ON CALIBRATION OF LARGE LANGUAGE MODELS: FROM RESPONSE TO CAPABILITY

Anonymous authors
Paper under double-blind review

ABSTRACT

Accurate confidence estimation is critical for reliable use of large language models (LLMs). Prior work on LLM calibration largely focuses on response-level confidence, which estimates the correctness of a single generated output. However, this formulation is misaligned with many practical settings where the central question is how likely a model is to solve a query overall. We show that this mismatch results from the stochastic nature of modern LLM decoding, under which single-response correctness fails to reflect underlying model capability. To address this issue, we introduce **capability calibration, a new evaluation framework** for measuring how well query-level confidence aligns with a model's expected accuracy on individual queries. We formally distinguish capability calibration (CC) from response calibration (RC) and show that the two differ both theoretically and empirically. We further show that CC is better suited than RC to applications like pass@k prediction and inference budget allocation. Finally, we evaluate common confidence estimation methods to understand the practical feasibility of CC.

1 INTRODUCTION

Large language models (LLMs) have reshaped human-AI interaction by enabling users to pose queries in natural language and receive informative responses Ouyang et al. (2022). This intuitive interface has driven their rapid adoption across a wide range of applications. However, despite their fluency, LLMs produce incorrect or misleading outputs without explicitly signaling uncertainty. This limitation makes accurate confidence estimation a critical component of reliable LLM deployment. Well-calibrated confidence scores enable users to better judge when to trust LLMs (Huang et al., 2024b; Aljohani et al., 2025), allow systems to selectively refuse or defer to human experts (Wu et al., 2024), and support performance prediction for downstream tasks.

Given the important role of confidence estimation, a natural question is how to evaluate its quality. Calibration (Niculescu-Mizil & Caruana, 2005; Guo et al., 2017) provides a principled framework by evaluating how well estimated confidence aligns with correctness probability. Existing work on LLM calibration (Geng et al., 2024) adopts a response-level formulation: given a query x and a generated response $\hat{y}$, a confidence estimator produces a score s as the probability that $\hat{y}$ is correct with respect to x. Under this formulation, calibration is evaluated independently for each generated response. We refer to this setting as response calibration (Figure 1a).

In many practical settings, however, what matters is not whether a particular response $\hat{y}$ is correct, but how likely the LLM is to solve a given query overall. This question naturally arises in applications such as allocating computational resources across queries (Chen et al., 2024; Ong et al., 2025) or predicting model performance in downstream pipelines. We refer to this quantity, *"how likely can the LLM answer this query correctly?"*, as query-level confidence. While response-level confidence and query-level confidence are equivalent for deterministic models (e.g., standard image classification where prediction is fixed), the two are fundamentally misaligned in LLMs due to the stochastic nature of text generation. Modern LLMs typically achieve better performance with stochastic decoding (Holtzman et al., 2020; Shi et al., 2024), such as non-zero temperature sampling (Renze, 2024; Li et al., 2025a), which can produce different responses given the same query across inference calls. As a result, the correctness of any single sampled response cannot accurately reflect the LLM's underlying capability on that query. This mismatch between single-response correctness and query-level performance lies beyond what response calibration can capture.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

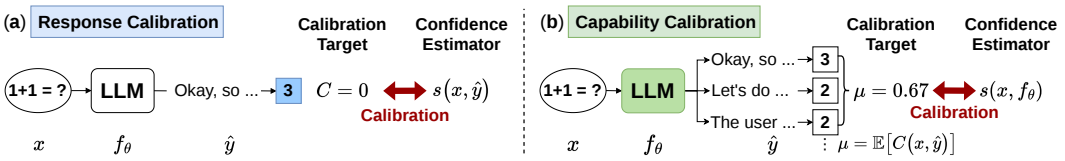

Figure 1: **(a) Response calibration.** Given an input x, a model f_θ, and its *single* sampled output $\hat{y}$, response calibration calibrates confidence $s(x, \hat{y})$ against correctness $\mathcal{C}$ of $\hat{y}$. **(b) Capability calibration.** We propose capability calibration, which calibrates confidence $s(x, f_\theta)$ against expected accuracy μ of f_θ's output distribution. While response calibration is limited by LLM stochastic decoding, capability calibration directly targets expected accuracy, providing a more reliable and efficient metric for tasks like forecasting test-time scaling performance and resource allocation.

Motivated by these observations, we introduce capability calibration (Figure 1b), which targets query-level expected accuracy: the probability that a response sampled from the model's output distribution conditioned on x would be correct. This formulation shifts the focus from whether a particular sampled response happens to be correct to the model's expected capability of solving the query. We formally distinguish capability calibration from response calibration and show that they differ both theoretically (§3.2) and empirically (§5.1). Notably, capability calibration is not merely the expectation of response calibration; the two quantities differ precisely by the variance of response correctness under the model's output distribution. Their key differences are summarized in Table 1.

We show that capability calibration is better suited than response calibration to practical applications that require query-level estimates of model capability (§4). We study two such applications: (1) pass@k estimation (Schaeffer et al., 2025; Kazdan et al., 2025), where confidence estimates are used to predict the pass@k success rate of individual queries without costly repeated sampling during inference, and (2) inference budget allocation (Snell et al., 2025; Damani et al., 2025), where confidence estimates guide the allocation of computational resources across queries. Empirically, capability-calibrated confidence yields stronger oracle performance than response-calibrated confidence, and practical capability-calibrated estimators consistently outperform standard baselines. We also discuss other applications where capability calibration may be beneficial.

After demonstrating the practical value of capability calibration, we study how to evaluate and estimate it. We benchmark commonly used training-free and training-based confidence estimation methods after adapting each to the capability calibration objective. Experimental results indicate that training linear probes on LLM activations offers a favorable tradeoff between online computational cost and confidence estimation performance (§5.3.2). We also conduct extensive experiments using our evaluation framework to stress-test existing confidence estimators, motivating future work to develop stronger capability-calibrated confidence estimators. By formally defining capability calibration, demonstrating its practical utility, and establishing it as **an evaluation framework**, our work offers a new perspective on LLM calibration that directly captures query-level model capability.

2 Related Works

2.1 LLM Confidence Estimation and Calibration

Confidence estimation focuses on estimating the probability that predictions are correct. Previous works (Niculescu-Mizil & Caruana, 2005; Guo et al., 2017) define calibration as the agreement between confidence and the correctness of an output. We call this definition **response calibration**. Denote x as the input, $\hat{y}$ as the model's output, estimated confidence as $s(x, \hat{y})$, and $\mathcal{C}(x, \hat{y}) \in \{0, 1\}$ as the correctness function. Formally, $\mathcal{C}(x, \hat{y}) = \mathbf{1}[\hat{y} \text{ is correct for } x]$. Perfect response calibration is defined as $\mathbb{P}\left[\mathcal{C}(x, \hat{y}) = 1 \mid s(x, \hat{y}) = p\right] = p$.

A commonly used evaluation metric is Brier score (Brier, 1950). Let $\mathcal{D} = \{x_i\}_{i=1}^N$ be a dataset of inputs, $\mathbf{s} = [s_1, s_2, ...s_i, ...s_N]$ be the estimated confidence for each query, and $\hat{y}_i$ be the sampled response for input x_i. The Brier score used in response calibration is

$$\mathcal{L}_{\text{Brier}}^{\text{response}}(\mathbf{s}) \triangleq \frac{1}{N} \sum_{i=1}^{N} \left(s_i - \mathcal{C}(x_i, \hat{y}_i)\right)^2. \tag{1}$$

Table 1: **Comparison of calibration definitions.** Unlike existing response calibration that assesses whether the confidence estimate s aligns with the correctness of one decoded answer $\hat{y}$, our proposed capability calibration evaluates whether s aligns with the model f_θ's capability to answer query x.

Definition	Calibration Target	Interpretation	Dependence on LM f_θ
Response Calibration	Accuracy of $\hat{y}$ given x	How likely is $\hat{y}$ correct?	**No.** Since $\hat{y}$ is already decoded, the estimation of $s(x, \hat{y})$ is decoupled from the generating model f_θ.
Capability Calibration	Expected accuracy of f_θ given x	How likely is f_θ to answer x correctly?	**Yes.** The expected accuracy for x depends on f_θ's capability.

To achieve response calibration for LLMs, training-free (Lin et al., 2022; Tian et al., 2023; Kadavath et al., 2022; Manakul et al., 2023) and training-based (Zhang et al., 2025a; Damani et al., 2026; Wu et al., 2025a; Li et al., 2025b) confidence estimation methods have been proposed. These methods typically operate post-hoc, estimating confidence only after the output is generated. In contrast, a body of work focuses on anticipating the performance of a single response, aiming to estimate the response's correctness before it is generated (Zhou et al., 2022; Cencerrado et al., 2025; Schellaert et al., 2025). Detailed descriptions of these methods are in Appendix C.1.

Despite the methodological differences in estimating confidence, their calibration target remains the same: they aim to estimate the correctness probability of a single response. This target is natural for deterministic models, such as standard image classifiers, where an `argmax` prediction is produced for each input. As a result, the correctness of a single prediction coincides with the model's expected accuracy, making response calibration and capability calibration equivalent. However, LLMs mark a shift from deterministic prediction to stochastic generation, where the same query yields different responses, some correct and some incorrect. Consequently, the correctness of any one sampled response is only a noisy realization of the model's underlying ability to solve the query. This shift makes response calibration insufficient for characterizing query-level model capability, and motivates **capability calibration** as **a new evaluation framework** for calibration of stochastic models.

2.2 LLM UNCERTAINTY QUANTIFICATION

LLM uncertainty quantification (UQ) studies methods for estimating how uncertain a model is on an input. Unlike calibration, which evaluates how well confidence scores align with correctness probabilities, UQ is often evaluated by the usefulness of uncertainty scores in downstream decision-making (Huang et al., 2024a), such as distinguishing correct from incorrect predictions or deciding when to abstain (Kamath et al., 2020; Chen et al., 2024; Kang et al., 2025). Therefore, common evaluation metrics are AUROC (Hendrycks & Gimpel, 2017) and the Risk-Coverage curve (Geifman & El-Yaniv, 2017). Details of common LLM UQ methods are in Appendix C.1.

Capability calibration and LLM UQ are related but have distinct goals: while UQ produces uncertainty scores, capability calibration evaluates if these score represent true probabilities of query-level success. Consequently, capability-calibrated scores can make UQ outputs more interpretable for applications.

3 CAPABILITY CALIBRATION

Capability calibration measures whether the estimated confidence on a query reflects the expected accuracy of responses sampled from an output distribution. We formalize this framework below.

3.1 DEFINITION

For a given input x, we define the model's **expected accuracy** as

$$\mu(x, f_\theta) \triangleq \mathbb{P}_{\hat{y}\sim f_\theta(\cdot|x)}[\mathcal{C}(x, \hat{y}) = 1] = \mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x)}[\mathcal{C}(x, \hat{y})], \tag{2}$$

which is equivalent to $\mu(x, f_\theta) = \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} \mathcal{C}(x, \hat{y}_i), \quad \hat{y}_i \sim f_\theta(\cdot \mid x)$.

Equation (2) defines the target of the capability calibration. As illustrated in Figure 1, the target expected accuracy $\mu(x, f_\theta)$ is defined as the frequency of correct outputs when the language model is sampled infinitely many times on the same input x. Notably, the concept of "expected accuracy" exists in different applications (Kazdan et al., 2025; Damani et al., 2025). However, they do not formalize it as confidence calibration target, nor use confidence estimators to estimate the quantity.

In capability calibration, an estimated confidence s is well calibrated if it is aligned with the expected accuracy $\mu(x, f_\theta)$. The perfect calibration for capability calibration is $s^* = \mu(x, f_\theta)$.

We adopt Brier score (Brier, 1950), one of the most common metrics used to evaluate query-level calibration, as the primary metric. Specifically, let μ_i be the expected accuracy $\mu(x_i, f_\theta)$, the capability calibration Brier score is

$$\mathcal{L}_{\text{Brier}}^{\text{capability}}(\mathbf{s}) \triangleq \frac{1}{N}\sum_{i=1}^{N}(s_i - \mu_i)^2. \quad (3)$$

This metric directly evaluates capability calibration by measuring how closely the estimated confidence matches the expected accuracy over a model's output distribution. In contrast, response calibration evaluates confidence against the correctness of a single sampled response. We further discuss their distinction below.

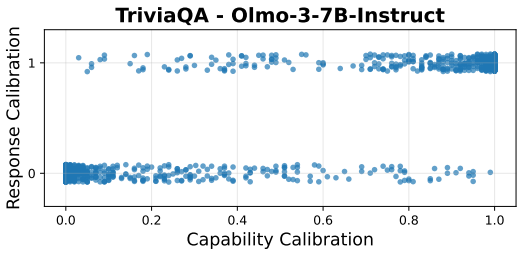

Figure 2: **Divergence of calibration targets.** We estimate the Response Calibration (RC) targets $\mathcal{C}(x, \hat{y})$ and Capability Calibration (CC) targets $\mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x)}[\mathcal{C}(x, \hat{y})]$. Since response stochasticity does not guarantee correctness stochasticity, the experiment result reveals divergence between the two targets: queries where the RC label is 0 exhibit CC values spanning the full [0, 1] range. The same observation for queries with an RC label of 1. This confirms that response-level outcomes do not reflect the model's true ability to answer a query.

3.2 Difference between response calibration and capability calibration

We argue that these two evaluations diverge in three key aspects:

- **Evaluation targets**: Response calibration targets correctness of a model's output $\mathcal{C}(x, \hat{y})$, whereas capability calibration targets the expected accuracy over all possible outputs $\mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x)}[\mathcal{C}(x, \hat{y})]$.
- **Conditional sets**: Response calibration is conditioned on both input and output $(x, \hat{y})$, estimating $\mathbb{P}(\text{Correct} \mid x, \hat{y})$. Capability calibration is conditioned on the input and the model's sampling distribution (x, f_θ), estimating $\mathbb{P}(\text{Correct} \mid x, f_\theta)$.
- **Optimal confidence**: Consequently, their optimal confidence scores differ, as proved in Appendix A.1. Optimal response-calibrated confidence is a binary indicator of a response's correctness, while optimal capability-calibrated confidence is a continuous probability representing capability. This theoretical divergence is empirically confirmed in Figure 2 and Section 5.1.

Having established that these objectives are distinct, we now formalize the connection between the response calibration loss function and the capability calibration loss function:

Theorem 1. ***(Decomposition of calibration losses).*** *Given a set of estimated confidence* $\mathbf{s}$*, define the expectation of response calibration loss* $\mathcal{L}_{\text{Brier}}^{\text{response}}$ *on the model's output distribution as*

$$\mathbb{E}\left[\mathcal{L}_{\text{Brier}}^{\text{response}}\right] \triangleq \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x_i)}\left[\left(s_i - \mathcal{C}(x_i, \hat{y})\right)^2\right].$$

Decoupling output correctness variance from response calibration, we get capability calibration:

$$\mathcal{L}_{\text{Brier}}^{\text{capability}} = \underbrace{\mathbb{E}[\mathcal{L}_{\text{Brier}}^{\text{response}}]}_{\textit{response calibration}} - \underbrace{\frac{1}{N}\sum_{i=1}^{N} Var(\mathcal{C}(x_i, \hat{y}))}_{\textit{output correctness variance}}, \quad (4)$$

where

$$Var(\mathcal{C}(x_i, \hat{y})) = \mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x_i)}[\mathcal{C}(x_i, \hat{y})^2] - \mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x_i)}[\mathcal{C}(x_i, \hat{y})]^2. \quad (5)$$

See Appendix A.2 for the proof. Theorem 1 demonstrates that when evaluating a set confidence estimation, capability calibration decouples the *model's output variance* from response calibration. While response calibration penalizes stochasticity of generated outputs, capability calibration targets the model's correctness probability. For strictly convex and differentiable losses, the difference $\mathbb{E}[\mathcal{L}^{response}] - \mathcal{L}^{capability}$ generalizes to Bregman information (Banerjee et al., 2005), which quantifies the gap (Gruber & Buettner, 2023) caused by output randomness (details in Appendix A.3).

4 Applications of capability calibration

We demonstrate that capability calibration benefits several downstream applications where response calibration is not optimal, either because output variance degrades its performance or because it requires full decoding before confidence can be estimated, creating an efficiency bottleneck.

4.1 Pass@k estimation

Test-time scaling via repeated sampling has been shown to enhance LLM capabilities (Brown et al., 2024), while simultaneously increasing computational burden (Schaeffer et al., 2025; Kazdan et al., 2025). Given these trade-offs, the ability to estimate resampling performance at a low inference cost is critical (Stroebl et al., 2024). A common approach to this problem, as proposed by Kazdan et al. (2025), is to predict pass@k performance by sampling only a small subset of outputs. Their method assumes that the expected accuracy of each query in a dataset follows a beta distribution.

We show that capability-calibrated confidence estimation can simulate the pass@k performance of each query without (1) sampling multiple outputs and (2) assuming a prior distribution over the dataset. Furthermore, by computing the pass@k success rate for each query, we can estimate the pass@k curve for the entire dataset. We discuss the simulation process in Appendix D.1.1.

We evaluate three confidence estimators: (1) Oracle Response-Calibrated (Oracle-RC), where confidence is the RC ground truth, (2) Oracle Capability-Calibrated (Oracle-CC), where confidence is the CC ground truth, and (3) Probe (Li et al., 2021), a common confidence estimator trained with CC target, which we will discuss in Section 5.2. We compute the real pass@k via the unbiased estimator of Chen et al. (2021) and use MSE to measure the query-level discrepancy between simulated and actual pass@k. Table 2 reports results on MATH-500; additional results are in Appendix D.1.2.

The results confirm that CC is a more suitable target than RC in pass@k estimation. Oracle-CC is expected accuracy (see Equation (2)), so it simulates ground-truth performance perfectly. In contrast, Oracle-RC focuses on single-response correctness, which is a noisy estimate of expected accuracy, causing MSE to increase at higher k. This underscores the fundamental limitations of response calibration: even a perfectly calibrated response-level estimator cannot capture pass@k performance. Notably, **non-oracle, CC target trained Probes could even outperform Oracle-RC** in most cases.

4.2 Inference budget allocation

Inference-time scaling has been shown to improve language model performance (Zhang et al., 2025c; Snell et al., 2025). In real-world scenarios, however, LLM providers often face budget allocation problems (Ong et al., 2025): maximizing total accuracy across a stream of queries while constrained by limited GPU hours or API costs. Damani et al. (2025) turns this challenge into the proxy:

In the best-of-k setting, how to solve as many problems as possible under a fixed sampling budget?

To tackle this problem, Damani et al. (2025) proposed an approach that involves distributing the total computational budget across a dataset of queries prior to generating answers. This strategy outperforms uniform allocation by learning a reward model to estimate the marginal "gain": the expected improvement in success rate per unit of additional compute. By prioritizing queries where this gain is highest, the system can strategically distribute resources to maximize overall performance.

The "gain" metric defined by Damani et al. (2025) relies directly on expected accuracy in Equation (2). Consequently, capability-calibrated confidence allows us to analytically estimate this gain and apply the greedy allocation algorithm detailed in Appendix D.2.1 for inference budget allocation. Besides the confidence estimators we compared in Section 4.1, we also test Verbalized Confidence (Verbalized), an estimator that is applicable to black-box models, which we discuss in Section 5.2.

Table 2: **Pass@k simulation error (MSE) on the MATH-500 dataset.** We evaluate the ability of different confidence estimations to simulate empirical pass@k performance. Perfectly capability-calibrated confidence (Oracle-CC) achieves near-perfect simulation, whereas the error of perfectly response-calibrated confidence (Oracle-RC) increases as k scales. Notably, the trained estimator (Probe) outperforms the Oracle-RC baseline by approximating the model's expected accuracy.

Method	pass@1	pass@4	pass@16	pass@64
Olmo-3-7B-Instruct				
Oracle-CC	0.0000	0.0000	0.0000	0.0002
Oracle-RC	0.0349	0.0497	0.0661	0.0804
Probe	0.0373	0.0336	0.0219	0.0148
Qwen3-8B				
Oracle-CC	0.0000	0.0000	0.0000	0.0005
Oracle-RC	0.0542	0.0884	0.1251	0.1536
Probe	0.0471	0.0417	0.0265	0.0134
gpt-oss-20b				
Oracle-CC	0.0000	0.0000	0.0000	0.0002
Oracle-RC	0.0243	0.0337	0.0400	0.0440
Probe	0.0216	0.0102	0.0068	0.0028

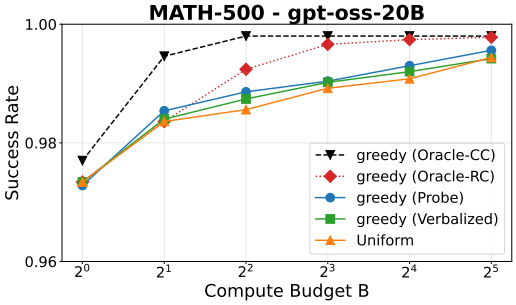

Figure 3: **Inference budget allocation performance of confidence estimations.** Given N questions, we evaluate the performance (success rate) of different methods under the fixed inference budget $N \times B$. The Oracle-CC achieves the best performance and consistently outperforms Oracle-RC. Meanwhile, confidence estimators (verbalized and Probe) both outperform the Uniform allocation in various budgets.

Experimental results validate the effectiveness of capability-calibrated confidence for inference budget allocation. Notably, Oracle-RC is fundamentally ill-suited for this task: since RC targets the confidence of a decoded response, obtaining an RC score consumes the very budget the system is trying to allocate. Figure 3 shows results for gpt-oss-20b on MATH-500; additional results for other models and datasets are provided in Appendix D.2.2. Oracle-CC yields the best performance across all compute budgets, and Probe consistently outperforms uniform allocation. Oracle-RC performs worse than Oracle-CC, confirming that hill-climbing CC is the better objective. Furthermore, verbalized confidence achieves results comparable to Probe without requiring access to internal model states, suggesting that the benefits of capability-calibrated confidence extend to API-based models.

4.3 Discussion on other applications

As an initial investigation, we prioritize two applications (§4.1, §4.2) where performance ties directly to expected accuracy. Beyond these, capability-calibrated confidence could potentially support selective prediction (Kamath et al., 2020), cost estimation (Wu et al., 2025b), curriculum learning (Zhang et al., 2026), and label-free benchmarking (Guha et al., 2024). Capability calibration is also more efficient than response calibration for applications such as model routing (Chen et al., 2024; Ong et al., 2025) and active query refinement (Wu et al., 2024). Appendix D.4 provides a conceptual discussion of these domains and capability calibration's advantages over response calibration.

Implication: choosing the right calibration metric. Capability calibration is preferable when the goal is to assess a model's expected accuracy, or to forecast model capability efficiently before generation. Response calibration, in contrast, is more appropriate for assessing the correctness of a single generated response. Different applications call for different metrics, and capability calibration fills the gap that response calibration cannot address for stochastic models like LLMs. Appendix D.3 further quantifies the results of using the wrong calibration framework in downstream applications.

5 Towards confidence estimation and evaluation with capability calibration

Given capability calibration's utility across diverse applications, we now ask: *how can capability-calibrated confidence be achieved?* We establish an empirical evaluation framework (Section 5.1), introduce confidence estimators commonly used for response calibration and adapt them for capability calibration (Section 5.2), and benchmark their performance (Section 5.3).

5.1 EVALUATION FRAMEWORK

For a given query x, an LLM's theoretical expected accuracy μ defined in Equation (2) is not directly accessible, so one has to estimate μ empirically. For a given x, we estimate the LLM's expected accuracy by sampling k_{eval} responses $\{\hat{y}_1, ..., \hat{y}_{k_{\text{eval}}}\}$ from $f_\theta(\cdot \mid x)$. Let c denote the number of correct responses. The estimated expected accuracy is $\hat{\mu} = c/k_{\text{eval}}$.

Target differs from single-response correctness. Since response stochasticity does not guarantee correctness stochasticity, we need to check whether estimated expected accuracy or single-response correctness are empirically different calibration targets. In Figure 2, we compare these two targets using Olmo-3-7B-Instruct on TriviaQA (see Section 5.3.1 for setup). Additional results are in Appendix E. Consistent with findings in Zhang et al. (2025e), the results show that LLM output correctness is rarely binary. This variance confirms that CC targets an objective different from RC.

Evaluation metrics: *Absolute calibration metric.* Since capability calibration targets query-level performance, we require a metric that preserves per-query granularity. Following the discussion in Section 3.1, we use Brier score to measure calibration quality. Given a dataset of N queries, we compute the empirical capability calibration Brier score following Equation (3) as the mean squared error between the confidence estimate s_i and the empirically estimated expected accuracy $\hat{\mu}_i$ for each query x_i. Lower Brier scores indicate better calibration.

Ranking metric. Beyond absolute calibration quality, ranking performance is also important for applications that require query-level discrimination. We report ranking performance in Section C.5.

5.2 METHODS FOR CONFIDENCE ESTIMATION

We next examine the practical feasibility of capability calibration. We first establish a baseline, then adapt confidence estimators commonly used for response calibration to the capability calibration setting and benchmark their performance. To make the training-free methods "verbalized confidence" and "P(True)" more competitive, we also augment them with isotonic regression on the training sets.

Uniform random baseline. To assess whether a method delivers meaningful performance, we establish a baseline that uses no information about the query. For each query x_i, we sample a confidence score from a uniform distribution $s_i \sim U(0, 1)$. This baseline admits an analytic expected loss as follows, where $\bar{\mu} = \frac{1}{N}\sum_i \hat{\mu}_i$ is the dataset's mean accuracy (see Appendix C.2 for derivation):

$$\mathbb{E}_{\mathbf{s}}\left[\mathcal{L}_{\text{Brier}}^{\text{capability}}(\mathbf{s})\right] = \frac{1}{N}\sum_{i=1}^{N}\left(\tfrac{1}{3} - \hat{\mu}_i + \hat{\mu}_i^2\right) = \tfrac{1}{3} - \bar{\mu} + \bar{\mu}^2 + \sigma^2, \tag{6}$$

Any useful confidence estimation method should outperform this baseline.

Response consistency (Wang et al., 2023). A straightforward way to estimate confidence is by measuring the consistency across multiple sampled responses. We sample k_c responses and compute the fraction that agree with the majority prediction. For example, if $k_c = 10$ and 7 responses are equivalent, the confidence estimate would be 0.7. Note that this method incurs a higher computational cost than other methods, as it requires sampling k_c full responses per query.

Verbalized confidence (Lin et al., 2022). We instruct the LLM to report a probability in $[0, 1]$ in natural language. Unlike prior work Lin et al. (2022); Tian et al. (2023), which asks for confidence in the *response* given the query, we ask for confidence in the *query* itself to measure query-level capability. See Appendix C.2 for the prompt.

P(True) (Kadavath et al., 2022). We ask the model if it can answer the *query* correctly by instructing it to respond with only "Yes" or "No". We extract the logprobs of these two tokens and use the softmax probability of "Yes" as the confidence estimate. Unlike prior work (Kadavath et al., 2022), we provide only the *query*, not with the *response*. We provide the prompt in Appendix C.2.

Probing LLMs' hidden states (Li et al., 2021). We train linear probes Li et al. (2021) on LLMs' internal representations to predict query-level confidence. We mean-pool activations from the last input token across transformer blocks to output a confidence score. This approach incurs low overhead, with an inference cost less than decoding a single token. See Appendix C.2 for details.

Cost. When measuring the cost of confidence estimators, we focus on *online inference cost*. In LLM deployments, preprocessing or training is typically a one-time offline expense, whereas inference cost

Table 3: **Capability calibration performance of existing confidence estimators with three LLMs on seven datasets.** We use **bold** to denote the **best** calibrated method, and underline to denote the second best. While the calibration of verbalized confidence and $P(\text{True})$ varies across models, applying isotonic regression yields improvements. Verbalized confidence achieves the best performance for gpt-oss-20b, while Probe is generally the most effective for the other two models.

Brier score (↓)	Domain	Factual knowledge		Mathematical reasoning			General exams	
Method	Cost	TriviaQA	SimpleQA	GSM8K	MATH	AIME	MMLU	GPQA
Olmo-3-7B-Instruct								
Uniform random baseline	N/A	0.2745	0.3133	0.3119	0.2940	0.2462	0.2565	0.2125
Verbalized confidence	L	0.2717	0.2741	0.0444	0.0632	0.2126	0.1556	0.2672
+ isotonic regression	L	0.1840	0.0138	0.0397	0.0501	0.1544	0.1200	0.1205
P(True)	1	0.1698	0.0416	0.1266	0.1392	0.1791	0.2446	0.1520
+ isotonic regression	1	0.1666	0.0149	0.0389	0.0523	0.1761	0.1228	0.1222
Probe	< 1	**0.1101**	**0.0130**	**0.0364**	**0.0373**	**0.1092**	**0.1004**	**0.1072**
Qwen3-8B								
Uniform random baseline	N/A	0.2865	0.3109	0.3144	0.2781	0.2800	0.2868	0.2113
Verbalized confidence	L	0.2459	0.4673	0.0455	0.0977	0.4093	0.1263	0.2719
+ isotonic regression	L	0.1661	**0.0271**	0.0405	0.0822	0.1327	0.1085	0.1351
P(True)	1	0.3253	0.6010	0.0483	0.1124	0.4728	0.1548	0.3422
+ isotonic regression	1	0.1476	0.0277	0.0391	0.0739	0.0904	0.1033	**0.1243**
Probe	< 1	**0.1163**	0.0306	**0.0377**	**0.0471**	**0.0578**	**0.0987**	0.1571
gpt-oss-20b								
Uniform random baseline	N/A	0.2639	0.3010	0.3195	0.3063	0.2369	0.3018	0.2388
Verbalized confidence	L	0.1204	0.1885	0.0247	**0.0186**	0.0581	0.0802	0.1066
+ isotonic regression	L	**0.0813**	**0.0216**	**0.0245**	0.0216	**0.0545**	**0.0752**	**0.0898**
P(True)	L	0.2435	0.5998	0.0299	0.0219	0.1446	0.1220	0.1933
+ isotonic regression	L	0.1417	0.0238	0.0259	0.0196	0.0852	0.0895	0.1183
Probe	< 1	0.1027	0.0264	0.0291	0.0216	0.0966	0.0878	0.1462

and latency accumulate continuously during real-world use. Thus, inference efficiency is the primary practical consideration. We provide a discussion of confidence estimation cost in Appendix C.8.

5.3 Experiments

5.3.1 Setup

Choice of k_{eval}. The estimated expected accuracy $\hat{\mu}$ is binomial with variance $\mu(1-\mu)/k_{\text{eval}}$, which decreases as k_{eval} increases. We investigate the effect of k_{eval} on evaluation reliability in Appendix B.1 and B.3. We set k_{eval} to 100 to balance cost and reliability. We also report $k_{\text{eval}} > 100$ results and error bars in Appendix B.3.

Models. We use Olmo-3-7B-Instruct Olmo Team et al. (2025), Qwen3-8B Yang et al. (2025), and gpt-oss-20b Agarwal et al. (2025) to ensure diversity. We adopt developer-recommended sampling hyperparameters (Appendix B.4). We also evaluate different decoding temperatures in Appendix B.5.

Datasets. We select datasets from three task domains: (1) *factual knowledge*, which tests parametric knowledge; (2) *mathematical reasoning*, where errors compound across multiple intermediate steps; and (3) *general exams*, which test both knowledge and reasoning in multiple subjects. For each type, we include datasets of different difficulty levels based on accuracy: ***Factual knowledge:*** We use TriviaQA Joshi et al. (2017) as the easy and SimpleQA verified Haas et al. (2025) as the hard one. ***Mathematical reasoning:*** We adopt GSM8K Cobbe et al. (2021) as the easiest dataset, MATH-500 Lightman et al. (2024) as the intermediate one, and AIME (questions from 2024 to 2026) as the hardest. ***General exams:*** We use MMLU Hendrycks et al. (2021a), which spans 57 subjects. We use GPQA Rein et al. (2024) as the harder one. We report train-test split details in Appendix C.2.

5.3.2 Results and discussion

Probe has the best online cost-performance tradeoff. A practically useful method should satisfy two properties: (1) *Acceptable inference cost*: no higher than decoding the response itself, otherwise the overhead would limit the method's practical utility (see §4). (2) *Good calibration performance*: at least better than the uniform random baseline, with lower Brier scores preferred. Figure 4 shows a

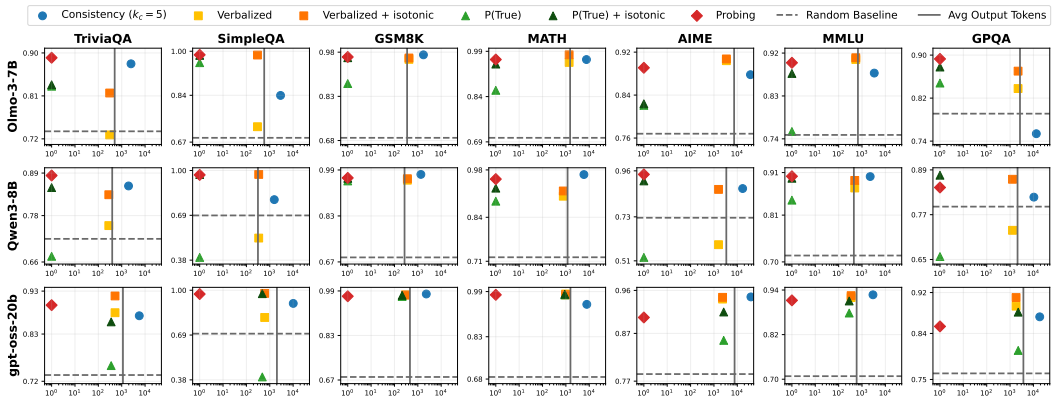

Figure 5: **Cost-performance tradeoff of different confidence estimation methods with three LLMs on seven datasets.** Following Figure 4, we compare inference cost (x-axis, average response tokens) against calibration performance (y-axis). Probes consistently outperform the random baseline with the lowest cost, while response consistency incurs a cost higher than decoding responses.

representative example, with full results shown in Figure 5. Among evaluated methods, Probe has the lowest inference cost while consistently outperforming the random baseline. We further analyze the effects of training data for probes in Appendix C.4.

Performance of verbalized confidence and P(True) differs across LLMs. As shown in Table 3, gpt-oss-20b performs strikingly well with verbalized confidence, achieving the best or second-best performance across datasets. In contrast, Qwen3-8B does not even consistently outperform the random uniform baseline with verbalized confidence. Moreover, verbalized confidence outperforms P(True) for Qwen3-8B and gpt-oss-20b, but not for Olmo-3-7B-Instruct. These results suggest that the effectiveness of these training-free methods is model-dependent. Isotonic regression enhances the performance of two methods by recalibrating confidence estimates based on training labels, which makes these methods become training-based. We also explored enhancing verbalized confidence through few-shot prompting, as detailed in Appendix C.7.

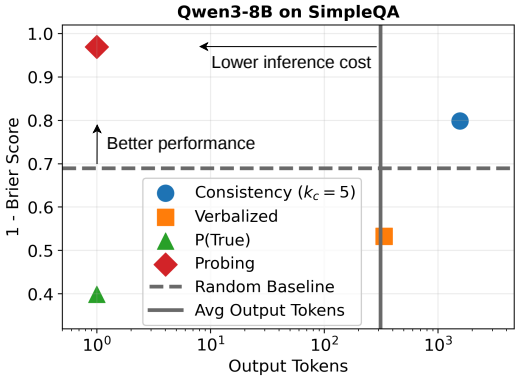

Figure 4: **Cost-performance tradeoff of different methods.** We compare inference cost (x-axis, log-scale) against calibration performance (y-axis, 1 - Brier score), where the upper-left corner is the ideal region. See Figure 5 for full results.

Response consistency costs more than responding. This method costs more than decoding the response itself (Figures 4 and 5), so it is impractical when confidence must be estimated efficiently, such as in inference budget allocation (§4.2). We report its performance in Appendix C.3.

6 Discussion

This work formalizes capability calibration and shows that it differs from existing response calibration due to the stochastic nature of LLM outputs. We demonstrate its superiority in downstream applications, including pass@k prediction and inference budget allocation. We also confirm that capability-calibrated confidence estimations are practically achievable.

Our study establishes a new calibration evaluation framework for confidence estimation rather than proposing new estimation methods; we envision future work developing confidence estimators that push the frontier of capability calibration, and extending the framework to applications such as model routing, human–AI collaboration, and trustworthy AI.

AI USE STATEMENT

In this work, we used generative AI tools for generating experiment code (designing or providing feedback on experiments, implementing methods). We have not used generative AI tools to develop the main idea, the conceptual framework, the literature survey, support qualitative and thematic data analysis, design or provide feedback on research methodology, help develop theoretical models, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, and interpret results. Generating synthetic data sets, proposing or refining hypotheses, assisting with translation, and cleaning and reformatting datasets are not applicable to this work. Additionally, we used generative AI tools to make the sentences more fluent and for grammar checking. We have reviewed all AI-assisted work. LLM-generated code was verified and tested for correctness by 2 authors. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This paper presents work whose goal is to advance the field of Machine Learning by improving the reliability and predictability of LLMs. As LLMs are increasingly deployed in real-world applications, ensuring they are trustworthy is paramount. Our framework for capability calibration enables models to more accurately assess their own limitations, allowing systems to abstain or seek human oversight when the model is unlikely to succeed. We believe this contributes to safer AI deployment by mitigating the risks associated with overconfidence and hallucination.

REPRODUCIBILITY STATEMENT

Experiment details are clearly described in the main paper and Appendix B to reproduce the empirical results. We also have provided the source code in the Supplementary Material of the submission form. The datasets used are all publicly available.

REFERENCES

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

Manar Aljohani, Jun Hou, Sindhura Kommu, and Xuan Wang. A comprehensive survey on the trustworthiness of large language models in healthcare. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, 2025.

Arindam Banerjee, Srujana Merugu, Inderjit S Dhillon, and Joydeep Ghosh. Clustering with bregman divergences. *Journal of machine learning research*, 2005.

Glenn W. Brier. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3, 1950.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

Nicola Cecere, Andrea Bacciu, Ignacio Fernández-Tobías, and Amin Mantrach. Monte Carlo temperature: a robust sampling strategy for LLM's uncertainty quantification methods. In *Proceedings of the 5th Workshop on Trustworthy NLP (TrustNLP 2025)*, 2025.

Iván Vicente Moreno Cencerrado, Arnau Padrés Masdemont, Anton Gonzalvez Hawthorne, David Demitri Africa, and Lorenzo Pacchiardi. No answer needed: Predicting llm answer accuracy from question-only linear probes. *arXiv preprint arXiv:2509.10625*, 2025.

Jiefeng Chen, Jinsung Yoon, Sayna Ebrahimi, Sercan Arik, Tomas Pfister, and Somesh Jha. Adaptation with self-evaluation to improve selective prediction in llms. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2023.

Lihu Chen, Gerard de Melo, Fabian M. Suchanek, and Gaël Varoquaux. Query-level uncertainty in large language models. In *International Conference on Learning Representations*, 2026.

Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*, 2024.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

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

Roi Cohen, Konstantin Dobler, Eden Biran, and Gerard de Melo. I don't know: Explicit modeling of uncertainty with an [idk] token. *Advances in Neural Information Processing Systems*, 2024.

Mehul Damani, Idan Shenfeld, Andi Peng, Andreea Bobu, and Jacob Andreas. Learning how hard to think: Input-adaptive allocation of LM computation. In *International Conference on Learning Representations*, 2025.

Mehul Damani, Isha Puri, Stewart Slocum, Idan Shenfeld, Leshem Choshen, Yoon Kim, and Jacob Andreas. Beyond binary rewards: Training LMs to reason about their uncertainty. In *International Conference on Learning Representations*, 2026.

Jinhao Duan, Hao Cheng, Shiqi Wang, Alex Zavalny, Chenan Wang, Renjing Xu, Bhavya Kailkhura, and Kaidi Xu. Shifting attention to relevance: Towards the predictive uncertainty quantification of free-form large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. *Advances in Neural Information Processing Systems*, 2017.

Jiahui Geng, Fengyu Cai, Yuxia Wang, Heinz Koeppl, Preslav Nakov, and Iryna Gurevych. A survey of confidence estimation and calibration in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 2024.

Sebastian Gruber and Florian Buettner. Uncertainty estimates of predictions via a general bias-variance decomposition. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, 2023.

Neel Guha, Mayee Chen, Trevor Chow, Ishan Khare, and Christopher Re. Smoothie: Label free language model routing. *Advances in Neural Information Processing Systems*, 2024.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, 2017.

Lukas Haas, Gal Yona, Giovanni D'Antonio, Sasha Goldshtein, and Dipanjan Das. Simpleqa verified: A reliable factuality benchmark to measure parametric knowledge. *arXiv preprint arXiv:2509.07968*, 2025.

Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations*, 2017.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021a.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021b.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020.

Hsiu-Yuan Huang, Yutong Yang, Zhaoxi Zhang, Sanwoo Lee, and Yunfang Wu. A survey of uncertainty estimation in llms: Theory meets practice. *arXiv preprint arXiv:2410.15326*, 2024a.

Yue Huang, Lichao Sun, Haoran Wang, Siyuan Wu, Qihui Zhang, Yuan Li, Chujie Gao, Yixin Huang, Wenhan Lyu, Yixuan Zhang, Xiner Li, Hanchi Sun, Zhengliang Liu, Yixin Liu, Yijue Wang, Zhikun Zhang, Bertie Vidgen, Bhavya Kailkhura, Caiming Xiong, Chaowei Xiao, Chunyuan Li, Eric P. Xing, Furong Huang, Hao Liu, Heng Ji, Hongyi Wang, Huan Zhang, Huaxiu Yao, Manolis Kellis, Marinka Zitnik, Meng Jiang, Mohit Bansal, James Zou, Jian Pei, Jian Liu, Jianfeng Gao, Jiawei Han, Jieyu Zhao, Jiliang Tang, Jindong Wang, Joaquin Vanschoren, John Mitchell, Kai Shu, Kaidi Xu, Kai-Wei Chang, Lifang He, Lifu Huang, Michael Backes, Neil Zhenqiang Gong, Philip S. Yu, Pin-Yu Chen, Quanquan Gu, Ran Xu, Rex Ying, Shuiwang Ji, Suman Jana, Tianlong Chen, Tianming Liu, Tianyi Zhou, William Yang Wang, Xiang Li, Xiangliang Zhang, Xiao Wang, Xing Xie, Xun Chen, Xuyu Wang, Yan Liu, Yanfang Ye, Yinzhi Cao, Yong Chen, and Yue Zhao. Position: TrustLLM: Trustworthiness in large language models. In *International Conference on Machine Learning*, 2024b.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. LLM-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2023.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017.

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

Amita Kamath, Robin Jia, and Percy Liang. Selective question answering under domain shift. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

Sungmin Kang, Yavuz Faruk Bakman, Duygu Nur Yaldiz, Baturalp Buyukates, and Salman Avestimehr. Uncertainty quantification for hallucination detection in large language models: Foundations, methodology, and future directions. *arXiv preprint arXiv:2510.12040*, 2025.

Joshua Kazdan, Rylan Schaeffer, Youssef Allouah, Colin Sullivan, Kyssen Yu, Noam Levi, and Sanmi Koyejo. Efficient prediction of pass@ k scaling in large language models. *arXiv preprint arXiv:2510.05197*, 2025.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *International Conference on Learning Representations*, 2023.

Belinda Z. Li, Maxwell Nye, and Jacob Andreas. Implicit representations of meaning in neural language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021.

Lujun Li, Lama Sleem, Niccolo' Gentile, Geoffrey Nichil, and Radu State. Exploring the impact of temperature on large language models: Hot or cold? *Procedia Comput. Sci.*, 2025a.

Yibo Li, Miao Xiong, Jiaying Wu, and Bryan Hooi. Conftuner: Training large language models to express their confidence verbally. In *Advances in Neural Information Processing Systems*, 2025b.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024.

Stephanie Lin, Jacob Hilton, and Owain Evans. Teaching models to express their uncertainty in words. *Transactions on Machine Learning Research*, 2022.

Potsawee Manakul, Adian Liusie, and Mark Gales. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. In *Conference on Empirical Methods in Natural Language Processing*, 2023.

Yuzhen Mao, Thibaut Durand, Nazanin Mehrasa, Jiawei He, and Martin Ester. Calibrating llms for selective prediction: Balancing coverage and risk. In *Socially Responsible and Trustworthy Foundation Models at NeurIPS 2025*, 2025.

Kaushal Kumar Maurya, KV Aditya Srivatsa, and Ekaterina Kochmar. Selectllm: Query-aware efficient selection algorithm for large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, 2025.

Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In *International Conference on Machine Learning*, 2005.

Olmo Team, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, et al. Olmo 3. *arXiv preprint arXiv:2512.13961*, 2025.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. RouteLLM: Learning to route LLMs from preference data. In *International Conference on Learning Representations*, 2025.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 2022.

Mark D Reid and Robert C Williamson. Information, divergence and risk for binary experiments. *Journal of Machine Learning Research*, 2011.

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

Matthew Renze. The effect of sampling temperature on problem solving in large language models. In *Findings of the association for computational linguistics: EMNLP 2024*, 2024.

Rylan Schaeffer, Joshua Kazdan, John Hughes, Jordan Juravsky, Sara Price, Aengus Lynch, Erik Jones, Robert Kirk, Azalia Mirhoseini, and Sanmi Koyejo. How do large language monkeys get their power (Laws)? In *International Conference on Machine Learning*, 2025.

Wout Schellaert, Fernando Martínez-Plumed, and José Hernández-Orallo. Analysing the predictability of language model performance. *ACM Transactions on Intelligent Systems and Technology*, 16(2): 1–26, 2025.

Chufan Shi, Haoran Yang, Deng Cai, Zhisong Zhang, Yifan Wang, Yujiu Yang, and Wai Lam. A thorough examination of decoding methods in the era of LLMs. In *Conference on Empirical Methods in Natural Language Processing*, 2024.

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

Benedikt Stroebl, Sayash Kapoor, and Arvind Narayanan. Inference scaling flaws: The limits of llm resampling with imperfect verifiers. *arXiv preprint arXiv:2411.17501*, 2024.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher Manning. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *Conference on Empirical Methods in Natural Language Processing*, 2023.

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

Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. Measuring short-form factuality in large language models. *arXiv preprint arXiv:2411.04368*, 2024.

Cheng-Kuang Wu, Zhi Rui Tam, Chao-Chung Wu, Chieh-Yen Lin, Hung-yi Lee, and Yun-Nung Chen. I need help! evaluating LLM's ability to ask for users' support: A case study on text-to-SQL generation. In *Conference on Empirical Methods in Natural Language Processing*, 2024.

Jiayun Wu, Jiashuo Liu, Zhiyuan Zeng, Tianyang Zhan, Tianle Cai, and Wenhao Huang. Mitigating llm hallucination via behaviorally calibrated reinforcement learning. *arXiv preprint arXiv:2512.19920*, 2025a.

Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. Inference scaling laws: An empirical analysis of compute-optimal inference for LLM problem-solving. In *International Conference on Learning Representations*, 2025b.

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

Anqi Zhang, Yulin Chen, Jane Pan, Chen Zhao, Aurojit Panda, Jinyang Li, and He He. Reasoning models know when they're right: Probing hidden states for self-verification. In *Second Conference on Language Modeling*, 2025a.

Jiamu Zhang, Jiayi Yuan, Andrew Wen, Hoang Anh Duy Le, Yu-Neng Chuang, Soo-Hyun Choi, Rui Chen, and Xia Hu. Reasonerrank: Redefining language model evaluation with ground-truth-free ranking frameworks. In *Findings of the Association for Computational Linguistics: ACL 2025*, 2025b.

Kexun Zhang, Shang Zhou, Danqing Wang, William Yang Wang, and Lei Li. Scaling LLM inference efficiently with optimized sample compute allocation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 2025c.

Shijie Zhang, Guohao Sun, Kevin Zhang, Xiang Guo, and Rujun Guo. Clpo: Curriculum learning meets policy optimization for llm reasoning. *arXiv preprint arXiv:2509.25004*, 2025d.

Wenbo Zhang, Hengrui Cai, and Wenyu Chen. Beyond the singular: The essential role of multiple generations in effective benchmark evaluation and analysis. *arXiv preprint arXiv:2502.08943*, 2025e.

Yang Zhang, Amr Mohamed, Hadi Abdine, Guokan Shang, and Michalis Vazirgiannis. Beyond random sampling: Efficient language model pretraining via curriculum learning. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2026.

Lexin Zhou, Fernando Martínez-Plumed, José Hernández-Orallo, Cèsar Ferri, and Wout Schellaert. Reject before you run: Small assessors anticipate big language models. *EBeM@ IJCAI*, 2022.

APPENDIX

Table of Contents

A Details of Capability Calibration **15**
- A.1 Optimal confidence differs under two calibration definitions . . . 15
- A.2 Proof of Theorem 1 . . . 16
- A.3 Connection between response calibration loss and capability calibration loss . . . 17

B Details of experiment setup **18**
- B.1 The choice of k_{eval} . . . 18
- B.2 Correcting Brier score for the measurement of expected accuracy . . . 19
- B.3 Stability of experiment findings under different k_{eval} . . . 20
- B.4 Sampling hyperparameters for each LLM. . . . 20
- B.5 Transferability of capability calibration across decoding temperatures . . . 20
- B.6 LLMs' mean expected accuracies across datasets . . . 22
- B.7 Compute resources for estimating expected accuracy . . . 22

C Details of confidence estimation methods **22**
- C.1 Detailed descriptions of existing methods . . . 22
- C.2 Implementation details of confidence estimators . . . 22
- C.3 Analysis of k_c for Response Consistency method . . . 25
- C.4 Probe performance in out-of-distribution data . . . 25
- C.5 Evaluating confidence estimators in ranking metric . . . 27
- C.6 Evaluating confidence estimators in ECE . . . 28
- C.7 Improving verbalized confidence . . . 30
- C.8 Cost of confidence estimators . . . 30

D Details of the applications **30**
- D.1 Pass@k simulation details . . . 30
- D.2 Inference budget allocation details . . . 33
- D.3 Estimator selection and downstream impact under RC and CC . . . 35
- D.4 Other potential applications . . . 36

E Targets difference **37**

A DETAILS OF CAPABILITY CALIBRATION

A.1 OPTIMAL CONFIDENCE DIFFERS UNDER TWO CALIBRATION DEFINITIONS

In this section, we show that the optimal confidence of response calibration is different from the optimal confidence capability calibration.

Theorem 2. ***(Divergence of targets and optima).*** *Let x be an input and $\hat{y} \sim f_\theta(\cdot \mid x)$ be a generated response. Minimizing the Brier scores for response calibration (Equation* (1)*) and capability calibration (Equation* (3)*) yields distinct optimal confidence estimators:*

$$\begin{aligned} s^*_{\text{resp}}(x, \hat{y}) &= \mathcal{C}(x, \hat{y}) \in \{0, 1\}, \\ s^*_{\text{cap}}(x, f_\theta) &= \mathbb{E}_{\hat{y} \sim f_\theta(\cdot|x)}[\mathcal{C}(x, \hat{y})] \in [0, 1]. \end{aligned} \tag{7}$$

Unless the model is deterministic, or its predictions are always correct or always incorrect, the evaluation targets differ, i.e., $\mathcal{C}(x, \hat{y}) \neq \mathbb{E}_{\hat{y} \sim f_\theta(\cdot|x)}[\mathcal{C}(x, \hat{y})]$, implying distinct optimal confidence values.

Proof. Let $f_\theta(\cdot \mid x)$ be a generative model. For each input x_i, let $\hat{y}_i \sim f_\theta(\cdot \mid x_i)$ be a single sampled output, and let $s_i \in [0,1]$ denote the predicted confidence.

Recall that the response calibration Brier score (Equation (1)) is defined as:

$$\mathcal{L}_{\text{Brier}}^{\text{response}}(s_1,\ldots,s_N) \triangleq \frac{1}{N}\sum_{i=1}^{N}\left(s_i - \mathcal{C}(x_i,\hat{y}_i)\right)^2, \quad \hat{y}_i \sim f_\theta(\cdot \mid x_i).$$

Define the confidence estimations of the dataset as $\mathbf{s} = (s_1, s_2, ..., s_n)$. Since the Brier score is convex, the stationary point is the global minimum, which is

$$\nabla \mathcal{L}_{\text{Brier}}^{\text{response}}(\mathbf{s}^*) = \mathbf{0},$$

i.e.,

$$\frac{\partial \mathcal{L}_{\text{Brier}}^{\text{response}}}{\partial s_i} = \frac{2}{N}(s_i - \mathcal{C}(x_i,\hat{y}_i)) = 0, \quad \forall i \in \{1,\ldots,N\}.$$

Thus, the optimal confidence estimation is

$$s_i^{*,\text{response}} = \mathcal{C}(x_i,\hat{y}_i), \quad \forall i \in \{1,\ldots,N\}.$$

Follow the definition of Equation (2)

$$\mu_i \triangleq \mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x_i)}[\mathcal{C}(x_i,\hat{y})],$$

we restate the capability calibration Brier score from Equation (3):

$$\mathcal{L}_{\text{Brier}}^{\text{capability}}(s_1,\ldots,s_N) \triangleq \frac{1}{N}\sum_{i=1}^{N}(s_i-\mu_i)^2 = \frac{1}{N}\sum_{i=1}^{N}\left(s_i - \mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x_i)}[\mathcal{C}(x_i,\hat{y})]\right)^2.$$

Following the same proof, the optimal confidence that minimize $\mathcal{L}_{\text{Brier}}^{\text{capability}}$ satisfies

$$s_i^{*,\text{capability}} = \mu_i, \quad \forall i \in \{1,\ldots,N\}.$$

For a generative model, $\mathcal{C}(x_i,\hat{y}_i) \in \{0,1\}$, while $\mu_i \in [0,1]$. Unless the model is deterministic or perfectly correct/incorrect on x_i, we have

$$\mathcal{C}(x_i,\hat{y}_i) \neq \mu_i \triangleq \mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x_i)}[\mathcal{C}(x_i,\hat{y})].$$

Therefore,

$$(s_1^{*,\text{response}},\ldots,s_N^{*,\text{response}}) \neq (s_1^{*,\text{capability}},\ldots,s_N^{*,\text{capability}}),$$

i.e., the two calibration objectives induce different optimal confidence predictors over the dataset. □

A.2 Proof of Theorem 1

Proof. First, expand the expectation of the Brier Score loss on the old calibration

$$\begin{aligned}\mathbb{E}\big[\mathcal{L}_{\text{Brier}}\big] &\triangleq \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x_i)}\left[\left(s_i - \mathcal{C}(x_i,\hat{y})\right)^2\right] \\ &= \frac{1}{N}\sum_{i=1}^{N}\Big(s_i^2 - 2s_i\cdot\mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x_i)}[\mathcal{C}(x_i,\hat{y})] + \mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x_i)}[\mathcal{C}(x_i,\hat{y})^2]\Big).\end{aligned} \tag{8}$$

Second, expand the Brier Score loss on expected calibration

$$\mathcal{L}_{\text{Brier}}^{\text{expected}} \triangleq \frac{1}{N}\sum_{i=1}^{N}(s_i-\mu_i)^2 = \frac{1}{N}\sum_{i=1}^{N}\left(s_i^2 - 2s_i\,\mu_i + \mu_i^2\right), \tag{9}$$

Subtracting Equation (9) - Equation (8), we obtain

$$\begin{aligned}\mathcal{L}_{\text{Brier}}^{\text{expected}} - \mathbb{E}[\mathcal{L}_{\text{Brier}}] &= \frac{1}{N}\sum_{i=1}^{N}\left(\mu_i^2 - 2s_i \cdot (\mu_i - \mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x_i)}[\mathcal{C}(x_i,\hat{y})]) - \mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x_i)}[\mathcal{C}(x_i,\hat{y})^2]\right) \\ &= \frac{1}{N}\sum_{i=1}^{N}\left(\mu_i^2 - \mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x_i)}[\mathcal{C}(x_i,\hat{y})^2]\right) \\ &= \frac{1}{N}\sum_{i=1}^{N}\left(\mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x_i)}[\mathcal{C}(x_i,\hat{y})]^2 - \mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x_i)}[\mathcal{C}(x_i,\hat{y})^2]\right) \\ &= -\frac{1}{N}\sum_{i=1}^{N}\text{Var}[\mathcal{C}(x_i,\hat{y})].\end{aligned} \tag{10}$$

□

A.3 Connection between response calibration loss and capability calibration loss

For the same loss function $\mathcal{L}(s,t)$ that evaluates the agreement between confidence s and target t, capability calibration and response calibration evaluate s with different t. Denote the loss function as $\mathcal{L}_s(t)$ when we discuss the impact of t.

Capability calibration takes the expected accuracy as the target. In general, capability calibration loss is

$$\mathcal{L}_s(\mathbb{E}_{\hat{y}\sim f(\cdot|x)}[\mathcal{C}(x,\hat{y})]),$$

we simplify it as $\mathcal{L}_s(\mathbb{E}[\mathcal{C}(x,\hat{y})])$.

Response calibration takes the individual response's correctness as the target. Therefore, the expectation of the response calibration loss is

$$\mathbb{E}_{\hat{y}\sim f(\cdot|x)}[\mathcal{L}_s(\mathcal{C}(x,\hat{y}))],$$

we simplify it as $\mathbb{E}[\mathcal{L}_s(\mathcal{C}(x,\hat{y}))]$.

The difference between the capability calibration loss and the response calibration loss

$$\mathbb{E}[\mathcal{L}_s(\mathcal{C}(x,\hat{y}))] - \mathcal{L}_s(\mathbb{E}[\mathcal{C}(x,\hat{y})]), \tag{11}$$

is known as Jensen Gap (Reid & Williamson, 2011). For loss functions that are strictly convex and differentiable, Jensen Gap is equivalent to the Bregman information (Banerjee et al., 2005), which measures the diversity of the random variable $\mathcal{C}(x,\hat{y})$ through the lens of the loss function $\mathcal{L}$. For square error loss, the Bregman information is the variance of $\mathcal{C}(x,\hat{y})$ (Banerjee et al., 2005).

B Details of experiment setup

B.1 The choice of k_{eval}

Theoretical insights. Let $f_\theta(\cdot \mid x)$ denote the LLM's conditional output distribution given a query x, and let $\mathcal{C}(x, \hat{y}) \in \{0, 1\}$ be a deterministic correctness function that indicates whether a sampled response $\hat{y}$ is correct for x. The *expected accuracy* (our capability-calibration target) is

$$\mu(x, f_\theta) \triangleq \mathbb{P}_{\hat{y} \sim f_\theta(\cdot|x)}[\mathcal{C}(x, \hat{y}) = 1].$$

Since $\mu(x, f_\theta)$ is not directly observable, we approximate it by drawing k_{eval} independent samples $\hat{y}_1, \ldots, \hat{y}_{k_{\text{eval}}} \sim f_\theta(\cdot \mid x)$ and computing the empirical mean

$$\hat{\mu} \triangleq \frac{1}{k_{\text{eval}}} \sum_{j=1}^{k_{\text{eval}}} \mathcal{C}(x, \hat{y}_j).$$

For convenience, define the per-sample correctness indicator $Z_j \triangleq \mathcal{C}(x, \hat{y}_j) \in \{0, 1\}$ and the number of correct samples

$$c \triangleq \sum_{j=1}^{k_{\text{eval}}} Z_j, \qquad \text{so that} \qquad \hat{\mu} = \frac{c}{k_{\text{eval}}}.$$

Binomial model and estimation variance. Under independent sampling randomness across repeated generations and a deterministic evaluator $\mathcal{C}$, we have

$$Z_j \mid x \sim \text{Bernoulli}(\mu), \qquad c \sim \text{Binomial}(k_{\text{eval}}, \mu).$$

Consequently, $\hat{\mu}$ is an unbiased estimator of μ with

$$\mathbb{E}[\hat{\mu}] = \mu, \qquad \text{Var}(\hat{\mu}) = \frac{\mu(1-\mu)}{k_{\text{eval}}}. \tag{12}$$

Notably, this binomial structure is induced by the binary correctness indicator and does not assume any particular parametric form for the LLM's raw text distribution.

Sample-size guidance (normal/Wald approximation). Equation equation 12 implies that the standard error of $\hat{\mu}$ decays as $O(k_{\text{eval}}^{-1/2})$. A common rule-of-thumb for selecting k_{eval} is to control the margin of error of $\hat{\mu}$. Using the asymptotic normal (Wald) approximation,

$$\hat{\mu} \approx \mathcal{N}\left(\mu, \frac{\mu(1-\mu)}{k_{\text{eval}}}\right),$$

a two-sided 95% confidence interval has approximate half-width (margin of error, MoE)

$$\text{MoE} \approx z_{0.975} \sqrt{\frac{\mu(1-\mu)}{k_{\text{eval}}}}, \qquad z_{0.975} = 1.96.$$

We interpret ϵ as a target absolute accuracy for per-query estimation: we aim for the 95% confidence interval of $\hat{\mu}$ to have half-width at most ϵ, i.e., $|\hat{\mu} - \mu| \leq \epsilon$ with approximately 95% confidence under the normal approximation. Thus, achieving $\text{MoE} \leq \epsilon$ suggests

$$k_{\text{eval}} \gtrsim \frac{z_{0.975}^2 \, \mu(1-\mu)}{\epsilon^2}. \tag{13}$$

If μ is unknown, the conservative worst case uses $\mu(1-\mu) \leq 1/4$ (attained at $\mu = 0.5$), yielding

$$k_{\text{eval}} \gtrsim \frac{z_{0.975}^2}{4\epsilon^2}. \tag{14}$$

Practical considerations and our choice of k_{eval}. While Equation equation 13–equation 14 are useful for intuition, Wald intervals can under-cover when μ is close to 0 or 1 and/or k_{eval} is small, and may produce bounds outside $[0, 1]$. We therefore use the Wald analysis only as a guideline for the scaling behavior in Equation equation 12, and complement it with a sensitivity analysis

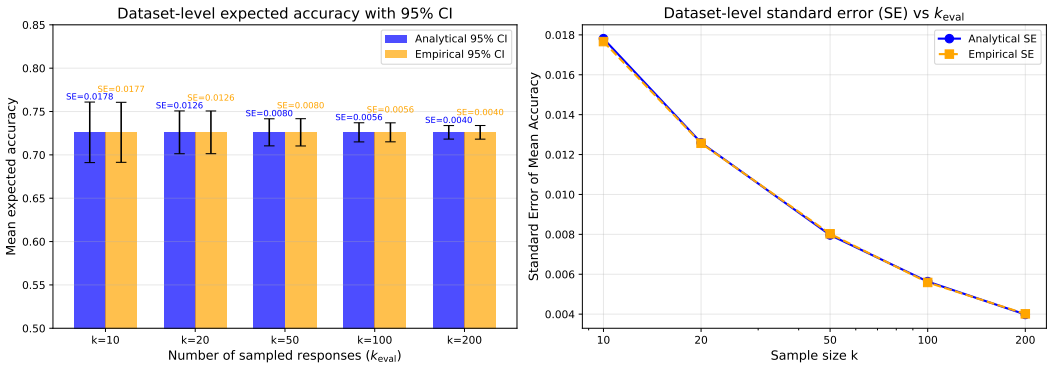

Figure 6: **Representative empirical results showing the standard error of expected accuracies obtained by different k_{eval}.**

over k_{eval}. Prior work (Zhang et al., 2025e) uses $k_{\text{eval}} = 50$ as the ground truth. We find that $k_{\text{eval}} = 100$ provides a stable per-query estimate of μ while keeping evaluation cost tractable; increasing k_{eval} beyond this point yields diminishing returns relative to the additional sampling cost. For completeness, one may alternatively adopt Wilson score intervals (which typically provide closer-to-nominal coverage under the same Bernoulli sampling assumptions) to select k_{eval} via a short numerical search; our empirical validation supports that $k_{\text{eval}} = 100$ is a reliable operating point in our setting.

Empirical results. We show representative empirical results of gpt-oss-20b Agarwal et al. (2025) on AIME25 in Figure 6. We found that the standard error of expected accuracies decreases to about 0.0056 when $k_{\text{eval}} = 100$. Although larger k_{eval} further decreases variance, it shows diminishing benefits. We finally chose $k_{\text{eval}} = 100$ to balance cost and reliability. Another supportive evidence is discussed in Section B.3.

B.2 Correcting Brier score for the measurement of expected accuracy

In this section, we show that correcting Brier scores to obtain an unbiased estimator does not change the ranking of confidence estimators.

From Equation 12 in Appendix B.1, we show that

$$\mathbb{E}[\hat{\mu}] = \mu, \qquad \mathrm{Var}(\hat{\mu}) = \frac{\mu(1-\mu)}{k_{\text{eval}}}.$$

Denote $k_{\text{eval}} = k$. The expected Brier score over the output distribution is

$$\mathbb{E}\left[(s_i - \hat{\mu}_i)^2\right] = (s_i - \mu_i)^2 + \frac{\mu_i(1-\mu_i)}{k},$$

which is a biased estimator of the true Brier loss. To obtain an unbiased estimator, we correct for the measurement noise in the empirical $\hat{\mu}_i$ by subtracting

$$\frac{\hat{\mu}_i(1-\hat{\mu}_i)}{k-1}.$$

Consequently,

$$\begin{aligned}\mathbb{E}\left[(s_i - \hat{\mu}_i)^2 - \frac{\hat{\mu}_i(1-\hat{\mu}_i)}{k-1}\right] &= (s_i - \mu_i)^2 + \frac{\mu_i(1-\mu_i)}{k} - \frac{\mu_i(1-\mu_i)}{k} \\ &= (s_i - \mu_i)^2.\end{aligned}$$

Importantly, the correction term $\frac{\hat{\mu}_i(1-\hat{\mu}_i)}{k-1}$ depends only on the label. Since all methods are evaluated on the same queries and labels, the same term is subtracted from every method. Therefore, the correction changes the absolute Brier scores but leaves method rankings within each model–dataset setting unchanged.

B.3 STABILITY OF EXPERIMENT FINDINGS UNDER DIFFERENT k_{eval}

We conduct additional experiments to examine how k_{eval} affects the stability of the calibration metrics. Using $k_{eval} = 200$ as the reference, we subsample to $k_{eval} = \{10, 20, ..., 190\}$, perform 1000 trials for each value, and measure both Brier score variability and the relative ranking stability of confidence estimation methods. In the resulting plots, solid lines show the mean and shaded regions show the 5th–95th percentile range. Full results are provided in Figure 7.

We find that across most model-dataset pairs, both Brier scores and rankings of methods are already stable at $k_{eval} = 100$, and in many cases even at smaller values. The experiment results show that $k_{eval} = 100$ is sufficient for the scenarios in this paper.

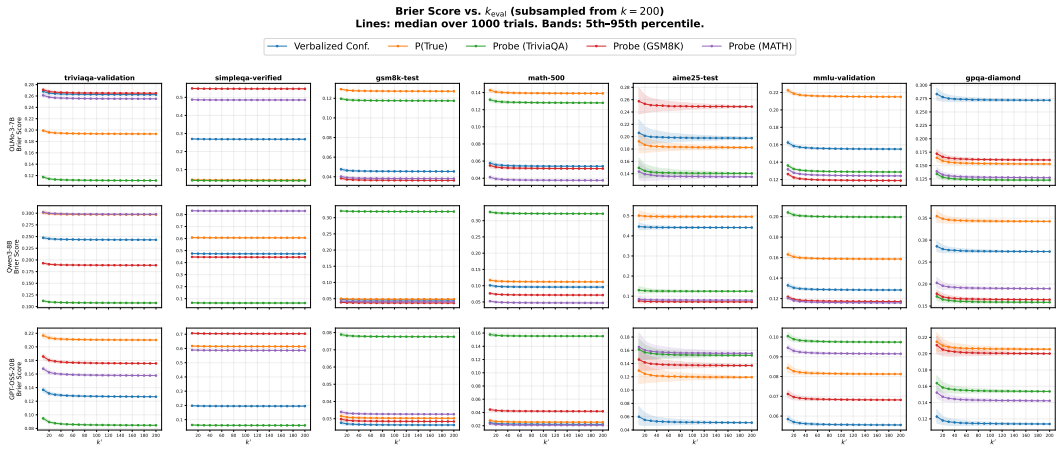

Figure 7: **Brier score of confidence estimators under different k_{eval}.** Probes are trained by training dataset of TriviaQA (Joshi et al., 2017), GSM8K (Cobbe et al., 2021), and MATH (Hendrycks et al., 2021b). Experimental results show that the Brier score is stable at $k_{eval} = 100$ for all scenarios.

B.4 SAMPLING HYPERPARAMETERS FOR EACH LLM.

For each LLM used in the experiments, we use the sampling hyperparameters recommended by its model developers. For Olmo-3-7B-Instruct Olmo Team et al. (2025), we use temperature=0.6 and top-p=0.95. For Qwen3-8B Yang et al. (2025), we use chat template of non-reasoning mode to save inference cost due to limited computational budget, and adopt temperature=0.7 and top-p=0.8. For gpt-oss-20b Agarwal et al. (2025), we use temperature=1.0 and top-p=1.0.

B.5 TRANSFERABILITY OF CAPABILITY CALIBRATION ACROSS DECODING TEMPERATURES

We now analyze the transferability of Capability Calibration across decoding . Across 3 datasets, 3 models, and 3 decoding temperatures, we find that Capability calibration's target, expected accuracy, is stable at the dataset level and broadly robust at the instance level. Meanwhile, The Brier score and the ranking of confidence estimators are likewise stable.

Experiment setup. For three models we benchmarked in the paper, we test two temperatures in addition to each model's recommended temperature. We test one representative dataset per task: TriviaQA, GSM8K, and MMLU.

The stability of capability calibration's target. As capability calibration is an evaluation framework for confidence estimations, we first check how the evaluation target (expected accuracy) varies across different decoding temperatures.

Table 4 shows that at the dataset level, the average expected accuracy is stable across different decoding temperatures. Meanwhile, Table 5 shows strong instance-level rank correlations (Spearman's ρ) for TriviaQA and MMLU. The correlations are lower for GSM8K, but the MSE remains consistently

small across all settings. MSE remains small because many targets are near 0 or 1 (Figure 13), making aggregate errors more stable.

Table 4: Average expected accuracy of three datasets.

Model	Temperature	TriviaQA	GSM8K	MMLU
Olmo-3-7B-Instruct	0.6 / 0.3 / 0.15	0.4884 / 0.4933 / 0.4957	0.9331 / 0.9340 / 0.9348	0.7006 / 0.6989 / 0.6989
Qwen3-8B-non-thinking	0.7 / 0.35 / 0.175	0.6040 / 0.6122 / 0.6140	0.9356 / 0.9355 / 0.9361	0.7940 / 0.7979 / 0.7976
gpt-oss-20b	1.0 / 0.5 / 0.25	0.6507 / 0.6625 / 0.6565	0.9576 / 0.9585 / 0.9566	0.8444 / 0.8429 / 0.8384

Table 5: Instance-level expected accuracy compared with the recommended temperature (Temp.).

Model	Temp. pair	TriviaQA		GSM8K		MMLU	
		Spearman's ρ	MSE	Spearman's ρ	MSE	Spearman's ρ	MSE
Olmo-3-7B-Instruct	$0.6 \leftrightarrow 0.3$	0.9721	0.0048	0.8654	0.0012	0.9763	0.0034
	$0.6 \leftrightarrow 0.15$	0.9522	0.0106	0.8443	0.0026	0.9654	0.0063
Qwen3-8B-non-thinking	$0.7 \leftrightarrow 0.35$	0.9642	0.0052	0.8680	0.0016	0.9596	0.0026
	$0.7 \leftrightarrow 0.175$	0.9348	0.0106	0.8129	0.0033	0.9404	0.0059
gpt-oss-20b	$1.0 \leftrightarrow 0.5$	0.9449	0.0120	0.6734	0.0011	0.8784	0.0035
	$1.0 \leftrightarrow 0.25$	0.9225	0.0208	0.6414	0.0027	0.8530	0.0063

Confidence estimator evaluation. After confirming the stability of the evaluation target, we next ask if the Brier score and the ranking of confidence estimators are stable as well. We test all confidence estimators in Table 3. Table 6 shows that estimator rankings mostly remain unchanged across temperatures, with Brier scores close to those at the recommended temperature (Table 3).

Table 6: **Brier score of the confidence estimator at different temperatures.** Iso denotes isotonic regression.

Olmo-3-7B-Instruct	**Temp. 0.3**			**Temp. 0.15**		
Method	**TriviaQA**	**GSM8K**	**MMLU**	**TriviaQA**	**GSM8K**	**MMLU**
Uniform random baseline	0.2748	0.3143	0.2594	0.2743	0.3205	0.2594
Verbalized confidence	0.2878	0.0483	0.1489	0.2950	0.0480	0.1576
Verbalized confidence+Iso	0.1684	**0.0382**	0.1276	0.1764	**0.0389**	0.1346
P(True)	0.1794	0.1306	0.2272	0.1858	0.1317	0.2306
P(True)+Iso	0.1663	0.0410	0.1283	0.1730	0.0416	0.1336
Probe	**0.1186**	0.0383	**0.1036**	**0.1318**	0.0390	**0.1114**
Qwen3-8B-non-thinking	**Temp. 0.35**			**Temp. 0.175**		
Method	**TriviaQA**	**GSM8K**	**MMLU**	**TriviaQA**	**GSM8K**	**MMLU**
Uniform random baseline	0.3015	0.3133	0.2909	0.3085	0.3201	0.2930
Verbalized confidence	0.2573	0.0531	0.1210	0.2607	0.0503	0.1260
Verbalized confidence+Iso	0.1761	0.0451	0.1078	0.1808	0.0460	0.1106
P(True)	0.3278	0.0515	0.1557	0.3326	0.0531	0.1585
P(True)+Iso	0.1540	0.0428	0.1051	0.1586	0.0447	0.1078
Probe	**0.1273**	**0.0412**	**0.0993**	**0.1350**	**0.0429**	**0.1044**
gpt-oss-20b	**Temp. 0.5**			**Temp. 0.25**		
Method	**TriviaQA**	**GSM8K**	**MMLU**	**TriviaQA**	**GSM8K**	**MMLU**
Uniform random baseline	0.2720	0.3252	0.2917	0.2887	0.3244	0.3064
Verbalized confidence	0.1634	0.0261	0.0901	0.1860	0.0283	0.0945
Verbalized confidence+Iso	**0.0816**	**0.0259**	**0.0801**	**0.0984**	**0.0270**	**0.0847**
P(True)	0.2653	0.0294	0.1249	0.2713	0.0332	0.1278
P(True)+Iso	0.1493	0.0271	0.0963	0.1611	0.0295	0.0981
Probe	0.1143	0.0331	0.0946	0.1208	0.0365	0.1013

Conclusion. In summary, both the capability calibration target and confidence estimator evaluations are stable across decoding temperatures. These results directly demonstrate the transferability of capability calibration across decoding temperatures.

B.6 LLMs' MEAN EXPECTED ACCURACIES ACROSS DATASETS

Table 7 shows the dataset-level expected accuracy of every model-dataset pair in this paper.

Table 7: Three LLMs' mean expected accuracies in seven datasets.

Model / Dataset	TriviaQA	SimpleQA	GSM8K	MATH-500	AIME	MMLU	GPQA
Olmo-3-7B-Instruct	48.84%	1.92%	93.31%	90.06%	47.29%	70.06%	43.64%
Qwen3-8B	60.40%	3.10%	93.56%	82.96%	20.97%	79.40%	49.65%
gpt-oss-20b	65.07%	3.90%	95.76%	95.66%	69.10%	84.44%	66.97%

B.7 COMPUTE RESOURCES FOR ESTIMATING EXPECTED ACCURACY

To estimate the expected accuracy of the models using $k_{eval} = 100$, all experiments are conducted on a single GPU (NVIDIA A40 or NVIDIA RTX Pro 6000). Each dataset takes at most 24 hours.

C DETAILS OF CONFIDENCE ESTIMATION METHODS

C.1 DETAILED DESCRIPTIONS OF EXISTING METHODS

In this section, we discuss the details of existing response calibration confidence estimators and LLM uncertainty quantification methods.

Response calibration confidence estimators. Training-free methods: verbalized confidence (Lin et al., 2022; Tian et al., 2023) that prompts the model to state its certainty, and token probability methods (Kadavath et al., 2022; Manakul et al., 2023). Specifically, $P(\text{True})$ (Kadavath et al., 2022) estimates confidence by measuring the probability assigned to confirmation tokens (e.g., "True"). These methods estimate confidence after the output is generated.

LLM Uncertainty Quantification (UQ) methods. We categorize LLM UQ methods into three kinds: token-based approaches, which derive confidence from token-level likelihoods or log probabilities (Kadavath et al., 2022; Duan et al., 2024); Sampling-based approaches, which analyze the consistency or entropy of multiple generated outputs (Wang et al., 2023; Kuhn et al., 2023; Cecere et al., 2025); and methods leveraging the models' internal signals. This includes training models to explicitly output an "I Don't Know" token (Cohen et al., 2024), probing hidden states, or estimating query-level uncertainty via internal self-evaluation (Chen et al., 2026).

C.2 IMPLEMENTATION DETAILS OF CONFIDENCE ESTIMATORS

Random baseline. The random baseline draws a prediction $s \sim \mathrm{Uniform}(0,1)$ independently of the target. For a single question i with target $\hat{\mu}_i$, the expected squared error is

$$\begin{aligned}\mathbb{E}_s\big[(s-\hat{\mu}_i)^2\big] &= \int_0^1 \left(s^2 - 2s\hat{\mu}_i + \hat{\mu}_i^2\right) ds \\ &= \left[\tfrac{1}{3}s^3 - s^2\hat{\mu}_i + s\hat{\mu}_i^2\right]_0^1 \\ &= \tfrac{1}{3} - \hat{\mu}_i + \hat{\mu}_i^2.\end{aligned}$$

Averaging over the N questions in the dataset and writing $\bar{\mu} = \frac{1}{N}\sum_i \hat{\mu}_i$ and $\sigma^2 = \frac{1}{N}\sum_i(\hat{\mu}_i - \bar{\mu})^2$ for the mean and variance of the targets, the expected loss of the random baseline is

$$\frac{1}{N}\sum_{i=1}^{N}\left(\tfrac{1}{3} - \hat{\mu}_i + \hat{\mu}_i^2\right) = \tfrac{1}{3} - \bar{\mu} + \bar{\mu}^2 + \sigma^2,$$

using $\frac{1}{N}\sum_i \hat{\mu}_i^2 = \bar{\mu}^2 + \sigma^2$.

Verbalized confidence prompt

Question: {question}

How likely are you to answer the question correctly? You may refer to the following probabilities P:
- 0.0-0.1: "Almost no chance"
- 0.1-0.2: "Highly unlikely"
- 0.2-0.3: "Chances are slight"
- 0.3-0.4: "Unlikely"
- 0.4-0.5: "Less than even"
- 0.5-0.6: "Better than even"
- 0.6-0.7: "Likely"
- 0.7-0.8: "Very good chance"
- 0.8-0.9: "Highly likely"
- 0.9-1.0: "Almost certain"

Reason about your uncertainty and confidence, and then provide a probability P between 0.0 and 1.0 in the format of \boxed{P}.

Figure 8: The prompt for verbalized confidence.

P(True) prompt

Question: {question}

Are you able to answer the question correctly?
Answer with only a single word: Yes or No.

Figure 9: The prompt for P(True).

Verbalized confidence and P(True). Prompts are provided in Figure 8 and Figure 9, respectively.

Training linear probes on LLM's hidden states. We used layer activations after the initial embedding layer and after each transformer block, so there are $\ell + 1$ layers of activations used, with ℓ being the number of transformer blocks in the LLM. In our preliminary experiments, we trained a linear probe on activations of each layer, as well as on max-pooled or mean-pooled activations. We found that mean-pooled activations performed the best on the validation sets of each dataset, so we train linear probes on the mean-pooled activations in our main experiments (results listed in §5.3.2). We also searched for other hyperparameters, including number of epochs = $\{100, 200, 500, 1000\}$, batch sizes = $\{32, 64, 128, 256\}$, weight decays = $\{0.1, 0.01, 0.001\}$, loss functions = $\{\mathrm{BCE}, \mathrm{MSE}\}$, input feature standardization = $\{\mathrm{False}, \mathrm{True}\}$, and learning rates = $\{1\mathrm{e}-2, 5\mathrm{e}-3, 2\mathrm{e}-3, 1\mathrm{e}-3, 5\mathrm{e}-4, 2\mathrm{e}-4, 1\mathrm{e}-4, 5\mathrm{e}-5, 2\mathrm{e}-5, 1\mathrm{e}-5, 5\mathrm{e}-6, 2\mathrm{e}-6, 1\mathrm{e}-6\}$ on the validation sets. The final chosen hyperparameters are listed in Table 8. We also tried training 2-layer MLP probes, but found that their performance did not differ from linear probes.

Train-test splits for training-based methods. For verbalized confidence + isotonic regression, P(True) + isotonic regression, or linear probe training, we adopt the same train-test splits as follows:

- TriviaQA Joshi et al. (2017): We sample 1,000 instances from the validation set as the test split, and sample 7,473 instances from the training set as the train split.
- SimpleQA Wei et al. (2024); Haas et al. (2025): We use the 1,000 instances from SimpleQA verified Haas et al. (2025) as the test split, and the non-overlapping instances in the original SimpleQA Wei et al. (2024) as the train split.
- GSM8K Cobbe et al. (2021): We use the 7,473 instances in the training set as the training split, and the 1,319 instances in the test set as the test split.
- MATH Hendrycks et al. (2021b): We use the 500 instances in MATH-500 Lightman et al. (2024) as the test split, and the remaining 12,000 instances from the original MATH dataset Hendrycks et al. (2021b) as the training split.

- AIME [1] [2] [3]: We use the 918 instances from AIME 1983-2023 as the training split, and the 90 instances from AIME 2024-2026 as the test split.
- MMLU Hendrycks et al. (2021a): We use all the 1,531 instances in the validation set as the training split, and sample 1,000 instances in the test set as the test split due to budget consideration.
- GPQA Rein et al. (2024): We use the 198 instances in the diamond subset as the test split, and the non-overlapping 348 instances in the extended subset as the train split.

Table 8: Hyperparameters or model information for linear probes trained on three LLMs.

Hyperparameter or Model Info.	**Olmo-3-7B-Ins.**	**Qwen3-8B**	**gpt-oss-20b**
Number of layer activations used	33	37	25
Hidden dimension	4096	4096	2880
Epochs	100	100	100
Batch size	32	32	32
Weight decay	0.01	0.01	0.01
Pooling method	Mean pooling	Mean pooling	Mean pooling
Loss function	BCE loss	BCE loss	BCE loss
Feature standardization	False	False	True
Learning rate (TriviaQA)	5×10^{-3}	2×10^{-4}	2×10^{-4}
Learning rate (SimpleQA)	1×10^{-2}	2×10^{-4}	1×10^{-2}
Learning rate (GSM8K)	1×10^{-2}	1×10^{-5}	1×10^{-3}
Learning rate (MATH)	1×10^{-2}	1×10^{-4}	2×10^{-3}
Learning rate (AIME)	5×10^{-3}	5×10^{-5}	2×10^{-4}
Learning rate (MMLU)	1×10^{-2}	2×10^{-5}	2×10^{-3}
Learning rate (GPQA)	5×10^{-3}	2×10^{-5}	1×10^{-4}

[1] https://huggingface.co/datasets/gneubig/aime-1983-2024
[2] https://huggingface.co/datasets/math-ai/aime25
[3] https://huggingface.co/datasets/MathArena/aime_2026

C.3 Analysis of k_c for Response Consistency method

In this section, we analyze how the sample size k_c impacts the **Response Consistency** confidence estimator. As shown in Table 9, a larger k_c generally leads to improved Brier scores by reducing estimation variance. However, this performance gain trades off with a higher estimation cost. We observe performance saturation because increasing k_c cannot correct for fundamental miscalibration, which is the main limitation of this confidence estimator. If the model's most frequent response is incorrect, the estimator converges to a confidence score for a failure case, preventing further reduction in Brier score.

Table 9: **Capability calibration Brier scores of Response Consistency across different numbers of samples** k_c. We use **bold** to denote the **best** calibrated method, and underline to denote the second best. Larger k_c generally improves performance at the cost of higher estimation overhead.

	Domain	Factual knowledge		Mathematical reasoning			General exams	
Method	**Cost**	**TriviaQA**	**SimpleQA**	**GSM8K**	**MATH**	**AIME**	**MMLU**	**GPQA**
Olmo-3-7B-Instruct								
Consistency ($k = 5$)	$5L$	0.1228	0.1605	0.0290	0.0373	0.1220	0.1220	0.2462
Consistency ($k = 10$)	$10L$	0.1121	0.1297	0.0272	0.0383	0.1097	0.1103	0.2246
Consistency ($k = 20$)	$20L$	**0.1046**	**0.1141**	**0.0264**	**0.0350**	**0.1048**	**0.1030**	**0.2079**
Qwen3-8B								
Consistency ($k = 5$)	$5L$	0.1433	0.2016	0.0255	0.0336	0.1285	0.0993	0.1830
Consistency ($k = 10$)	$10L$	0.1301	0.1657	0.0241	0.0304	0.1282	0.0922	0.1656
Consistency ($k = 20$)	$20L$	**0.1239**	**0.1521**	**0.0237**	**0.0276**	**0.1069**	**0.0892**	**0.1485**
gpt-oss-20b								
Consistency ($k = 5$)	$5L$	**0.1274**	0.0912	0.0210	**0.0553**	0.0536	0.0728	0.1275
Consistency ($k = 10$)	$10L$	0.1382	0.0624	0.0202	0.0554	**0.0391**	0.0673	0.1066
Consistency ($k = 20$)	$20L$	0.1406	**0.0520**	**0.0188**	0.0593	0.0402	**0.0629**	**0.1007**

C.4 Probe performance in out-of-distribution data

Table 10 provides the performance of Probe in out-of-distribution and out-of-domain scenario. It shows that probing performs well under in-domain, in-distribution settings. However, some applications may require applying a confidence estimator to (1) *same-domain but out-of-distribution* queries (e.g., different factual knowledge datasets), or (2) *out-of-domain* queries (e.g., training on factual knowledge but applying to mathematical reasoning). Overall, probing generalizes reasonably well under in-domain, out-of-distribution settings, especially for the mathematical reasoning domain. However, it does not consistently generalize to out-of-domain settings. Existing works (Cencerrado et al., 2025; Zhang et al., 2025a) also find that probes struggles with out-of-domain settings for response calibration. Since this paper is focused on the calibration (evaluation) framework of confidence estimator instead of confidence estimator itself, the out-of-domain generalization of probes is out-of-scope for this paper. Developing generalizable methods for capability calibration remains an important direction for future work.

C.4.1 Mixing training dataset from different domains

In this section, we discuss the effect of mixing training datasets from different domains. The experiment results are in Table 11. While single-dataset probes typically perform best on their specific in-distribution tasks, the mixed probe (TriviaQA + GSM8K) often achieves the best or second-best Brier scores across all in-domain out-of-distribution categories, such as MATH and SimpleQA. This suggests that data diversity improves the probe's ability to generalize to in-domain tasks. However, this trend does not hold in out-domain settings like MMLU and GPQA, where the performance of mixed versus single probes varies by model, indicating that mixing training data does not guarantee improved calibration for entirely unrelated domains.

Table 10: **Capability calibration performance (Brier score) of linear probes trained with different training sets.** For probes, we use different colors to indicate in-domain in-distribution, in-domain out-of-distribution, and out-domain performance. We use **bold** to denote the **best** calibrated method, and underline to denote the second best. Probe performs best under in-domain in-distribution settings and generalizes reasonably well under in-domain out-distribution settings.

Brier score (↓)	Domain	Factual knowledge		Mathematical reasoning			General exams	
Method	**Cost**	**TriviaQA**	**SimpleQA**	**GSM8K**	**MATH**	**AIME**	**MMLU**	**GPQA**
Olmo-3-7B-Instruct								
Probe (train on TriviaQA)	< 1	**0.1101**	0.0397	0.1377	0.1547	0.1315	0.1209	0.1230
Probe (train on SimpleQA)	< 1	0.3337	**0.0130**	0.5342	0.4510	0.1894	0.3187	0.1557
Probe (train on GSM8K)	< 1	0.2924	0.6038	**0.0364**	0.0496	0.1622	0.1142	0.1635
Probe (train on MATH)	< 1	0.2628	0.4810	0.0382	**0.0373**	0.1195	0.1271	0.1255
Probe (train on AIME)	< 1	0.2015	0.3686	0.0382	0.0443	**0.1092**	0.1170	0.1167
Probe (train on MMLU)	< 1	0.2058	0.2349	0.0383	0.0527	0.2755	**0.1004**	0.1149
Probe (train on GPQA)	< 1	0.1935	0.1587	0.3428	0.1726	0.1883	0.2492	**0.1072**
Qwen3-8B								
Probe (train on TriviaQA)	< 1	**0.1163**	0.0591	0.2411	0.1269	0.1075	0.1256	**0.1366**
Probe (train on SimpleQA)	< 1	0.4255	**0.0306**	0.4370	0.2226	0.4496	0.3955	0.1619
Probe (train on GSM8K)	< 1	0.2164	0.5177	**0.0377**	0.0678	0.1570	0.1121	0.1602
Probe (train on MATH)	< 1	0.3235	0.8566	0.0411	**0.0471**	0.0774	0.1358	0.2383
Probe (train on AIME)	< 1	0.2210	0.4383	0.0993	0.1216	**0.0578**	0.1527	0.1607
Probe (train on MMLU)	< 1	0.2638	0.6218	0.0402	0.0697	0.2971	**0.0987**	0.1570
Probe (train on GPQA)	< 1	0.2117	0.2707	0.3009	0.4568	0.1207	0.1746	0.1571
gpt-oss-20b								
Probe (train on TriviaQA)	< 1	**0.1027**	0.0894	0.0273	0.0284	0.1508	0.1071	0.1514
Probe (train on SimpleQA)	< 1	0.5694	**0.0264**	0.8525	0.7829	0.3848	0.7277	0.5702
Probe (train on GSM8K)	< 1	0.2574	0.8625	0.0291	0.0395	0.1252	0.1329	0.2273
Probe (train on MATH)	< 1	0.2274	0.8025	0.0317	**0.0216**	0.2530	0.1037	0.1541
Probe (train on AIME)	< 1	0.2385	0.7793	**0.0272**	0.0319	**0.0966**	0.0985	0.1515
Probe (train on MMLU)	< 1	0.1739	0.2729	0.0278	0.0224	0.1889	**0.0878**	0.1487
Probe (train on GPQA)	< 1	0.2218	0.7578	0.0327	0.0378	0.1305	0.0954	**0.1462**

Table 11: **Capability calibration Brier scores of linear probes trained on single and mixed datasets.** Results reported by Brier scores (↓). For linear probes trained with different datasets, we use different colors to indicate in-domain in-distribution, in-domain out-of-distribution, and out-domain performance. We use **bold** to denote the **best** calibrated method, and underline to denote the second best. Results show that training probes on a mixture of datasets (TriviaQA + GSM8K) generally yields the most robust calibration across both in-distribution and in-domain OOD tasks (e.g., MATH, SimpleQA). However, this benefit is less consistent for out-domain datasets (MMLU, GPQA), where specialized single-dataset probes occasionally maintain an edge.

	Domain	Factual knowledge		Mathematical reasoning			General exams	
Method	**Cost**	**TriviaQA**	**SimpleQA**	**GSM8K**	**MATH**	**AIME**	**MMLU**	**GPQA**
Olmo-3-7B-Instruct								
Probe (train on TriviaQA)	< 1	**0.1101**	0.0397	0.1377	0.1547	0.1315	0.1209	0.1230
Probe (train on GSM8K)	< 1	0.2924	0.6038	**0.0364**	**0.0496**	0.1622	**0.1142**	0.1635
Probe (TriviaQA + GSM8K)	< 1	0.1147	**0.0396**	0.0371	0.0545	**0.1127**	0.1298	**0.1141**
Qwen3-8B								
Probe (train on TriviaQA)	< 1	0.1163	**0.0591**	0.2411	0.1269	0.1075	0.1256	**0.1366**
Probe (train on GSM8K)	< 1	0.2164	0.5177	**0.0377**	**0.0678**	0.1570	**0.1121**	0.1602
Probe (TriviaQA + GSM8K)	< 1	**0.1079**	0.0607	0.0408	0.1378	**0.0952**	0.1422	0.1683
gpt-oss-20b								
Probe (train on TriviaQA)	< 1	0.1027	0.0894	**0.0273**	**0.0284**	0.1508	0.1071	0.1514
Probe (train on GSM8K)	< 1	0.2574	0.8625	0.0291	0.0395	0.1252	0.1329	0.2273
Probe (TriviaQA + GSM8K)	< 1	**0.0902**	**0.0707**	0.0277	0.0554	**0.1181**	**0.0818**	**0.1256**

C.5 EVALUATING CONFIDENCE ESTIMATORS IN RANKING METRIC

Ranking metrics are valuable for tasks needing query-level discrimination, which a constant predictor lacks. Since the target of CC is a continuous value in $[0, 1]$, a suitable ranking metric would be Spearman's correlation, which measures whether the confidence scores preserve the ranking of examples by their correctness probability. We evaluated the Spearman's r for existing confidence estimation methods in Table 12, whose findings are consistent with the Brier score Section 5.3.2: Verbalized confidence achieves the best performance for gpt-oss-20b, while Probe is generally the most effective for the other two models.

Table 12: **Capability calibration performance (Spearman's ρ) of different methods with three LLMs on seven datasets.** We use **bold** to denote the **best** calibrated method, and underline to denote the second best. The findings are consistent with the Brier score result in Table 3.

Spearman's ρ (↑)	**Domain**	**Factual knowledge**		**Mathematical reasoning**			**General exams**	
Method	**Cost**	**TriviaQA**	**SimpleQA**	**GSM8K**	**MATH**	**AIME**	**MMLU**	**GPQA**
Olmo-3-7B-Instruct								
Uniform random baseline	N/A	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Verbalized confidence	L	0.3424	0.0982	0.1293	0.1476	0.4013	0.1709	0.1552
+ isotonic regression	L	0.3443	0.0928	0.1325	0.1576	0.3593	0.1696	0.1694
P(True)	1	0.3944	0.1336	0.2474	0.1862	0.1123	0.3163	0.0211
+ isotonic regression	1	0.3951	0.1256	0.2499	0.1889	0.0993	0.3178	0.0565
Probe	< 1	**0.6418**	**0.1576**	**0.4551**	**0.6493**	**0.5885**	**0.5573**	**0.4001**
Qwen3-8B								
Uniform random baseline	N/A	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Verbalized confidence	L	0.3197	0.0225	0.0606	0.1954	0.1297	0.3075	0.0829
+ isotonic regression	L	0.3192	**0.0352**	0.0636	0.1891	0.1682	0.3069	0.0811
P(True)	1	0.4795	-0.0556	0.3043	0.3989	0.4042	0.3863	**0.3261**
+ isotonic regression	1	0.4876	-0.0599	0.3051	0.4019	0.4212	0.3840	0.2949
Probe	< 1	**0.6411**	0.0320	**0.4058**	**0.7058**	**0.5711**	**0.4681**	-0.0077
gpt-oss-20b								
Uniform random baseline	N/A	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Verbalized confidence	L	0.6674	**0.3286**	**0.2526**	0.3565	0.7237	0.5059	0.5828
+ isotonic regression	L	**0.6680**	0.3254	0.2506	0.3558	**0.7254**	**0.5075**	**0.5912**
P(True)	L	0.2625	0.0901	0.1885	0.0205	0.1975	0.1811	0.2937
+ isotonic regression	L	0.2608	0.1089	0.1859	0.0289	0.2135	0.1890	0.2904
Probe	< 1	0.6283	0.1607	0.2283	**0.5372**	0.4854	0.2814	0.2088

C.6 EVALUATING CONFIDENCE ESTIMATORS IN ECE

In this section, we report the ECE score of the confidence estimators. As shown in Table 13, verbalized confidence after isotonic regression is competitive with the linear probe. Figure 10 further visualizes the ECE metric as reliability diagrams.

Table 13: **Capability calibration performance (ECE) of different methods with three LLMs on seven datasets.** We use **bold** to denote the **best** calibrated method, and underline to denote the second best. The findings are consistent with the Brier score results in Table 3.

ECE (↓)	Domain	Factual knowledge		Mathematical reasoning			General exams	
Method	**Cost**	**TriviaQA**	**SimpleQA**	**GSM8K**	**MATH**	**AIME**	**MMLU**	**GPQA**
Olmo-3-7B-Instruct								
Uniform random baseline	N/A	0.2511	0.4667	0.4365	0.4110	0.2644	0.2905	0.2601
Verbalized confidence	L	0.3222	0.3803	0.0143	0.0309	0.2913	0.1877	0.3714
+ isotonic regression	L	0.0382	0.0141	0.0192	**0.0066**	0.1597	**0.0153**	0.0446
P(True)	1	0.0985	0.0847	0.2483	0.2455	0.2062	0.3126	0.1540
+ isotonic regression	1	0.1098	0.0137	0.0186	0.0131	0.1829	0.0235	**0.0423**
Probe	< 1	**0.0306**	**0.0095**	**0.0085**	0.0218	**0.1116**	0.0164	0.0652
Qwen3-8B								
Uniform random baseline	N/A	0.2615	0.4496	0.4386	0.3591	0.3440	0.3368	0.2539
Verbalized confidence	L	0.2758	0.5494	0.0181	0.0877	0.5210	0.1112	0.3817
+ isotonic regression	L	**0.0275**	**0.0211**	0.0171	0.0148	0.1747	**0.0183**	**0.0947**
P(True)	1	0.3782	0.6576	0.0663	0.1671	0.5917	0.1993	0.4657
+ isotonic regression	1	0.0340	0.0240	**0.0035**	0.0268	0.1029	0.0297	0.1038
Probe	< 1	0.0296	0.0399	0.0080	**0.0146**	**0.0532**	0.0255	0.1463
gpt-oss-20b								
Uniform random baseline	N/A	0.2731	0.4426	0.4579	0.4575	0.2950	0.3683	0.2829
Verbalized confidence	L	0.1691	0.3593	0.0118	**0.0054**	0.1086	0.0726	0.1264
+ isotonic regression	L	0.0241	0.0252	**0.0052**	0.0257	**0.0455**	**0.0399**	**0.0564**
P(True)	L	0.3119	0.6334	0.0441	0.0426	0.2620	0.1634	0.2886
+ isotonic regression	L	**0.0156**	**0.0217**	0.0054	0.0200	0.0803	0.0465	0.1053
Probe	< 1	0.0275	0.0557	0.0322	0.0462	0.1089	0.0498	0.1368

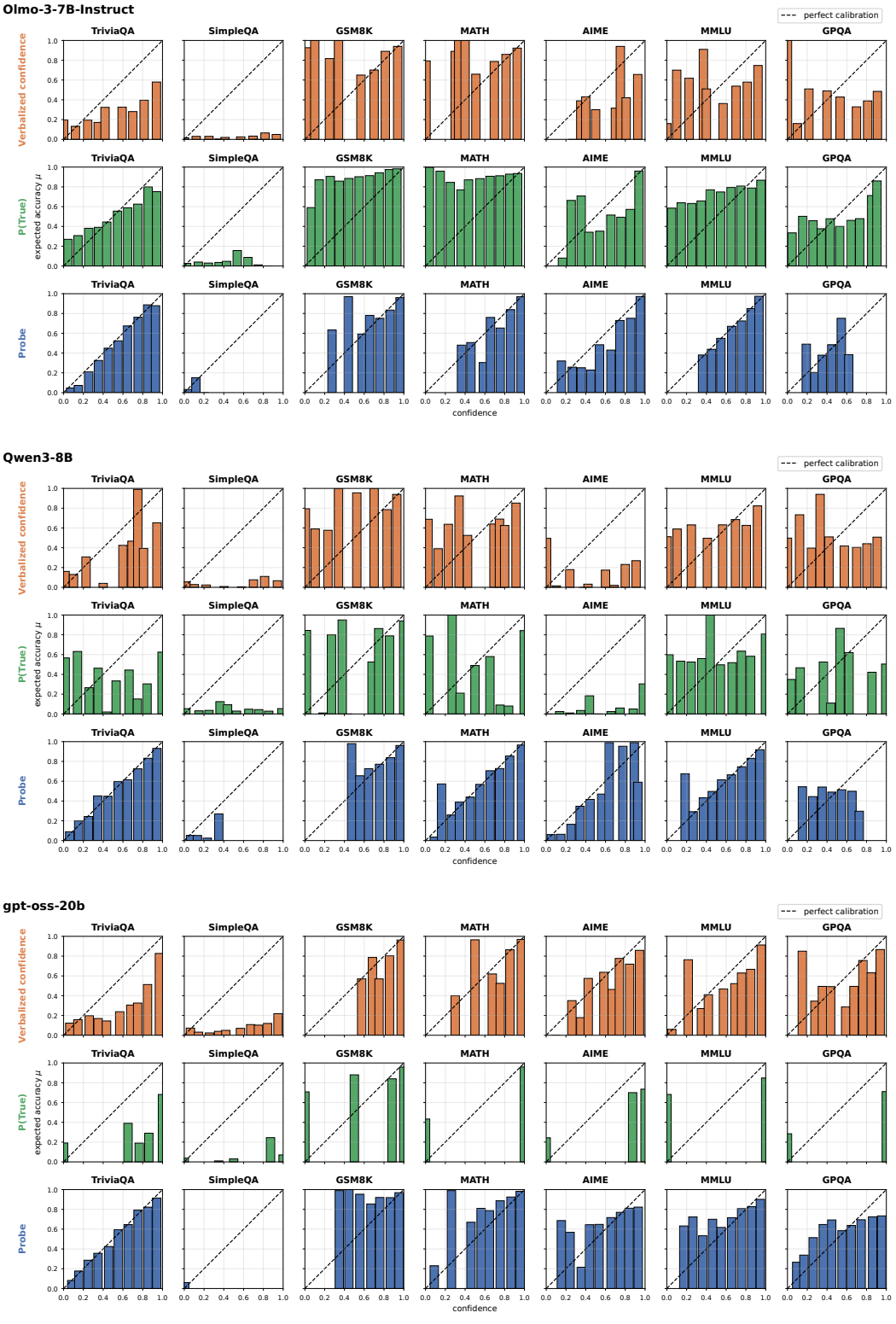

Figure 10: **Reliability diagrams for the three models.**

C.7 IMPROVING VERBALIZED CONFIDENCE

In this section, we investigate whether few-shot examples help verbalized confidence. Experiment results are in Table 14. The performance of few-shot verbalized confidence differs across LLMs and datasets; few-shot examples do not always improve the performance.

Table 14: **Performance of few-shot verbalized confidence on TriviaQA (left) and GSM8K (right) with reduced offline training costs**. Few-shot examples are from the training set of TriviaQA and GSM8K, respectively. Three few-shot examples are provided. Other confidence estimators are listed for comparison. Results are reported by Brier scores (↓). While few-shot verbalized confidence consistently improves upon the zero-shot baseline on GSM8K, this advantage is not observed consistently on TriviaQA.

Method / Model	Olmo-3-7B-Instruct	Qwen3-8B	gpt-oss-20b
Uniform random baseline	0.2745	0.2865	0.2639
Verbalized confidence	0.2717	0.2459	0.1204
Verbalized confidence (few-shot)	0.2802	0.2724	0.1217
P(True)	0.1698	0.3253	0.2435
Probe	**0.1101**	**0.1163**	**0.1027**

Method / Model	Olmo-3-7B-Instruct	Qwen3-8B	gpt-oss-20b
Uniform random baseline	0.3119	0.3144	0.3195
Verbalized confidence	0.0444	0.0455	0.0247
Verbalized confidence (few-shot)	0.0423	0.0429	**0.0237**
P(True)	0.1266	0.0483	0.0299
Probe	**0.0364**	**0.0377**	0.0291

C.8 COST OF CONFIDENCE ESTIMATORS

The cost for confidence estimators stems from **(1)** the one-time offline preprocessing/training cost and **(2)** the recurring online inference cost. We detail two types of costs below:

For **(1)** offline training, only the linear probe requires offline cost for all confidence estimators we discussed in Section 5.2. The cost comes from (a) constructing training labels for each query by repeated sampling to estimate the expected accuracy, and (b) extracting hidden states. However, once the hidden states are cached, training the probe itself only requires fitting on frozen hidden states rather than backpropagating through the full LLM. In our case, hidden-state extraction, caching, and probe training were all done on a single 20GB GPU.

For **(2)** online cost, all methods require one forward pass to process the query. Linear probe only add a single linear-layer computation, which is much smaller and is the main advantage of linear probes. By contrast, methods such as verbalized confidence require autoregressive decoding, often requiring hundreds of forward passes. Therefore, linear probing is substantially cheaper during online deployment, where latency is crucial and inference cost scales with usage.

Meanwhile, Training cost of linear probes can be further reduced: While we use $k_{eval} = 100$, probes can be trained with a smaller k_{train}. We show this by using $k_{train} = 10$ for building training sets, and find that resulting probes still achieve similar performance while reducing data construction cost to one-tenth of the original setup where $k_{train} = 100$. Experiment results of probes trained on TriviaQA and GSM8K are presented in Table 15.

Table 15: **Performance of linear probes on TriviaQA (left) and GSM8K (right) with reduced offline training costs**. Results are reported by Brier scores (↓). Other confidence estimators are listed for comparison. Results show that linear probes trained with $k_{train} = 10$ achieve nearly identical performance to the full setup ($k_{train} = 100$) across all evaluated models.

Method / Model	Olmo-3-7B-Instruct	Qwen3-8B	gpt-oss-20b
Uniform random baseline	0.2745	0.2865	0.2639
Verbalized confidence	0.2717	0.2459	0.1204
P(True)	0.1698	0.3253	0.2435
Probe ($k_{train} = 100$)	**0.1101**	**0.1163**	**0.1027**
Probe ($k_{train} = 10$)	0.1117	0.1083	0.0859

Method / Model	Olmo-3-7B-Instruct	Qwen3-8B	gpt-oss-20b
Uniform random baseline	0.3119	0.3144	0.3195
Verbalized confidence	0.0444	0.0455	**0.0247**
P(True)	0.1266	0.0483	0.0299
Probe ($k_{train} = 100$)	**0.0364**	**0.0377**	0.0291
Probe ($k_{train} = 10$)	0.0370	0.0371	0.0290

D DETAILS OF THE APPLICATIONS

D.1 PASS@k SIMULATION DETAILS

D.1.1 PROCESS OF SIMULATING PASS@k CURVE

Pass@k score at each k. Define the capability-calibrated confidence as p. For each query i that is sampled k times, the success rate, i.e., pass it or not, follows a Bernoulli distribution. The

Bernoulli distribution has mean p and variance $p(1-p)$. Therefore, for each query i that samples k times, the success rate is $P(\text{success@}k)_i = S_{i,k} = 1-(1-p_i)^k$, the variance is $S_{i,k}(1-S_{i,k})$. Generalized to the dataset level. For a dataset $\mathcal{D} = \{x_i\}_{i=1}^N$, the pass@$k$ score's mean and variance are $\mu_k = \frac{1}{N}\sum_{i=1}^N S_{i,k}$ and $\text{std}_k^2 = \frac{1}{N^2}\sum_{i=1}^N S_{i,k}(1-S_{i,k})$ respectively. Here, we assume that each query is independent, so the covariance is zero.

Drawing the simulation curve. The mean and variance of the pass@k score are estimated for each k along the curve. Based on the Central Limit Theorem, the distribution of pass@k is treated as a normal distribution, allowing for the calculation of a 95% confidence interval. This interval represents the region where the true curve is expected to be.

D.1.2 More simulation results

In this section, we discuss more pass@k simulation results. First of all, the MSE of AIME simulation results are in Table 16. The experiment results are consistent with the simulation results on MATH-500.

Figure 11 presents the pass@k simulation curve at the dataset level. While Oracle Capability-Calibrated Confidence (Oracle-CC) fits the actual pass@k curve near perfectly, Probe does not fit well in most scenarios. The experiment results encourage future work on developing confidence estimators that can capture the model's capability on the dataset.

Table 16: **Pass@k simulation error (MSE) on the AIME dataset.** The experiment findings are consistent with the MATH-500 simulation results at Table 2.

Method	pass@1	pass@4	pass@16	pass@64
Olmo-3-7B-Instruct				
Oracle-CC	0.0000	0.0000	0.0001	0.0014
Oracle-RC	0.0967	0.1523	0.2541	0.3395
Probe	0.1092	0.2002	0.1928	0.1577
Qwen3-8B				
Oracle-CC	0.0000	0.0000	0.0002	0.0036
Oracle-RC	0.0522	0.0988	0.2307	0.4170
Probe	0.0578	0.1386	0.2495	0.2992
gpt-oss-20b				
Oracle-CC	0.0000	0.0000	0.0000	0.0006
Oracle-RC	0.1068	0.1520	0.1959	0.2223
Probe	0.0966	0.0968	0.0744	0.0584

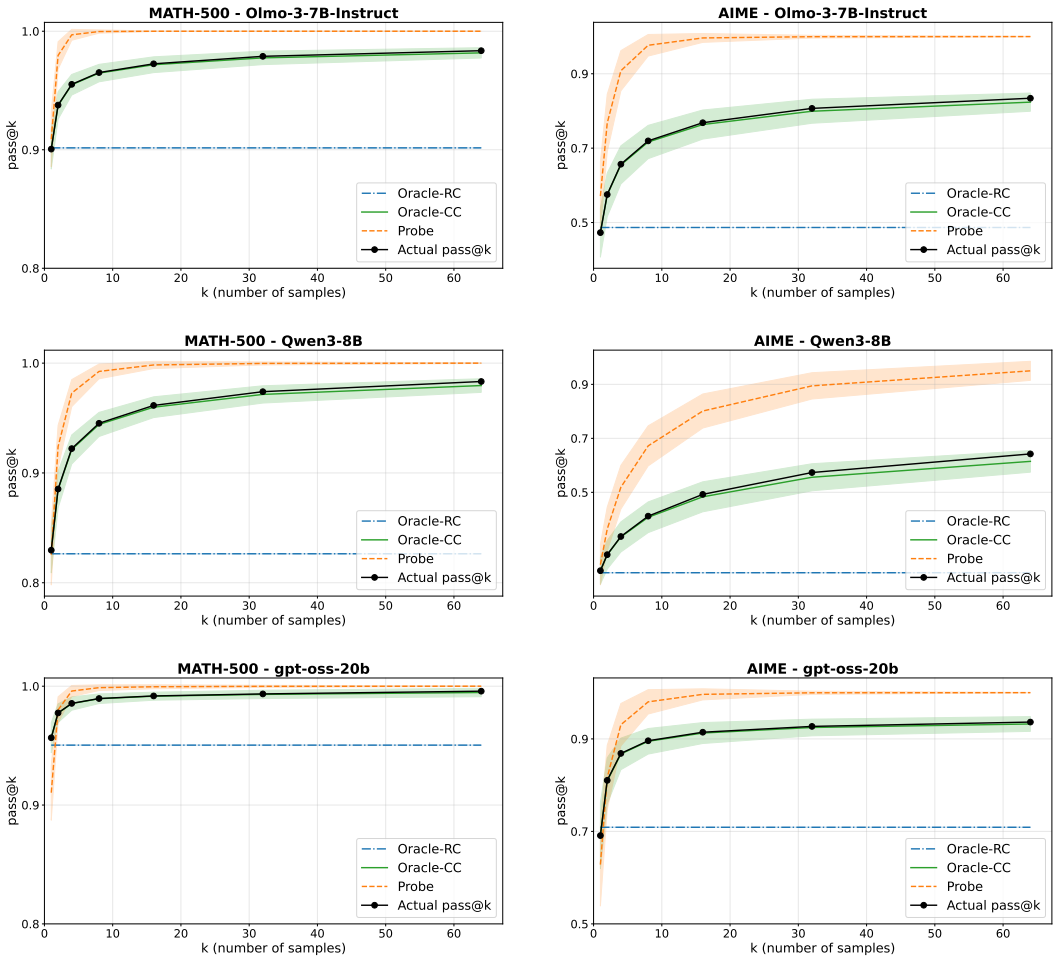

Figure 11: **Pass@k simulation results across 3 models on MATH-500 and AIME datasets.** While Oracle-CC can simulate the actual pass@k near-perfectly, Oracle-RC performs poorly as it is not measuring the models' capabilities. However, Probe does not fit well in many scenarios.

D.2 Inference budget allocation details

D.2.1 Greedy algorithm for test-time compute allocation (Damani et al., 2025)

To allocate the inference budget efficiently, we maximize the expected number of solved questions (best-of-k). Let N be the number of questions, $N \times B$ be the total budget, and p_i be the capability-calibrated confidence estimation for question i.

The total expected score S is the sum of the probabilities that each question is solved at least once:

$$S = \sum_{i=1}^{N} \left[1 - (1 - p_i)^{k_i}\right] \tag{15}$$

where k_i is the number of samples allocated to question i.

To optimize this, we analyze the marginal improvement of adding a single sample to question i, given that it has already been allocated k_i samples:

$$\begin{aligned} \text{Gain}_i &= S(\text{with } k_i + 1 \text{ samples}) - S(\text{with } k_i \text{ samples}) \\ &= \left[1 - (1 - p_i)^{k_i+1}\right] - \left[1 - (1 - p_i)^{k_i}\right] \\ &= (1 - p_i)^{k_i} - (1 - p_i)^{k_i+1} \\ &= (1 - p_i)^{k_i} \left[1 - (1 - p_i)\right] \\ &= p_i(1 - p_i)^{k_i} \end{aligned} \tag{16}$$

Because the gain function is strictly decreasing with respect to k_i, a greedy strategy that iteratively assigns the next budget unit to the question with the highest current Gain_i results in better allocation.

D.2.2 More experiment results

Full experiment results of inference budget allocation are available at Figure 12. We observe that confidence estimators with lower Brier scores have better inference budget allocation performance.

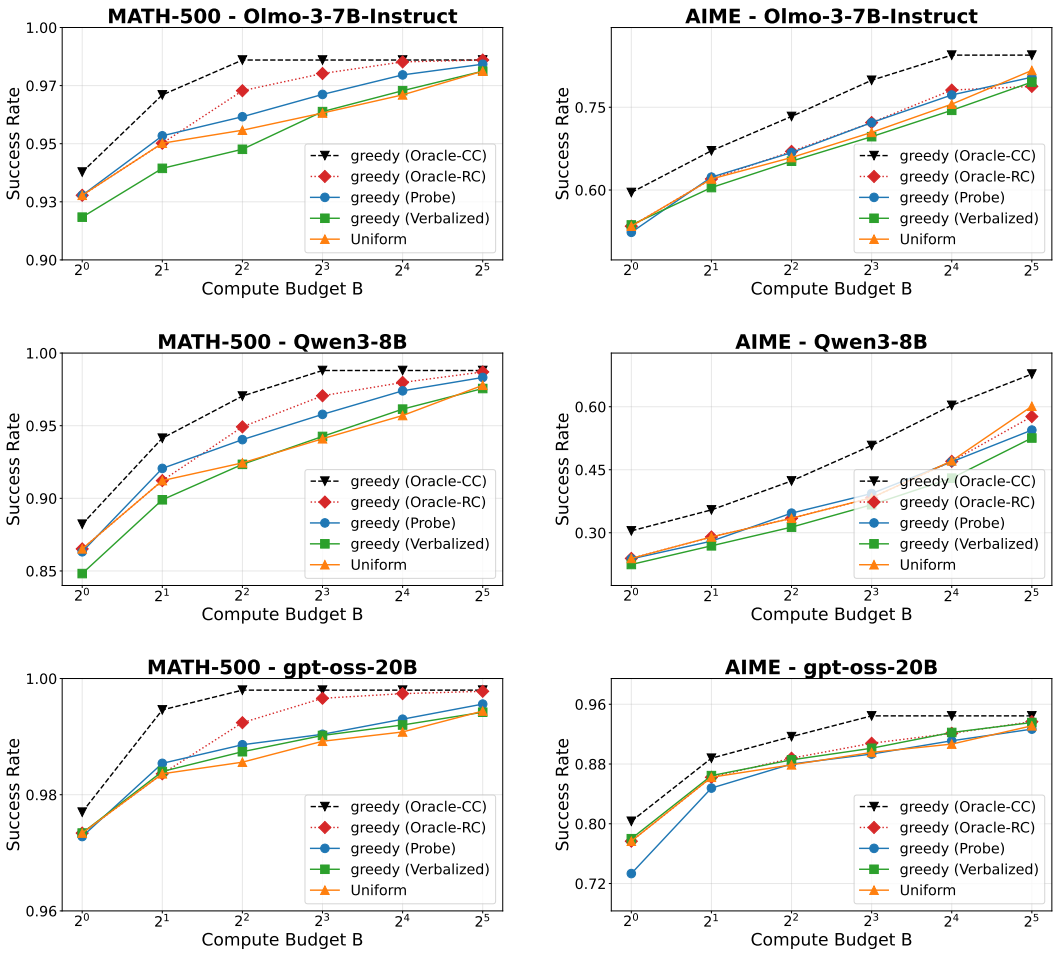

Figure 12: **Budget allocation results across 3 models on MATH-500 and AIME datasets.** Oracle confidence consistency reaches the best performance in all scenarios. Verbalized confidence and Probe often outperform the uniform allocation. If the Brier score is good (see Table 3 for the Brier score), the estimated confidence will have better performance in inference budget allocation.

D.3 Estimator selection and downstream impact under RC and CC

In this section, we ask:

As two evaluation frameworks, do RC and CC select different confidence estimators?

We consider the 21 model–dataset settings in Table 3 and focus on query-level confidence estimation, where estimators receive only the query. At the dataset level, we test whether using expected accuracy as CC's target and single-response correctness as RC's target leads to different estimator selections. We use Brier score as the evaluation metric.

For CC, we sample $k_{eval} = 100$ responses and use Table 3 to identify the CC-selected estimator in each setting, whose ranking is stable. For RC, the selected estimator depends on the sampled response correctness. We therefore conduct 100,000 simulations to estimate each estimator's selection probability.

Among Verbalized confidence, P(True), and Probe, RC can select a different estimator from CC in six model–dataset settings. These results show that RC can select a different estimator from CC with non-negligible probability. When estimators' performances are close, RC is more likely to select a different method because it evaluates against a noisy target induced by stochastic decoding. When one estimator clearly dominates, as Probe does in several settings, RC and CC select the same estimator. Table 17 further reports the gpt-oss-20b settings where an alternative RC selection occurs in at least 5% of trials. A low disagreement probability reflects estimator dominance, not framework equivalence.

Table 17: **CC selections and alternative single-response RC selections for gpt-oss-20b.** We report an alternative RC selection (**RC alt.**) only when it is selected in at least 5% of trials. P and V denote Probe and Verbalized confidence, respectively.

CC choice; RC alt. (%)	TriviaQA	SimpleQA	GSM8K	MATH	AIME	MMLU	GPQA
gpt-oss-20b	P; P	P; P	V; V	**V; P (12.3%)**	**V; P (6.0%)**	V; V	V; V

D.3.1 Impact on Downstream Applications

We next ask:

Which evaluation framework selects better estimators for downstream applications, RC or CC?

We focus on gpt-oss-20b on MATH and AIME, the two settings in Table 17 with disagreement probabilities above 5%. For Pass@k prediction, Table 18 shows that the CC-selected estimator can perform better. For inference-budget allocation, Figure 12 shows that on AIME, the CC-selected Verbalized confidence outperforms the alternative RC selection, Probe.

Table 18: **Pass@k prediction MSE for the CC-selected estimator and the alternative RC selection.** Lower is better. RC selection probabilities are shown in parentheses.

Data	Selection	Estimator	Pass@1	Pass@4	Pass@16	Pass@64
MATH	CC choice	Verbalized	0.0186	0.0102	0.0067	0.0028
	RC alternative	Probe (12.3%)	0.0216	0.0102	0.0068	0.0028
AIME	CC choice	Verbalized	0.0581	0.0717	0.0701	0.0584
	RC alternative	Probe (6.0%)	0.0966	0.0968	0.0744	0.0584

On MATH, the two estimators perform similarly because their confidence estimates are similar, not because RC and CC are equivalent. On AIME, the CC-selected Verbalized confidence achieves lower prediction error than the alternative RC selection. Section 4.3 provides further discussion of evaluation framework selection.

D.4 Other potential applications

In this section, we first discuss how capability-calibrated confidence relates to other potential applications.

Resource Routing and Estimation. As discussed in Damani et al. (2025), by estimating the ranking of a group of models' capability on answering a question, capability-calibrated confidence can be directly applied to LLM Routing (Maurya et al., 2025; Jiang et al., 2023; Chen et al., 2024; Ong et al., 2025). Additionally, capability-calibrated confidence enables cost estimation. By estimating the expected accuracy μ (difficulty of the queries), we can predict the expected sampling budget $\frac{1}{\mu}$ required to generate a correct response (Wu et al., 2025b).

Inference-Time Reliability and Active Refinement. By defining a confidence threshold, one can implement reliable systems that perform Selective Prediction (Mao et al., 2025; Duan et al., 2024; Chen et al., 2023; Kamath et al., 2020), where systems could abstain or seek human assistance when the model is uncertain (Wu et al., 2024; Chen et al., 2026) or perform query rewriting when the confidence is low.

Enhanced Learning and Evaluation. Accurately estimating expected accuracy serves as a proxy for query difficulty. It allows Curriculum Learning (Zhang et al., 2026; 2025d) that sorts training data by difficulty, or identifies the effects of easy and hard queries. Meanwhile, capability-calibrated confidence enables the evaluation of models on unlabeled test sets, which is called Label-free Benchmarking (Guha et al., 2024; Zhang et al., 2025b). We can derive model rankings that align with ground-truth evaluations.

D.4.1 The Latency Issue of RC

Beyond accuracy, RC creates a systemic latency issue. Because RC evaluates a specific generated response, the system must wait for the LLM to autoregressively decode the full answer before estimating confidence. We now discuss more details of the scenarios we mentioned in Section 4.3:

Model Routing: In model routing (Chen et al., 2024; Damani et al., 2025; Maurya et al., 2025), systems need to decide beforehand whether to invoke a cheap model or a powerful, expensive one. If the system must wait for the cheap model to fully decode a response just to extract a response-calibrated confidence score, the latency and compute have already been wasted. Therefore, Damani et al. (2025) uses lightweight probes to predict "expected accuracy" for model routing and budget allocation rather than predicting single-response correctness. Similarly, Maurya et al. (2025) trains a lightweight router to estimate the likelihood of a correct response by using training labels derived from the majority vote of 10 responses, which is a direct approximation of capability calibration. While neither work formally names the concept or links it to the broader confidence calibration or uncertainty quantification literature, both implicitly perform "query-level capability estimation" - a practically necessary step for model routing and budget allocation.

Active Query Refinement (Wu et al., 2024): Systems that rewrite vague prompts need to know if the model is capable of answering the query. Waiting for the model to generate a flawed answer before triggering refinement is highly inefficient. In contrast, the benchmarked CC method (linear probe) estimates the model's capability before generation, using only a single forward pass over the query.

E TARGETS DIFFERENCE

In this section, we compare the targets of response calibration (single-response correctness) and capability calibration (the expected accuracy of the output distribution) across three models and seven datasets. As illustrated in Figure 13, a clear divergence between these two targets persists across all evaluated scenarios.



Figure 13: **Divergence of calibration targets across three models and seven datasets.** The data show that the Response Calibration (RC) target $\mathcal{C}(x, \hat{y})$ and Capability Calibration (CC) targets $\mathbb{E}_{\hat{y}\sim f_\theta(\cdot|x)}[\mathcal{C}(x, \hat{y})]$ differs at each model-dataset pair.