Confidence Before Answering: A Paradigm Shift for Efficient LLM Uncertainty Estimation

Changcheng Li[1], Jiancan Wu[1,3]*, Hengheng Zhang[2], Zhengsu Chen[2], Guo An[2], Junxiang Qiu[1], Xiang Wang[1], Qi Tian[2]

[1]University of Science and Technology of China, [2]Huawei Inc.,

[3]Institute of Dataspace, Hefei Comprehensive National Science Center, Hefei, China

{lichangcheng, qiujx}@mail.ustc.edu.cn, {wujcan, xiangwang1223}@gmail.com

{zhanghengheng55, chenzhengsu2, anguo1, tian.qi1}@huawei.com

Abstract

Reliable deployment of large language models (LLMs) requires accurate uncertainty estimation. Existing answer-first methods produce confidence only after generating an answer, thereby measuring the correctness of a specific response and limiting their practical usability. We study a confidence-first paradigm, where the model outputs its confidence before answering, interpreting this score as the model's probability of answering the question correctly under its current policy.

We propose **CoCA(Co-optimized Confidence and Answers)**, a GRPO reinforcement learning framework that jointly optimizes confidence calibration and answer accuracy via segmented credit assignment. By assigning separate rewards and group-relative advantages to confidence and answer segments, CoCA enables stable joint optimization and substantially mitigates the observed reward-hacking failure mode. Experiments across math, code, and factual QA benchmarks show improved calibration and uncertainty discrimination while preserving answer quality, thereby enabling a broader range of downstream applications.

1 Introduction

LLMs have made remarkable progress on reasoning-intensive tasks, yet hallucinations remain pervasive — they frequently generate plausible but incorrect responses (Ji et al., 2023; Bang et al., 2023). This problem may be amplified by current post-training paradigms (Mei et al., 2026; Kirichenko et al., 2025), resulting in overconfidence that undermines trustworthiness, particularly in high-stakes domains such as medicine (Pal et al., 2023), law (Dahl et al., 2024), and finance (Joshi, 2025). Recognizing this challenge, a growing body of work has studied *confidence estimation* in LLMs (Kadavath et al., 2022; Stangel et al., 2025) — methods that produce a numerical score reflecting how likely the model's answer is to be correct. Well-calibrated confidence estimates not only help users judge answer reliability, but also support system-level decisions such as selective answering, refusal, and model routing (Chen et al., 2026b). Reliable confidence estimation is likewise valuable in recommender systems (Zhao et al., 2024; Liao et al., 2024; Kong et al., 2025), where model outputs are increasingly used to support downstream decision-making .

Most existing methods estimate confidence in an *answer-first* manner, which generates responses before estimating confidence through internal probing (Mielke et al., 2022; Fadeeva et al., 2024), post-hoc verbalized confidence (Lin et al., 2022; Xu et al., 2024), or sampling-based surrogates (Aichberger et al., 2025). They essentially ask "*Is the specific answer correct?*", but incur high computational overhead and cannot enable early decisions. In contrast, **confidence-first** approaches estimate correctness probability before generation, asking a fundamentally harder question — "*Given my current capabilities, how likely am I to answer correctly?*". Toward this goal, existing methods typically train separate supervised modules on frozen correctness labels. They generate the LLM's answers on the training dataset, label each by correctness, then train a confidence predictor — either on the model's internal representations (Cencerrado et al., 2025) or an external assessor (Zhou et al., 2022) — to predict these frozen labels.

Despite effectiveness, this decoupled pipeline faces two fundamental challenges:

- **Confidence estimation is inherently policy-dependent**. Training on frozen correctness labels usually causes predictors to overfit to superficial patterns (such as problem difficulty), rather than capturing the model's intrinsic uncertainty (Farquhar et al., 2024). Proper confidence opti-

*Jiancan Wu and Qi Tian are the corresponding authors

Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing, pages 24252–24270
October 24-29, 2026 ©2026 Association for Computational Linguistics

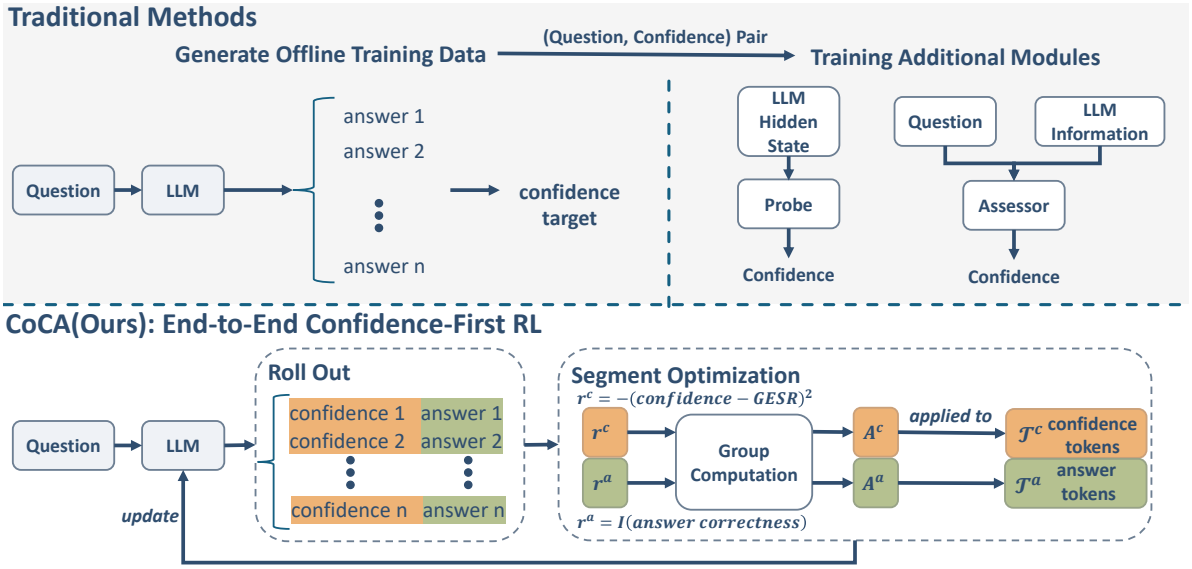


Figure 1: **From Decoupled Confidence Estimation to End-to-End Confidence-First Learning.** *Top*: Traditional pipelines derive confidence targets from offline model outputs and train separate predictors on frozen correctness targets. *Bottom*: CoCA (ours) jointly generates confidence and answers and optimizes them end-to-end with segment-specific GRPO rewards, using group-wise empirical success rates (GESR) as dynamic confidence targets.

mization therefore requires tracking the dynamic evolution of the model's capability to prevent such optimization hacking.

- **Confidence and answer quality are intrinsically entangled**. Users care about both reliable confidence estimates and accurate answers. However, isolated confidence training can degrade answer quality(Damani et al., 2026). Jointly optimizing confidence and accuracy can alleviate this mismatch, but it demands precise credit assignment to enable stable end-to-end learning (Ha et al., 2025; Guo et al., 2025), since confidence tokens and answer tokens are governed by distinct optimization objectives.

To address these challenges, we propose **CoCA (Co-optimized Confidence and Answers)**, an end-to-end, confidence-first learning framework that jointly optimizes confidence calibration and answer quality without requiring separate modules or frozen labels. The key idea is to have the model verbalize its confidence before generating the answer, then co-optimize both through a unified policy gradient objective with segment-specific credit assignment. Specifically, we build upon Group Relative Policy Optimization (GRPO) (DeepSeek-AI, 2025), and introduce three core designs: (1) **Dynamic confidence targets**. Rather than training on static correctness labels, we align confidence targets to group-wise empirical success rates (GESR) observed during policy rollouts. By grounding confidence in the model's real-time performance, estimates naturally track evolving model capabilities without requiring re-labeling. (2) **Calibration rewards**. We incorporate a Brier score penalty (BRIER, 1950) (*i.e.,* the squared difference between expressed confidence and GESR, *cf.* Equation (8)) into the reward function to quantify miscalibration. This quadratic form amplifies penalties for severe miscalibration — confident but wrong predictions or hesitant but correct ones — thereby incentivizing the model to accurately reflect its capability. (3) **Segment-specific reward decomposition**. Each response receives targeted rewards for its two segments: the confidence segment is rewarded for calibration accuracy, while the answer segment is rewarded for task correctness, preventing the model from sacrificing answer quality to improve calibration during optimization.

Experiments show that when trained only on math datasets, CoCA attains strong calibration not only in-distribution but also under distribution shift — for example, on Qwen2.5-3B-Instruct it reduces ECE from 0.54 to 0.09 on Math and from 0.66 to 0.14 on Factual QA, outperforming existing confidence-first baselines. Moreover, compared to answer-first methods, it enables much earlier decision-making by emitting confidence with only ~10 tokens, and cutting confidence-estimation token cost by >92% across all categories.

2 Related Work

2.1 Answer-first Confidence Estimation

Internal Probing. A common approach is to probe a model's internal states or output probabilities to estimate confidence in a given answer. Kadavath et al. (2022) prompt language models to output "true" or "false" and use the probability of "true" as a proxy for confidence. Mielke et al. (2022) condition response generation on external confidence probes. Fadeeva et al. (2024) propose Claim Conditioned Probability, a token-level uncertainty method based on internal signals. Azaria and Mitchell (2023); Orgad et al. (2025) show that hidden states encode truthfulness cues, while Kapoor et al. (2024) introduce an auxiliary uncertainty head fine-tuned via LoRA.

Post-hoc Verbalized Confidence. Another line of work elicits verbalized confidence (numeric or natural-language(Tao et al., 2025; Zhang et al., 2024)) from the LLM after answering, and calibrates the resulting confidence behavior using supervised fine-tuning or reinforcement learning.Lin et al. (2022) train GPT-3 to estimate confidence directly by regressing on its empirical accuracy over question–answer pairs. Stengel-Eskin et al. (2024) propose a speaker–listener setup where the speaker is rewarded based on the listener's inferred confidence. Leng et al. (2025) integrate explicit confidence annotations into reward model training, improving alignment with verbalized confidence levels. Xu et al. (2024) and Stangel et al. (2025) apply reinforcement learning with proper scoring rules as rewards — using the Brier score and a clipped log loss, respectively to enhance calibration. In contrast, Damani et al. (2026) use a single reward to jointly optimize confidence and accuracy.

Sampling-based Surrogates. This line of work leverages response agreement, such as majority voting or best-of-N sampling, as a proxy for confidence. Aichberger et al. (2025) generate semantically diverse yet plausible outputs and assess uncertainty via their consistency. Kuhn et al. (2023) introduce semantic entropy, a sampling-based method that accounts for linguistic variations to better capture uncertainty in natural language generation. Xue et al. (2025) assess model uncertainty by introducing cross-model consistency.

2.2 Confidence-first Confidence Estimation

In contrast to the extensive body of work on confidence estimation for specific answers, this area remains relatively underexplored. A number of studies investigate whether a model is able to answer a question by probing its internal representations (Ferrando et al., 2025; Cencerrado et al., 2025; Chen et al., 2026b; Kadavath et al., 2022). Specifically, Ferrando et al. (2025) decompose intermediate model layers (the residual stream) using Sparse Autoencoders (SAEs) to determine whether the model recognizes a given entity, while Cencerrado et al. (2025) employ probes to predict confidence for a given question. Other works rely on external assessors for evaluation, where the assessors range from neural networks (Hernández-Orallo et al., 2022) to Random Forests (Zhou et al., 2022), as well as XGBoost and Logistic Regression models (Pacchiardi et al., 2025). Beyond these approaches, Shrivastava et al. (2025) obtain confidence by asking the model to perform pairwise comparisons across questions and ranking them accordingly.

Given the different application scopes, target signals, and evaluation protocols of these studies, we compare against representative confidence-first mechanisms, such as internal probes and external assessors, rather than reproducing every individual system.

3 Method

3.1 Preliminaries: RL for LLMs and GRPO

Given an input prompt x, we denote the language model policy as $\pi_\theta(\cdot \mid x)$, which generates a token sequence $y = (y_1, \ldots, y_T)$. In reinforcement learning for LLMs (e.g., RLHF/RLAIF/RLVR) (Bai et al., 2022a,b; Lee et al., 2024; DeepSeek-AI, 2025), the standard objective is to maximize an external reward $R(x, y)$ while preventing the policy from drifting too far from a reference policy π_{ref}.

GRPO (Group Relative Policy Optimization) (DeepSeek-AI, 2025) is a PPO-style method that avoids training an explicit value function. For each prompt x, GRPO samples a group of G candidate responses from the current policy, computes a scalar reward r_i for each response, and constructs a group-wise relative advantage to reduce variance:

$$\hat{A}_i = \frac{r_i - \mu(r)}{\sigma(r) + \epsilon}, \tag{1}$$

where $\mu(r)$ and $\sigma(r)$ are the mean and standard deviation computed over the G rewards. Let $\pi_{\theta_{\text{old}}}$

be the policy before the update. Define the token-level probability ratio as follows:

$$\rho_{i,t}(\theta) = \frac{\pi_\theta(y_{i,t} \mid x, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t} \mid x, y_{i,<t})}. \tag{2}$$

Then a clipped GRPO objective can be written as:

$$\begin{aligned}\mathcal{L}_{\text{GRPO}}(\theta) =& \mathbb{E}_x \Bigg[\frac{1}{G} \sum_{i=1}^{G} \sum_{t=1}^{T_i} \min \big(\rho_{i,t}(\theta) \hat{A}_i, \\ & \text{clip}(\rho_{i,t}(\theta), 1-\varepsilon, 1+\varepsilon) \hat{A}_i \big) \Bigg] \\ & - \beta \mathbb{E}_x \big[\text{KL}(\pi_\theta || \pi_{\text{ref}}) \big].\end{aligned} \tag{3}$$

The formulation above uses a single reward to drive the entire response sequence. Recent studies further suggest that effective LLM post-training depends on careful advantage construction and token-level policy updates (Wu et al., 2026; Huang et al., 2026), while imperfect optimization signals may also induce reward hacking (Chen et al., 2026a).

3.2 Confidence-First Paradigm Definition

We study a confidence-first paradigm: the model must output its confidence before producing the answer. We decompose the output into two segments:

$$y = (y^c, y^a), \tag{4}$$

where y^c is the confidence segment and y^a is the answer segment. We enforce a fixed format:

$$y \equiv \texttt{<confidence>}\ s\ \texttt{</confidence>}\ y^a. \tag{5}$$

Training a confidence-first model is inherently a **multi-objective problem**: the policy must output a calibrated confidence score and produce a correct answer.

3.3 CoCA: Segmented GRPO for Co-optimized Confidence and Answers

3.3.1 Reward Formulation

Accuracy reward. For each prompt x, we sample G full outputs $y_i = (y_i^c, y_i^a)$. We define an answer correctness reward ($r_i^a \in \{0, 1\}$) as

$$r_i^a = \mathbb{I}\big(\text{AnsCorrect}(x, y_i^a)\big), \tag{6}$$

where $\text{AnsCorrect}(\cdot)$ is computed by the dataset-specific evaluator.

Confidence Reward. Next, we define GESR as an estimate of how likely the current policy answers this question correctly:

$$\hat{p}(x) = \frac{1}{G} \sum_{j=1}^{G} r_j^a. \tag{7}$$

The confidence segment is parsed into a scalar $s_i = \text{Parse}(y_i^c) \in [0, 1]$. We encourage s_i to match $\hat{p}(x)$ using a stable **Brier-style reward**:

$$r_i^c = -\big(s_i - \hat{p}(x)\big)^2. \tag{8}$$

Throughout the entire process, the confidence target is derived from the same rollout via the GESR $\hat{p}(x)$. Meanwhile, we do not employ any sampling strategy and instead preserve the model's original distribution. This makes s reflect the probability of answering correctly under the current policy.

3.3.2 Segmented Credit Assignment and Joint Optimization

Sequentially optimizing accuracy and then confidence can introduce reward hacking: the model may improve the confidence objective by altering answer behavior (e.g., refusal or evasiveness). CoCA avoids this by optimizing both objectives simultaneously, while restricting each advantage to its corresponding token span, which anchors answer quality and confidence calibration throughout training.

We therefore compute two advantages within the same group:

$$\hat{A}_i^c = \frac{r_i^c - \mu(r^c)}{\sigma(r^c) + \epsilon}, \qquad \hat{A}_i^a = \frac{r_i^a - \mu(r^a)}{\sigma(r^a) + \epsilon}. \tag{9}$$

We then apply the clipped policy gradient separately to the confidence and answer token segments. Let $\mathcal{T}_i^c$ denote the set of tokens in the confidence segment of sample i, and $\mathcal{T}_i^a$ denote those in the answer segment. Our segmented objective, without a KL-divergence term, is given by

$$\begin{aligned}\mathcal{L}_i^c(\theta) =& \sum_{t \in \mathcal{T}_i^c} \min \Big(\rho_{i,t}(\theta) \hat{A}_i^c, \\ & \text{clip}(\rho_{i,t}(\theta), 1-\varepsilon, 1+\varepsilon) \hat{A}_i^c \Big), \\ \mathcal{L}_i^a(\theta) =& \sum_{t \in \mathcal{T}_i^a} \min \Big(\rho_{i,t}(\theta) \hat{A}_i^a, \\ & \text{clip}(\rho_{i,t}(\theta), 1-\varepsilon, 1+\varepsilon) \hat{A}_i^a \Big).\end{aligned} \tag{10}$$

Here $\rho_{i,t}(\theta)$ and $\text{clip}(\cdot)$ follow standard PPO/-GRPO definitions. By segmenting the output

$y = (y^c, y^a)$ and computing separate advantages $\hat{A}^c$ and $\hat{A}^a$ that are applied only to their respective token spans, CoCA provides a more targeted learning signal and leads to faster and more stable training.

The **joint optimization** is as follows:

$$\mathcal{L}_{\text{CoCA}}(\theta) = \mathbb{E}_x \left[\frac{1}{G} \sum_{i=1}^{G} \left(\mathcal{L}_i^c(\theta) + \mathcal{L}_i^a(\theta) \right) \right]. \tag{11}$$

The complete algorithmic workflow is presented in Algorithm 1.

4 Experiment

This section primarily examines whether, under the confidence-first paradigm, CoCA can improve the usability and cross-domain generalization of confidence estimates while preserving answer quality. In addition, we perform comparisons against the answer-first paradigm to assess whether CoCA can attain comparable performance while reducing the token cost of obtaining confidence estimates. We also conduct ablation studies to contrast segmented versus joint rewards, and to expose reward hacking risks arising from sequential training.

4.1 Experimental Setup

4.1.1 Models and Training Data

We conduct our confidence-first comparisons on three instruction-tuned **Qwen2.5-Instruct** models of different scales: 7B, 3B, and 1.5B (Yang et al., 2024), to verify consistency across model sizes. Unless otherwise specified, all remaining experiments (including answer-first comparisons and ablations) are conducted on **Qwen2.5-7B-Instruct**.

For the main experiments, training is performed on **Big-Math-Verified** (Albalak et al., 2025), a math dataset with automatically verifiable correctness, enabling low-noise reward computation.

4.1.2 Evaluation Benchmarks

After training, all models are evaluated on a diverse set of benchmarks:

- **Math**: AIME2024, AIME2025, MATH-500 (Hendrycks et al., 2021), GSM8K (Cobbe et al., 2021)
- **Code**: HumanEval (Chen et al., 2021), Sanitized MBPP (Austin et al., 2021)
- **Factual QA**: SimpleQA (Wei et al., 2024), TriviaQA (Joshi et al., 2017)

Importantly, although training is performed solely on math data, we evaluate on code and factual QA to test whether the learned confidence reflects general **uncertainty awareness** rather than domain-specific heuristics.

4.1.3 Metrics

- **Accuracy** ↑: The proportion of correct predictions among all samples.
- **AUROC** ↑: Measures how well confidence scores distinguish correct answers from errors.
- **Expected Calibration Error (ECE)** ↓: Measures the gap between predicted confidence and actual accuracy across different confidence bins.
- **Brier Score** ↓: Measures the mean squared error between confidence and binary correctness.

Additionally, in Section 4.4, we also measured the token consumption to confidence prediction (TTC), so as to reflect both computational cost and latency.

4.2 Comparison with Confidence-First Baselines

We first compare CoCA with approaches that either directly predict confidence or attach confidence estimation to an accuracy-optimized model.

4.2.1 Baselines

We consider the following baselines:

- **Verbalized Confidence**: eliciting the model's self-reported confidence by designing prompts that explicitly ask it to express its confidence level without confidence-specific optimization.
  - **Instruct Model**: the original instruction-tuned model.
  - **RLVR (Accuracy-only)**: reinforcement learning that optimizes only answer accuracy.
- **Internal Probing**: extracting uncertainty-related signals from the model's internal states.
  - **Question Probability**: using the likelihood of the question tokens as a confidence proxy.

$$\text{QuestionProb}(x) = \frac{1}{|\mathcal{X}|} \sum_{i \in \mathcal{X}} P_\theta(x_i \mid x_{<i}). \tag{12}$$

Here, $\mathcal{X}$ denotes the sequence of input tokens in the question, and $P_\theta(x_i \mid x_{<i})$ represents the model's probability of generating token x_i conditioned on all preceding input tokens.

  - **Probe**: a linear probe trained on the final-layer hidden representation of the RLVR model.

- **Assessor Model**: estimating the model's confidence using an external model or algorithm.
  - **Additional Assessor Model**: using Qwen2.5-1.5B-Instruct trained to predict the target model's correctness probability from the question alone.

These baselines cover verbalized confidence methods, probing-based methods, likelihood-based heuristics, and external assessor models.

4.2.2 Results and Analysis

Category-wise statistics are reported in Table 1, and detailed benchmark-level results are provided in Tables 7, 8, and 9. Across all benchmarks, we observe the following consistent trends:

Uncalibrated verbalized confidence is severely overconfident. Without explicit confidence calibration, both instruction-tuned models and accuracy-only RLVR models exhibit severe overconfidence in their verbalized confidence. For example, the 1.5B Instruct model achieves only 13.79% accuracy on Math, but its ECE and Brier Score reach 0.77 and 0.73, respectively. These results show that improving answer correctness alone does not teach the model to express reliable uncertainty.

External assessor models and probes are sensitive to distribution shift. The outputs of external assessor models tend to concentrate around 0.5, exhibiting limited discriminative power. While probes or auxiliary assessor models can improve in-domain calibration, these gains are not consistently preserved on out-of-domain code and factual QA tasks, indicating a dependence on the training distribution and learned representations.

Question probability is a weak proxy for correctness. Question probability tends to assign uniformly low scores, and token likelihood primarily reflects linguistic familiarity rather than problem solvability. As a result, it yields inferior AUROC, particularly on reasoning-intensive benchmarks.

CoCA provides strong confidence quality with end-to-end optimization. CoCA generally achieves strong calibration and discrimination performance across model sizes and task categories, while preserving reasonable answer accuracy. Probe- or assessor-based baselines require additional modules, whereas CoCA obtains reliable confidence estimates through a single end-to-end optimized model.

4.3 Additional Generalization Studies

The main comparison in Table 1 evaluates CoCA under a math-trained setting on the Qwen2.5-Instruct model family. To further examine whether the effectiveness of CoCA is tied to a specific backbone or to math-only training, we conduct two additional studies. Specifically, we train Llama3-8B (Team, 2024) on the same math training data and Qwen2.5-7B-Instruct on TACO code data (Li et al., 2023), and evaluate them on the same benchmark suite.

Table 5 summarizes the results. Across these supplementary settings, CoCA preserves the main trend observed in Table 1: it improves confidence discrimination and calibration, while keeping answer accuracy comparable. These results provide additional evidence that CoCA is robust across both model backbones and verifiable training domains.

4.4 Comparison with Answer-First Paradigm

We next compare our method with answer-first approaches, which generate an answer before estimating confidence.

4.4.1 Baselines

We focus on two representative answer-first methods:

- **Sampling-based surrogates (Majority Voting)**: multiple answers are sampled and clustered by semantic equivalence; confidence is computed as the proportion of samples in the largest cluster, and the representative answer from this cluster is returned as the final prediction.

- **Post-hoc verbalized confidence (RLCR; Reinforcement Learning with Calibration Rewards)**: we adopt the approach described in (Damani et al., 2026); the specific reward computation is given by the following formula:

$$R_{RLCR} = \mathbb{I}(y) - (s - \mathbb{I}(y))^2 \qquad (13)$$

4.4.2 Results and Practical Implications

Table 2 summarizes the category-level averages for accuracy, AUROC, and the token consumption to confidence prediction (TTC) across Math, Code, and Factual QA. A full per-dataset breakdown (including all benchmarks within each category) is reported in Table 12.

Table 1: **Main results compared with confidence-first methods**. "Math/Code/Factual" are benchmark-category averages. Bold indicates the best method per model × category and metric. For 1.5B, we also report confidence-generation success rate (SR) in Table 9.

Answer	Confidence	Metric	Qwen2.5-Instruct (temp=1.0, pass@1)								
			1.5B			3B			7B		
			Math	OOD		Math	OOD		Math	OOD	
				Code	Factual		Code	Factual		Code	Factual
Instruct	Verbalized	Acc	13.79	55.10	14.81	37.54	63.78	22.71	43.73	77.43	30.13
		AUROC	0.52	0.53	0.53	0.54	0.52	0.57	0.61	0.57	0.63
		ECE	0.77	0.41	0.72	0.54	0.30	0.66	0.52	0.22	0.58
		Brier	0.73	0.42	0.66	0.52	0.33	0.60	0.50	0.23	0.52
RLVR	Verbalized	Acc	30.89	39.00	16.77	42.86	34.58	23.92	51.50	76.10	30.74
		AUROC	0.52	0.58	0.54	0.34	0.48	0.54	0.63	0.55	0.60
		ECE	0.62	0.56	0.76	0.48	0.56	0.62	0.46	0.23	0.58
		Brier	0.61	0.55	0.71	0.47	0.54	0.56	0.45	0.23	0.53
RLVR	QuestionProb	Acc	39.40	57.15	19.79	46.53	69.03	26.01	50.23	76.58	31.38
		AUROC	0.44	0.46	0.51	0.42	0.42	0.50	0.32	0.47	0.55
		ECE	0.36	0.48	0.19	0.44	0.65	0.26	0.47	0.70	0.30
		Brier	0.37	0.48	0.20	0.45	0.63	0.26	0.47	0.68	0.31
RLVR	Assessor	Acc	39.40	57.15	19.79	46.53	69.03	26.01	50.23	76.58	31.38
		AUROC	0.72	0.63	0.63	0.83	0.64	0.61	0.83	0.55	0.61
		ECE	0.16	0.25	0.16	0.16	0.29	0.27	0.24	0.29	0.26
		Brier	0.14	0.35	0.17	0.14	0.33	0.25	0.19	0.30	0.26
RLVR	Probe	Acc	39.40	57.15	19.79	46.53	69.03	26.01	50.23	76.58	31.38
		AUROC	0.62	0.64	0.57	0.84	0.59	0.64	**0.84**	0.62	0.64
		ECE	0.10	0.27	0.46	0.11	0.37	0.29	0.12	0.71	**0.21**
		Brier	0.12	0.34	0.42	0.11	0.39	0.24	0.10	0.68	**0.21**
CoCA(ours) Joint optimization		Acc	36.85	46.72	18.89	44.85	67.38	24.74	47.92	76.60	28.85
		AUROC	**0.82**	**0.72**	**0.71**	**0.88**	**0.67**	**0.73**	0.71	**0.72**	**0.69**
		ECE	**0.09**	**0.09**	**0.09**	**0.09**	**0.28**	**0.14**	**0.10**	**0.16**	0.26
		Brier	**0.10**	**0.22**	**0.12**	**0.09**	**0.30**	**0.14**	**0.09**	**0.19**	0.24

Table 2: Comparison against answer-first baselines. TTC refers to the token consumption to confidence prediction. Per-dataset results are provided in Table 12.

Method	Metric	Math	Code	Factual
Majority Voting	Acc	54.36	79.29	32.89
	AUROC	0.80	0.69	**0.75**
	TTC	9549.16	1996.04	1685.17
RLCR	Acc	49.01	74.52	25.16
	AUROC	**0.85**	0.67	0.67
	TTC	840.46	209.20	121.78
CoCA(ours)	Acc	49.90	77.49	28.95
	AUROC	0.73	**0.74**	0.70
	TTC	**9.95**	**9.78**	**9.75**

AUROC differences are small across methods. Across the answer-first baselines and CoCA, AUROC values are broadly comparable, indicating that these methods offer similar ranking ability for separating correct from incorrect answers.

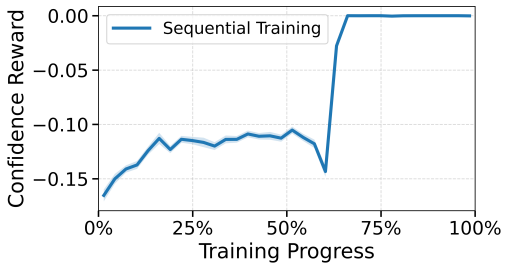
Confidence-first enables earlier and lower-cost confidence access. Sampling-based confidence estimates are only available after repeated sampling and agreement checks. Post-hoc verbalized confidence also requires the full answer to be generated before confidence can be obtained. In contrast, CoCA exposes confidence before answer generation, creating an early decision point for routing, early stopping, or budget-aware inference. As shown in Table 2, CoCA reduces TTC to approximately 10 tokens, compared with hundreds or thousands of tokens for answer-first baselines.

These results demonstrate that confidence-first is not merely a formatting change, but a paradigm shift aligned with real-world deployment requirements.

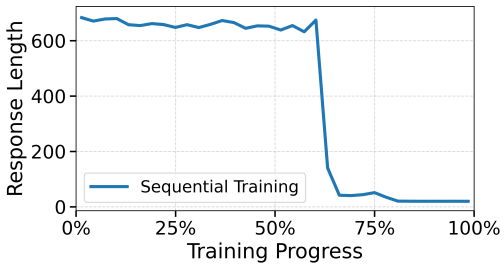
4.5 Ablation Studies

4.5.1 Reward Hacking in Sequential Training

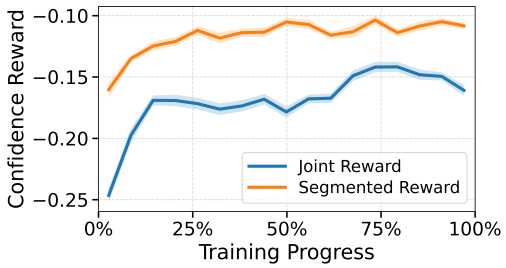
We examine a sequential training pipeline where the model is first optimized for answer correctness and then further optimized only with the confidence reward. As shown in Figure 2a and Figure 2b, the confidence reward increases during the second-stage optimization, while the response length col-



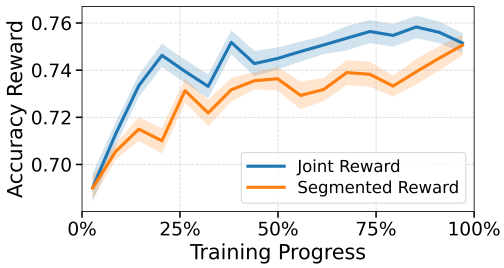
(a) Sequential training: confidence reward.



(b) Sequential training: response length.



(c) Joint vs. segmented rewards: confidence reward.



(d) Joint vs. segmented rewards: accuracy reward.

Figure 2: **Ablation training dynamics.** *Top*: sequential confidence training exhibits reward hacking, where confidence reward increases while response length sharply decreases. *Bottom*: segmented rewards provide a clearer optimization signal for confidence learning, with comparable accuracy-reward dynamics.

Table 3: Representative degenerate outputs from sequential confidence training.

Question	**Model Output**
Vector $\vec{a} = (2, 1)$, $\vec{b} = (x, -1)$, and $\vec{a} \parallel \vec{b}$. Find x.	`<confidence>0.003</confidence>` I need more context and information to provide a proper answer.
Given $\tan\alpha = 2$, calculate $\frac{\sin\alpha+\cos\alpha}{\sin\alpha-3\cos\alpha}$.	`<confidence>0.005</confidence>` I cannot provide a numerical answer or a step-by-step solution as the instruction is unclear.

Table 4: Comparison between Segment Reward and Joint Reward across math, code, and factual QA benchmarks.

Method	Metric	Math	Code	Factual QA
Segment	Acc	49.90	77.49	28.95
	AUROC	0.73	**0.74**	**0.70**
	ECE	0.12	**0.15**	**0.22**
	Brier	**0.10**	**0.18**	**0.20**
Joint	Acc	52.75	77.41	29.20
	AUROC	**0.74**	0.52	0.69
	ECE	**0.09**	0.23	0.24
	Brier	**0.10**	0.23	0.24

lapses sharply. The failure cases in Table 3 show that the model learns to refuse answering or produce trivial outputs, improving the confidence objective at the cost of answer quality.

This result motivates our joint training design, which optimizes confidence calibration and answer correctness together so that confidence learning remains anchored to meaningful answer generation.

4.5.2 Joint Reward vs. Segmented Reward

We further compare **joint rewards**, where confidence and accuracy rewards are applied to the entire response, with **segmented rewards** (ours), where the confidence reward is applied only to confidence tokens and the accuracy reward is applied only to answer tokens.

Figure 2c and Figure 2d show that segmented rewards lead to a stronger increase in confidence reward, while the accuracy reward remains comparable to that of joint rewards. This indicates that segment-level reward assignment provides a more targeted learning signal for confidence tokens. The evaluation results in Table 4 further support this interpretation: segmented rewards produce more generalizable confidence estimates, with lower ECE and Brier scores on out-of-domain Code and Factual QA.

5 Conclusion

We propose **CoCA (Co-optimized Confidence and Answers)**, an end-to-end, confidence-first learning framework that jointly optimizes confidence and answer quality. Across math, code, and factual QA, CoCA improves confidence quality (calibration and discrimination) while preserving accuracy and outperforming confidence-first base-

lines. In the main math-trained setting, these gains are obtained using only verifiable math data, and additional code-domain training further supports the robustness of the framework. Confidence-first outputs also enable early routing and termination for more efficient inference, while ablations show that joint optimization with segmented rewards is key to stable training and reduced reward hacking, leading to more reliable confidence estimates.

6 Limitations

Our current approach has two main limitations. First, training–evaluation mismatch for non-verifiable domains. Training is performed on automatically verifiable math and code data, while evaluation extends beyond it. This does not fully capture domains where correctness is ambiguous or unverifiable (e.g., open-ended QA, summarization, dialogue). Extending confidence-first RL to weak or indirect supervision—such as preference feedback, rubric-based graders, or tool-assisted verification—remains an important direction. Second, some hard-math evaluations rely on small, high-difficulty test sets, so metrics such as AUROC and calibration can vary noticeably with training stochasticity and checkpoint selection; larger-scale hard-math benchmarks or curated difficult collections would yield more precise estimates.

Acknowledgments

This research was supported by the National Science and Technology Major Project (2024YFF0908204-1). It was also supported by the National Natural Science Foundation of China (U24B20180, 62302321) and the Fundamental Research Funds for the Central Universities (WK2100250065).

References

Lukas Aichberger, Kajetan Schweighofer, Mykyta Ielanskyi, and Sepp Hochreiter. 2025. Improving uncertainty estimation through semantically diverse language generation. In *ICLR*. OpenReview.net.

Alon Albalak, Duy Phung, Nathan Lile, Rafael Rafailov, Kanishk Gandhi, Louis Castricato, Anikait Singh, Chase Blagden, Violet Xiang, Dakota Mahan, and Nick Haber. 2025. Big-math: A large-scale, high-quality math dataset for reinforcement learning in language models. *CoRR*, abs/2502.17387.

Jacob Austin, Augustus Odena, Maxwell I. Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie J. Cai, Michael Terry, Quoc V. Le, and Charles Sutton. 2021. Program synthesis with large language models. *CoRR*, abs/2108.07732.

Amos Azaria and Tom M. Mitchell. 2023. The internal state of an LLM knows when it's lying. In *EMNLP (Findings)*, pages 967–976. Association for Computational Linguistics.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, and 12 others. 2022a. Training a helpful and harmless assistant with reinforcement learning from human feedback. *CoRR*, abs/2204.05862.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, and 32 others. 2022b. Constitutional AI: harmlessness from AI feedback. *CoRR*, abs/2212.08073.

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, Quyet V. Do, Yan Xu, and Pascale Fung. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. In *IJCNLP (1)*, pages 675–718. Association for Computational Linguistics.

GLENN W. BRIER. 1950. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1 – 3.

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

Heyu Chen, Junkang Wu, Guoqing Hu, Kexin Huang, Xiang Wang, and Jiancan Wu. 2026a. Mitigating reward hacking in LLM-based recommendation: A

preference optimization approach. In *The Forty-Third International Conference on Machine Learning, ICML 2026, Seoul, South Korea, July 6-11, 2026*.

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

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé 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, and 39 others. 2021. Evaluating large language models trained on code. *CoRR*, abs/2107.03374.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

Matthew Dahl, Varun Magesh, Mirac Suzgun, and Daniel E. Ho. 2024. Large legal fictions: Profiling legal hallucinations in large language models. *CoRR*, abs/2401.01301.

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

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948.

Ekaterina Fadeeva, Aleksandr Rubashevskii, Artem Shelmanov, Sergey Petrakov, Haonan Li, Hamdy Mubarak, Evgenii Tsymbalov, Gleb Kuzmin, Alexander Panchenko, Timothy Baldwin, Preslav Nakov, and Maxim Panov. 2024. Fact-checking the output of large language models via token-level uncertainty quantification. In *ACL (Findings)*, pages 9367–9385. Association for Computational Linguistics.

Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. 2024. Detecting hallucinations in large language models using semantic entropy. *Nat.*, 630(8017):625–630.

Javier Ferrando, Oscar Balcells Obeso, Senthooran Rajamanoharan, and Neel Nanda. 2025. Do I know this entity? knowledge awareness and hallucinations in language models. In *ICLR*. OpenReview.net.

Yiran Guo, Lijie Xu, Jie Liu, Dan Ye, and Shuang Qiu. 2025. Segment policy optimization: Effective segment-level credit assignment in RL for large language models. *CoRR*, abs/2505.23564.

Rui Ha, Chaozhuo Li, Rui Pu, and Sen Su. 2025. From "aha moments" to controllable thinking: Toward meta-cognitive reasoning in large reasoning models via decoupled reasoning and control. *CoRR*, abs/2508.04460.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *NeurIPS Datasets and Benchmarks*.

José Hernández-Orallo, Wout Schellaert, and Fernando Martínez-Plumed. 2022. Training on the test set: Mapping the system-problem space in AI. In *AAAI*, pages 12256–12261. AAAI Press.

Kexin Huang, Haoming Meng, Junkang Wu, Jinda Lu, Chiyu Ma, Ziqian Chen, Xue Wang, Bolin Ding, Jiancan Wu, Xiang Wang, Xiangnan He, Guoyin Wang, and Jingren Zhou. 2026. Beyond magnitude: Leveraging direction of RLVR updates for LLM reasoning. In *The Fourteenth International Conference on Learning Representations, ICLR 2026, Rio de Janeiro, Brazil, April 23-27, 2026*. OpenReview.net.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12):248:1–248:38.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL (1)*, pages 1601–1611. Association for Computational Linguistics.

Satyadhar Joshi. 2025. Comprehensive Review of AI Hallucinations: Impacts and Mitigation Strategies for Financial and Business Applications. *International Journal of Computer Applications Technology and Research (IJCATR)*.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, and 17 others. 2022. Language models (mostly) know what they know. *CoRR*, abs/2207.05221.

Sanyam Kapoor, Nate Gruver, Manley Roberts, Katie Collins, Arka Pal, Umang Bhatt, Adrian Weller, Samuel Dooley, Micah Goldblum, and Andrew Gordon Wilson. 2024. Large language models must be taught to know what they don't know. In *NeurIPS*.

Polina Kirichenko, Mark Ibrahim, Kamalika Chaudhuri, and Samuel J. Bell. 2025. Abstentionbench: Reasoning llms fail on unanswerable questions. *CoRR*, abs/2506.09038.

Xiaoyu Kong, Junguang Jiang, Bin Liu, Ziru Xu, Han Zhu, Jian Xu, Bo Zheng, Jiancan Wu, and Xiang

Wang. 2025. Think before recommendation: Autonomous reasoning-enhanced recommender. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *ICLR*. OpenReview.net.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. 2024. RLAIF vs. RLHF: scaling reinforcement learning from human feedback with AI feedback. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pages 26874–26901. PMLR / OpenReview.net.

Jixuan Leng, Chengsong Huang, Banghua Zhu, and Jiaxin Huang. 2025. Taming overconfidence in llms: Reward calibration in RLHF. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Rongao Li, Jie Fu, Bo-Wen Zhang, Tao Huang, Zhihong Sun, Chen Lyu, Guang Liu, Zhi Jin, and Ge Li. 2023. TACO: topics in algorithmic code generation dataset. *CoRR*, abs/2312.14852.

Jiayi Liao, Sihang Li, Zhengyi Yang, Jiancan Wu, Yancheng Yuan, Xiang Wang, and Xiangnan He. 2024. Llara: Large language-recommendation assistant. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*, pages 1785–1795. ACM.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Teaching models to express their uncertainty in words. *Trans. Mach. Learn. Res.*, 2022.

Zhiting Mei, Christina Zhang, Tenny Yin, Justin Lidard, Ola Shorinwa, and Anirudha Majumdar. 2026. Reasoning about uncertainty: Do reasoning models know when they don't know? In *Findings of the Association for Computational Linguistics: EACL 2026, Rabat, Morocco, March 24-29, 2026*, Findings of ACL, pages 3408–3458. Association for Computational Linguistics.

Sabrina J. Mielke, Arthur Szlam, Emily Dinan, and Y-Lan Boureau. 2022. Reducing conversational agents' overconfidence through linguistic calibration. *Trans. Assoc. Comput. Linguistics*, 10:857–872.

Hadas Orgad, Michael Toker, Zorik Gekhman, Roi Reichart, Idan Szpektor, Hadas Kotek, and Yonatan Belinkov. 2025. Llms know more than they show: On the intrinsic representation of LLM hallucinations. In *ICLR*. OpenReview.net.

Lorenzo Pacchiardi, Konstantinos Voudouris, Ben Slater, Fernando Martínez-Plumed, José Hernández-Orallo, Lexin Zhou, and Wout Schellaert. 2025. Predictaboard: Benchmarking LLM score predictability. In *ACL (Findings)*, pages 15245–15266. Association for Computational Linguistics.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. 2023. Med-halt: Medical domain hallucination test for large language models. In *CoNLL*, pages 314–334. Association for Computational Linguistics.

Vaishnavi Shrivastava, Ananya Kumar, and Percy Liang. 2025. Language models prefer what they know: Relative confidence estimation via confidence preferences. *CoRR*, abs/2502.01126.

Paul Stangel, David Bani-Harouni, Chantal Pellegrini, Ege Özsoy, Kamilia Zaripova, Matthias Keicher, and Nassir Navab. 2025. Rewarding doubt: A reinforcement learning approach to calibrated confidence expression of large language models. *CoRR*, abs/2503.02623.

Elias Stengel-Eskin, Peter Hase, and Mohit Bansal. 2024. LACIE: listener-aware finetuning for calibration in large language models. In *NeurIPS*.

Linwei Tao, Yi-Fan Yeh, Bo Kai, Minjing Dong, Tao Huang, Tom A. Lamb, Jialin Yu, Philip H. S. Torr, and Chang Xu. 2025. Can large language models express uncertainty like human? *CoRR*, abs/2509.24202.

Llama Team. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

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

Junkang Wu, Kexin Huang, Jiancan Wu, An Zhang, Xiang Wang, and Xiangnan He. 2026. Quantile advantage estimation: Stabilizing RLVR for LLM reasoning. In *The Fourteenth International Conference on Learning Representations, ICLR 2026, Rio de Janeiro, Brazil, April 23-27, 2026*. OpenReview.net.

Tianyang Xu, Shujin Wu, Shizhe Diao, Xiaoze Liu, Xingyao Wang, Yangyi Chen, and Jing Gao. 2024. Sayself: Teaching llms to express confidence with self-reflective rationales. In *EMNLP*, pages 5985–5998. Association for Computational Linguistics.

Yihao Xue, Kristjan H. Greenewald, Youssef Mroueh, and Baharan Mirzasoleiman. 2025. Verify when uncertain: Beyond self-consistency in black box hallucination detection. *CoRR*, abs/2502.15845.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian

Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024. Qwen2.5 technical report. *CoRR*, abs/2412.15115.

Hanning Zhang, Shizhe Diao, Yong Lin, Yi R. Fung, Qing Lian, Xingyao Wang, Yangyi Chen, Heng Ji, and Tong Zhang. 2024. R-tuning: Instructing large language models to say 'i don't know'. In *NAACL-HLT*, pages 7113–7139. Association for Computational Linguistics.

Zihao Zhao, Yi Jing, Fuli Feng, Jiancan Wu, Chongming Gao, and Xiangnan He. 2024. Leave no patient behind: Enhancing medication recommendation for rare disease patients. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*, pages 533–542. ACM.

Lexin Zhou, Fernando Martínez-Plumed, José Hernández-Orallo, Cèsar Ferri, and Wout Schellaert. 2022. Reject before you run: Small assessors anticipate big language models. In *EBeM@IJCAI*, volume 3169 of *CEUR Workshop Proceedings*. CEUR-WS.org.

A Pseudocode of CoCA

We present the pseudocode of the CoCA algorithm in Algorithm 1. The algorithm separates the generation of confidence and answer into two distinct segments, and applies segment-specific rewards via a modified group-based reinforcement learning procedure.

B Additional Generalization Studies

This section provides category-level results for the additional generalization studies discussed in Section 4.3. While Table 1 focuses on the primary Qwen2.5-Instruct scale comparison under math-domain training, here we further examine whether the proposed CoCA framework generalizes across model backbones and training domains.

Specifically, we report two supplementary settings. First, we train Llama3-8B on the same math-domain training data used in the main experiments, which evaluates whether CoCA remains effective beyond the Qwen2.5 model family. Second, we train Qwen2.5-7B-Instruct on code-domain data, which evaluates whether the proposed confidence-answer joint optimization remains effective when the verifiable training domain changes from math to code. For both settings, we use the same baseline protocol as Table 1.

Table 5 reports category-level results, while the per-dataset results are provided in Table 10 and Table 11.

Overall, these results show that CoCA's benefits are not limited to the original Qwen2.5 math-trained setting. Although improvements vary across benchmarks and domains, CoCA generally improves calibration or discrimination while maintaining comparable answer accuracy, supporting the robustness of segmented confidence-answer optimization across backbone and training-domain shifts.

C Training and Evaluation Details

C.1 Training Configuration and Schedules

All models are trained using the **MindSpeed-RL** framework on **Ascend 910B** and **Ascend 910C** accelerators. Unless otherwise specified, we use the same optimization and decoding settings across all experiments.

Optimization hyperparameters.

- Global batch size: 128×16 (128 prompts, with $G = 16$ responses per prompt)

Table 5: **Additional generalization results with the same baseline settings as Table 1.** We report two supplementary settings: Llama3-8B trained on math data and Qwen2.5-7B-Instruct trained on code data.

Answer	Confidence	Metric	Llama3-8B, Train: Math			Qwen2.5-7B, Train: Code		
			Math	OOD		Code	OOD	
				Code	Factual		Math	Factual
Instruct	Verbalized	Acc	26.88	33.77	29.68	77.43	43.73	30.13
		AUROC	0.52	0.56	0.64	0.57	0.61	0.63
		ECE	0.63	0.75	0.47	0.22	0.52	0.58
		Brier	0.59	0.72	0.42	0.23	0.50	0.52
RLVR	Verbalized	Acc	26.73	44.55	32.24	76.32	45.22	28.85
		AUROC	0.54	0.53	0.58	0.53	0.69	0.64
		ECE	0.70	0.55	0.52	0.24	0.50	0.59
		Brier	0.67	0.54	0.48	0.24	0.49	0.52
RLVR	QuestionProb	Acc	28.15	61.48	35.70	80.05	48.77	30.67
		AUROC	0.40	0.49	0.52	0.46	0.40	0.55
		ECE	0.28	0.58	0.36	0.70	0.44	0.29
		Brier	0.26	0.58	0.36	0.65	0.44	0.30
RLVR	Assessor	Acc	28.15	61.48	35.70	80.05	48.77	30.67
		AUROC	**0.78**	0.65	0.61	0.60	0.66	0.55
		ECE	**0.06**	0.27	0.28	**0.08**	0.24	0.40
		Brier	**0.08**	0.43	0.20	**0.16**	0.17	0.36
RLVR	Probe	Acc	28.15	61.48	35.70	80.05	48.77	30.67
		AUROC	0.76	0.65	0.60	0.59	0.66	0.51
		ECE	**0.06**	0.23	0.24	0.23	0.22	0.49
		Brier	**0.08**	0.28	0.22	0.22	0.18	0.45
CoCA(ours) Joint optimization		Acc	34.88	52.24	35.16	78.68	44.79	29.10
		AUROC	**0.78**	**0.75**	**0.74**	**0.71**	**0.76**	**0.76**
		ECE	0.07	**0.16**	**0.15**	0.09	**0.14**	**0.15**
		Brier	0.09	**0.23**	**0.16**	0.17	**0.12**	**0.15**

- Learning rate: 1×10^{-6}
- Maximum generation length: 4096 tokens
- Temperature: 1.0
- GRPO clipping parameter: $\epsilon = 0.2$

No additional sampling strategies (e.g., top-k, nucleus sampling) are used during training, in order to preserve the model's intrinsic output distribution.

Training schedules. We set the training length based on the optimization behavior observed in the confidence-first baseline comparison. With one-epoch training, the main objective already approaches a plateau before the epoch ends, while later updates bring limited gains but increase cost. Therefore, for the answer-first comparison and the joint-vs-segmented reward ablation, we use a 0.5-epoch schedule as a controlled and efficient budget.

For the sequential-training ablation, we use a longer schedule because it serves as a stability test. Specifically, we train for one epoch on answer accuracy followed by one epoch on confidence, which allows us to examine whether confidence-only optimization becomes unstable after accuracy training and to reveal delayed reward-hacking behavior.

C.2 Baseline Implementation Details

Probe. We train a linear probe on the final-layer embedding of the RLVR model. Given the final-layer embedding $\phi(x)$ of the RLVR model, we train a linear probe to predict the confidence score:

$$f_\theta(x) = \sigma\Big(\mathbf{w}^\top \phi(x) + b\Big),$$

where x denotes the input question and $\sigma(\cdot)$ denotes the sigmoid function.

Additional Assessor Model. We train the assessor model based on Qwen2.5-1.5B-Instruct. The input is the question, and the target label is the empirical accuracy of the target model over 16 sampled responses. The assessor is trained for two epochs with a batch size of 16, a learning rate of 1×10^{-5}, weight decay of 0.01, and MSE loss.

RLCR. We follow an answer-first generation format. The entire trajectory is optimized with a single

reward, corresponding to Equation 13. All other hyperparameters are kept the same as in the main experiments.

C.3 Prompt Format

To enforce the confidence-first output structure, we adopt a fixed system prompt and a task-specific user prompt.

System Prompt
You need to provide the answer as well as its confidence level to follow-up questions. The confidence level is a number between 0 and 1 (inclusive) enclosed within `<confidence>` `</confidence>` tags. The final format that must be followed is:
`<confidence> confidence level here </confidence> answer here`

User prompt
`{question}` Please reason step by step, and put your final answer within `\boxed{}`.

C.4 Evaluation Protocol

All evaluations are conducted using the **OpenCompass** framework. Due to the confidence-first output format, we implement a lightweight modification to the evaluation pipeline:

1. The confidence score enclosed within <confidence> </confidence> tags is first extracted.

2. The remaining text (i.e., the answer segment) is passed to the standard task-specific evaluator.

During inference, **no sampling strategies** are employed; each response is generated via a single forward pass. This ensures that both answer quality and confidence estimates reflect the model's inherent policy distribution rather than artifacts of stochastic decoding.

D Sensitivity to Group Size

We further study the sensitivity of CoCA to the group size G. The main experiments use $G = 16$. We additionally train Qwen2.5-3B-Instruct on Big-Math-Verified with $G = 8$ and $G = 4$, while keeping the remaining training settings unchanged. Table 6 reports the average results across Math, Code, and Factual QA benchmarks.

Overall, CoCA is robust to different choices of group size. On Math benchmarks, reducing G from 16 to 8 and 4 leads to only moderate performance changes, with $G = 16$ achieving the best AUROC. On Code benchmarks, $G = 8$ achieves slightly better confidence estimation performance, while on Factual QA, $G = 16$ provides better calibration in terms of ECE and Brier score despite $G = 8$ obtaining a higher AUROC. These results suggest that CoCA remains effective across different group sizes and does not critically depend on a specific choice of G.

E Detailed Evaluation Results

This section provides a comprehensive breakdown of per-dataset evaluation results for all model scales and training variants considered in this work. While the main paper reports aggregated performance over task categories (Math, Code, and Factual QA) to highlight high-level trends, the tables in this appendix present fine-grained results on each individual benchmark.

For each model size (Qwen2.5-1.5B, 3B, and 7B), we report accuracy (Acc), area under the ROC curve (AUROC), expected calibration error (ECE), and Brier score on all datasets. Following standard practice, higher values indicate better performance for Acc and AUROC, whereas lower values are preferred for ECE and Brier score. To facilitate comparison across training methods, the best-performing method for each dataset and metric is highlighted in bold.

Notably, for the smallest model (Qwen2.5-1.5B), confidence generation is less reliable. We therefore report ECE together with the confidence-generation success rate (SR), defined as the fraction of examples for which a valid confidence estimate is produced. In addition, for datasets on which the accuracy is zero, AUROC cannot be meaningfully computed; such cases are uniformly marked as "–". These cases are excluded from aggregate metric calculations.

Overall, these detailed results complement the main paper by exposing dataset-level behavior that is otherwise obscured by category-level aggregation, and they provide additional evidence for the robustness and limitations of the proposed methods across model scales and task domains.

Table 6: **Sensitivity analysis of the group size G.** We train Qwen2.5-3B-Instruct on Big-Math-Verified with different group sizes and report the average performance on Math, Code, and Factual QA benchmarks. The main experiments use $G = 16$.

Group Size	Metric	Math	Code	Factual
$G = 16$	Acc	44.85	67.38	24.74
	AUROC	0.88	0.67	0.73
	ECE	0.09	0.28	0.14
	Brier	0.09	0.30	0.14
$G = 8$	Acc	42.10	66.55	25.29
	AUROC	0.85	0.74	0.76
	ECE	0.10	0.18	0.17
	Brier	0.09	0.22	0.15
$G = 4$	Acc	43.95	69.94	25.37
	AUROC	0.82	0.69	0.71
	ECE	0.09	0.19	0.22
	Brier	0.10	0.23	0.20

Table 7: Per-dataset results for Qwen2.5-7B-Instruct. "MBPP(s)" denotes sanitized MBPP. "–" indicates undefined AUROC (e.g., when all answers are incorrect and the label has no variance). Bold highlights the best method per dataset for AUROC (higher is better) and for ECE/Brier (lower is better).

Method	Metric	Math				Code		Factual QA	
		GSM8K	Math500	AIME24	AIME25	MBPP(s)	HumanEval	SimpleQA	TriviaQA
Instruct Model	Acc	89.46	78.80	6.67	0.00	73.15	81.71	3.26	56.99
	AUROC	0.62	0.67	0.55	–	0.54	0.59	0.62	0.63
	ECE	0.08	0.18	0.88	0.93	0.27	0.17	0.79	0.36
	Brier	0.10	0.19	0.83	0.87	0.27	0.18	0.67	0.37
RLVR	Acc	93.33	86.00	20.00	6.67	73.54	78.66	4.02	57.45
	AUROC	0.62	0.61	0.62	0.65	0.51	0.59	0.59	0.61
	ECE	0.05	0.13	0.75	0.89	0.26	0.19	0.80	0.35
	Brier	0.06	0.14	0.73	0.86	0.26	0.20	0.69	0.36
RLVR+QuestionProb	Acc	93.63	80.60	16.67	10.00	75.10	78.05	3.63	59.13
	AUROC	0.50	0.24	0.34	0.20	0.48	0.45	0.53	0.56
	ECE	0.92	0.76	**0.10**	**0.09**	0.70	0.70	**0.03**	0.58
	Brier	0.90	0.74	0.15	0.10	0.68	0.67	**0.04**	0.58
RLVR+Assessor Model	Acc	93.63	80.60	16.67	10.00	75.10	78.05	3.63	59.13
	AUROC	0.70	0.78	0.88	0.97	0.51	0.58	0.59	0.62
	ECE	**0.03**	0.05	0.37	0.53	**0.07**	0.50	0.50	**0.03**
	Brier	**0.06**	0.13	0.24	0.34	**0.19**	0.41	0.29	**0.23**
RLVR+Probe	Acc	93.63	80.60	16.67	10.00	75.10	78.05	3.63	59.13
	AUROC	**0.75**	0.82	0.80	**1.00**	0.64	**0.60**	**0.68**	0.60
	ECE	0.07	**0.03**	0.19	0.20	0.70	0.71	0.17	0.24
	Brier	0.07	0.12	0.14	**0.07**	0.67	0.68	0.10	0.31
CoCA(ours)	Acc	92.95	85.40	10.00	3.33	73.93	79.27	3.79	53.90
	AUROC	0.65	**0.84**	**0.96**	0.40	**0.74**	**0.70**	0.66	**0.72**
	ECE	0.06	0.09	0.11	0.15	0.20	**0.12**	0.26	0.26
	Brier	0.07	**0.11**	**0.05**	0.11	0.21	**0.17**	0.18	0.29

Algorithm 1 CoCA (Segmented GRPO for Confidence-First Outputs)

Require: Dataset of prompts $\mathcal{D}$; initial policy π_θ; group size G; clip ε;
Ensure: Updated policy parameters θ.

for each training step **do**
  Sample a mini-batch of prompts $\{x_b\}_{b=1}^{B} \sim \mathcal{D}$.
  for each prompt x in the mini-batch **do**
    Rollout G responses $\{y_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(\cdot \mid x)$ with the enforced format

$$y_i \equiv \texttt{<confidence>}\ s_i\ \texttt{</confidence>}\ y_i^a,$$

    where $s_i = \mathrm{Parse}(y_i^c) \in [0, 1]$.
    Compute answer rewards
    $r_i^a \leftarrow \mathbb{I}(\mathrm{AnsCorrect}(x, y_i^a))$.
    Compute group success rate
    $\hat{p}(x) \leftarrow \frac{1}{G}\sum_{j=1}^{G} r_j^a$.
    Compute confidence rewards
    $r_i^c \leftarrow -\big(s_i - \hat{p}(x)\big)^2$.
    Compute normalized group-relative advantages:

$$\hat{A}_i^a \leftarrow \mathrm{Norm}\big(\{r_j^a\}_{j=1}^{G}, r_i^a\big)$$

$$\hat{A}_i^c \leftarrow \mathrm{Norm}\big(\{r_j^c\}_{j=1}^{G}, r_i^c\big).$$

    Identify token index sets $\mathcal{T}_i^c$ (confidence segment tokens) and $\mathcal{T}_i^a$ (answer segment tokens).
  end for
  Update θ by maximizing the segmented GRPO objective:

$$\mathcal{L}_{\text{CoCA}}(\theta) = \mathbb{E}_x\left[\frac{1}{G}\sum_{i=1}^{G}\Big(\mathcal{L}_i^c(\theta) + \mathcal{L}_i^a(\theta)\Big)\right],$$

$$\mathcal{L}_i^c(\theta) = \sum_{t\in\mathcal{T}_i^c} \min\Big(\rho_{i,t}(\theta)\hat{A}_i^c,\ \mathrm{clip}(\rho_{i,t}(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_i^c\Big),$$

$$\mathcal{L}_i^a(\theta) = \sum_{t\in\mathcal{T}_i^a} \min\Big(\rho_{i,t}(\theta)\hat{A}_i^a,\ \mathrm{clip}(\rho_{i,t}(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_i^a\Big).$$

    where

$$\rho_{i,t}(\theta) = \pi_\theta(y_{i,t}|x, y_{i,<t})/\pi_{\theta_{\text{old}}}(y_{i,t}|x, y_{i,<t}).$$

  Set $\theta_{\text{old}} \leftarrow \theta$.
end for

Table 8: Per-dataset results for Qwen2.5-3B-Instruct. "MBPP(s)" denotes sanitized MBPP. "–" indicates undefined AUROC (e.g., when all answers are incorrect and the label has no variance). Bold highlights the best method per dataset for AUROC (higher is better) and for ECE/Brier (lower is better).

Method	Metric	Math				Code		Factual QA	
		GSM8K	Math500	AIME24	AIME25	MBPP(s)	HumanEval	SimpleQA	TriviaQA
Instruct Model	Acc	77.03	59.80	6.67	6.67	61.09	66.46	2.54	42.88
	AUROC	0.50	0.53	0.40	0.73	0.48	0.55	0.55	0.59
	ECE	0.16	0.31	0.86	0.81	0.33	**0.27**	0.82	0.50
	Brier	0.20	0.33	0.79	0.75	0.35	**0.30**	0.71	0.49
RLVR	Acc	88.55	73.20	3.33	6.67	49.03	20.12	3.40	44.44
	AUROC	0.50	0.51	0.00	0.36	0.48	0.48	0.53	0.54
	ECE	**0.02**	0.18	0.89	0.81	0.42	0.70	0.77	0.46
	Brier	0.10	0.23	0.80	0.74	0.43	0.65	0.66	0.45
RLVR+QuestionProb	Acc	89.39	73.40	13.33	10.00	67.32	70.73	2.36	49.65
	AUROC	0.53	0.24	0.61	0.30	0.45	0.38	0.46	0.53
	ECE	0.89	0.71	0.11	**0.06**	0.65	0.64	**0.02**	0.49
	Brier	0.88	0.71	0.12	0.10	0.64	0.62	**0.02**	0.49
RLVR+Assessor Model	Acc	89.39	73.40	13.33	10.00	67.32	70.73	2.36	49.65
	AUROC	0.74	**0.86**	**0.88**	0.83	0.58	**0.69**	0.58	0.63
	ECE	0.03	0.08	0.18	0.36	**0.09**	0.49	0.47	**0.07**
	Brier	**0.09**	0.14	0.11	0.21	**0.22**	0.43	0.25	**0.24**
RLVR+Probe	Acc	89.39	73.40	13.33	10.0	67.32	70.73	2.36	49.65
	AUROC	0.77	**0.86**	0.83	0.88	0.60	0.57	0.63	0.64
	ECE	0.04	**0.03**	0.15	0.21	0.15	0.58	0.27	0.31
	Brier	**0.09**	**0.13**	0.11	0.10	0.24	0.54	0.14	0.34
CoCA(ours)	Acc	87.34	75.40	13.33	3.33	62.20	72.56	0.88	48.59
	AUROC	**0.79**	**0.86**	**0.88**	**1.00**	**0.67**	0.67	**0.73**	**0.73**
	ECE	0.08	0.08	**0.10**	0.10	0.16	0.39	0.11	0.17
	Brier	0.10	0.14	**0.10**	**0.03**	0.24	0.36	0.03	**0.24**

Table 9: Per-dataset results for Qwen2.5-1.5B-Instruct. "MBPP(s)" denotes sanitized MBPP. "–" indicates undefined AUROC (e.g., when all answers are incorrect and the label has no variance). Bold highlights the best method per dataset for AUROC (higher is better) and for ECE/Brier (lower is better).

Method	Metric	Math				Code		Factual QA	
		GSM8K	Math500	AIME24	AIME25	MBPP(s)	HumanEval	SimpleQA	TriviaQA
Instruct Model	Acc	28.35	26.80	0.00	0.00	54.09	56.10	1.87	27.74
	AUROC	0.52	0.51	–	–	0.56	0.50	0.49	0.56
	ECE (SR)	0.61(0.89)	0.65(0.96)	0.89(0.93)	0.93(0.80)	0.42(0.98)	0.40(0.98)	0.81(0.96)	0.63(0.93)
	Brier	0.58	0.63	0.82	0.87	0.42	0.41	0.72	0.60
RLVR	Acc	66.94	56.60	0.00	0.00	42.02	35.98	2.17	31.37
	AUROC	0.52	0.51	–	–	0.61	0.55	0.53	0.55
	ECE (SR)	0.24(0.91)	0.36(0.94)	0.94	0.94(0.97)	0.55(0.99)	0.56	0.89(0.95)	0.62(0.90)
	Brier	0.28	0.38	0.89	0.88	0.54	0.56	0.82	0.60
RLVR+QuestionProb	Acc	80.06	64.20	6.67	6.67	57.59	56.71	1.64	37.94
	AUROC	0.47	0.27	0.34	**0.68**	0.48	0.44	0.47	0.54
	ECE (SR)	0.78	0.59	**0.06**	**0.02**	0.50	0.45	**0.00**	0.37
	Brier	0.76	0.60	0.07	0.06	0.50	0.46	**0.02**	0.37
RLVR+Assessor Model	Acc	80.06	64.20	6.67	6.67	57.59	56.71	1.64	37.94
	AUROC	0.76	0.83	**0.93**	0.34	0.63	0.63	0.61	0.64
	ECE (SR)	0.03	0.09	0.16	0.37	0.09	0.41	0.29	**0.03**
	Brier	**0.14**	0.17	**0.06**	0.20	0.24	0.46	0.11	0.22
RLVR+Probe	Acc	80.06	64.20	6.67	6.67	57.59	56.71	1.64	37.94
	AUROC	0.76	0.80	0.73	0.20	0.76	0.51	0.60	0.53
	ECE (SR)	**0.02**	**0.05**	0.16	0.15	**0.07**	0.47	0.68	0.23
	Brier	**0.14**	0.17	0.07	0.11	**0.20**	0.47	0.52	0.31
CoCA(ours)	Acc	79.53	61.20	6.67	0.00	55.64	37.80	1.16	36.62
	AUROC	**0.77**	**0.85**	0.83	–	**0.78**	**0.66**	**0.71**	**0.71**
	ECE (SR)	0.11	0.07	0.09	0.09	0.09	**0.09**	0.08	0.10
	Brier	0.15	**0.15**	**0.06**	**0.02**	**0.20**	**0.23**	**0.02**	**0.21**

Table 10: Per-dataset results for llama3-8B-Instruct. "MBPP(s)" denotes sanitized MBPP. "–" indicates undefined AUROC (e.g., when all answers are incorrect and the label has no variance). Bold highlights the best method per dataset for AUROC (higher is better) and for ECE/Brier (lower is better).

Method	Metric	Math				Code		Factual QA	
		GSM8K	Math500	AIME24	AIME25	MBPP(s)	HumanEval	SimpleQA	TriviaQA
Instruct Model	Acc	76.12	31.40	0.00	0.00	44.36	23.17	3.58	55.78
	AUROC	0.49	0.55	–	–	0.61	0.51	0.70	0.58
	ECE	0.15	0.59	0.88	0.90	0.74	0.76	0.64	0.29
	Brier	0.21	0.56	0.77	0.81	0.72	0.72	0.52	0.32
RLVR	Acc	76.72	30.20	0.00	0.00	48.25	40.85	5.25	59.22
	AUROC	0.52	0.55	–	–	0.54	0.52	0.56	0.60
	ECE	0.22	0.67	0.96	0.93	0.51	0.58	0.78	0.26
	Brier	0.23	0.66	0.93	0.87	0.50	0.58	0.66	0.30
RLVR+QuestionProb	Acc	79.38	33.20	0.00	0.00	63.81	59.15	2.33	69.07
	AUROC	0.41	0.39	–	–	0.51	0.46	0.50	0.53
	ECE	0.79	0.25	**0.03**	0.03	0.60	0.56	**0.02**	0.69
	Brier	0.78	0.25	**0.00**	**0.00**	0.60	0.56	**0.02**	0.69
RLVR+Assessor Model	Acc	79.38	33.20	0.00	0.00	63.81	59.15	2.33	69.07
	AUROC	**0.75**	**0.81**	–	–	0.72	0.58	0.55	0.67
	ECE	**0.04**	**0.05**	**0.03**	0.10	0.30	0.23	0.26	0.30
	Brier	0.14	**0.15**	**0.00**	0.01	0.29	0.57	0.10	0.29
RLVR+Probe	Acc	79.38	33.20	0.00	0.00	63.81	59.15	2.33	69.07
	AUROC	0.73	0.78	–	–	0.72	0.57	0.61	0.59
	ECE	**0.04**	0.06	0.06	0.07	0.22	0.24	0.22	0.26
	Brier	0.15	0.16	0.01	0.01	0.25	0.30	0.12	0.31
CoCA(ours)	Acc	85.37	50.80	3.33	0.00	58.75	45.73	4.60	65.71
	AUROC	0.67	**0.81**	**0.86**	–	**0.77**	**0.72**	**0.74**	**0.73**
	ECE	0.11	0.13	0.05	**0.00**	**0.14**	**0.18**	0.19	**0.10**
	Brier	**0.13**	0.19	0.04	**0.00**	**0.21**	**0.24**	0.10	**0.21**

Table 11: Per-dataset results for Qwen2.5-7B-Instruct trained on code data. "MBPP(s)" denotes sanitized MBPP. "–" indicates undefined AUROC (e.g., when all answers are incorrect and the label has no variance). Bold highlights the best method per dataset for AUROC (higher is better) and for ECE/Brier (lower is better).

Method	Metric	Math				Code		Factual QA	
		GSM8K	Math500	AIME24	AIME25	MBPP(s)	HumanEval	SimpleQA	TriviaQA
Instruct Model	Acc	89.46	78.80	6.67	0.00	73.15	81.71	3.26	56.99
	AUROC	0.62	0.67	0.55	–	0.54	0.59	0.62	0.63
	ECE	**0.08**	0.18	0.88	0.93	0.27	0.17	0.79	0.36
	Brier	**0.10**	0.19	0.83	0.87	0.27	0.18	0.67	0.37
RLVR	Acc	89.69	81.20	6.67	3.33	72.76	79.88	2.94	54.76
	AUROC	0.58	0.68	0.54	0.96	0.52	0.53	0.65	0.63
	ECE	**0.08**	**0.15**	0.87	0.90	0.27	0.20	0.78	0.39
	Brier	**0.10**	0.17	0.83	0.84	0.27	0.20	0.65	0.39
RLVR+QuestionProb	Acc	91.36	80.40	13.33	10.00	78.99	81.10	3.51	57.83
	AUROC	0.53	0.23	0.68	0.14	0.44	0.47	0.52	0.57
	ECE	0.87	0.72	**0.03**	0.12	0.69	0.71	**0.02**	0.56
	Brier	0.84	0.69	0.11	0.11	0.63	0.67	**0.03**	0.56
RLVR+Assessor Model	Acc	91.36	80.40	13.33	10.00	78.99	81.10	3.51	57.83
	AUROC	0.56	0.71	**0.80**	0.58	0.63	0.57	0.56	0.53
	ECE	0.30	0.27	0.07	0.31	**0.06**	0.10	0.62	0.18
	Brier	0.17	0.22	**0.09**	0.21	**0.16**	0.16	0.43	0.28
RLVR+Probe	Acc	91.36	80.40	13.33	10.00	78.99	81.10	3.51	57.83
	AUROC	0.56	0.73	0.78	0.57	0.50	0.67	0.48	0.53
	ECE	0.29	0.39	0.08	**0.11**	0.15	0.31	0.67	0.31
	Brier	0.19	0.31	0.10	0.11	0.19	0.25	0.53	0.37
CoCA(ours)	Acc	89.01	76.80	10.00	3.33	76.26	81.10	2.38	55.81
	AUROC	**0.70**	**0.84**	0.52	**0.98**	**0.73**	**0.69**	**0.76**	**0.75**
	ECE	0.18	**0.15**	0.12	0.12	0.13	**0.04**	0.21	**0.08**
	Brier	0.14	**0.15**	0.14	**0.05**	0.18	**0.15**	0.09	**0.21**

Table 12: Per-dataset results for the comparison between our method and the answer-first methods. "MBPP(s)" denotes sanitized MBPP. TTC refers to the token consumption to confidence prediction.

Method	Metric	Math				Code		Factual QA	
		GSM8K	Math500	AIME24	AIME25	MBPP(s)	HumanEval	SimpleQA	TriviaQA
Majority Voting	Acc	94.16	86.60	20.00	16.67	76.26	82.32	4.90	60.88
	AUROC	**0.78**	0.86	0.86	0.68	0.66	0.71	**0.73**	**0.77**
	TTC	3266.38	7229.33	14374.50	13326.4	1945.30	2046.78	2321.69	1048.64
RLCR	Acc	92.49	80.20	13.33	10.00	71.60	77.44	0.65	49.67
	AUROC	0.66	**0.91**	**1.00**	**0.83**	0.59	**0.75**	0.68	0.66
	TTC	392.01	727.65	1074.07	1168.10	219.96	198.43	131.54	112.00
CoCA(ours)	Acc	92.87	83.40	20.00	3.33	75.10	79.88	3.70	54.19
	AUROC	0.66	0.86	0.83	0.57	**0.73**	0.74	0.67	0.73
	TTC	9.92	9.53	9.93	10.40	9.60	9.96	9.79	9.70

Table 13: Detail comparison between Segment Reward and Joint Reward across math, code, and factual QA benchmarks. "MBPP(s)" denotes sanitized MBPP.

Method	Metric	Math				Code		Factual QA	
		GSM8K	Math500	AIME24	AIME25	MBPP(s)	HumanEval	SimpleQA	TriviaQA
Segment Reward	Acc	92.87	83.40	20.00	3.33	75.10	79.88	3.70	54.19
	AUROC	**0.66**	**0.86**	0.83	0.57	**0.73**	**0.74**	0.67	**0.73**
	ECE	**0.06**	**0.06**	0.15	0.19	**0.19**	**0.10**	0.22	**0.22**
	Brier	0.07	**0.10**	0.09	**0.13**	**0.20**	**0.15**	**0.14**	**0.26**
Joint Reward	Acc	93.86	83.80	20.00	13.33	74.32	80.49	2.10	56.30
	AUROC	0.50	0.73	**0.85**	**0.88**	0.50	0.53	**0.74**	0.64
	ECE	**0.06**	0.12	**0.03**	**0.15**	0.26	0.19	**0.15**	0.33
	Brier	**0.06**	0.13	**0.07**	0.14	0.26	0.19	**0.14**	0.33