widehat{u}) > t\}$$

and take probabilities. Combine the result of Theorem 3.2 (i.e., $\mathbb{P}(R(\widehat{u}) > \epsilon) \le \alpha$) with the law of total probability and Lemma E.1 to conclude the claim:

$$\begin{aligned}\mathbb{P}(\widehat{R}(\widehat{u}) > \epsilon + t) &\le \mathbb{P}(R(\widehat{u}) > \epsilon) + \mathbb{P}(\widehat{R}(\widehat{u}) - R(\widehat{u}) > t) \\ &= \mathbb{P}(R(\widehat{u}) > \epsilon) + \mathbb{E}\big[\mathbb{P}\big(\widehat{R}(\widehat{u}) - R(\widehat{u}) > t \,\big|\, \widehat{u}\big)\big] \\ &\le \alpha + \exp\Big(-\tfrac{2Nt^2}{(b-a)^2}\Big).\end{aligned}$$

This is equivalent to the stated guarantee. □

F Transductive PAC Reasoning

The transductive target is the average loss on a fixed pool, not the population risk on future inputs. Condition on the finite collection

$$\mathcal{F} = \{(x_i, U_i, \tilde{y}_i, y_i)\}_{i=1}^n, \qquad \tilde{y}_i = \tilde{f}(x_i), \quad y_i = f(x_i).$$

The expert answers may be unknown to the algorithm, but they are fixed values for this analysis. If generation is random, fix the generated outputs or their random seeds as part of $\mathcal{F}$. Let A denote the randomness from index sampling, expert-query decisions, and any further UCB randomization. For the same strict-threshold routing rule as Equation (1), define

$$L_{\mathcal{F}}(u) = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, \tilde{y}_i)\mathbf{1}\{U_i < u\}.$$

The loss is nonnegative, so $L_{\mathcal{F}}(u)$ is non-decreasing and $L_{\mathcal{F}}(0) = 0$.

Choose a finite ordered grid $\Lambda_{\mathcal{F}} = \{0 < u_1 < \cdots < u_K \le 1\}$ and fix its order before the random sampling and expert queries used for calibration. Unlike the inductive grid, this grid may depend on the known scores of the fixed pool. What matters is that it is fixed conditional on $\mathcal{F}$, and is not chosen or reordered from the realized query losses or UCB values. For each candidate, require the design-conditional pointwise bound

$$\mathbb{P}_A\Big(L_{\mathcal{F}}(u_k) \le \widehat{L}_{u_k}(\alpha) \mid \mathcal{F}\Big) \ge 1 - \alpha, \qquad k = 1, \ldots, K. \tag{8}$$

For a fixed draw count m, use uniform index sampling with replacement and fresh independent Bernoulli query decisions with fixed probabilities $\pi_i > 0$ in Algorithms 1 and 3. The weighted draw

$$Z_j(u) = \frac{\xi_j}{\pi_{i_j}} \ell(y_{i_j}, \tilde{y}_{i_j})\mathbf{1}\{U_{i_j} < u\}$$

then has conditional mean $L_{\mathcal{F}}(u)$. Cached expert answers may be reused, but the Bernoulli decision is redrawn at every index draw. If $0 \le \ell \le B$, these draws lie in $[0, B/\min_i \pi_i]$. Writing $C = B/\min_i \pi_i$, Hoeffding's inequality (Hoeffding, 1994) gives

$$\mathbb{P}_A\left(L_{\mathcal{F}}(u) > \frac{1}{m}\sum_{j=1}^{m} Z_j(u) + C\sqrt{\frac{\log(2/\alpha)}{2m}} \,\middle|\, \mathcal{F}\right) \le \frac{\alpha}{2} \le \alpha.$$

This is the bound used in Algorithm 3 and establishes Equation (8) under this sampling design. A CLT bound gives only an asymptotic version when its conditional variance and normal-approximation conditions hold. Algorithm 4 separates this pointwise-bound construction from the fixed-sequence selection rule.

Algorithm 4 Transductive PAC Reasoning

Input: Fixed inputs and scores $\{(x_i, U_i, \tilde{y}_i)\}_{i=1}^n$, expert f, loss ℓ, tolerance ϵ, error level α, fixed query design, grid $\Lambda_{\mathcal{F}}$ fixed before calibration queries, UCB procedure satisfying Equation (8)
Output: Outputs $\{\widehat{y}_i\}_{i=1}^n$ and threshold $\widehat{u}$

Draw the calibration queries and construct the pointwise UCBs using the fixed query design.
Initialize $\widehat{u} \leftarrow u_0 = 0$.
for $k = 1, \dots, K$ **do**
  if $\widehat{L}_{u_k}(\alpha) > \epsilon$ **then**
    break
  end if
  $\widehat{u} \leftarrow u_k$.
end for
for $i = 1, \dots, n$ **do**
  if $U_i \ge \widehat{u}$ **then**
    Obtain y_i from the expert, reusing a cached answer if available, and set $\widehat{y}_i \leftarrow y_i$.
  else
    $\widehat{y}_i \leftarrow \tilde{y}_i$.
  end if
end for
Return $\{\widehat{y}_i\}_{i=1}^n$ and $\widehat{u}$.

Theorem F.1 (Transductive PAC guarantee)**.** *For a fixed pool $\mathcal{F}$ and a grid fixed before its calibration queries, suppose Equation (8) holds at every candidate. Then Algorithm 4, with its first-failure stopping rule and expert-only fallback, satisfies*

$$\mathbb{P}_A\left(\frac{1}{n}\sum_{i=1}^{n} \ell(y_i, \widehat{y}_i) = L_{\mathcal{F}}(\widehat{u}) \le \epsilon \,\middle|\, \mathcal{F}\right) \ge 1 - \alpha.$$

Proof. Conditional on $\mathcal{F}$, the risks and the candidate sequence are fixed. If every candidate has loss at most ϵ, all possible outputs, including $u_0 = 0$, are safe. Otherwise, let $k^* = \min\{k : L_{\mathcal{F}}(u_k) > \epsilon\}$. Selecting an unsafe candidate requires reaching an index at least k^* and hence passing the test at k^*. Therefore,

$$\begin{aligned}\mathbb{P}_A\big(L_{\mathcal{F}}(\widehat{u}) > \epsilon \mid \mathcal{F}\big) &\le \mathbb{P}_A\big(\widehat{L}_{u_{k^*}}(\alpha) \le \epsilon \mid \mathcal{F}\big) \\ &\le \mathbb{P}_A\big(\widehat{L}_{u_{k^*}}(\alpha) < L_{\mathcal{F}}(u_{k^*}) \mid \mathcal{F}\big) \le \alpha.\end{aligned}$$

The last inequality is Equation (8) at the single fixed first unsafe candidate. The final expert queries only supply the outputs required by the selected policy; they do not change the fixed pool or threshold. The loss identity follows from $\ell(y_i, y_i) = 0$ and the common strict-threshold rule. □

***Remark* 3** (Comparison with the inductive setting)**.** The inductive guarantee controls the population risk of the selected policy, with probability over calibration data and algorithm randomness. An independent test sample is used to evaluate that policy, not to define its population-risk guarantee. The transductive guarantee instead controls loss on the given fixed pool, with probability over the query design. No i.i.d. assumption on that pool is needed for this conditional statement, but it does not imply risk control on a new population. With only asymptotic conditional pointwise validity on a fixed grid, the same first-unsafe-index inclusion gives an asymptotic transductive guarantee.

G Multi-level PAC Reasoning

We extend the selection argument to a fixed sequence of multi-level policies. Fix an expert f, two fast models $\tilde{f}_1, \tilde{f}_2$, and a score function $U : \mathcal{X} \to [0,1]$ before calibration. For $y = f(x)$ and $0 \le u_1 \le u_2 \le 1$, define the policy

$$T_{u_1,u_2}(x) = \begin{cases} \tilde{f}_1(x), & U(x) < u_1, \\ \tilde{f}_2(x), & u_1 \le U(x) < u_2, \\ f(x), & U(x) \ge u_2. \end{cases}$$

The intervals do not overlap, even when $u_1 = u_2$ or a score equals a threshold. In particular, $T_{0,0} = f$ is a safe fallback, including for inputs with score zero. Let $d_k(x) = \ell(y, \tilde{f}_k(x))$ for $k = 1, 2$. The population risk is

$$R(u_1, u_2) = \mathbb{E}_{x\sim P}[d_1(x)\mathbf{1}\{U(x) < u_1\} + d_2(x)\mathbf{1}\{u_1 \le U(x) < u_2\}] .$$

Ordering models by cost or average accuracy does not imply that this risk is non-decreasing in every threshold. For example, at fixed u_2 and $0 \le a \le b \le u_2$,

$$R(b, u_2) - R(a, u_2) = \mathbb{E}[(d_1(x) - d_2(x))\mathbf{1}\{a \le U(x) < b\}] ,$$

which can have either sign. We therefore do not rely on coordinate-wise risk monotonicity or search for a maximum over all pairs that happen to pass.

Fixed policy sequence Before observing $\mathcal{D}_{cal}$, fix a finite ordered list of threshold pairs $\mathbf{u}^{(1)}, \dots, \mathbf{u}^{(J)}$, with $J \ge 1$ and $0 \le u_1^{(j)} \le u_2^{(j)} \le 1$. Set $T_j = T_{\mathbf{u}^{(j)}}$, $T_0 = f$, and $r_j = \mathbb{E}_{x\sim P}[\ell(y, T_j(x))]$, so $r_0 = 0$. Assume that the expert-relative losses of the fixed fast models are nonnegative and integrable, with $\ell(y, y) = 0$. For each fixed candidate, require a pointwise population-risk bound $B_j(\alpha)$ with

$$\mathbb{P}\big(r_j \le B_j(\alpha)\big) \ge 1 - \alpha, \qquad j = 1, \dots, J, \tag{9}$$

under the full calibration and UCB sampling law. To estimate a candidate risk, its losses must use exactly the same tier intervals as T_j. The UCB construction must satisfy Equation (9) for the population risk of each policy.

Test $B_j(\alpha) \le \epsilon$ in this fixed order, stop at the first failure, and return the preceding policy. In symbols,

$$\tau = \min(\{j \in \{1, \dots, J\} : B_j(\alpha) > \epsilon\} \cup \{J + 1\}) , \qquad \widehat{j} = \tau - 1, \quad \widehat{f} = T_{\widehat{j}}.$$

A first-candidate failure returns $T_0 = f$, and passing all tests returns T_J.

Proposition G.1 (Fixed-sequence multi-level guarantee)**.** *For the fixed policy list above, assume $\epsilon > 0$, $\alpha \in (0, 1)$, and Equation (9). Then the first-failure rule satisfies*

$$\mathbb{P}\big(r_{\widehat{j}} \le \epsilon\big) \ge 1 - \alpha.$$

No monotonicity of $r_1, \dots, r_J$ is required.

Proof. If no candidate is unsafe, the selected policy and the expert fallback are safe. Otherwise, let $j^* = \min\{j : r_j > \epsilon\}$. This is a fixed index because the policy list and population risks do not depend on calibration data. If $r_{\widehat{j}} > \epsilon$, then $\widehat{j} \ge j^*$ by the definition of the first unsafe index, regardless of whether later risks increase or decrease. Returning $T_{\widehat{j}}$ requires all tests $1, \dots, \widehat{j}$ to pass, including the test at j^*. Consequently,

$$\{r_{\widehat{j}} > \epsilon\} \subseteq \{B_{j^*}(\alpha) \le \epsilon < r_{j^*}\},$$

whose probability is at most α by Equation (9). □

The guarantee applies to the last accepted policy in the fixed sequence. For a fixed finite policy list, asymptotic pointwise bounds give the corresponding asymptotic conclusion by the same event inclusion.

Table 3: Prompt for the verbalized confidence scores.

System prompt: You are a reasoning assistant. For each question and proposed answer, you must estimate how likely the proposed answer is correct.
User prompt:
Question: {QUESTION}
Answer: {ANSWER}
Provide a probability (between 0.0 and 1.0) that your answer is correct. Only output the probability.

Extension to more tiers Let $M \geq 2$ be the number of models, with fast models $\tilde{f}_1, \ldots, \tilde{f}_{M-1}$ and expert f. For thresholds $0 \leq u_1 \leq \cdots \leq u_{M-1} \leq 1$, use

$$T_{\mathbf{u}}(x) = \begin{cases} \tilde{f}_1(x), & U(x) < u_1, \\ \tilde{f}_k(x), & u_{k-1} \leq U(x) < u_k, \quad k = 2, \ldots, M-1, \\ f(x), & U(x) \geq u_{M-1}. \end{cases}$$

With $d_k(x) = \ell(y, \tilde{f}_k(x))$, its risk is

$$R(\mathbf{u}) = \mathbb{E}\left[d_1(x)\mathbf{1}\{U(x) < u_1\} + \sum_{k=2}^{M-1} d_k(x)\mathbf{1}\{u_{k-1} \leq U(x) < u_k\} \right].$$

An all-zero threshold vector is expert-only. Fix a finite ordered list of such vectors before calibration and require a pointwise population-risk UCB for each associated policy. Proposition G.1 then gives the risk guarantee for the selected policy.

H Experimental details

Hyperparameter settings of LLMs In this study, we configure the decoding parameters as follows: for Qwen/Qwen3-4B-Instruct-2507, we set *Temperature* = 0.7, *TopP* = 0.8, *TopK* = 20, and *MinP* = 0; for Qwen/Qwen3-4B-Thinking-2507, we set *Temperature* = 0.6, *TopP* = 0.95, *TopK* = 20, and *MinP* = 0. Experiments were run on one NVIDIA RTX A6000 Graphics Card.

The prompt for verbalized uncertainty score In Table 3, we present the prompt used to elicit the verbalized confidence scores. After ten trials, we obtained the average confidence score and defined the verbalized uncertainty score as 1 minus this average confidence.

Details of Datasets Table 4 summarizes the datasets employed in our experiments, together with their corresponding splitting strategies. For each dataset, we report its type, overall size, and the partitioning into PAC calibration and PAC test sets.

Table 4: The details of datasets and splitting settings for PAC experiments

Dataset	Dataset Type	Dataset Size	Split Setting	Size
MATH-500	Math Reasoning	500	PAC Calibration	300
			PAC Test	200
ZebraLogic	Text reasoning	1000	PAC Calibration	500
			PAC Test	500
Arena-Hard	Alignment Task	750	PAC Calibration	450
			PAC Test	300

H.1 Baselines

We compare empirical-mean calibration, learned routing (Feng et al., 2025), prompting-based efficient reasoning (Xu et al., 2025), and reasoning-free generation (Ma et al., 2025). The following definitions specify the implementations used in our experiments.

Naive threshold A simple baseline that routes each query to the thinking model whenever its uncertainty score U_i exceeds a fixed threshold, and to the non-thinking model otherwise. We set the threshold to 0.05 in our experiments.

Naive control This baseline calibrates the routing threshold using the empirical mean loss. Specifically, given a target error tolerance ϵ, this baseline selects the largest threshold u such that the empirical loss

$$L(u) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, \tilde{y}_i) \mathbf{1}\{U_i \leq u\}$$

is at most ϵ on the calibration set. It compares the empirical loss directly with ϵ, while PAC Reasoning tests a pointwise UCB in the fixed order of Algorithm 2.

Router We train a learned router for the model pair using the open-source library LLM-ROUTER (Feng et al., 2025). Specifically, we sample 2,000 instances from eight widely used benchmarks, including ARC-Challenge (Clark et al., 2018), CommonsenseQA (Talmor et al., 2019), GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021b), HumanEval (Chen et al., 2021), MMLU (Hendrycks et al., 2021a), NaturalQA (Kwiatkowski et al., 2019), and TriviaQA (Joshi et al., 2017). The router is implemented as a lightweight classifier that takes the input prompt as features and outputs a routing score, representing the predicted probability that invoking the thinking model is necessary. At inference time, inputs are routed to the thinking model if the predicted probability exceeds a fixed threshold of 0.5, and otherwise handled by the non-thinking model. Such a router does not provide explicit guarantees on the resulting performance loss after routing.

Chain of Draft Chain of Draft (CoD) (Xu et al., 2025) is a prompting-based method designed to improve reasoning efficiency by explicitly constraining the verbosity of intermediate reasoning steps. Instead of generating full chain-of-thoughts, CoD enforces a compact "draft" for each reasoning step, typically limited to a few words, thereby reducing token usage while preserving a minimal reasoning structure. The specific prompt template used in our experiments is shown in Table 5. As a prompting strategy, CoD focuses on reducing reasoning length, but does not control the performance loss induced by truncated reasoning.

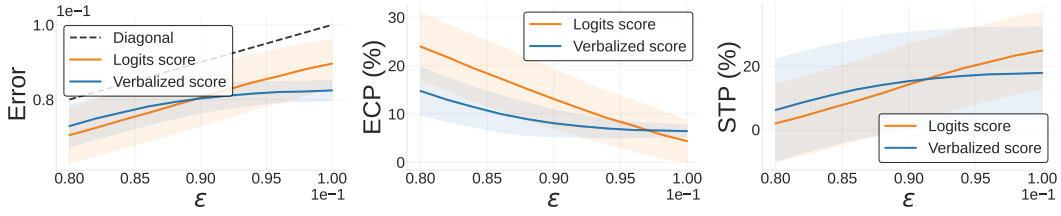
Table 5: Prompt for Chain of Draft.

System prompt: Think step by step, but only keep a minimum draft for each thinking step, with at most five words. Finally, put your final answer within `\boxed{}`. **User prompt:** Question: {QUESTION}

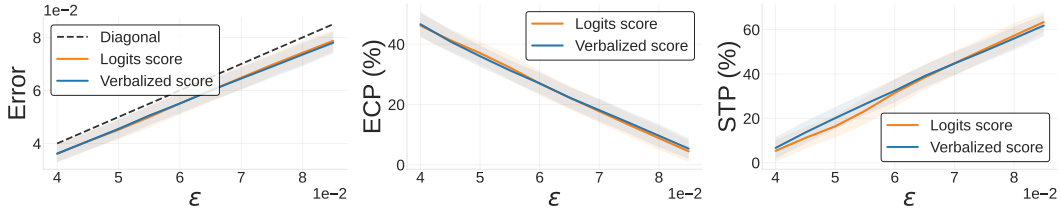
NoThinking NoThinking (Ma et al., 2025) eliminates explicit chain-of-thought reasoning by removing special reasoning markers (e.g., `<think>`) and relying on direct answer generation combined with simple aggregation strategies. The method shows that, for certain reasoning tasks, models can achieve competitive performance without explicitly generating intermediate reasoning steps, leading to reduced inference cost and latency. However, NoThinking does not incorporate uncertainty-aware decision mechanisms, nor does it provide explicit control or theoretical guarantees over the resulting performance degradation.

H.2 Choice of loss functions

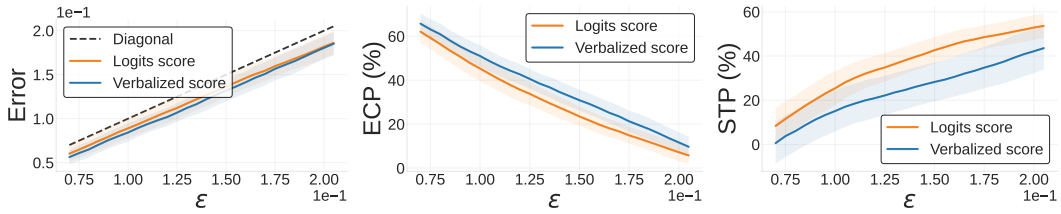
The loss functions, shown as in Equation (6) and Equation (7), serve distinct purposes in evaluating the PAC Reasoning. The semantic cosine distance captures the degree of semantic alignment between the PAC Reasoning's prediction and reference outputs. It is particularly suitable for tasks where nuanced differences in meaning are critical, such as natural language understanding or generation tasks. By using the "Qwen/Qwen3-Embedding4B" model, we ensure that the embeddings capture rich contextual information and compare semantic content in high-dimensional spaces. In contrast, the binary 0–1 loss is designed for scenarios where the correctness of the generated answer is verifiable, such as in mathematical problem-solving or multiple-choice question answering. This loss function is effective for evaluating the framework's ability to produce exact matches to ground-truth answers, emphasizing precision in verifiable tasks. By testing PAC Reasoning on these two loss functions, we can assess its semantic quality and factual accuracy across diverse tasks.



(a) MATH-500



(b) ZebraLogic



(c) Arena-Hard

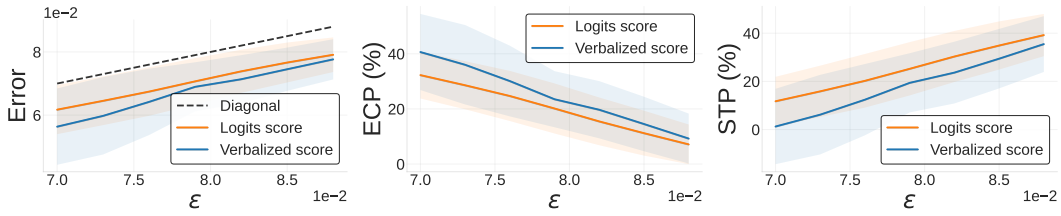
Figure 2: **Error control, ECP and STP of PAC Reasoning** for semantic loss across three benchmarks under a confidence level of $\alpha = 0.05$, on the Llama-3.1-8B–based models. Uncertainty score includes the logits-based score and the verbalized score. All experiments are repeated 100 times, and the shaded areas represent standard deviations.

I Additional experiments

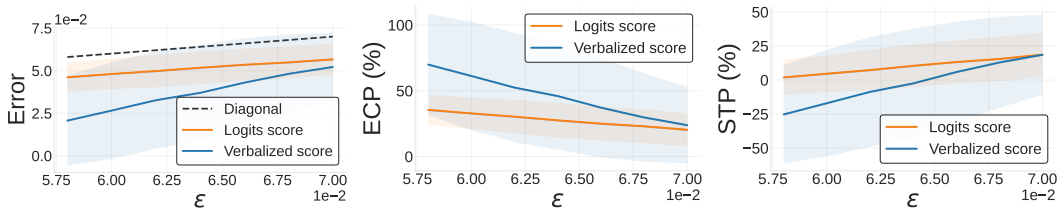
I.1 Experimental results of Llama-based LLMs

In this section, we evaluate PAC Reasoning on additional LLM architectures and larger-scale models to further verify the generalizability of our framework. Specifically, we conduct experiments using Llama-3.1-8B–based models: the "DeepSeek-R1-Distill-Llama-8B" as the thinking model and "Llama-3.1-8B-Instruct" as the lower-performance non-thinking model. We configure the decoding parameters as follows: for Llama-3.1-8B-Instruct, we set *Temperature* = 0.6, *TopP* = 0.95, *TopK* = 20, *MinP* = 0, and *max_tokens* = 4096. For DeepSeek-R1-Distill-Llama-8B, we set *Temperature* = 0.6, *TopP* = 0.95, *TopK* = 20, and *MinP* = 0. Experiments were run on one NVIDIA RTX A6000 Graphics Card. Other experimental details follow Appendix H.

Figure 2 summarizes the performance of PAC Reasoning on Llama-3.1-8B–based models. Across all three benchmarks, PAC Reasoning consistently maintains valid performance loss control, with empirical performance loss staying below the diagonal reference line. For uncertainty estimation, the logits-based score exhibits tighter calibration and lower ECP than the verbalized score, particularly under smaller ϵ values, while the verbalized score shows slightly higher variance but still adheres to theoretical bounds. In terms of efficiency, both scores achieve substantial STP, demonstrating that PAC Reasoning can reliably identify confident cases and reduce unnecessary calls to the thinking model. Overall, the results confirm that PAC Reasoning generalizes well to larger LLM architectures and continues to deliver stable performance loss control and efficiency gains.



(a) GPQA



(b) HumanEval

Figure 3: Error control, ECP and STP of PAC Reasoning for semantic loss for GPQA and HumanEval at a confidence level of $\alpha = 0.05$. Uncertainty score includes the logits-based score and the verbalized score. The red dashed line ϵ denotes the target tolerance, and the shaded areas represent standard deviations. All experiments are repeated 100 times.

I.2 Additional benchmarks

In this section, we further validate the effectiveness of our framework on additional datasets, including GPQA (Rein et al., 2024) and HumanEval (Chen et al., 2021). We choose the semantic loss function to evaluate the performance loss. The experimental results shown in Figure 3 evaluate the performance of PAC Reasoning on these benchmarks using two types of uncertainty scores: the logits-based score and the verbalized score. Across both datasets, the two uncertainty scores demonstrate valid error control. The ECP decreases steadily as the tolerance level ϵ grows, with the verbalized score exhibiting slightly worse ECP performance. For STP, the logits-based score consistently achieves higher performance compared to the verbalized score. Overall, these observations indicate that PAC Reasoning remains effective and reliable when applied to a broader range of datasets.

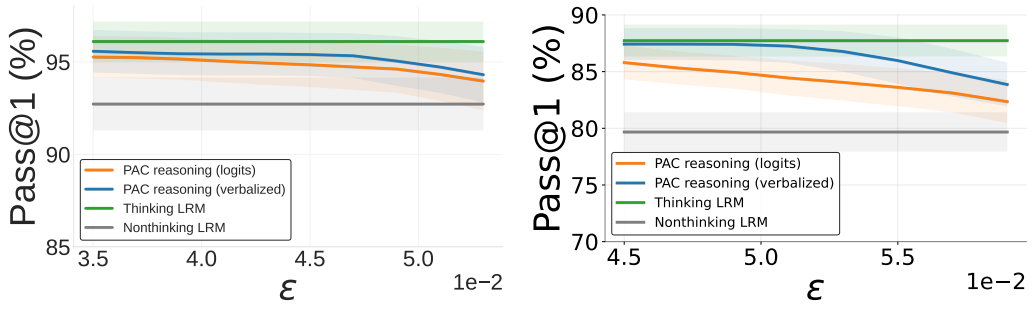
I.3 Accuracy of PAC Reasoning

In this section, we investigate the effectiveness of the PAC Reasoning framework when controlled by semantic loss. As shown in Figure 4, applying semantic loss to regulate the PAC filtering process leads to consistently improved final accuracies across both MATH-500 and ZebraLogic. Both logits-based and verbalized uncertainty scores yield higher Pass@1 performance than the baseline non-thinking model, with the verbalized score performing the best. The results indicate that PAC Reasoning controlled by semantic loss reliably improves output accuracy while remaining robust to different ϵ settings.

I.4 Experimental results for the binary loss

Empirical test-loss success rates. Table 6 reports the success counts from 100 repeated runs on MATH-500 and ZebraLogic. A run is counted as successful when its held-out empirical test loss is at most ϵ. We report the number of successful runs out of 100 and a two-sided exact 95% Clopper–Pearson binomial interval for the run-level success probability. These intervals describe success under the repeated experimental protocol on the fixed benchmark data, rather than population-risk coverage.

Wall-clock latency and dollar cost. We further report total wall-clock latency, token usage, and dollar cost of PAC Reasoning on MATH-500 and ZebraLogic, summed over the test set, following the experimental setup of Table 1 in the main manuscript. Dollar costs are computed based on Alibaba Cloud Model Studio pricing: Qwen3-4B-Thinking-2507 costs $1.26 per 1M output tokens



(a) MATH-500 (b) ZebraLogic

Figure 4: Accuracy of PAC Reasoning based on semantic loss across reasoning benchmarks at a confidence level of $\alpha = 0.05$. Uncertainty score includes the logits-based score and the verbalized score. All experiments are repeated 100 times, and the shaded areas represent standard deviations.

Table 6: Empirical test-loss success counts and two-sided exact 95% binomial intervals over 100 runs. Qwen uses $\epsilon = 0.03$ on MATH-500 and $\epsilon = 0.08$ on ZebraLogic; the Llama pair uses $\epsilon = 0.1$ on both datasets. The intervals are in percent.

	MATH-500		ZebraLogic	
Method / score	Success / 100	95% interval	Success / 100	95% interval
Qwen pair (logits score)				
PAC Reasoning	89	[81.17, 94.38]	85	[76.47, 91.35]
Naive threshold	15	[8.65, 23.53]	100	[96.38, 100.00]
Naive control	36	[26.64, 46.21]	0	[0.00, 3.62]
CoD	0	[0.00, 3.62]	100	[96.38, 100.00]
NoThinking	0	[0.00, 3.62]	100	[96.38, 100.00]
Router	16	[9.43, 24.68]	66	[55.85, 75.18]
Llama pair (PAC Reasoning)				
Logits	92	[84.84, 96.48]	91	[83.60, 95.80]
Verbalized	87	[78.80, 92.89]	88	[79.98, 93.64]

and Qwen3-4B-Instruct-2507 costs $0.42 per 1M output tokens, with both models sharing an input price of $0.11 per 1M tokens.

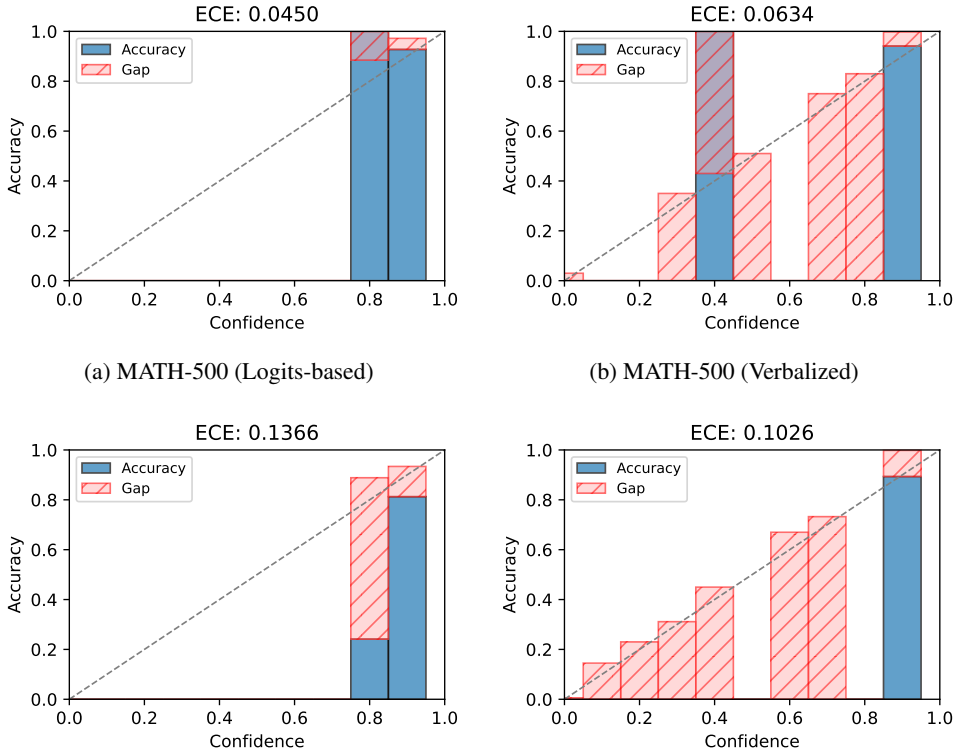
As shown in Table 7, PAC Reasoning significantly reduces both wall-clock latency and dollar cost. On MATH-500, latency is reduced from 266.4k s to 82.2k s and cost from 1.71 to 0.78, corresponding to reductions of 69.23% and 54.46%, respectively. On ZebraLogic, latency decreases from 59.8k s to 43.8k s and cost from 4.64 to 1.98, yielding reductions of 26.76% and 57.37%.

I.5 Expected calibration error of two uncertainty scores

We evaluate the calibration quality of uncertainty estimates on MATH-500 and ZebraLogic using Qwen3-4B-Instruct-2507. We consider two uncertainty scores: a logits-based score derived from the model's predictive distribution and a verbalized score elicited from the model's self-reported confidence. Expected calibration error (ECE) (Guo et al., 2017) quantifies the discrepancy between predicted confidence and empirical accuracy via binning and a weighted average of absolute gaps, where smaller values indicate better calibration. Across both benchmarks, the logits-based score exhibits consistently lower ECE and smoother reliability than the verbalized score, indicating tighter calibration of uncertainty estimates. The verbalized score shows higher variance and mild overconfidence in high-confidence bins. These findings support the use of the logits-based score within PAC Reasoning and motivate improved elicitation methods for verbalized confidence. Figure 5 summarizes the reliability plots and aggregated ECE. All experiments are repeated 100 times under the decoding configuration described in Section 4.1.

Table 7: Wall-clock latency, token usage, and dollar cost comparison.

Dataset	Method	Total Latency (s)	Dollar Cost ($)	Cheap Model Tokens	Thinking Model Tokens
MATH-500	Only Thinking Model	266.4k	1.71	–	1358.9k
	PAC Reasoning	82.2k	0.78	724.4k	125.5k
ZebraLogic	Only Thinking Model	59.8k	4.64	–	3670.1k
	PAC Reasoning	43.8k	1.98	1713.3k	981.6k



(a) MATH-500 (Logits-based)

(b) MATH-500 (Verbalized)

(c) ZebraLogic (Logits-based)

(d) ZebraLogic (Verbalized)

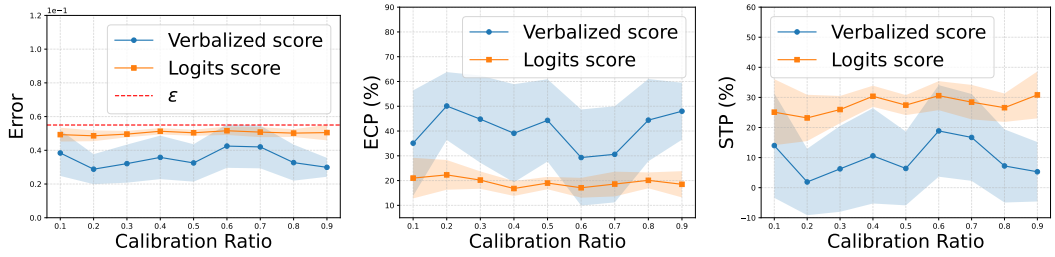
Figure 5: Expected Calibration Error across two reasoning benchmarks.

I.6 Ablation study on the size of the calibration set

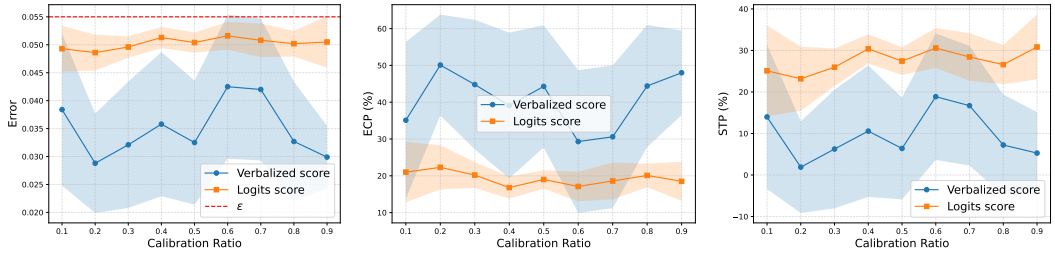
We conduct experiments to investigate the stability of efficiency gains under different calibration set sizes. Specifically, we repeat the experiments with varying calibration ratios to examine how the size of the correction set influences performance. The results are presented in Figure 6. PAC Reasoning maintains stable error control and consistent uncertainty calibration across all benchmarks. Both uncertainty scoring methods control the target performance loss, though the verbalized score exhibits larger variance. Moreover, the logits-based score consistently outperforms the verbalized uncertainty score, achieving lower ECP and higher STP. These findings demonstrate that PAC Reasoning can maintain valid performance loss control and stable efficiency gains under varying calibration dataset sizes.

I.7 Reward score as an alternative uncertainty score

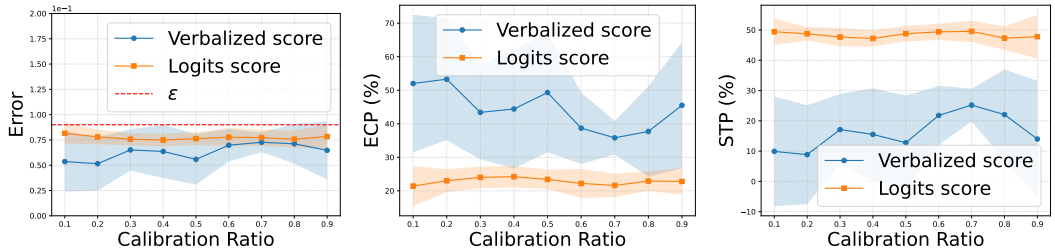
In this part, we evaluate whether PAC Reasoning remains valid when replacing the original uncertainty score with the reward score. Concretely, we apply PAC Reasoning to MATH-500 and ZebraLogic using the reward score as the uncertainty estimate, and we report its error control based on the semantic cosine distance, ECP, and STP under varying ϵ. We follow the experimental setting described in Section 4.1, and the reward model is "Qwen2.5-Math-PRM-7B" (Zhang et al., 2025c).



(a) MATH-500



(b) ZebraLogic



(c) Arena-Hard

Figure 6: Error control, ECP and STP of PAC Reasoning for semantic loss for different calibration ratios at a confidence level of $\alpha = 0.05$. Uncertainty score includes the logits-based score and the verbalized score. The red dashed line ϵ denotes the target tolerance, and the shaded areas represent standard deviations. All experiments are repeated 100 times.

The results are presented in Figure 7. Across both benchmarks, the observed error curves remain below the diagonal baseline, indicating that PAC Reasoning still satisfies the theoretical error guarantee even with this alternative scoring method. For efficiency, ECP consistently decreases as ϵ increases, showing that the method becomes more selective. These results show that PAC Reasoning is robust to the choice of uncertainty score: using the reward score still ensures valid error control and provides reasonable efficiency gains.

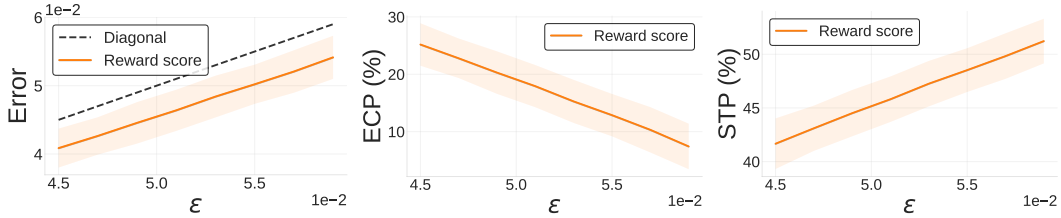
I.8 Empirical behavior under distribution shift

While the inductive population-risk guarantee assumes calibration and deployment draws from the same target law, we provide empirical observations on how PAC Reasoning behaves under a moderate distribution shift. Specifically, we calibrate on 300 examples from MATH-500 and evaluate on mixed test sets of 200 examples that include 20, 40, or 60 AIME problems. Each setting is repeated 100 times under the binary loss. As shown in Table 8, the **empirical error of PAC Reasoning remains below the target tolerance across all settings**.

Moreover, a key observation is a conservative escalation effect: AIME problems tend to induce higher uncertainty, causing them to exceed the threshold calibrated on MATH-500 more frequently. Consequently, the escalation rate (ECP) increases while the token savings (STP) decrease as the proportion of AIME problems grows. Despite this shift, the method still maintains non-trivial efficiency gains, with time savings ranging from 61.73% to 69.02% and monetary savings from 21.17% to 37.24%.



(a) MATH-500



(b) ZebraLogic

Figure 7: Error control, ECP and STP of PAC Reasoning for semantic loss across two reasoning benchmarks at a confidence level of $\alpha = 0.05$. Uncertainty score is the **reward score**. All experiments are repeated 100 times, and the shaded areas represent standard deviations.

Table 8: Empirical performance under distribution shift with varying proportions of AIME problems. Calibration is performed on MATH-500 ($\epsilon = 0.03$).

Uncertainty Score	AIME Size	Error	STP (%)	ECP (%)	Time Saving (%)	Money Saving (%)
Logits-based score	20	0.0193	19.02	26.92	69.02	36.50
	40	0.0220	6.56	32.55	67.38	26.51
	60	0.0211	0.51	36.97	66.00	21.17
Verbalized score	20	0.0208	19.76	30.18	64.62	37.24
	40	0.0278	7.27	35.49	63.04	27.21
	60	0.0289	1.19	39.58	61.73	21.86

I.9 Accuracy–cost operating points

We vary ϵ to obtain operating points at expert-call rates of about 10%, 15%, 40%, and 65% on MATH-500 and ZebraLogic. Tables 9 and 10 compare PAC Reasoning with Naive control, using $\alpha = 0.05$ and 100 random calibration/test splits. ECP measures the fraction of queries sent to the thinking model. The methods use different tolerances to reach similar call rates. Their accuracies are close at these operating points, while the mean loss stays below the stated tolerance for PAC Reasoning and exceeds it for Naive control in each reported row. Table 11 gives the accuracy of CoD and NoThinking, which have no escalation stage.

Table 9: Accuracy–cost operating points on MATH-500 with 300 calibration and 200 test examples. Entries are mean $\pm$ standard deviation over 100 splits; ECP and accuracy are percentages.

Target ECP	Method	ϵ	Loss	ECP (%)	Accuracy (%)
$\approx 10\%$	PAC Reasoning	0.050	0.031 ± 0.015	8.8 ± 8.9	93.7 ± 1.7
	Naive control	0.030	0.031 ± 0.014	7.8 ± 2.8	93.6 ± 1.7
$\approx 15\%$	PAC Reasoning	0.040	0.026 ± 0.013	13.4 ± 12.1	94.1 ± 1.6
	Naive control	0.020	0.022 ± 0.012	13.6 ± 6.8	94.4 ± 1.5
$\approx 40\%$	PAC Reasoning	0.020	0.014 ± 0.010	38.5 ± 21.9	95.1 ± 1.4
	Naive control	0.010	0.013 ± 0.009	40.9 ± 17.0	95.1 ± 1.3
$\approx 65\%$	PAC Reasoning	0.010	0.009 ± 0.008	63.3 ± 22.1	95.5 ± 1.2
	Naive control	0.005	0.007 ± 0.007	65.5 ± 9.8	95.6 ± 1.2

Table 10: Accuracy–cost operating points on ZebraLogic with 500 calibration and 500 test examples. Entries are mean ± standard deviation over 100 splits; ECP and accuracy are percentages.

Target ECP	Method	ϵ	Loss	ECP (%)	Accuracy (%)
$\approx 10\%$	PAC Reasoning	0.105	0.083 ± 0.021	10.3 ± 5.8	83.2 ± 2.1
	Naive control	0.082	0.084 ± 0.016	9.7 ± 3.1	83.1 ± 1.6
$\approx 15\%$	PAC Reasoning	0.090	0.071 ± 0.020	15.0 ± 6.4	84.2 ± 2.0
	Naive control	0.068	0.070 ± 0.015	14.8 ± 3.3	84.2 ± 1.5
$\approx 40\%$	PAC Reasoning	0.043	0.032 ± 0.012	39.5 ± 7.8	87.1 ± 1.3
	Naive control	0.030	0.031 ± 0.010	40.6 ± 4.8	87.2 ± 1.2
$\approx 65\%$	PAC Reasoning	0.009	0.006 ± 0.006	66.2 ± 10.4	88.7 ± 1.0
	Naive control	0.004	0.006 ± 0.005	64.1 ± 4.1	88.7 ± 1.0

Table 11: Accuracy of the fixed baselines without escalation, in percent. Entries are mean ± standard deviation.

Dataset	CoD	NoThinking
MATH-500	54.93 ± 2.70	43.95 ± 2.38
ZebraLogic	88.06 ± 1.06	88.12 ± 1.11

Naive-control tolerance sweep Figure 8 compares PAC Reasoning and naive control on MATH-500 using logits scores and the semantic loss of Figure 1. The tolerance ranges from 0.035 to 0.053, with 100 random calibration/test splits at each value. Both mean-loss curves lie below the tolerance line in this experiment. PAC Reasoning leaves a larger margin below the tolerance, while naive control tracks the line more closely.

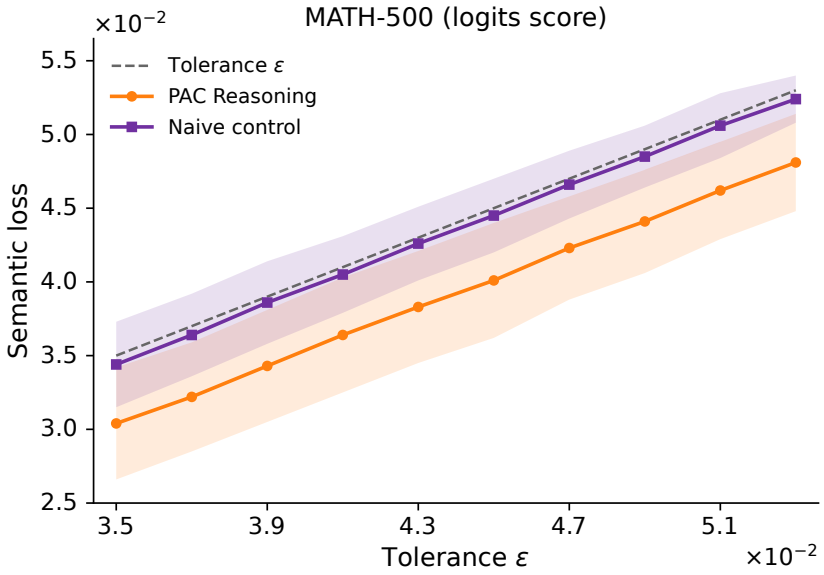


Figure 8: MATH-500 comparison with naive control using logits scores. The curves show mean semantic loss over 100 random calibration/test splits, and the shaded areas show one standard deviation. The dashed line marks the target tolerance.

I.10 Additional distribution-shift results

Mathematics-to-code transfer We calibrate on 300 MATH-500 examples and evaluate on HumanEval with $\epsilon = 0.1$ over 100 repetitions. Table 12 reports the loss and routing costs for both uncertainty scores. The logits-based score uses the expert more often and gives lower loss, while the verbalized score gives larger token savings in this setting.

Table 12: MATH-500 calibration followed by HumanEval evaluation at $\epsilon = 0.1$. Loss is mean $\pm$ standard deviation over 100 repetitions; ECP and STP are percentages.

Score	Loss	ECP (%)	STP (%)
Logits	0.0305 ± 0.0021	14.63	23.33
Verbalized	0.0732 ± 0.0018	0.61	45.82

AIME-only evaluation Table 13 compares MATH-500 and AIME-only operating points after MATH-500 calibration. The AIME-only evaluation uses $\epsilon = 0.02$. On AIME alone, ECP rises to 98.33% for logits scores and 78.33% for verbalized scores, while STP becomes -33.20% and -14.29%, respectively. The fast-model output is generated before escalation, so frequent expert calls can make the combined token cost exceed that of the thinking model alone.

Table 13: MATH-500 reference results and AIME-only evaluation with MATH-500 calibration at $\epsilon = 0.02$ for the AIME-only setting. Values are point summaries; ECP and STP are percentages.

Score	Test set	Loss	ECP (%)	STP (%)
Logits	MATH-500	0.0209	20.32	39.10
	AIME	0.0000	98.33	-33.20
Verbalized	MATH-500	0.0198	25.32	34.44
	AIME	0.0100	78.33	-14.29

I.11 Additional results for the Llama pair

Table 14 reports PAC Reasoning and the baselines with DeepSeek-R1-Distill-Llama-8B and Llama-3.1-8B-Instruct at $\epsilon = 0.1$ on both datasets. It complements the semantic-loss sweeps in Appendix I.1 and the success counts in Table 6. For PAC Reasoning, the mean loss is below 0.1 in all four settings, with positive mean token savings.

Table 14: Results for the Llama-based model pair at $\epsilon = 0.1$. Loss is mean $\pm$ standard deviation; ECP and STP are mean percentages over 100 runs. SR is the reported percentage of runs with empirical test loss at most ϵ; a dash marks an entry not reported in the source table.

Method	Score	Loss	ECP (%)	STP (%)	SR (%)
MATH-500					
Naive threshold	Logits	0.0000 ± 0.0000	100.00	-10.06	100
	Verbalized	0.2575 ± 0.1129	4.75	81.34	3
Naive control	Logits	0.1713 ± 0.1026	14.62	72.83	14
	Verbalized	0.2550 ± 0.1158	7.00	79.62	4
PAC Reasoning	Logits	0.0898 ± 0.1137	68.88	11.75	92
	Verbalized	0.0788 ± 0.1246	54.00	36.47	87
Router	–	0.0000 ± 0.0000	74.50	14.68	56
CoD	–	0.1349 ± 0.0125	–	23.44	100
NoThinking	–	0.0938 ± 0.0817	–	0.76	37
ZebraLogic					
Naive threshold	Logits	0.0745 ± 0.0086	52.32	-30.35	100
	Verbalized	0.1040 ± 0.0095	70.32	-76.30	40
Naive control	Logits	0.2405 ± 0.0152	5.23	1.77	0
	Verbalized	0.2262 ± 0.0144	23.29	-27.30	0
PAC Reasoning	Logits	0.0826 ± 0.0217	49.93	1.26	91
	Verbalized	0.0825 ± 0.0199	76.78	3.66	88
Router	–	0.0108 ± 0.0031	95.62	0.53	48
CoD	–	0.1278 ± 0.0110	–	1.90	1
NoThinking	–	0.1124 ± 0.0097	–	-1.82	22

I.12 Score resolution and threshold sensitivity

We use a synthetic experiment to separate score compression, quantization, and sparse-loss effects. For each observation, draw $S \sim \mathrm{Unif}(0,1)$ and a binary loss $D \mid S = s \sim \mathrm{Bernoulli}(p_\lambda(s))$, where

$$p_\lambda(s) = \lambda(0.01 + 0.20s^2), \qquad \epsilon_\lambda = 0.04\lambda, \qquad \lambda \in \{1, 0.25\}.$$

At latent cutoff t, the population risk and fast-route rate are

$$R_\lambda(t) = \lambda\left(0.01t + \frac{0.20}{3}t^3\right), \qquad A(t) = t.$$

The fast route incurs loss D, while the expert route incurs zero loss. In the continuous-score conditions, we fix the latent candidate cutoffs $t_k = k/1000$ before calibration. We use $n \in \{500, 2000\}$, $\alpha = 0.05$, and 1,000 paired repetitions per condition. Each repetition uses the losses of all n original observations and applies fixed-sequence selection with CLT, Hoeffding, or exact one-sided binomial UCBs. We use the known population risk to estimate the violation probability $\mathbb{P}(R_\lambda(\widehat{t}) > \epsilon_\lambda)$ across repetitions.

Continuous compression We define $U^{(c)} = 0.05 + c(S - 1/2)$ for $c \in \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$ and transform the pre-specified candidate thresholds by the same map. This keeps the candidate routing sets fixed while changing the numerical scale of the scores. Across 12,000 paired comparisons, the selected latent rank is unchanged.

Quantization and finite precision We quantize the score into $K \in \{100, 20, 5\}$ levels and average over four bin phases. Tied score blocks enter the fast-route set together. Table 15 reports the fast-route rate and the two-sided one-level routing jump, measured between the candidates one level below and one level above the selected threshold. Coarser levels produce larger routing jumps and lower fast-route rates in this experiment. With fixed decimal precision of 10^{-3}, compressing the score width from 10^{-1} to 10^{-2} lowers the mean fast-route rate from 0.6716 to 0.6246; at width 10^{-3}, selection returns the expert-only action.

Table 15: Score quantization for the CLT rule with $n = 500$ and $\lambda = 1$. Values average four bin phases; rates and jumps are fractions.

Levels K	Tie mass	Two-sided routing jump	Mean fast-route rate
100	0.01	0.02	0.6716
20	0.05	0.10	0.6516
5	0.20	0.40	0.5760

Sparse losses and UCB choice For regular losses ($\lambda = 1$), the CLT violation rate is 0.067 at $n = 500$ and 0.050 at $n = 2000$. For sparse losses ($\lambda = 0.25$) at $n = 500$, it rises to 0.117 with an exact 95% interval of $[0.0977, 0.1386]$, while the exact-binomial UCB gives 0.043 with interval $[0.0313, 0.0575]$. The Hoeffding rule has no observed violations in these continuous-score conditions; at $n = 500$, it selects the expert-only action in both loss regimes. Table 16 also reports the $n = 2000$ results. These full-observation experiments separate the effect of score resolution from the small-sample behavior of the UCB approximation. Figure 9 summarizes the routing and violation results.

Table 16: Population-risk violation rates and two-sided exact 95% binomial intervals for continuous scores. All entries are percentages, with 1,000 repetitions per setting and $\alpha = 0.05$.

n	λ	CLT	Exact binomial	Hoeffding
500	1	6.70 [5.23, 8.43]	4.10 [2.96, 5.52]	0.00 [0.00, 0.37]
500	0.25	11.70 [9.77, 13.86]	4.30 [3.13, 5.75]	0.00 [0.00, 0.37]
2000	1	5.00 [3.73, 6.54]	3.80 [2.70, 5.18]	0.00 [0.00, 0.37]
2000	0.25	5.80 [4.43, 7.43]	3.40 [2.37, 4.72]	0.00 [0.00, 0.37]

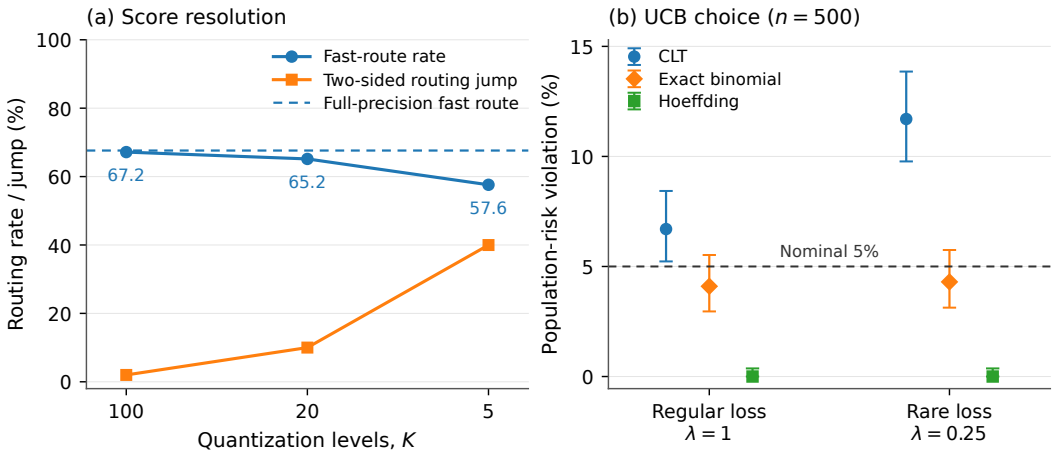


Figure 9: Score resolution and UCB choice in the full-observation synthetic experiment, with $\alpha = 0.05$ and 1,000 repetitions. (a) Mean fast-route rate and two-sided one-level routing jump for $n = 500$, $\lambda = 1$, and the CLT UCB, averaged over four bin phases. The dashed line is the full-precision fast-route rate. (b) Population-risk violation rates for continuous scores at $n = 500$, with two-sided exact 95% binomial intervals. The dashed line marks the nominal 5% level.

I.13 Calibration choices and amortized cost

Choosing the tolerance and model pair The tolerance ϵ expresses the application's acceptable loss relative to the chosen thinking model. Its numerical scale depends on the loss, task, model pair, and their disagreement. The values 0.03 and 0.08 in Table 1 are representative binary-loss operating points for MATH-500 and ZebraLogic. The tolerance sweeps in Figures 1, 2, 3, and 7 show how routing and savings change within each setting. The framework uses two callable models, an expert-relative loss, and an uncertainty score, with no requirement that the models share a family. The current experiments cover the Qwen pair and the Llama-based pair described in Section 4.1. Useful pair choices balance the thinking model's quality, the fast model's cost, and the observed disagreement between their outputs. If all queries are escalated after fast-model generation, ECP is 100% and the extra fast-model tokens can make STP negative.

Query events and cached expert calls Importance weighting supports partial or unequal-probability expert queries while preserving the conditional pool mean. For uniform sampling of indices with replacement and a fresh query decision of probability π on each draw, the expected number of query events is $m\pi$. With one cached expert answer per calibration item, the expected number of distinct expert calls is

$$\mathbb{E}[N_{\text{unique}}] = n\left[1 - \left(1 - \frac{\pi}{n}\right)^m\right].$$

Indeed, a fixed item is queried on one draw with probability π/n, so the chance that it is never queried in m draws is $(1 - \pi/n)^m$. At the experimental choice $m = n/\pi$, the expected number of query events is n, while the expected number of distinct calls is approximately $(1 - e^{-1})n \approx 0.632n$ for large n. The number of distinct cached expert calls is at most n. The choice $\pi = 0.5$ is an experimental setting; changing π changes the sampling variance, weighted range, and query cost. With $\pi = 1$, every sampled index is queried, whereas full-observation calibration evaluates each of the n original observations once.

Deployment amortization Table 17 reports mean token use, including calibration for PAC Reasoning. On MATH-500, the reported means are 6.06k for PAC Reasoning, 6.63k for CoD, and 9.63k for NoThinking. On ZebraLogic, the corresponding means are 7.53k, 7.34k, and 7.38k.

Let C_{cal} be the one-time calibration cost, and let C_{deploy} and C_{think} be the mean cost per deployment query for PAC Reasoning and the thinking model alone, measured in the same unit. For N deployment queries,

$$C_{\text{total}}(N) = C_{\text{cal}} + NC_{\text{deploy}}, \qquad C_{\text{avg}}(N) = \frac{C_{\text{cal}}}{N} + C_{\text{deploy}}.$$

Table 17: Reported mean token use, in thousands of tokens. PAC Reasoning includes calibration-stage token use.

Method	MATH-500	ZebraLogic
NoThinking	9.63	7.38
CoD	6.63	7.34
PAC Reasoning	6.06	7.53

When $C_{\mathrm{deploy}} < C_{\mathrm{think}}$, the costs match at

$$N^* = \frac{C_{\mathrm{cal}}}{C_{\mathrm{think}} - C_{\mathrm{deploy}}},$$

and PAC Reasoning has lower total cost for $N > N^*$. If $C_{\mathrm{deploy}} \geq C_{\mathrm{think}}$, amortization does not produce a finite break-even point. The calibration cost includes fast-model generation, uncertainty scoring, distinct expert queries, and loss evaluation. The test-stage measurements in Table 7 provide the deployment component of this accounting.

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 abstract and introduction state the main contribution as PAC-style control of performance loss for efficient reasoning, and this claim is matched by the theoretical results in Sections 3–4 and the experiments in Sections 5–6.

   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 6.1 discusses the sampling conditions, calibration cost, distribution shift, and the expert-relative loss target.

   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: Section 3 states the assumptions for the fixed-sequence selection theorem, including pointwise population-risk validity, and gives proof sketches. Appendix D gives the selection proof and its asymptotic version under the stated pointwise validity condition. The CLT result in Appendix C is conditional on a fixed calibration pool; it does not by itself establish population-risk coverage.

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 and Appendix H describe the model pairs, uncertainty scores, data splits, loss functions, metrics, prompts, and decoding settings. The public code repository is `https://github.com/Jianguo99/PAC-Reasoning`. Its README provides installation instructions, data-access requirements, and commands for generation, scoring, calibration, and evaluation.

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: The code is publicly available at https://github.com/Jianguo99/PAC-Reasoning under the Apache-2.0 license. The repository README describes the software environment, data preparation, and scripts for the main experiments. It also identifies the access requirements for gated datasets, including ZebraLogic and GPQA.

   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: Section 4.1 and Appendix H describe the calibration/test splits, model choices, decoding settings, prompts, repeated-run protocol, and baselines. The public repository linked above gives commands for running generation, scoring, calibration, and evaluation with these settings.

   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: The figures and tables report means with standard deviations over 100 repeated experiments, and the captions explicitly state that shaded areas represent standard deviations.

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: The appendix specifies the compute worker as one NVIDIA RTX A6000 graphics card, and the paper further reports wall-clock latency and dollar cost for representative experiments.

   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: To the best of our knowledge, the research follows the NeurIPS Code of Ethics and uses existing benchmark datasets and models.

   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: [No]

    Justification: The current draft does not include a discussion of broader societal impacts, either positive or negative.

    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 studies a routing method on existing models and benchmarks, and it does not release a new high-risk model or scraped dataset that would require special release safeguards.

    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: [No]

Justification: The public code repository uses the Apache-2.0 license and credits the evaluation tools and prompts on which it builds. The paper cites the benchmark datasets and model families, but it does not provide a full list of licenses and terms of use for the third-party datasets, models, and software used in the experiments.

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: [N/A]

Justification: The paper does not present a new released dataset, model, or benchmark asset as a primary contribution in the current submission text.

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 report crowdsourcing experiments or research with human subjects conducted by the authors.

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 research by the authors, 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: [Yes]

    Justification: The method uses existing LLMs as the thinking and non-thinking models in the proposed routing framework. In addition, LLM-generated uncertainty estimates, such as verbalized confidence scores, are used as one possible scoring function for calibration and routing. The paper describes these components, including the model choices, prompts, and decoding settings, in the experimental setup and appendix.

    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.