BEYOND VERBALIZED CONFIDENCE: CALIBRATING REASONERS WITH DIFFERENTIABLE READOUTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning with verifiable rewards (RLVR) trains reasoning models to produce correct answers, but does not ensure that their stated confidence is calibrated. The resulting models are systematically overconfident. Recent methods train calibration inside the RLVR loop by having the model state a numerical confidence alongside its answer, but they all obtain the confidence by sampling it as text. This choice imposes two costs: a sampled confidence introduces variance and in practice collapses to a handful of distinct values, and sampling makes the confidence non-differentiable, forcing the calibration loss through a scalar reward. We propose **CREDO** (Confidence REaDOut) to replace sampling with a deterministic readout. While RLVR optimizes correctness, CREDO reads the confidence from a dedicated token pair in the model's output distribution and trains it by differentiable regression. CREDO further turns the trained confidence into a signal for accuracy, weighting rollouts by how far confidence and outcome disagree, so that accuracy and calibration improve together. Across mathematical and code reasoning, CREDO attains the best accuracy and calibration, and the gains extend to abstention and selective prediction. Code for CREDO and all baselines is available at `https://anonymous.4open.science/r/CREDO-5678`.

1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) has become the standard recipe for training reasoning models (Shao et al., 2024; Guo et al., 2025). Answers in mathematics and code can be checked automatically, so the training reward is simply whether the answer is correct. This produces strong reasoners, but the reward carries no signal about how confident the model should be in any particular answer. Models trained this way are systematically overconfident (Yang et al., 2024; Bereket & Leskovec, 2025; Mei et al., 2025).

Overconfidence has practical costs: downstream systems route answers by their stated confidence, so a confidently wrong answer passes through the filters meant to catch it (Geifman & El-Yaniv, 2017; Kirichenko et al., 2025). The remedy is calibration: a model's stated confidence should match how often it is actually correct, so that when it reports 0.9, roughly nine in ten of those answers are right (DeGroot & Fienberg, 1983; Guo et al., 2017). Calibration does not emerge from a correctness reward, so it has to be trained for.

For calibration to be trainable, the model must produce a confidence estimate alongside each answer (Figure 1a). One approach reads confidence from the model's generation probabilities (Fomicheva et al., 2020), but these reflect the likelihood of the generated text rather than the probability of being correct. Recent methods instead ask the model to state a numerical confidence after its answer, scored by a reward based on a proper scoring rule (Gneiting & Raftery, 2007). Several methods along this route have refined the reward design and the balance between correctness and calibration (Damani et al., 2026; Ma et al., 2026; Xu et al., 2024; Bani-Harouni et al., 2026; Leng et al., 2025). Despite their differences, these methods share one choice: the confidence is a *numeral sampled as text*, which forces the calibration loss to work through a scalar reward (Figure 1b).

This choice imposes two costs. First, a sampled confidence introduces variance that persists as long as the report is stochastic. In practice, the reported confidence collapses to a handful of distinct values, and correct and incorrect answers end up sharing the same confidence (Figure 1c). Second, sampling makes the confidence non-differentiable, forcing the calibration loss to work through a

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.

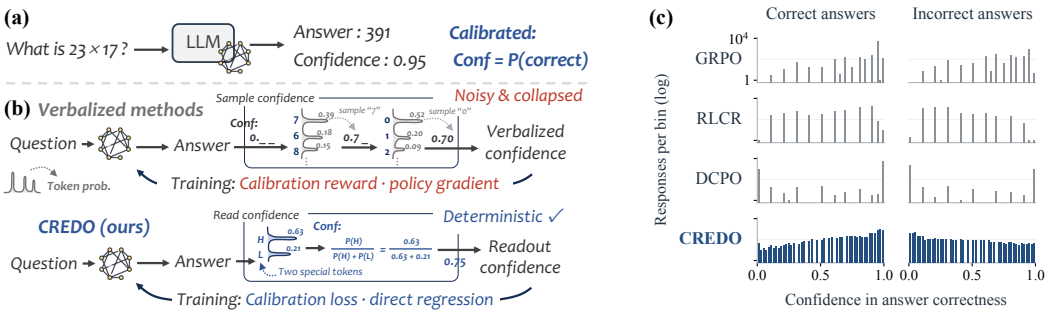


Figure 1: **(a)** The calibration task: a model answers and reports a confidence; calibration means the reported probability matches observed correctness. **(b)** Verbalized methods sample the confidence as text and train it through a scalar reward. CREDO reads the relative probability of a pair of reserved tokens and trains it by direct regression from the calibration loss. **(c)** Confidence distributions on code. Verbalized reports (GRPO, RLCR, DCPO) concentrate on a few values shared by correct and incorrect answers, even when calibration is trained explicitly (RLCR, DCPO). The readout (CREDO) spreads across the full range.

scalar reward rather than a direct gradient. The resulting gradient estimate carries additional variance from the report, making calibration harder to learn. Both costs follow from verbalizing the confidence; redesigning the reward cannot remove them.

We propose CREDO (Confidence REaDOut),[1] which reads the confidence directly from the output distribution rather than sampling it. While RLVR optimizes correctness, CREDO sets aside a pair of reserved tokens and takes their relative probability as the confidence, trained by differentiable regression (Figure 1b). The readout is deterministic and directly differentiable, so the calibration loss trains it by gradient descent. CREDO further turns the trained confidence into a signal for accuracy. A confident mistake or an unexpected success is more informative than a rollout where confidence and outcome already agree, so CREDO weights learning on the answer by the size of the discrepancy. Accuracy and calibration improve together in both domains.

Our main contributions are as follows:

- We show that sampling the confidence as text imposes two costs: the report introduces variance and in practice collapses to a handful of distinct values, and the sampling step makes the confidence non-differentiable, forcing the calibration loss through a scalar reward.
- We propose CREDO, which reads the confidence as the relative probability between a pair of reserved tokens and trains it by differentiable regression, giving the calibration loss a direct gradient path. The policy gradient continues to handle correctness and credit assignment.
- We turn the trained confidence into a signal for accuracy, weighting each rollout by the gap between confidence and outcome. The weighting improves accuracy in both domains.
- Across mathematical and code reasoning, CREDO attains the best accuracy and calibration. Ablations isolate the contributions of the readout and the weighting, and the improved calibration pays off downstream in abstention and selective prediction.

2 Confidence Channels and Their Costs

We start from the RLVR setup and the calibration objective (§2.1). The rest of the section identifies the channel choice behind verbalized confidence (§2.2), analyzes its two costs (§2.3), and derives three requirements for removing them (§2.4).

[1]The name also reads as the Latin *credo*, "I believe."

2.1 SETUP: RLVR AND THE CALIBRATION OBJECTIVE

We work in the GRPO form of RLVR (Shao et al., 2024). For a question x, the policy π_θ samples a group of G rollouts $o_1, \dots, o_G$, and a verifier scores each one for correctness, $a_i = \mathbb{1}[y_i \equiv y^*]$, where y_i is the answer extracted from o_i and y^* the reference answer. The policy maximizes the clipped objective of Schulman et al. (2017), using the group as the baseline:

$$\mathcal{J}(\theta) = \mathbb{E}\Big[\tfrac{1}{G}\sum_{i=1}^{G}\tfrac{1}{|o_i|}\sum_{k\in o_i}\min\left(\rho_{i,k}\hat{A}_{i,k},\ \mathrm{clip}(\rho_{i,k}, 1\pm\epsilon)\,\hat{A}_{i,k}\right)\Big], \quad \hat{A}_{i,k} = \hat{A}_i = \frac{r_i - \bar{r}}{\mathrm{std}(r)}. \tag{1}$$

Here $\rho_{i,k}$ is the token-level importance ratio, ϵ the clipping range, and $\bar{r}$ and $\mathrm{std}(r)$ the mean and standard deviation of the group's rewards. The reward is the correctness score, $r_i = a_i$. In standard GRPO every token of a rollout shares the same $\hat{A}_i$. The verifier's signal is not differentiable in the parameters, so training on it runs through the policy gradient.

A calibrated model must also report a scalar $q \in [0, 1]$ alongside its answer, and the report must be right as often as it claims:

$$\mathbb{E}[a \mid q = v] = v, \qquad \forall v \in [0, 1]. \tag{2}$$

The standard way to train toward equation 2 is a strictly proper scoring rule, whose expectation is uniquely minimized at $\mathbb{E}[a \mid q]$ (Gneiting & Raftery, 2007); the Brier score $(q - a)^2$ is the canonical example (Brier, 1950). RLVR on its own does not target equation 2. Its reward depends only on correctness and is indifferent to the model's confidence, so nothing in the objective pushes a confident error down (Damani et al., 2026).

2.2 THE VERBALIZED CHANNEL

A confidence report has two design choices: where the confidence comes from and how it is trained.

Definition 1 (confidence channel)**.** A *confidence channel* is a pair $(\mathcal{E}, \mathcal{T})$: an emission map $\mathcal{E}$ from the rollout prefix to a scalar report, and a training path $\mathcal{T}$, the route by which the calibration objective reaches the report's value.

The methods that train a model to state its confidence inside RLVR make the same pair of choices. The confidence is written out as a numeral in the text,

$$\mathcal{E}^{\mathrm{verb}}: \quad \hat{q} = \mathrm{parse}(s), \qquad s \sim \pi_\theta(\cdot \mid h), \tag{3}$$

where $h = (x, y, \dots)$ is the rollout prefix up to the confidence slot. The model generates a rollout containing a solution y to x, samples a numeral s (say "0.9") at the confidence slot, and parses it into a report $\hat{q}$. We call a channel whose emission includes a sampling step a *verbalized channel*.

This choice of $\mathcal{E}$ determines $\mathcal{T}$. Sampling makes the realized value of $\hat{q}$ non-differentiable in the parameters, so the calibration signal can only enter equation 1 as a scalar reward. RLCR (Damani et al., 2026) and DCPO (Ma et al., 2026) are the two representative methods along this route,

$$\begin{aligned} &r^{\mathrm{RLCR}} = a - (\hat{q} - a)^2, \\ &r^{\mathrm{DCPO}}_{\mathrm{reason}} = a, \qquad r^{\mathrm{DCPO}}_{\mathrm{conf}} = -\big|\hat{q} - R_{IG}\big|, \qquad R_{IG} = \lambda\tilde{R}_G + (1-\lambda)a, \end{aligned} \tag{4}$$

where $\tilde{R}_G$ is the group's mean accuracy and λ its mixing weight. The first couples the calibration term into a single reward; the second scores the two segments separately and mixes the target. The disagreements are all internal to the reward, but the channel is shared: every verbalized variant uses the same emission and training path (Xu et al., 2024; Bani-Harouni et al., 2026; Leng et al., 2025; Stengel-Eskin et al., 2024; Zhang et al., 2026).

In both analyses, the optimal report is calibrated and the calibration term does not hurt accuracy (Damani et al., 2026; Ma et al., 2026). These analyses validate the reward design but leave the channel unexamined.

2.3 COSTS OF THE VERBALIZED CHANNEL

Each component of the verbalized channel carries a cost. The first falls on the emission $\mathcal{E}$ (proofs of all propositions are in Appendix A).

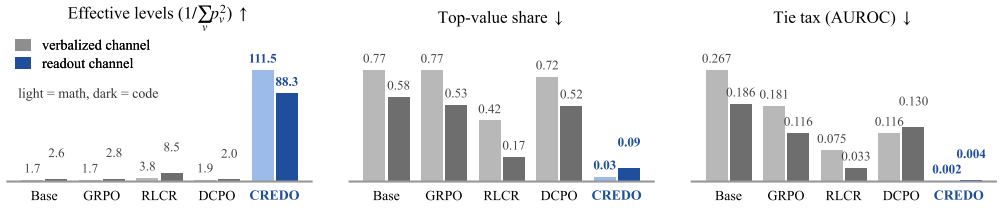


Figure 2: Collapse of the verbalized channel, with CREDO for contrast. Effective levels $= 1/\sum_v p_v^2$ (equiprobable-level equivalent); top-value share = pool fraction of the most frequent value; tie tax = AUROC lost to exactly equal reports.

Proposition 1 (a verbalized report pays a variance tax or determinizes)**.** *Fix a rollout prefix and let $\hat{q}$ be the parsed report of equation 3, with conditional mean $\bar{q}$ and variance v. Every verbalized policy either pays a per-prefix tax that persists as long as the report is stochastic, or determinizes its per-prefix report distribution. Under the squared loss the tax is exactly v, since $\mathbb{E}[(\hat{q}-t)^2] = (\bar{q}-t)^2 + v$ for any target $t \in [0,1]$; under any strictly convex loss ℓ, Jensen's inequality gives $\mathbb{E}[\ell(\hat{q},t)] - \ell(\bar{q},t) > 0$.*

Proposition 1 leaves two outcomes: variance or determinization. Determinization avoids the tax but does not by itself prevent different prefixes from using different values. Empirically, however, reports cluster onto a handful of values, so correct and incorrect answers share the same confidence (Figure 2). The collapse is visible already in an untrained base model, and training does not spread the values out: GRPO, RLCR, and DCPO all have effective levels in single digits.

The variance can be avoided if the report is not sampled but computed. We call a channel whose report is a deterministic functional of the output distribution a *readout channel* (the rightmost group of Figure 2). Even the conditional mean of a verbalized channel is such a functional and removes the variance, but it cannot be optimized through the reward, which only observes sampled values. Removing the variance is not enough; the calibration loss must also reach the report directly.

The second cost falls on $\mathcal{T}$: how the calibration loss trains the report.

Proposition 2 (gradient availability)**.** *Fix a calibration loss ℓ. A sampled report admits no unbiased pathwise gradient estimator, so ℓ can be queried only through its value. If instead the report c is a differentiable function of the parameters at a single position, the exact derivative $\frac{\partial \ell}{\partial c}\nabla_\theta c$ is available, with zero estimator variance given the rollout. Exact pathwise training also requires the report to be computable inside the forward pass that generates the rollout. A verbalized numeral does not satisfy this: its realized value is a sample, and its conditional mean requires marginalizing over the numeral span.*

For the same loss ℓ, the two kinds of report give gradient estimators

$$\underbrace{\hat{g}_{\mathrm{SF}} = \big(\ell(\hat{q},t) - b\big)\,\nabla_\theta \log \pi_\theta(\hat{q} \mid \cdot)}_{\text{sampled report: zero-order query}} \qquad \text{vs.} \qquad \underbrace{\hat{g}_{\mathrm{PW}} = \frac{\partial \ell(c,t)}{\partial c}\,\nabla_\theta c}_{\text{deterministic report: exact derivative}} \tag{5}$$

where b is a baseline. The left side is the score-function estimator of Williams (1992); the right is the pathwise form, available when the report is differentiable in the parameters (Kingma & Welling, 2014; Mohamed et al., 2020).

The two costs reinforce each other: once the grid collapses, the channel can no longer express fine-grained values, and the scalar reward provides no direct gradient to recover them.

2.4 Design requirements

The two costs come from a single choice. A report is either sampled, and then the calibration signal reaches it only through a score-function estimate, or computed deterministically, and then the calibration gradient reaches it directly.

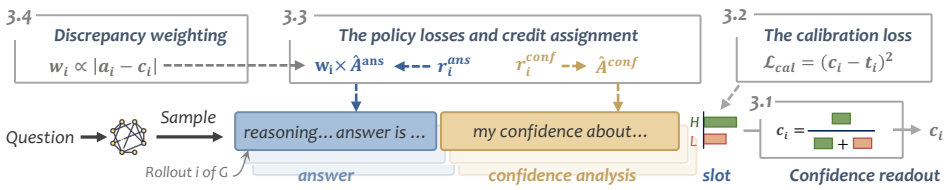


Figure 3: Overview of CREDO. Each rollout contains an answer, a confidence analysis, and a confidence slot. The readout c is the relative probability of two reserved tokens at the slot. The calibration loss trains c by differentiable regression toward a hybrid target t. The policy gradient scores the answer and analysis segments through their text. Discrepancy weighting gives more weight to rollouts where the outcome and the readout disagree.

The two costs then convert into three requirements, two on $\mathcal{E}$ and one on $\mathcal{T}$:

(i) the report is a deterministic functional of the output distribution rather than a sample from it, removing the variance term of Proposition 1;

(ii) its value is not confined to the vocabulary grid, removing the tie tax of Figure 2;

(iii) the calibration loss is differentiable in the report, and the report is computable exactly in the forward pass that generates the rollout, which yields the right-hand branch of equation 5.

A pair of reserved tokens and their relative probability is a simple functional of the output distribution that meets all three. The next section develops such a channel and trains its report directly.

3 CREDO: INSTANTIATING THE READOUT CHANNEL

CREDO operates within the RLVR loop of §2.1 (Figure 3), changing the confidence channel. It takes the report from a pair of reserved tokens (§3.1) and trains it by a regression that uses the calibration gradient directly (§3.2), meeting the three requirements of §2.4. The policy gradient keeps the verifier's correctness judgment and the credit assignment over the text (§3.3), and the trained readout weights learning on the answer (§3.4).

3.1 ROLLOUT FORMAT AND THE READOUT

A rollout ends in a fixed format with three segments: an answer, a confidence analysis, and the confidence slot. At the first generated position inside the slot, a pair of reserved tokens $\{H, L\}$ set aside in the vocabulary serves as the two events of the readout (Chuang et al., 2025). Writing h' for the prefix up to the slot, we restrict the next-token distribution to the pair and renormalize,

$$c \;=\; \frac{\pi_\theta(H \mid h')}{\pi_\theta(H \mid h') + \pi_\theta(L \mid h')} \;=\; \sigma(z_H - z_L) \;\in (0,1), \tag{6}$$

where z_H and z_L are the logits of the two tokens and σ is the logistic function. c depends on the output distribution alone and varies continuously, which gives (i) and (ii). Because the readout is finished inside the forward pass that generates the rollout, the second half of (iii) holds as well. The training in §3.2 gives c its meaning as a confidence.

3.2 THE CALIBRATION LOSS

The readout is differentiable, which gives the first half of (iii). We train it by a regression whose target takes the within-group hybrid form of Ma et al. (2026),

$$\mathcal{L}_{\text{cal}} = \frac{1}{|\mathcal{V}|} \sum_{i \in \mathcal{V}} \big(c_i - t_i\big)^2, \qquad t_i = \gamma\, a_i + (1-\gamma)\, \bar{a}_{\text{group}}, \tag{7}$$

where $\mathcal{V}$ collects the rollouts whose readout is available, $\gamma \in [0,1]$ is the mixing weight, and $\bar{a}_{\text{group}}$ is the mean correctness over all G rollouts. The group term shrinks the binary target toward the

group mean, reducing target variance conditional on the question. The gradient follows the right-hand branch of equation 5 into the readout position: $\nabla_\theta \mathcal{L}_{\text{cal}} \propto 2(c-t)\,c(1-c)\,\nabla_\theta(z_H - z_L)$, one term per sample, with no estimator variance given the rollout.

Proposition 3 (the Bayes-optimal readout)**.** *Fix the rollout policy and assume independent rollouts given x. Let $\mathcal{I}$ comprise x and this rollout's prefix up to the readout, and $\mu = \mathbb{E}[a \mid \mathcal{I}]$ the conditional correctness probability. The Bayes-optimal readout for the unfiltered per-rollout risk of equation 7 is $c^* = \lambda\mu + (1-\lambda)\,p(x)$, $\lambda = \gamma + (1-\gamma)/G$, and $\mathbb{E}[c^* \mid x] = p(x)$ for every γ.*

Here $p(x) = \mathbb{E}_{\pi_\theta}[a \mid x]$ is prompt-level accuracy under the current policy. Decreasing γ reduces target variance conditional on the question but shrinks c^* toward $p(x)$, biasing it relative to μ. Prompt-level mean consistency does not, in general, imply calibration as defined in equation 2, so we treat the hybrid target as regularized correctness supervision. §4.3 evaluates sensitivity to γ; Appendix A contains the proof and quantifies target variance and shrinkage.

3.3 Policy losses and credit assignment

With the report's value trained by the regression, the policy gradient is left with two jobs: whether the answer is right, and whether the confidence analysis deserves reinforcement. The answer and analysis segments are scored separately, on the token sets ans_i and conf_i, and $F_i \in \{0,1\}$ records whether all three segments are present and parseable,

$$r_i^{\text{ans}} \;=\; \beta\,F_i + a_i, \qquad r_i^{\text{conf}} \;=\; F_i\,\big(1-(t_i-c_i)^2\big), \tag{8}$$

where β weights the format term. Each reward is standardized within its own group into $\hat{A}^{\text{ans}}$ and $\hat{A}^{\text{conf}}$, each acting only on its own segment; the segmented credit assignment is inherited from Ma et al. (2026). The policy gradient thus scores the two text segments, while the slot's value is trained by the exact derivative of $\mathcal{L}_{\text{cal}}$; the policy gradient and the calibration loss share parameters but no credit.

3.4 Discrepancy weighting: the readout as a training signal

A rollout whose outcome falls far from what the model expected has more to teach about the answer. We therefore turn that distance into a per-sample weight. *Discrepancy weighting* centers it within the group and lets the high-discrepancy rollouts count for more in the answer segment,

$$s_i = |a_i - c_i|, \qquad w_i = \text{clip}\big(1+\kappa\,(s_i - \bar{s}_{\text{group}}),\; w_-,\; w_+\big), \qquad \hat{A}_i^{\text{ans}} \;\leftarrow\; w_i\,\hat{A}_i^{\text{ans}}, \tag{9}$$

where $s_i = |a_i - c_i|$ is the discrepancy, $\kappa \geq 0$ sets the strength, $\bar{s}_{\text{group}}$ is its group mean, and $w_\pm$ are clipping bounds with $w_- > 0$. The standardization uses the unweighted rewards, and the weights have group mean one before clipping, so the weighting redistributes learning within the group. Calibration information thus becomes a per-sample signal inside the training loop.

The training objective is

$$\mathcal{L} \;=\; -\,\mathcal{J} \;+\; \alpha\,\mathcal{L}_{\text{cal}}, \qquad \hat{A}_{i,k} \;=\; \mathbb{1}[k \in \text{ans}_i]\,w_i\hat{A}_i^{\text{ans}} \;+\; \mathbb{1}[k \in \text{conf}_i]\,\hat{A}_i^{\text{conf}}, \tag{10}$$

where $\mathcal{J}$ is the objective of equation 1, α weights the calibration loss, and the per-token advantage $\hat{A}_{i,k}$ of equation 1 now takes the form on the right.

4 Experiments

We compare the methods in and out of domain (§4.2), locate the gain by removing one component at a time (§4.3), and measure what the calibration is worth to a user who can abstain (§4.4).

4.1 Experimental setup

Datasets and metrics. Mathematics trains on DeepScaleR (Luo et al., 2025b) and is evaluated on seven sets: its in-distribution split, MATH-500 (Hendrycks et al., 2021; Lightman et al., 2024), AIME 2024–2026, and AMC 2023–2024. Code trains on DeepCoder (Luo et al., 2025a) and is

Table 1: Accuracy and calibration in and out of domain, each entry macro-averaged over evaluation sets and training seeds. Column groups are the evaluation domain. Baselines are read through their own verbalized numeral and again through a digit expectation (Appendix B.3), in the rows named `-d`; CREDO is read through its readout. **Bold** marks the best mean in each column and any within one joint standard error of it.

		Mathematics				Code			
	Method	Acc ↑	ECE ↓	AUROC ↑	Brier ↓	Acc ↑	ECE ↓	AUROC ↑	Brier ↓
IN DOMAIN	Base	.439±.001	.472±.002	.644±.004	.453±.002	.455±.001	.418±.001	.690±.004	.395±.002
	GRPO	.706±.009	.199±.027	.775±.062	.204±.029	.581±.007	.182±.070	.753±.061	.205±.066
	RLCR	.682±.010	.159±.012	.831±.009	.171±.009	.553±.013	**.075**±.005	.815±.014	.146±.006
	DCPO	.722±.008	.217±.008	.848±.012	.198±.003	.576±.003	.153±.024	.776±.025	.153±.024
	Base-d	.439±.001	.442±.001	.765±.010	.424±.001	.455±.001	.389±.002	.782±.003	.373±.002
	GRPO-d	.706±.009	.192±.025	.864±.020	.203±.023	.581±.007	.155±.022	.833±.034	.184±.025
	RLCR-d	.682±.010	.163±.005	.852±.006	.171±.009	.553±.013	.080±.004	.818±.011	.145±.005
	DCPO-d	.722±.008	.188±.010	.858±.006	.191±.013	.576±.003	.152±.025	**.849**±.027	.152±.025
	CREDO	**.752**±.013	**.103**±.013	**.933**±.007	**.086**±.004	**.600**±.018	**.064**±.020	**.850**±.027	**.127**±.016
OUT OF DOMAIN	GRPO	**.564**±.003	.335±.019	.734±.030	.325±.020	.495±.004	.315±.041	.721±.032	.308±.040
	RLCR	.515±.011	.170±.004	.815±.016	.186±.007	**.502**±.002	.228±.014	.792±.005	.229±.010
	DCPO	.548±.003	.205±.021	.755±.024	.205±.020	**.508**±.014	.330±.040	.746±.038	.318±.032
	GRPO-d	**.564**±.003	.311±.017	.826±.004	.308±.015	.495±.004	.298±.028	.792±.026	.299±.028
	RLCR-d	.515±.011	.172±.006	.833±.014	.184±.006	**.502**±.002	.232±.013	.816±.003	.226±.011
	DCPO-d	.548±.003	.203±.020	.835±.004	.203±.020	**.508**±.014	.275±.027	.778±.021	.283±.017
	CREDO	**.560**±.007	**.117**±.030	**.855**±.013	**.142**±.013	**.504**±.021	**.138**±.016	**.836**±.009	**.165**±.008

evaluated on four: its in-distribution split, LiveCodeBench v5 and v6 (Jain et al., 2025), and HumanEval+ (Liu et al., 2023). We report accuracy and three calibration measures: ECE (Guo et al., 2017) on the reported values, AUROC on the ranking they induce, and Brier (Brier, 1950) on both.

Baselines and training. On Qwen3-8B (Yang et al., 2025a) we compare the untrained base model with GRPO (Shao et al., 2024), which trains no calibration term, and with RLCR (Damani et al., 2026) and DCPO (Ma et al., 2026), which train one. Training runs for two epochs at learning rate 2×10^{-6} with eight rollouts per question and temperature 1.0. Every run uses seed 43, and the main comparisons add seeds 44 and 45.

Appendix B provides prompt templates, training hyperparameters, the evaluation protocol, and other implementation details.

4.2 ACCURACY AND CALIBRATION IN AND OUT OF DOMAIN

Table 1 compares each method three ways: in its own domain, with only the readout replaced, and on the other domain with nothing adapted.

In domain, CREDO leads on all four metrics in both mathematics and code. Among the baselines, DCPO is the most accurate on mathematics but still trails CREDO on calibration; RLCR trades accuracy for a tighter ECE on code, where it is the only baseline to match CREDO on any calibration metric. All baselines use the verbalized channel and incur both costs of §2.3 (Figure 2).

The `-d` rows replace the sampled numeral with the expectation over each baseline's digit distribution, giving a continuous confidence without retraining. Every swap improves ranking metrics such as AUROC, confirming that sampling adds noise to an otherwise informative signal. None of the swaps closes the calibration gap to CREDO: the reported values were shaped by the training path, and changing the emission alone does not recalibrate them.

The OUT OF DOMAIN rows move each checkpoint to the other domain. Accuracy is comparable across methods, but CREDO's calibration advantage persists and remains the largest. RLCR was the only baseline to match CREDO's code ECE in domain; that match does not survive the domain shift. The calibration advantage transfers across domains.

Appendix C.1 repeats the mathematics comparison at 1.7B and 4B. CREDO leads on all four metrics at every scale, and its calibration advantage widens as models grow larger. Per-set breakdowns and calibration curves appear in Appendices D and C.5.

Table 2: Leave-one-out ablation of CREDO (seed 43).

	Mathematics				Code			
	Acc ↑	ECE ↓	AUROC ↑	Brier ↓	Acc ↑	ECE ↓	AUROC ↑	Brier ↓
CREDO	.753	.094	.933	.088	.614	.052	.869	.117
w/o differentiable regression (α=0)	.696	.241	.880	.232	.611	.172	.857	.172
w/o uncertainty-analysis segment	.709	.096	.902	.107	.587	.048	.868	.117
w/o confidence-segment advantage ($\hat{A}^{\text{conf}}\equiv$0)	.696	.099	.909	.102	.576	.047	.867	.126
w/o discrepancy weighting (κ=0)	.721	.122	.904	.102	.572	.051	.843	.132

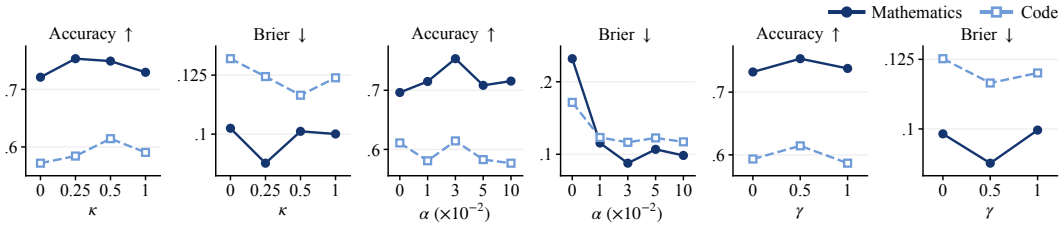


Figure 4: Accuracy and Brier for the two evaluation domains as each coefficient (κ, α, γ) of §3 is varied on seed 43, the others held fixed. Exact values are in Table 15.

4.3 Ablation and attribution

Table 2 removes one component at a time. Setting α=0 disables the calibration loss entirely: ECE and Brier degrade sharply in both domains, and accuracy drops on mathematics. Without differentiable regression the readout loses its direct supervision and calibration collapses. The next two rows restore the regression but remove the confidence reasoning: dropping the analysis text, or keeping it but withholding its policy-gradient credit, both lower accuracy in both domains while calibration is largely unaffected. This suggests that reasoning about confidence improves the model's answers, not only its confidence estimates. Appendix C.2 goes further, independently swapping the emission and training path; neither axis alone recovers the full method's performance.

Discrepancy weighting is the only component feeding the readout back into the answer segment. Removing it causes the largest accuracy drop on code and a substantial drop on mathematics. Calibration also degrades on mathematics, though the weighting acts only on the answer advantage. The confidence and answer channels are therefore not independent: the readout shapes how the model learns from its answers. Appendix C.3 adds the same weighting to RLCR and DCPO, driven by the parsed numeral or a continuous digit expectation; in neither case are the gains as large or as consistent as CREDO's.

Figure 4 varies each coefficient in turn. Any moderate discrepancy weight κ outperforms κ=0, confirming that the weighting itself, not just the readout, contributes to accuracy. The calibration weight α has a threshold effect: α=0 disables calibration training and both ECE and Brier collapse, while all tested positive values restore calibration. The target mixture γ trades variance against shrinkage (§3.2); among tested values, γ=0.5 gives the best accuracy and Brier in both domains, though γ=1 achieves lower code ECE. Appendix C.4 compares two alternative reward shapes; the additive form is the best or tied in mathematics and competitive in code.

4.4 Downstream benefits of calibration

Figure 5 plots the error among answered questions as the model keeps only its most confident fraction (Geifman & El-Yaniv, 2017). In mathematics CREDO is below every baseline throughout. In code RLCR is lower at very low coverage; above it CREDO is lowest. At the evaluated coverage budgets (Table 3), CREDO answers the most accurately and has the lowest aggregate error in both domains; only the two tightest code budgets are statistically tied with DCPO. Selective prediction relies on setting a confidence threshold, and the channel's granularity determines how finely it can be tuned. Verbalized reports concentrate on a few values (Figure 2), leaving a threshold policy only a dozen or two operating points (Table 3); the readout supplies hundreds. On code, DCPO's most confident value already covers half the pool.

Table 3: AURC, selective accuracy at three coverages, and the number of operable thresholds.

	AURC (%) ↓	acc@50 ↑	acc@80 ↑	acc@90 ↑	Thr. ↑
MATHEMATICS					
CREDO	**2.62**±0.04	**.986**±.001	**.962**±.001	**.927**±.002	**372**
DCPO	4.98±0.65	.958±.006	.941±.002	.909±.005	22
RLCR	5.88±0.67	.952±.009	.910±.009	.878±.007	15
GRPO	7.92±0.67	.927±.003	.915±.023	.888±.011	21
Base	29.64±0.30	.712±.001	.705±.005	.666±.001	34
CODE					
CREDO	**16.80**±1.57	**.885**±.013	**.686**±.025	**.617**±.019	**932**
DCPO	21.26±0.95	.863±.016	**.679**±.003	**.611**±.003	10
RLCR	21.36±1.00	.809±.015	.608±.014	.562±.014	12
GRPO	21.50±3.08	.797±.081	.652±.041	.599±.023	17
Base	45.10±0.50	.572±.005	.478±.002	.437±.002	16

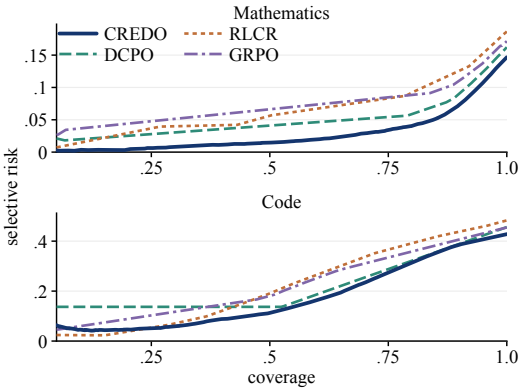


Figure 5: Risk–coverage frontiers, pooled over three seeds; Base omitted.

5 Related work

Confidence estimation. The most direct way to obtain a confidence estimate is to ask for one: Lin et al. (2022) fine-tune a model to state a number, but such numbers are systematically overconfident (Xiong et al., 2024; Tian et al., 2023). Instead of asking, the confidence can be read from the model's own probabilities or internal representations (Fomicheva et al., 2020; Kadavath et al., 2022; Azaria & Mitchell, 2023; Orgad et al., 2025), or estimated from sample agreement (Kuhn et al., 2023; Aichberger et al., 2025; Geng et al., 2024). A dedicated readout can also be trained directly: Chuang et al. (2025) learn a confidence-token readout under supervised finetuning, applied to routing and rejection. CREDO trains the same form of readout on-policy inside RLVR, against a hybrid calibration target, and feeds it back as a signal for answer learning.

Calibration in RLVR. Inside RLVR, one line of work incorporates confidence derived from response likelihood or token-level statistics into policy optimization (Wang et al., 2026; Liu et al., 2025; Zhao et al., 2026). Another trains the model to state a numerical confidence and scores it with a reward. RLCR (Damani et al., 2026) folds a proper scoring rule into the reward, while DCPO (Ma et al., 2026) separates credit by scoring answer and confidence tokens; other work extends to question answering, preference tuning, and generative recommendation (Xu et al., 2024; Bani-Harouni et al., 2026; Stengel-Eskin et al., 2024; Leng et al., 2025; Fan et al., 2026). In the numerical-report methods, the confidence is sampled as text and trained through a scalar reward (§2). A dedicated readout can also serve as a self-reward signal: LaSeR (Yang et al., 2025b) aligns a last-token score with verifier rewards during RLVR and mixes score-derived advantages into policy updates. CREDO targets calibration rather than self-reward: it regresses a normalized two-token readout toward a hybrid calibration target and uses confidence–outcome discrepancies to rescale answer advantages.

6 Conclusion

In this work, we examined the confidence channel shared by all verbalized calibration methods. We analyzed the variance cost of sampling the report, observed that it collapses to a handful of values in the baselines we evaluated, and showed that sampling forces the calibration loss through a scalar reward. We therefore proposed CREDO, which reads the confidence as a deterministic function of the output distribution and trains it by differentiable regression, while the policy gradient optimizes correctness. The same readout also serves as a training signal: it weights learning on the answer by how far each outcome departs from the model's own confidence, and this improves accuracy in both domains. On mathematical and code reasoning tasks CREDO attains the best accuracy and calibration, and its confidence translates into finer abstention control. Because CREDO trains confidence by regression, it is not tied to binary correctness labels; open-ended tasks with graded reward models are a natural next step. Looking further, feeding the confidence signal back during generation would let uncertain steps be reconsidered before the answer is finalized, moving toward reasoners that know when they are right.

AI USE STATEMENT

Generative AI tools assisted with writing mathematical proofs based on proof strategies developed by the authors and with translating portions of the manuscript. We also used these tools to draft sections of the manuscript and revise the prose for clarity, concision, and terminological consistency. They further supported the retrieval of potentially relevant literature and bibliographic metadata, reference formatting, and iterative figure design.

The mathematical claims, proof strategies, theoretical framework, and hypotheses were developed by the authors without generative AI assistance. The authors likewise designed the research methodology and experiments, implemented CREDO, and interpreted the results without such assistance. Generative AI tools were not used to generate, clean, or reformat datasets; qualitative and thematic data analysis was not applicable to this work.

The authors reviewed and revised all AI-assisted material, independently verified each proof, and checked every cited work against the original publication and authoritative metadata. The authors take responsibility for the final content of this paper.

REPRODUCIBILITY STATEMENT

We release a unified codebase at `https://anonymous.4open.science/r/CREDO-5678` that implements CREDO and all baselines under the same training framework, together with the configuration files and datasets needed to reproduce every experiment. The base model, system prompts, full hyperparameter table, random seeds, and evaluation protocol are documented in §4.1 and Appendix B. Appendix A contains complete proofs of the three propositions, and Appendices C–D provide supporting analyses and disaggregated per-dataset results.

REFERENCES

Lukas Aichberger, Kajetan Schweighofer, Mykyta Ielanskyi, and Sepp Hochreiter. Improving uncertainty estimation through semantically diverse language generation. In *International Conference on Learning Representations*, 2025.

Amos Azaria and Tom Mitchell. The internal state of an LLM knows when it's lying. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 967–976, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.68. URL `https://aclanthology.org/2023.findings-emnlp.68/`.

David Bani-Harouni, Chantal Pellegrini, Paul Stangel, Ege Özsoy, Kamilia Zaripova, Nassir Navab, and Matthias Keicher. Rewarding doubt: A reinforcement learning approach to calibrated confidence expression of large language models. In *International Conference on Learning Representations*, 2026.

Michael Bereket and Jure Leskovec. Uncalibrated reasoning: GRPO induces overconfidence for stochastic outcomes. *arXiv preprint arXiv:2508.11800*, 2025.

Glenn W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1–3, 1950. doi: 10.1175/1520-0493(1950)078⟨0001:VOFEIT⟩2.0.CO;2.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Yu-Neng Chuang, Prathusha Kameswara Sarma, Parikshit Gopalan, John Boccio, Sara Bolouki, Xia Hu, and Helen Zhou. Learning to route LLMs with confidence tokens. In *International Conference on Machine Learning*, 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.

Morris H. DeGroot and Stephen E. Fienberg. The comparison and evaluation of forecasters. *The Statistician*, 32(1/2), 1983. doi: 10.2307/2987588.

Chenxiao Fan, Chongming Gao, Yaxin Gong, Haoyan Liu, Fuli Feng, and Xiangnan He. Uncertainty-aware generative recommendation. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1015–1026. ACM, 2026. doi: 10.1145/3770855.3817975.

Marina Fomicheva, Shuo Sun, Lisa Yankovskaya, Frédéric Blain, Francisco Guzmán, Mark Fishel, Nikolaos Aletras, Vishrav Chaudhary, and Lucia Specia. Unsupervised quality estimation for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8: 539–555, 2020. doi: 10.1162/tacl_a_00330.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 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)*, pp. 6577–6595, Mexico City, Mexico, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.366. URL https://aclanthology.org/2024.naacl-long.366/.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007. doi: 10.1198/016214506000001437.

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

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638, 2025. doi: 10.1038/s41586-025-09422-z.

James A. Hanley and Barbara J. McNeil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143(1):29–36, 1982. doi: 10.1148/radiology.143.1.7063747.

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 *Advances in Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021.

Hugging Face. Math-Verify: A robust mathematical expression evaluation system. https://github.com/huggingface/Math-Verify, 2025.

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

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, Deep Ganguli, Danny Hernandez, Josh Jacobson, Jackson Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom Brown, Jack Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

Polina Kirichenko, Mark Ibrahim, Kamalika Chaudhuri, and Samuel J. Bell. AbstentionBench: Reasoning LLMs fail on unanswerable questions. In *Advances in Neural Information Processing Systems*, 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.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, 2023.

Jixuan Leng, Chengsong Huang, Banghua Zhu, and Jiaxin Huang. Taming overconfidence in LLMs: Reward calibration in RLHF. In *International Conference on Learning Representations*, 2025.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri 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. URL https://openreview.net/forum?id=8s8K2UZGTZ.

Haotian Liu, Shuo Wang, and Hongteng Xu. C^2GSPG: Confidence-calibrated group sequence policy gradient towards self-aware reasoning. *arXiv preprint arXiv:2509.23129*, 2025.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? rigorous evaluation of large language models for code generation. In *Advances in Neural Information Processing Systems*, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Michael Luo, Sijun Tan, Roy Huang, Ameen Patel, Alpay Ariyak, Qingyang Wu, Xiaoxiang Shi, Rachel Xin, Colin Cai, Maurice Weber, Ce Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. DeepCoder: A fully open-source 14B coder at O3-mini level. https://pretty-radio-b75.notion.site/DeepCoder-A-Fully-Open-Source-14B-Coder-at-O3-mini-Level-1cf81902c14680b3bee5eb349a512a51, 2025a. Notion Blog.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica. DeepScaleR: Surpassing O1-preview with a 1.5B model by scaling RL. https://pretty-radio-b75.notion.site/DeepScaleR-Surpassing-O1-Preview-with-a-1-5B-Model-by-Scaling-RL-19681902c1468005bed8ca303013a4e2, 2025b. Notion Blog.

Zhengzhao Ma, Xueru Wen, Boxi Cao, Yaojie Lu, Hongyu Lin, Jinglin Yang, Min He, Xianpei Han, and Le Sun. Decoupling reasoning and confidence: Resurrecting calibration in reinforcement learning from verifiable rewards. In *International Conference on Machine Learning*, 2026.

Zhiting Mei, Christina Zhang, Tenny Yin, Justin Lidard, Ola Shorinwa, and Anirudha Majumdar. Reasoning about uncertainty: Do reasoning models know when they don't know? *arXiv preprint arXiv:2506.18183*, 2025.

Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte carlo gradient estimation in machine learning. *Journal of Machine Learning Research*, 21(132):1–62, 2020.

Hadas Orgad, Michael Toker, Zorik Gekhman, Roi Reichart, Idan Szpektor, Hadas Kotek, and Yonatan Belinkov. LLMs know more than they show: On the intrinsic representation of LLM hallucinations. In *International Conference on Learning Representations*, 2025.

Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015. doi: 10.1609/aaai.v29i1.9602.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Elias Stengel-Eskin, Peter Hase, and Mohit Bansal. LACIE: Listener-aware finetuning for calibration in large language models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 43080–43106. Curran Associates, Inc., 2024. doi: 10.52202/079017-1364.

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

Ziqi Wang, Xingzhou Lou, Meiqi Wu, Zhengqi Wen, and Junge Zhang. Calibration-aware policy optimization for reasoning LLMs. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 18375–18390, San Diego, California, United States, 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.836. URL https://aclanthology.org/2026.acl-long.836/.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3–4):229–256, 1992. doi: 10.1007/BF00992696.

Edwin B. Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212, 1927. doi: 10.1080/01621459.1927.10502953.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. Can LLMs express their uncertainty? an empirical evaluation of confidence elicitation in LLMs. In *International Conference on Learning Representations*, 2024.

Tianyang Xu, Shujin Wu, Shizhe Diao, Xiaoze Liu, Xingyao Wang, Yangyi Chen, and Jing Gao. SaySelf: Teaching LLMs to express confidence with self-reflective rationales. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 5985–5998, Miami, Florida, USA, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.343. URL https://aclanthology.org/2024.emnlp-main.343/.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a.

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

Wenkai Yang, Weijie Liu, Ruobing Xie, Yiju Guo, Lulu Wu, Saiyong Yang, and Yankai Lin. LaSeR: Reinforcement learning with last-token self-rewarding. *arXiv preprint arXiv:2510.14943*, 2025b.

Caiqi Zhang, Xiaochen Zhu, Chengzu Li, Nigel Collier, and Andreas Vlachos. LoVeC: Reinforcement learning for better verbalized confidence in long-form generation. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 33336–33363, San Diego, California, United States, 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.1539. URL https://aclanthology.org/2026.acl-long.1539/.

Qiannian Zhao, Chen Yang, Jinhao Jing, Yunke Zhang, Xuhui Ren, Lu Yu, Shijie Zhang, and Hongzhi Yin. Know what you know: Metacognitive entropy calibration for verifiable RL reasoning. *arXiv preprint arXiv:2602.22751*, 2026.

A PROOFS

For the verbalized channel, h denotes the rollout prefix up to the confidence slot, and the report is $\hat{q} = \mathrm{parse}(s)$ for a numeral $s \sim \pi_\theta(\cdot \mid h)$ sampled from the policy, taking values in a finite set $\mathcal{Q} \subset [0, 1]$. Expectations are conditional on h and on the other rollouts of the group, which fixes any target $t \in [0, 1]$ determined by them.

A.1 PROOF OF PROPOSITION 1

Fix a rollout prefix and let $\hat{q}$ be the parsed report of equation 3, with conditional mean $\bar{q}$ and variance v. Every verbalized policy either pays a per-prefix tax that persists as long as the report is stochastic, or determinizes its per-prefix report distribution. Under the squared loss the tax is exactly v, since $\mathbb{E}[(\hat{q} - t)^2] = (\bar{q} - t)^2 + v$ for any target $t \in [0, 1]$; under any strictly convex loss ℓ, Jensen's inequality gives $\mathbb{E}[\ell(\hat{q}, t)] - \ell(\bar{q}, t) > 0$.

Proof. The tax is the excess of scoring the sampled report over scoring the same policy's conditional mean, $\tau = \mathbb{E}[\ell(\hat{q}, t)] - \ell(\bar{q}, t)$.

Under the squared loss, t and $\bar{q}$ are fixed given h, and expanding about $\bar{q}$ gives

$$\mathbb{E}\big[(\hat{q} - t)^2\big] = \mathbb{E}\big[(\hat{q} - \bar{q})^2\big] + 2(\bar{q} - t)\,\mathbb{E}\big[\hat{q} - \bar{q}\big] + (\bar{q} - t)^2 = v + (\bar{q} - t)^2, \tag{11}$$

where the middle term vanishes. So the tax is the conditional variance itself, $\tau = v$, and it does not involve the target. Changing calibration moves $\bar{q}$, and with it only the $(\bar{q} - t)^2$ term.

For a strictly convex ℓ, Jensen's inequality gives $\mathbb{E}[\ell(\hat{q}, t)] \geq \ell(\bar{q}, t)$, with equality only if $\hat{q}$ is almost surely constant given h.

Both branches follow. A non-degenerate per-prefix report distribution pays $\tau > 0$. A degenerate one concentrates the report on a single element of $\mathcal{Q}$ and pays nothing — the determinized case. □

A.2 PROOF OF PROPOSITION 2

Fix a calibration loss ℓ. A sampled report admits no unbiased pathwise gradient estimator, so ℓ can be queried only through its value. If instead the report c is a differentiable function of the parameters at a single position, the exact derivative $\frac{\partial \ell}{\partial c} \nabla_\theta c$ is available, with zero estimator variance given the rollout. Exact pathwise training also requires the report to be computable inside the forward pass that generates the rollout. A verbalized numeral does not satisfy this: its realized value is a sample, and its conditional mean requires marginalizing over the numeral span.

Proof. A pathwise estimator (Kingma & Welling, 2014) writes the draw as $s = S_\theta(\xi)$, with noise ξ whose law does not depend on θ, and differentiates the composition, $\frac{\partial \ell}{\partial \hat{q}} \nabla_\theta \,\mathrm{parse}(S_\theta(\xi))$. The map $\theta \mapsto \mathrm{parse}(S_\theta(\xi))$ takes values in the finite set $\mathcal{Q}$, so it is locally constant wherever it is continuous and its gradient is zero wherever it exists. The estimator is therefore zero, while the gradient

$$\nabla_\theta\, \mathbb{E}\big[\ell(\hat{q}, t)\big] = \sum_s \ell\big(\mathrm{parse}(s), t\big)\, \nabla_\theta \pi_\theta(s \mid h) \tag{12}$$

is not whenever two values in $\mathcal{Q}$ carry different losses and the policy can shift probability between them. In this identity ℓ enters only through its values on $\mathcal{Q}$, never through $\partial \ell / \partial \hat{q}$. Equivalently, the same gradient has the score-function form (Williams, 1992) $\mathbb{E}[\ell(\hat{q}, t)\, \nabla_\theta \log \pi_\theta(s \mid h)]$, the left-hand branch of equation 5.

Now let c be differentiable in θ at a single position, with ℓ differentiable in the report. Given the rollout, c is a deterministic function of θ, so the chain-rule derivative $\nabla_\theta \ell(c, t) = \frac{\partial \ell}{\partial c} \nabla_\theta c$ is exact and carries no variance.

Exact pathwise training also needs a differentiable value inside the pass that generates the rollout, and a verbalized numeral has none. Its realized value is a sample, which carries no pathwise derivative. Its conditional mean,

$$\bar{q} = \sum_s \pi_\theta(s \mid h)\, \mathrm{parse}(s), \tag{13}$$

is differentiable in θ, but the sum ranges over every string the numeral span can produce, because parse couples the span's positions. The generating pass produces one of those strings, not the sum. A report read at a single position is instead a function of that position's logits, which the same pass has already computed. □

A.3 Proof of Proposition 3

Fix the rollout policy and assume independent rollouts given x. Let $\mathcal{I}$ comprise x and this rollout's prefix up to the readout, and $\mu = \mathbb{E}[a \mid \mathcal{I}]$ the conditional correctness probability. The Bayes-optimal readout for the unfiltered per-rollout risk of equation 7 is $c^* = \lambda\mu + (1-\lambda)\,p(x)$, $\lambda = \gamma + (1-\gamma)/G$, and $\mathbb{E}[c^* \mid x] = p(x)$ for every γ.

Proof. We analyze the unfiltered per-rollout population risk $R(c) = \mathbb{E}[(c(\mathcal{I}) - t)^2]$, which coincides with the expected loss of equation 7 when every rollout has an available readout. Expectations are over the group drawn at a question x, and $p(x) = \mathbb{E}[a \mid x]$ denotes the current policy's accuracy on it. The target is $t = \gamma a + (1-\gamma)\bar{a}_{\mathrm{group}}$, with $\bar{a}_{\mathrm{group}}$ the mean correctness over all G rollouts. Each squared error is minimized pointwise by the conditional mean of its target, so the best readout among functions of $\mathcal{I}$ is $c^* = \mathbb{E}[t \mid \mathcal{I}]$.

By definition $\mathbb{E}[a \mid \mathcal{I}] = \mu$. The other rollouts are drawn independently given x, and $\mathcal{I}$ is a function of x and this rollout alone, so each of the other $G-1$ scores has conditional mean $p(x)$, and

$$\mathbb{E}\big[\bar{a}_{\mathrm{group}} \mid \mathcal{I}\big] = \frac{\mu}{G} + \frac{G-1}{G}\,p(x) = p(x) + \frac{\mu - p(x)}{G}. \tag{14}$$

Therefore

$$c^* = \gamma\,\mu + (1-\gamma)\,p(x) + (1-\gamma)\,\frac{\mu - p(x)}{G} = \lambda\mu + (1-\lambda)\,p(x), \qquad \lambda = \gamma + \frac{1-\gamma}{G}. \tag{15}$$

Averaging over rollouts at the same question, $\mathbb{E}[\mu \mid x] = \mathbb{E}[\mathbb{E}[a \mid \mathcal{I}] \mid x] = p(x)$, so $\mathbb{E}[c^* \mid x] = \lambda\,p(x) + (1-\lambda)\,p(x) = p(x)$ for every γ. □

Target variance and shrinkage bias. Writing $t_i = \lambda a_i + \frac{1-\gamma}{G}\sum_{j\neq i} a_j$ and using conditional independence given x:

Target variance.

$$\mathrm{Var}(t_i \mid x) = \left[\gamma^2 + \frac{1-\gamma^2}{G}\right] p(x)\big(1 - p(x)\big). \tag{16}$$

At $G{=}8$ and $\gamma{=}0.5$ the bracketed coefficient is 0.344, compared with 1 at $\gamma{=}1$. This is a reduction in marginal target variance conditional on the question; it does not by itself establish lower prediction error or gradient variance.

Shrinkage bias. Relative to the conditional correctness probability μ,

$$c^*_\gamma - \mu = (1-\lambda)\big(p(x) - \mu\big). \tag{17}$$

For fixed $\gamma{<}1$, the limit as $G \to \infty$ is $(1{-}\gamma)(p(x){-}\mu)$: increasing the group size does not generally eliminate the bias.

Brier cost. Because μ minimizes Brier risk over $\mathcal{I}$-measurable predictors and the cross term vanishes ($\mathbb{E}[a{-}\mu \mid \mathcal{I}] = 0$),

$$\mathbb{E}\big[(c^*_\gamma - a)^2\big] - \mathbb{E}\big[(\mu - a)^2\big] = (1-\lambda)^2\,\mathbb{E}\big[(\mu - p(x))^2\big]. \tag{18}$$

The hybrid target thus reduces target variance conditional on x while introducing a population-level Brier cost relative to μ. Whether this regularization improves the learned readout under finite data and limited optimization is an empirical question (§4.3).

B EXPERIMENTAL SETUP AND IMPLEMENTATION DETAILS

B.1 MODEL AND PROMPTS

All experiments use Qwen3-8B (Yang et al., 2025a). The reserved tokens H and L of §3.1 are added to the vocabulary as special tokens, with embeddings copied from the rows of "high" and "low".

The system prompts of the two channels fix the three-segment format of §3.1: they differ only in the slot instruction, and the coding versions differ only in carrying the answer in a `python` block.

Verbalized channel. A conversation between User and Assistant. The user asks a question, and the Assistant solves it. Solve the problem step by step, and put your final answer within `\boxed{}`. After the final answer, analyze the uncertainty of your solution within `<analysis> </analysis>` tags. This analysis is the basis for the confidence level you will report next, and must follow these rules: (1) point out specific steps that could be wrong or ambiguous, including alternative approaches that might lead to different answers; (2) do not solve the problem again and do not revise or change your answer; (3) be specific — if you cannot find more uncertainties, say so explicitly. Then provide your confidence that the final answer is correct, as a decimal number between 0 and 1 (e.g. 0.3 or 0.8), within `<confidence> </confidence>` tags. The final format that must be followed is: {step-by-step solution with `\boxed{final answer}`} `<analysis>` uncertainty analysis here `</analysis>` `<confidence>` confidence here `</confidence>`

Readout channel (CREDO). A conversation between User and Assistant. The user asks a question, and the Assistant solves it. Solve the problem step by step, and put your final answer within `\boxed{}`. After the final answer, analyze the uncertainty of your solution within `<analysis> </analysis>` tags. This analysis is the basis for the confidence level you will report next, and must follow these rules: (1) point out specific steps that could be wrong or ambiguous, including alternative approaches that might lead to different answers; (2) do not solve the problem again and do not revise or change your answer; (3) be specific — if you cannot find more uncertainties, say so explicitly. Then, within `<confidence> </confidence>` tags, output exactly one token: `<CONF_HIGH>` if your final answer is more likely correct than not, otherwise `<CONF_LOW>`. The final format that must be followed is: {step-by-step solution with `\boxed{final answer}`} `<analysis>` uncertainty analysis here `</analysis>` `<confidence><CONF_HIGH></confidence>`

B.2 TRAINING

Table 4: Training configuration. Values that differ by domain are given as mathematics / code. Rollouts are generated with vLLM (Kwon et al., 2023).

SHARED			
Base model	Qwen3-8B	Precision	bf16
Seeds	43, 44, 45	Rollouts per question G	8
Learning rate	2×10^{-6}	Per-device batch size	2
Schedule	constant, no warmup	Gradient accumulation	64
Optimizer	AdamW (Loshchilov & Hutter, 2019)	Epochs	2
Rollout temperature	1.0	Rollout top-k	50
Prompt budget	1,024 / 2,048	Clipping range ϵ	0.2
Completion budget	8,192 / 6,144	KL coefficient	0
CREDO			
Target mixture γ	0.5	Weighting strength κ	0.25 / 0.50
Calibration weight α	0.03	Weight clip $[w_-, w_+]$	$[0.5,\ 2.0]$
Format weight β	0.5	Warmup for α and κ	30 steps

B.3 EVALUATION AND METRICS

Mathematics trains on DeepScaleR, code on DeepCoder, and each domain is evaluated on the sets of Table 5. We evaluate at temperature 0.7, generating each trained run under its own training seed; the base model has no training seed, so its error bars come from the three evaluation seeds.

Table 5: Datasets. n is the number of samples drawn per problem at evaluation. AIME and AMC are the corresponding years' competition problems; HumanEval+ strengthens the tests of HumanEval (Chen et al., 2021).

Mathematics			Code		
Set	Problems	n	Set	Problems	n
DeepScaleR (train)	9,500	—	DeepCoder (train)	9,500	—
DeepScaleR test	500	4	DeepCoder test	500	4
MATH-500	500	4	HumanEval+	163	4
AIME 2024	30	8	LiveCodeBench v5	279	4
AIME 2025	30	8	LiveCodeBench v6	131	4
AIME 2026	30	8			
AMC 2023	46	8			
AMC 2024	45	8			

Write $\{(c_i, a_i)\}_{i=1}^{n}$ for an evaluation pool, with c_i the reported confidence and $a_i \in \{0, 1\}$ the correctness, decided by symbolic equivalence in mathematics (Hugging Face, 2025) and by executing the program against the problem's tests in code.

Digit-probability expectation. The `-d` rows throughout the paper replace the sampled numeral with a continuous value derived from the model's token probabilities at the confidence slot. Given the verbalized format `<confidence>0.d</confidence>`, we teacher-force the prefix up to the confidence position and read the logits at two slots:

- *Integer position* (candidates "0" and "1"), renormalized: $\hat{p}_0,\ \hat{p}_1$.
- *Decimal position* (candidates "0" through "9"), renormalized: $\hat{q}_d$ for $d = 0, \ldots, 9$.

The confidence is a mixture over the two integer outcomes:

$$c \;=\; \hat{p}_1 \cdot 1.0 \;+\; \hat{p}_0 \cdot \frac{\mathbb{E}[d]}{10}, \qquad \mathbb{E}[d] = \sum_{d=0}^{9} d\, \hat{q}_d\,. \tag{19}$$

The mixture covers $[0, 1]$ continuously. Probabilities are renormalized within each candidate set, and the logits are read from the existing rollout without an additional forward pass.

We report the following metrics over each evaluation pool:

- **Accuracy** is the mean of a_i.
- **ECE** uses ten equal-width bins $B_1, \ldots, B_{10}$ (Pakdaman Naeini et al., 2015; Guo et al., 2017), with 0 and 1 falling in the first and last bin:

$$\mathrm{ECE} = \sum_{b=1}^{10} \frac{|B_b|}{n} \left|\overline{a}_{B_b} - \overline{c}_{B_b}\right|, \tag{20}$$

  where $\overline{a}_B$ and $\overline{c}_B$ are the bin's mean correctness and mean confidence.
- **Brier** is the mean squared error, $\frac{1}{n}\sum_i (c_i - a_i)^2$ (Brier, 1950).
- **AUROC** is the area under the ROC curve (Hanley & McNeil, 1982) obtained by scoring a with c.
- **Selective prediction** (AURC, acc@τ, Thr.) (Geifman & El-Yaniv, 2017) answers questions in decreasing order of c. Equal confidences cannot be separated by a threshold, so the achievable coverages are the boundaries of the tie groups: for the distinct values $v_1 > \cdots > v_K$ of c,

$$\mathrm{cov}_k = \frac{|\{i : c_i \geq v_k\}|}{n}, \qquad \mathrm{acc}_k = \frac{1}{|\{i : c_i \geq v_k\}|} \sum_{i:\, c_i \geq v_k} a_i, \tag{21}$$

  and Thr. is the number K of achievable points. AURC integrates the risk $1-\mathrm{acc}$ against coverage over $[0, 1]$ by the trapezoid rule, extended as a constant below cov_1, and $\mathrm{acc@}\tau$ interpolates $(\mathrm{cov}_k, \mathrm{acc}_k)$ linearly in coverage.

C ADDITIONAL ANALYSES

C.1 ADDITIONAL MODEL SCALES

The main table uses Qwen3-8B. Table 6 runs the same mathematics comparison at 1.7B and 4B (seed 43), with the 8B seed-43 run included for direct comparison; three-seed averages for 8B appear in Table 1. Accuracy grows with model size for every method. CREDO leads on all four metrics at every scale, and its calibration advantage widens as models grow larger. The digit-expectation rows (-d) consistently outrank their verbalized counterparts in AUROC at all three scales, indicating that sampling noise is not specific to 8B.

Table 6: Mathematics results at three model scales (seed 43). All methods follow the same protocol as Table 1. `-d` rows evaluate through the digit expectation; CREDO through its logit readout. **Bold** marks the best in each column.

	Qwen3-1.7B				Qwen3-4B				Qwen3-8B			
	Acc ↑	ECE ↓	AUROC ↑	Brier ↓	Acc ↑	ECE ↓	AUROC ↑	Brier ↓	Acc ↑	ECE ↓	AUROC ↑	Brier ↓
Base	.268	.531	.691	.477	.425	.468	.641	.447	.440	.474	.640	.455
GRPO	.460	.391	.782	.357	.660	.221	.813	.220	.717	.174	.820	.179
RLCR	.494	.145	.853	.155	.663	.186	.836	.185	.682	.167	.837	.181
DCPO	.501	.203	.823	.189	.651	.220	.797	.196	.715	.222	.846	.195
Base-d	.268	.557	.717	.495	.425	.446	.765	.428	.440	.443	.772	.425
GRPO-d	.460	.379	.806	.350	.660	.208	.851	.211	.717	.170	.886	.184
RLCR-d	.494	.152	.867	.154	.663	.191	.866	.190	.682	.164	.855	.180
DCPO-d	.501	.214	.873	.191	.651	.183	.871	.186	.715	.184	.852	.180
CREDO	**.514**	**.115**	**.905**	**.122**	**.689**	**.116**	**.884**	**.128**	**.753**	**.094**	**.933**	**.088**

C.2 EMISSION AND TRAINING-PATH DECOMPOSITION

Table 2 removes one component at a time while keeping the rest of CREDO intact. To disentangle the emission from the training path, we fill the two remaining cells of the 2×2 design space (Table 7) and add one fully decoupled reference. All arms follow the protocol of Table 1.

Table 7: Design space for the confidence channel.

	Score-function reward	Differentiable regression
Verbalized numeral	RLCR / DCPO (Table 1)	Digit-exp. emission
$\{H, L\}$ readout	Readout + reward only	CREDO (Table 1)

- **Digit-expectation emission.** The confidence slot produces a verbalized numeral, but the value used for training and evaluation is the continuous expectation over the digit probabilities rather than the $\{$H,L$\}$ logit readout. The regression loss, target, and discrepancy weighting are all kept; only the emission differs from CREDO.
- **Readout + reward only.** The $\{$H,L$\}$ readout is kept, but differentiable regression is removed and the readout tokens are trained through the reward signal via policy gradient, as in verbalized methods. Only the training path differs from CREDO.
- **GRPO + auxiliary head.** A standard GRPO model whose answer reward uses accuracy alone. The confidence channel plays no role in policy training; a separate BCE head at the readout position learns to predict per-rollout correctness.

Changing either axis alone degrades performance. Replacing the dedicated token pair with a digit-probability expectation while keeping the regression causes the largest drop: the verbalized digit distribution, though continuous, provides a weaker training signal. Keeping the readout but training it only through the reward also hurts, which confirms that differentiable regression is what drives the calibration gains. A standalone BCE head on top of GRPO provides a natural lower bound; it achieves reasonable calibration, but CREDO outperforms it on every metric—optimizing the readout inside the training loop works better than attaching a head afterward.

Table 8: Emission and training-path decomposition (seed 43). **Bold** marks the best in each column.

	Mathematics				Code			
	Acc ↑	ECE ↓	AUROC ↑	Brier ↓	Acc ↑	ECE ↓	AUROC ↑	Brier ↓
CREDO (full)	**.753**	**.094**	**.933**	**.088**	**.614**	**.052**	**.869**	**.117**
Digit-exp. emission	.676	.251	.671	.218	.587	.180	.724	.229
Readout + reward only	.695	.262	.869	.252	.588	.187	.818	.186
GRPO + auxiliary head	.702	.106	.902	.111	.599	.125	.832	.155

C.3 Discrepancy weighting on verbalized baselines

Table 2 shows that discrepancy weighting contributes substantially to accuracy. A natural question is whether the same weighting transfers to verbalized baselines. Table 9 adds it to RLCR and DCPO, varying the source of the confidence value c in the weight $w \propto |a - c|$:

- **Parsed numeral.** The weight uses each baseline's own parsed confidence value. With only a handful of distinct values per group, rollouts within a group share one of only a few distinct weights.
- **Digit-probability expectation.** The weight uses the continuous expectation over each baseline's own digit logits, so that rollouts within a group receive distinct weights even when their verbalized reports coincide.

Adding discrepancy weighting to RLCR or DCPO does not reproduce the accuracy improvement that CREDO obtains from the same mechanism (Table 2), because the verbalized channel cannot supply the per-rollout confidence variation the weighting needs. With parsed numerals, collapsed values leave rollouts in a group sharing one of only a few distinct weights, so the reweighting barely acts. Digit-expectation confidence gives the weights more variation, but does not close the gap either: for DCPO, accuracy drops in both domains, and calibration worsens in mathematics though it improves on several code metrics. Whether the confidence is read as a parsed numeral or as a continuous expectation, the verbalized channel does not give the weighting enough signal to redistribute learning. CREDO's readout varies continuously across rollouts, which lets the weighting concentrate updates on those where confidence and outcome disagree most—and that is where the accuracy gains come from.

Table 9: Verbalized baselines with discrepancy weighting (seed 43). The Eval column indicates the confidence source at evaluation. Shaded rows evaluate through the digit-probability expectation; CREDO through its logit readout; unshaded rows through the parsed verbalized numeral. **Bold** marks the best in each column.

		Mathematics				Code			
	Eval	Acc ↑	ECE ↓	AUROC ↑	Brier ↓	Acc ↑	ECE ↓	AUROC ↑	Brier ↓
RLCR	verbalized	.682	.167	.837	.181	.539	.070	.820	.144
RLCR + weighting (parsed)	verbalized	.701	.142	.837	.152	.545	.104	.818	.155
RLCR + weighting (digit-exp)	verbalized	.697	.141	.847	.148	.542	.085	.810	.149
RLCR	digit-exp	.682	.164	.855	.180	.539	.078	.822	.144
RLCR + weighting (parsed)	digit-exp	.701	.151	.857	.152	.545	.108	.823	.153
RLCR + weighting (digit-exp)	digit-exp	.697	.152	.866	.147	.542	.090	.813	.147
DCPO	verbalized	.715	.222	.846	.195	.579	.180	.747	.179
DCPO + weighting (parsed)	verbalized	.716	.223	.816	.203	.581	.150	.791	.154
DCPO + weighting (digit-exp)	verbalized	.690	.277	.780	.262	.560	.153	.801	.153
DCPO	digit-exp	.715	.184	.852	.180	.579	.179	.842	.178
DCPO + weighting (parsed)	digit-exp	.716	.187	.865	.187	.581	.147	.850	.151
DCPO + weighting (digit-exp)	digit-exp	.690	.221	.785	.233	.560	.149	.818	.147
CREDO	readout	**.753**	**.094**	**.933**	**.088**	**.614**	**.052**	**.869**	**.117**

C.4 Shape of the answer reward

We compare the answer-segment reward with two variants of it:

- **additive** (CREDO, equation 8): $r^{\text{ans}} = \beta F + a$;
- **gated**: $r^{\text{ans}} = F\,(\beta + a)$;
- **no format term**: $r^{\text{ans}} = a$.

The additive form is best or tied on every metric in mathematics, and neither variant dominates it in code (Table 10). Under the gated form, a format failure zeroes out the entire reward including the correctness signal, which slows accuracy learning. Removing the format term has a milder effect on accuracy but yields mixed results on calibration.

Table 10: Answer-reward shapes on seed 43; all else follows Table 4.

	Mathematics				Code			
r^{ans}	Acc ↑	ECE ↓	AUROC ↑	Brier ↓	Acc ↑	ECE ↓	AUROC ↑	Brier ↓
$\beta F + a$	.753	.094	.933	.088	.614	.052	.869	.117
$F\,(\beta + a)$	.714	.105	.910	.110	.589	.038	.871	.113
a	.723	.113	.924	.091	.583	.042	.889	.102

C.5 Calibration curves

Figure 6 shows calibration curves for all methods in both domains. The verbalized baselines resolve only a handful of points, reflecting the collapse documented in Figure 2: with few distinct confidence values, each bin absorbs a large share of the evaluation pool. Most of these points fall below the diagonal, indicating systematic overconfidence. CREDO's readout, by contrast, populates the full confidence range and tracks the diagonal closely. We use equal-mass bins here so that every marker represents a comparable number of samples; the quantitative ECE in Table 1 uses the standard ten equal-width bins defined in Appendix B.3.

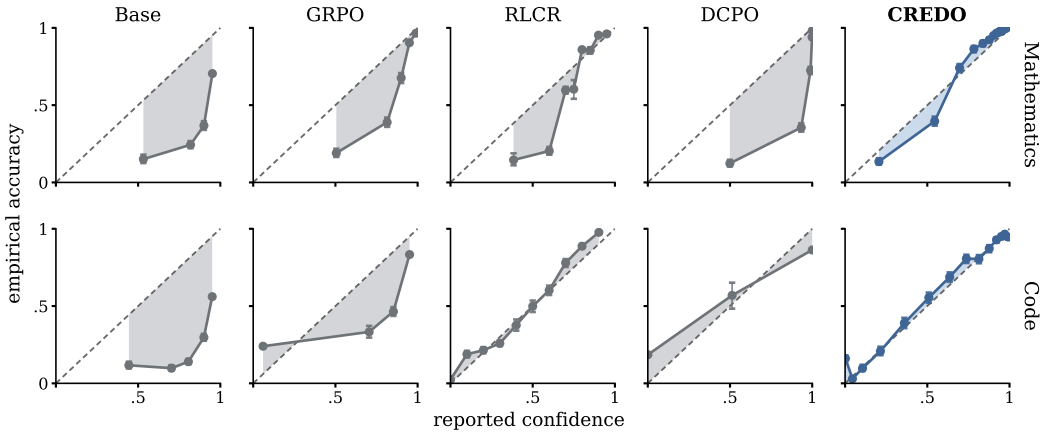


Figure 6: Calibration curves, pooled over the three seeds and over each domain's evaluation sets. Baselines report through their verbalized numeral, CREDO through its readout. Equal-mass bins; bars are Wilson 95% intervals (Wilson, 1927).

D Extended quantitative results

D.1 Per-set results

Tables 11–14 decompose Table 1 by evaluation set, in domain and under transfer. Entries are mean ± std over the three seeds. The Average column reproduces Table 1. DeepScaleR and DeepCoder name the in-distribution test splits.

Table 11: Per-set results on mathematics.

	DeepScaleR	MATH-500	AIME-24	AIME-25	AIME-26	AMC-23	AMC-24	Average
ACCURACY ↑								
Base	.603±.005	.821±.004	.240±.013	.172±.009	.136±.002	.599±.010	.503±.008	.439±.001
GRPO	.836±.006	.939±.001	.610±.021	.451±.021	.475±.030	.856±.014	.779±.014	.706±.009
RLCR	.821±.011	.932±.006	.564±.021	.382±.010	.465±.039	.851±.021	.762±.008	.682±.010
DCPO	.845±.003	.944±.007	.624±.035	.478±.009	.521±.019	.864±.018	.780±.029	.722±.008
Base-d	.603±.005	.821±.004	.240±.013	.172±.009	.136±.002	.599±.010	.503±.008	.439±.001
GRPO-d	.836±.006	.939±.001	.610±.021	.451±.021	.475±.030	.856±.014	.779±.014	.706±.009
RLCR-d	.821±.011	.932±.006	.564±.021	.382±.010	.465±.039	.851±.021	.762±.008	.682±.010
DCPO-d	.845±.003	.944±.007	.624±.035	.478±.009	.521±.019	.864±.018	.780±.029	.722±.008
CREDO	**.850±.003**	**.953±.002**	**.682±.032**	**.504±.015**	**.561±.042**	**.909±.015**	**.801±.008**	**.752±.013**
ECE ↓								
Base	.317±.005	.118±.004	.660±.011	.714±.018	.751±.005	.329±.009	.416±.013	.472±.002
GRPO	.093±.017	**.028±.007**	.254±.054	.396±.066	.388±.037	**.083±.019**	.149±.028	.199±.027
RLCR	.054±.003	.078±.033	.196±.029	.345±.030	.266±.032	.103±.017	**.072±.024**	.159±.012
DCPO	.129±.002	.048±.006	.279±.039	.388±.014	.379±.029	.107±.015	.189±.019	.217±.008
Base-d	.276±.005	.113±.006	.626±.015	.694±.010	.731±.005	.284±.010	.374±.011	.442±.001
GRPO-d	.058±.012	.076±.011	.252±.036	.390±.047	.368±.021	**.088±.053**	.109±.018	.192±.025
RLCR-d	.070±.005	.101±.017	.201±.002	.336±.025	.256±.024	.107±.014	**.073±.011**	.163±.005
DCPO-d	.065±.012	.083±.012	.249±.021	.359±.037	.329±.029	.121±.021	.108±.015	.188±.010
CREDO	**.033±.016**	.051±.021	**.128±.017**	**.129±.021**	**.171±.007**	.111±.029	.099±.026	**.103±.013**
AUROC ↑								
Base	.619±.018	.663±.018	.674±.015	.653±.025	.641±.011	.620±.024	.640±.017	.644±.004
GRPO	.699±.044	.732±.031	.820±.090	.826±.072	.840±.069	.787±.102	.722±.031	.775±.062
RLCR	.737±.021	.843±.008	.863±.029	.871±.003	.877±.037	.848±.018	.781±.017	.831±.009
DCPO	.752±.009	.813±.037	.913±.033	.882±.017	.921±.024	.863±.042	.793±.049	.848±.012
Base-d	.703±.007	.829±.009	.791±.020	.779±.020	.768±.021	.770±.007	.711±.039	.765±.010
GRPO-d	.768±.027	.866±.013	.900±.029	.900±.013	.906±.024	.907±.024	.804±.024	.864±.020
RLCR-d	.754±.011	.870±.009	.877±.020	.891±.010	.901±.024	.877±.019	.798±.007	.852±.006
DCPO-d	.749±.009	.847±.038	.903±.028	.903±.007	.938±.019	.887±.053	.776±.043	.858±.006
CREDO	**.846±.011**	**.926±.017**	**.961±.002**	**.941±.027**	**.968±.007**	**.954±.009**	**.935±.013**	**.933±.007**
BRIER ↓								
Base	.326±.004	.150±.002	.608±.010	.657±.018	.691±.006	.336±.010	.406±.011	.453±.002
GRPO	.133±.009	.051±.002	.251±.053	.359±.071	.348±.045	.116±.015	.172±.021	.204±.029
RLCR	.130±.009	.057±.005	.214±.012	.291±.024	.250±.018	.106±.011	.151±.006	.171±.009
DCPO	.130±.001	.047±.006	.250±.029	.345±.005	.330±.048	.104±.014	.181±.012	.198±.003
Base-d	.304±.004	.143±.003	.565±.013	.619±.010	.648±.006	.310±.008	.376±.009	.424±.001
GRPO-d	.127±.008	.054±.001	.262±.041	.360±.054	.340±.029	.111±.017	.169±.015	.203±.023
RLCR-d	.129±.008	.059±.004	.218±.015	.288±.022	.245±.016	.107±.011	.150±.004	.171±.009
DCPO-d	.122±.002	.050±.004	.249±.019	.339±.045	.308±.048	.106±.011	.165±.006	.191±.013
CREDO	**.093±.004**	**.030±.003**	**.091±.007**	**.122±.010**	**.114±.007**	**.060±.012**	**.092±.007**	**.086±.004**

Table 12: Per-set results on code, models trained on mathematics.

	Accuracy ↑					ECE ↓				
	DeepCoder	HumanEval+	LCB v5	LCB v6	Average	DeepCoder	HumanEval+	LCB v5	LCB v6	Average
GRPO	**.389±.015**	.852±.003	.400±.002	.338±.007	.495±.004	.418±.058	.086±.011	.360±.042	.396±.057	.315±.041
RLCR	**.388±.006**	.858±.003	**.400±.009**	**.365±.007**	**.502±.002**	.311±.027	**.054±.017**	.271±.017	.276±.022	.228±.014
DCPO	**.394±.022**	**.861±.004**	**.415±.025**	**.363±.019**	**.508±.014**	.436±.059	.153±.017	.344±.047	.387±.049	.330±.040
GRPO-d	**.389±.015**	.852±.003	.400±.002	.338±.007	.495±.004	.405±.040	**.050±.002**	.345±.031	.391±.042	.298±.028
RLCR-d	**.388±.006**	.858±.003	**.400±.009**	**.365±.007**	**.502±.002**	.315±.022	**.056±.011**	.270±.016	.285±.023	.232±.013
DCPO-d	**.394±.022**	**.861±.004**	**.415±.025**	**.363±.019**	**.508±.014**	.384±.036	**.050±.021**	.304±.031	.361±.039	.275±.027
CREDO	**.401±.025**	**.853±.016**	**.406±.030**	**.357±.018**	**.504±.021**	**.178±.019**	.062±.010	**.148±.016**	**.165±.025**	**.138±.016**

	AUROC ↑					Brier ↓				
	DeepCoder	HumanEval+	LCB v5	LCB v6	Average	DeepCoder	HumanEval+	LCB v5	LCB v6	Average
GRPO	.765±.033	.611±.036	.733±.026	.777±.041	.721±.032	.390±.058	.127±.005	.349±.039	.365±.059	.308±.040
RLCR	.832±.005	.717±.025	.795±.003	.826±.011	.792±.005	.279±.017	.118±.003	.263±.011	.255±.015	.229±.010
DCPO	.776±.052	.626±.058	.784±.025	.799±.031	.746±.038	.415±.047	.153±.018	.338±.036	.366±.044	.318±.032
GRPO-d	.827±.029	.702±.035	.794±.019	.846±.024	.792±.026	.380±.041	.122±.003	.338±.028	.358±.042	.299±.028
RLCR-d	.850±.006	**.754±.026**	.821±.006	.838±.017	.816±.003	.281±.015	**.110±.002**	.258±.011	.255±.017	.226±.011
DCPO-d	.803±.023	.688±.022	.807±.026	.812±.030	.778±.021	.366±.025	.119±.004	.309±.019	.337±.027	.283±.017
CREDO	**.869±.010**	**.735±.033**	**.858±.013**	**.883±.003**	**.836±.009**	**.191±.008**	.120±.011	**.176±.009**	**.173±.010**	**.165±.008**

Table 13: Per-set results on code.

	Accuracy ↑					ECE ↓				
	DeepCoder	HumanEval+	LCB v5	LCB v6	Average	DeepCoder	HumanEval+	LCB v5	LCB v6	Average
Base	.318±.004	.829±.007	.345±.012	.330±.007	.455±.001	.558±.005	.105±.008	.503±.012	.506±.001	.418±.001
GRPO	.475±.006	**.882±.011**	.522±.008	**.443±.007**	.581±.007	.229±.070	.113±.098	.187±.083	.199±.036	.182±.070
RLCR	.449±.008	**.872±.019**	.480±.017	.412±.016	.553±.013	**.049±.014**	.111±.017	**.068±.010**	**.071±.016**	**.075±.005**
DCPO	.477±.016	**.878±.016**	.525±.013	.422±.010	.576±.003	.167±.031	.112±.015	.178±.041	.156±.013	.153±.024
Base-d	.318±.004	.829±.007	.345±.012	.330±.007	.455±.001	.528±.004	.074±.017	.474±.012	.481±.003	.389±.002
GRPO-d	.475±.006	**.882±.011**	.522±.008	**.443±.007**	.581±.007	.218±.038	**.040±.019**	.150±.022	.213±.030	.155±.022
RLCR-d	.449±.008	**.872±.019**	.480±.017	.412±.016	.553±.013	**.056±.013**	.115±.014	**.066±.012**	**.082±.017**	.080±.004
DCPO-d	.477±.016	**.878±.016**	.525±.013	.422±.010	.576±.003	.164±.031	.112±.015	.176±.041	.157±.013	.152±.025
CREDO	**.509±.027**	**.881±.006**	**.563±.016**	**.448±.024**	**.600±.018**	**.049±.012**	**.062±.046**	**.069±.009**	**.076±.017**	**.064±.020**
	AUROC ↑					Brier ↓				
	DeepCoder	HumanEval+	LCB v5	LCB v6	Average	DeepCoder	HumanEval+	LCB v5	LCB v6	Average
Base	.701±.010	.628±.019	.695±.006	.735±.012	.690±.004	.513±.006	.146±.006	.464±.009	.454±.007	.395±.002
GRPO	.791±.094	.605±.033	.794±.083	.822±.057	.753±.061	.237±.067	.156±.097	.215±.074	.211±.035	.205±.066
RLCR	.845±.003	**.763±.028**	.825±.017	.825±.020	.815±.014	.156±.003	.110±.004	.166±.008	.150±.012	.146±.006
DCPO	.834±.032	.603±.040	.829±.040	.839±.017	.776±.025	.168±.030	.113±.014	.177±.041	.155±.014	.153±.024
Base-d	.796±.010	.719±.022	.782±.006	.832±.006	.782±.003	.482±.004	.141±.006	.437±.009	.432±.006	.373±.002
GRPO-d	.874±.033	.713±.047	.861±.031	.884±.027	.833±.034	.230±.036	**.098±.010**	.192±.024	.214±.030	.184±.025
RLCR-d	.848±.001	**.775±.027**	.826±.017	.821±.018	.818±.011	.155±.001	.110±.003	.165±.008	.151±.012	.145±.005
DCPO-d	**.883±.034**	.730±.081	**.887±.051**	**.896±.031**	**.849±.027**	.165±.031	.112±.015	.175±.041	.155±.013	.152±.025
CREDO	**.889±.025**	.736±.045	**.888±.023**	**.887±.029**	**.850±.027**	**.131±.012**	**.115±.037**	**.132±.010**	**.129±.010**	**.127±.016**

Table 14: Per-set results on mathematics, models trained on code.

	DeepScaleR	MATH-500	AIME-24	AIME-25	AIME-26	AMC-23	AMC-24	Average
ACCURACY ↑								
GRPO	**.743±.004**	**.892±.005**	.374±.016	**.276±.009**	**.288±.004**	**.728±.003**	**.644±.010**	**.564±.003**
RLCR	.701±.016	.873±.009	.324±.009	.233±.026	.207±.016	.688±.024	.579±.025	.515±.011
DCPO	.721±.009	**.894±.002**	.357±.010	.240±.024	.253±.013	**.727±.004**	**.644±.017**	.548±.003
GRPO-d	**.743±.004**	**.892±.005**	.374±.016	**.276±.009**	**.288±.004**	**.728±.003**	**.644±.010**	**.564±.003**
RLCR-d	.701±.016	.873±.009	.324±.009	.233±.026	.207±.016	.688±.024	.579±.025	.515±.011
DCPO-d	.721±.009	**.894±.002**	.357±.010	.240±.024	.253±.013	**.727±.004**	**.644±.017**	.548±.003
CREDO	.717±.021	**.888±.013**	**.401±.038**	**.268±.017**	.269±.023	**.737±.020**	**.642±.027**	**.560±.007**
ECE ↓								
GRPO	.178±.010	**.048±.009**	.496±.035	.587±.038	.571±.033	.200±.012	.263±.007	.335±.019
RLCR	**.034±.002**	.102±.010	.249±.014	.306±.020	.328±.019	**.055±.018**	**.117±.003**	.170±.004
DCPO	.224±.006	.092±.009	.237±.044	**.197±.061**	.237±.034	.216±.015	.228±.023	.205±.021
GRPO-d	.133±.011	.063±.019	.468±.032	.569±.023	.551±.022	.165±.014	.227±.008	.311±.017
RLCR-d	.049±.005	.115±.007	.250±.008	.308±.015	.331±.018	**.056±.014**	**.098±.020**	.172±.006
DCPO-d	.224±.006	.091±.008	.234±.039	**.196±.059**	.238±.031	.214±.016	.225±.023	.203±.020
CREDO	.127±.071	.109±.086	**.100±.036**	**.146±.071**	**.126±.067**	.118±.073	**.093±.063**	**.117±.030**
AUROC ↑								
GRPO	.682±.037	.739±.048	.773±.014	.780±.039	.767±.035	.682±.046	.712±.028	.734±.030
RLCR	.734±.013	.857±.005	.837±.029	.889±.041	.843±.017	.747±.020	.797±.032	.815±.016
DCPO	.686±.025	.747±.027	.769±.037	.837±.050	.815±.008	.711±.022	.720±.035	.755±.024
GRPO-d	**.755±.003**	.857±.015	.848±.024	.887±.009	.841±.004	.819±.014	.776±.015	.826±.004
RLCR-d	.748±.010	**.876±.007**	.856±.018	.907±.036	**.861±.025**	.762±.018	**.819±.027**	.833±.014
DCPO-d	.750±.005	.859±.028	.818±.019	**.916±.047**	**.866±.016**	**.830±.022**	**.803±.036**	.835±.004
CREDO	.736±.022	**.874±.011**	**.900±.009**	**.944±.013**	**.850±.026**	**.845±.028**	**.834±.049**	**.855±.013**
BRIER ↓								
GRPO	.207±.006	**.090±.006**	.452±.036	.525±.042	.510±.039	.220±.011	.268±.008	.325±.020
RLCR	**.176±.008**	**.090±.002**	.217±.011	.216±.011	.234±.012	.177±.008	.191±.008	.186±.007
DCPO	.228±.005	**.091±.008**	.235±.043	.197±.060	.237±.031	.216±.015	.229±.024	.205±.020
GRPO-d	.193±.007	**.087±.007**	.426±.031	.503±.028	.485±.030	.208±.005	.257±.006	.308±.015
RLCR-d	**.175±.007**	**.089±.002**	.213±.012	.214±.009	.232±.013	.175±.009	.186±.006	.184±.006
DCPO-d	.227±.006	**.091±.008**	.232±.039	.195±.059	.235±.030	.213±.015	.226±.023	.203±.020
CREDO	**.188±.034**	**.092±.042**	**.130±.007**	**.111±.025**	**.153±.019**	**.155±.032**	**.164±.029**	**.142±.013**

D.2 SENSITIVITY TO THE COEFFICIENTS

Table 15 gives every evaluated setting of the three coefficients.

Table 15: Every evaluated setting of the three coefficients of §3, all on seed 43.

	Mathematics				Code			
	Acc ↑	ECE ↓	AUROC ↑	Brier ↓	Acc ↑	ECE ↓	AUROC ↑	Brier ↓
WEIGHTING STRENGTH κ								
0	.721	.122	.904	.102	.572	.051	.843	.132
0.25	.753	.094	.933	.088	.584	.044	.854	.124
0.5	.749	.093	.924	.101	.614	.052	.869	.117
1	.730	.108	.924	.100	.591	.049	.858	.124
CALIBRATION WEIGHT α								
0	.696	.241	.880	.232	.611	.172	.857	.172
0.01	.715	.123	.912	.115	.581	.044	.851	.123
0.03	.753	.094	.933	.088	.614	.052	.869	.117
0.05	.708	.103	.910	.107	.583	.042	.857	.123
0.10	.716	.089	.915	.098	.577	.042	.867	.117
TARGET MIXTURE γ								
0	.732	.130	.918	.098	.593	.054	.854	.125
0.5	.753	.094	.933	.088	.614	.052	.869	.117
1	.738	.101	.924	.100	.587	.034	.855	.120