WHEN SELF-BELIEF MISLEADS: ACTIVE LABEL ACQUISITION FOR REINFORCEMENT LEARNING WITH VERIFIABLE REWARDS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large Language Models (LLMs) have achieved remarkable advancements in reasoning capabilities empowered by Reinforcement Learning with Verifiable Rewards (RLVR). Nonetheless, RLVR intrinsically relies on ground-truth labels for reward computation, the acquisition of which is often prohibitively expensive in real-world scenarios. While unsupervised RLVR paradigms attempt to circumvent this by training on pseudo-labels, they are notoriously susceptible to training collapse. Moreover, different samples often exhibit varying annotation values. In this paper, we propose Reinforcement Learning with Active Verifiable Rewards (RLAVR), which actively acquires ground-truth labels for a small set of selected samples and integrates them with pseudo-labels, thereby stabilizing training dynamics and improving performance under limited annotation budgets. To identify valuable samples, we propose the Corrective Advantage Gap (CAG) metric and analyze the sample-level supervision value. Building on this, we introduce Correction-Aware Reliability Estimation for RLAVR (CARE), which translates the oracle CAG criterion into a practical pre-query acquisition policy to substantially improve training stability. Extensive experiments across diverse domains, model families, and model scales demonstrate the effectiveness and generality of our approach. Our code is available at `https://anonymous.4open.science/r/CARE476-3B6B`.

1 INTRODUCTION

From DeepSeek-R1 to the GPT-O series, Reinforcement Learning with Verifiable Rewards (RLVR) (Wang et al., 2026; Guan et al., 2025; Yang et al., 2025b; Chen et al., 2025) has emerged as a critical method for advancing the reasoning capabilities of Large Language Models (LLMs). By utilizing stable feedback from the verifiers (e.g., rule-based verifiers), RLVR enables the LLM policy to continuously refine its behavior through an iterative cycle of exploration and policy optimization. Although effective, RLVR faces a fundamental challenge in its dependence on ground-truth labels to compute rewards, such as standard answers for math problems or test cases for coding tasks. These labels are often costly to obtain, and the resulting reliance on external supervision constrains the policy's ability to self-improve and explore autonomously.

Unsupervised RLVR (Zuo et al., 2025; He et al., 2026) takes a significant step toward eliminating the reliance on supervised data. It leverages intrinsic model signals, such as majority voting (Zhang et al., 2025d; Liao et al., 2026) and self-consistency (Zhao et al., 2025; Ghimire et al., 2026), as pseudo-labels for reward computation. While promising, recent studies suggest that the model's internal beliefs are not always reliable (He et al., 2026; Tan et al., 2025; Zhou et al., 2025). Pure internal signal-based supervision may form a dead loop that reinforces erroneous beliefs and ultimately leads to training collapse (shown in section 4.1). This failure mode suggests the need for external supervision, yet the supervision must be used sparingly: acquiring ground-truth labels for all samples would lead to an expensive annotation cost. Consequently, a natural question arises:

How can we actively acquire sparse ground-truth labels for a small set of critical samples and combine them with pseudo-labels to provide reliable training signals?

To answer this question, this paper makes the following three technical contributions.

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.

New Problem and Justification. Motivated by this need, we introduce Reinforcement Learning with Active Verifiable Rewards (RLAVR), to our knowledge, the first active label-acquisition problem for online RLVR. Specifically, we formulate RLAVR as a budget-constrained label assignment problem: under a limited annotation budget, the learner actively selects which samples to query for ground-truth annotations and combines the verified labels with pseudo-labels to construct mixed training signals. To justify this problem setting, we examine the supervision value of different samples, measured by the performance gain obtained from correcting erroneous pseudo-labels with ground-truth labels. Our analysis reveals that samples exhibit highly heterogeneous supervision value, suggesting that sparse annotations should be allocated selectively rather than uniformly.

CAG-Guided Acquisition Framework. To solve the proposed problem, we first propose the Corrective Advantage Gap (CAG), a metric that characterizes the discrepancy between the advantage vectors induced by ground-truth labels and pseudo-labels. Empirically, CAG faithfully reflects supervision value: across different settings, selecting samples with larger CAG consistently yields stronger performance gains than alternative selection strategies (shown in section 4.2). However, CAG requires ground-truth labels and therefore cannot be directly computed before annotation. Therefore, we further propose Correction-Aware Reliability Estimation for RLAVR (CARE), a method for introducing sparse supervision into RLAVR by predicting sample reliability and supervision value before querying. Specifically, CARE trains a dedicated two-stage classifier network: the first-stage classifier estimates pseudo-label reliability and retains reliable pseudo-label signals, while the second-stage classifier predicts expected CAG for unreliable samples. Guided by the classifiers, CARE prioritizes samples whose pseudo-labels are likely to induce high supervision value, thereby fully exploiting reliable pseudo-labeled samples while suppressing erroneous internal reward signals.

Theoretical and Empirical Validation. Theoretically, we justify CAG by deriving a CAG-dependent upper bound on gradient alignment: larger advantage deviations imply a smaller guaranteed cosine alignment between ground-truth-induced and pseudo-label-induced GRPO gradients. This explains why high-CAG samples can introduce more misaligned optimization signals if left uncorrected. Empirically, with only a 20% annotation budget, CARE consistently improves over TTRL across Qwen3-Base models at 1.7B, 4B, and 8B scales on six mathematical reasoning benchmarks, while substantially narrowing the gap to fully supervised RLVR. Further ablations validate the effectiveness of reliability estimation, CAG prediction, and sample dropping.

2 Related Work

2.1 Unsupervised and Semi-supervised RLVR

The success of DeepSeek-R1 (Guo et al., 2025) has demonstrated the substantial potential of RLVR (Shao et al., 2024) for enhancing the reasoning capabilities of LLMs through reinforcement learning. Building on this insight, a growing body of work (Lu et al., 2026; Lin et al., 2025b;a) has explored the RLVR paradigm and achieved promising results. However, standard RLVR relies on ground-truth labels to compute reward signals for training, whereas in many practical scenarios such labels are difficult or costly to obtain. To enable training in unlabeled settings, numerous studies have investigated unsupervised RLVR, where pseudo-labels are constructed from signals such as majority voting (Zuo et al., 2025; Wang et al., 2025), self-certainty (Zhao et al., 2025; Ghimire et al., 2026), and entropy (Zhang et al., 2025c; Agarwal et al., 2025; Prabhudesai et al., 2025). Some methods further introduce more refined designs by modeling answer distributions (Yu et al., 2025b) or exploiting discrepancies between correct and incorrect reasoning trajectories (Zhang et al., 2025b), thereby yielding additional performance gains. Concurrently, several works have noted that fully label-free paradigms are prone to training instability or even collapse (He et al., 2026), and have mitigated this by designing specific reward mechanisms (Tan et al., 2025; Zhou et al., 2025) or trained verifiers (Pan et al., 2026). As an alternative approach, semi-supervised RLVR methods (Yang et al., 2025c; Luo et al., 2026) incorporate a limited amount of ground-truth supervision to guide sample selection or pseudo-label generation, leading to more effective training. Although RLAVR also involves mixed training with pseudo-labels and ground-truth labels, its focus lies in actively selecting valuable samples for annotation. Semi-supervised RLVR methods typically assume a given labeled set and focus on exploiting existing labeled and unlabeled data. In contrast, RLAVR focuses on deciding which unlabeled samples to annotate and combining their ground-truth labels with pseudo-labels to improve training under limited annotation budgets.

2.2 Active Learning

Active learning (Aggarwal et al., 2014) aims to reduce annotation costs by selecting the most informative samples for labeling, thereby maximizing model improvement under limited supervision. To this end, a variety of acquisition strategies have been explored in the literature, including uncertainty sampling (Wang & Shang, 2014; Geifman & El-Yaniv, 2017), query-by-committee (Kee et al., 2018), expected model change (Cai et al., 2013), expected error reduction (Mussmann et al., 2022), and density-aware methods (Wang et al., 2021). In the LLM era, where high-quality human annotation is substantially more expensive, active learning has become increasingly attractive as a mechanism for improving data efficiency while maintaining or even enhancing downstream performance. Recent studies have incorporated active selection into Supervised Fine-Tuning (SFT) (Bayer et al., 2026), alignment and post-training pipelines, particularly in Reinforcement Learning from Human Feedback (RLHF) (Bai et al., 2022), where samples are selected based on signals such as response redundancy (Feng et al., 2025) or model uncertainty (Duan et al., 2025) to improve preference data collection efficiency (Liu et al., 2024; Mehta et al., 2023). Related post-training work includes active label acquisition for Direct Preference Optimization (DPO) (Rafailov et al., 2023; Ji et al., 2024) and active data filtering for stabilizing RL (Yi et al., 2026). However, the latter requires labels for all samples before filtering. In contrast, RLAVR integrates active learning into online RL training to selectively acquire labels under a limited budget while continuously improving capabilities.

3 Background and Problem Formulation

3.1 Group Relative Policy Optimization

Group Relative Policy Optimization (GRPO) (Shao et al., 2024) is a popular policy optimization method for RLVR, which first introduces the group-based advantage estimator. Specifically, for each prompt x_i, the current policy samples a group of G rollouts, denoted by $\mathcal{O}_i = \{o_{i,g}\}_{g=1}^{G}$, and each rollout $o_{i,g}$ is assigned a scalar reward $r_{i,g}$ by a rule-based verifier. GRPO computes group-relative advantages by normalizing rewards within each rollout group,

$$A_{i,g} = \frac{r_{i,g} - \mu_i(r)}{\sigma_i(r) + \epsilon}, \tag{1}$$

where $\mu_i(r)$ and $\sigma_i(r)$ denote the mean and standard deviation of $\{r_{i,g}\}_{g=1}^{G}$, respectively, and ϵ is a small constant for numerical stability. Since rewards are defined at the rollout level, the same advantage $A_{i,g}$ is shared across all tokens in $o_{i,g}$. The policy is then optimized by

$$\mathcal{J}_{\mathrm{GRPO}}(\theta) = \mathbb{E}_{x_i,\mathcal{O}_i}\left[\frac{1}{G}\sum_{g=1}^{G}\frac{1}{|o_{i,g}|}\sum_{t=1}^{|o_{i,g}|}\Big(\min\big(\rho_{i,g,t}A_{i,g}, \mathrm{clip}(\rho_{i,g,t}, 1-\delta, 1+\delta)A_{i,g}\big)]\Big)\right], \tag{2}$$

where $\rho_{i,g,t} = \pi_\theta(o_{i,g,t} \mid x_i, o_{i,g,<t}) / \pi_{\theta_{\mathrm{old}}}(o_{i,g,t} \mid x_i, o_{i,g,<t})$ is the token-level importance ratio, and δ controls the clipping range.

3.2 Reinforcement Learning with Active Verifiable Rewards

We introduce Reinforcement Learning with Active Verifiable Rewards (RLAVR). The studied problem can be formulated as a budget-constraint label assignment problem. Formally, assume the training proceeds for T rounds. At each round t, a batch of prompts $B_t = \{x_1^t, \ldots, x_n^t\}$ is first sampled. Then, for each prompt x_i^t, the policy π_{θ_t} generates a group of responses $\{o_{i,g}^t\}_{g=1}^{G}$, resulting in a set of prompt-response pairs $C_t = \{(x_i^t, \{o_{i,g}^t\}_{g=1}^{G})\}_{i=1}^{n}$. We hope to select a subset of informative samples $I_t^L \subseteq [n]$ for label querying and obtain ground-truth labels y_i^t for $i \in I_t^L$, while for the remaining samples $i \notin I_t^L$, pseudo-labels $\hat{y}_i^t = \mathcal{A}(x_i^t, \{o_{i,g}^t\})$ are constructed via an aggregation function. All labeled samples, including both queried and pseudo-labeled ones, are then used to form a training batch for reward and advantage computation, based on which the policy is updated from π_{θ_t} to $\pi_{\theta_{t+1}}$ according to Eq. (2). Our goal is to maximize the performance of the best checkpoint

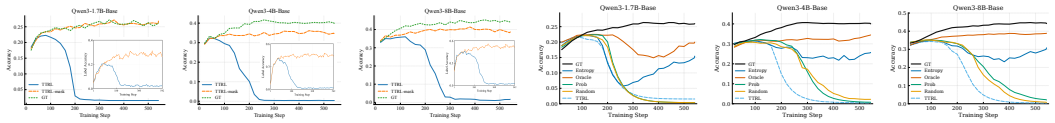

Figure 1: **(Left)** Training dynamics across three Qwen3 models on math tasks. **(Right)** Comparison of label acquisition strategies across three Qwen3 models on math tasks.

over T rounds under a predefined labeling budget constraint p:

$$\max_{\{I_t^L\}_{t=1}^T} \max_{t\in[T]} \mathrm{Perf}(\pi_{\theta_t}) \quad \text{s.t.} \quad \frac{1}{nT}\sum_{t=1}^{T} |I_t^L| \le p, \tag{3}$$

where $\mathrm{Perf}(\pi_{\theta_t})$ denotes the evaluation performance of checkpoint π_{θ_t}. It is worth noting that RLAVR differs from semi-supervised RLVR, where labeled samples are predefined. Instead, the goal is to actively select samples for annotation during online training.

4 Preliminary Analysis of RLAVR

4.1 Reliability Analysis for Unsupervised RLVR Training

To assess the reliability of the model's internal beliefs, we compared the GRPO curves trained with true labels, pseudo-labels generated through majority voting, and pseudo-labels with errors masked (Details and additional results are provided in D.7.) As shown in figure 1 (Left), the performance of the model trained with pseudo-labels rapidly deteriorates after an initial phase of improvement, aligning with previous studies (He et al., 2026; Tan et al., 2025; Zhou et al., 2025) that the model's internal signals are not always reliable. Notably, masking erroneous pseudo-labels stabilizes the model's performance, further attributing the primary cause of training collapse to the misguidance of incorrect internal beliefs. The change in pseudo-label accuracy reveals the underlying mechanism: before the sharp performance decline, pseudo-label accuracy steadily decreases, creating a vicious cycle that reinforces incorrect internal beliefs and ultimately leads to training collapse.

Takeaway : Unsupervised RLVR driven by internal beliefs is not always reliable. Erroneous beliefs can be continuously reinforced, causing the policy to deviate from the correct optimization direction and eventually collapse.

4.2 Supervision Value Analysis of Erroneous Samples

The preceding results indicate that erroneous pseudo-labels are the main driver of training collapse. As a preliminary study, we further ask whether correcting different erroneous pseudo-labels leads to similar performance gains, which we interpret as their supervision value. Specifically, given a limited annotation budget, we compare the effect of correcting different subsets of incorrect pseudo-labels.

Intuitively, an erroneous sample should have a high supervision value if its pseudo-label induces a learning signal that differs substantially from the one induced by the ground-truth label. Following this idea, we quantify supervision value by the advantage difference between ground-truth labels and pseudo-labels. Specifically, for the G responses to a prompt x_i, let $A_i = (A_{i,1}, \ldots, A_{i,G})$ and $\tilde{A}_i = (\tilde{A}_{i,1}, \ldots, \tilde{A}_{i,G})$ denote the advantage vectors computed with ground-truth and pseudo-labels, respectively, as per Eq. (1). We define the *Corrective Advantage Gap* (CAG) score as

$$\mathrm{s}_i = \|A_i - \tilde{A}_i\|_2. \tag{4}$$

As shown in figure 1 (Right), assigning ground-truth labels by CAG (Oracle) yields the largest performance gain among the compared selection criteria, including random and response-distribution-based selection. Notably, the other strategies still suffer from training collapse, suggesting that CAG more faithfully identifies erroneous pseudo-labels with high supervision value. Surprisingly, even when the remaining pseudo-labeled samples are masked out and training uses only the acquired ground-truth

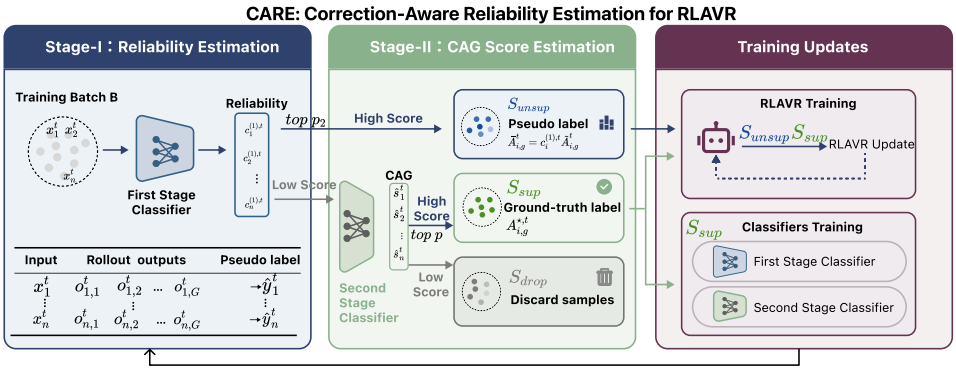

Figure 2: Overall pipeline of CARE. The Stage-I classifier predicts sample reliability, the Stage-II classifier predicts CAG to select informative samples for annotation, and the resulting labeled and pseudo-labeled data are used to jointly train the LLM and the classifiers.

labels, the CAG-selected samples still achieve strong performance gains (see Appendix D.8 for details and more results). This further shows that high-CAG samples are not only useful for removing harmful pseudo-labels, but also consistently provide informative supervision signals for learning.

Takeaway: Different samples carry substantially different supervision value, and CAG provides a reliable signal for identifying samples whose annotation yields larger learning gains.

5 METHODOLOGY

Based on the preceding analysis, CAG provides an effective criterion for measuring the supervision value of correcting a pseudo-label. However, it is an oracle quantity at acquisition time, as CAG requires the ground-truth label. We therefore propose Correction-Aware Reliability Estimation for RLAVR (CARE), which turns the oracle CAG criterion into a practical pre-query acquisition policy, as illustrated in Figure 2. CARE estimates pseudo-label reliability and expected supervision value from rollout signals, retaining reliable pseudo-labeled samples for unsupervised optimization while querying unreliable, high-value samples. Algorithm 2 summarizes the full procedure.

5.1 OVERALL FRAMEWORK

Although CAG shows promise as an annotation-value criterion, applying it uniformly to all samples ignores the hierarchical nature of the budgeted decision. Specifically, when a pseudo-label is correct, it induces the same advantage vector as the ground-truth label and thus has zero CAG; such samples should be retained for low-cost unsupervised training rather than queried. CAG ranking is meaningful only within the unreliable region: among samples with incorrect pseudo-labels, high-CAG ones are worth annotating because their ground-truth labels would bring a large performance gain, while low-CAG ones can be discarded due to their wrong learning signals. CARE therefore decomposes acquisition into two conditional steps. Stage-I estimates pseudo-label reliability to retain reliable samples, and Stage-II ranks the predicted unreliable samples by expected CAG for annotation.

5.1.1 CASCADED CLASSIFIERS

The cascade implements the conditional acquisition rule described above. At round t, for each prompt $x_i^t \in \mathcal{B}^t$ with rollouts $\{o_{i,g}^t\}_{g=1}^G$, CARE first builds a rollout-derived representation $\phi_i^t \in \mathbb{R}^d$ by

$$\phi_i^t = E(x_i^t, \{o_{i,g}^t\}_{g=1}^G), \tag{5}$$

where $E(\cdot)$ is a lightweight feature encoder detailed in Section 5.2. CARE then applies two neural classifiers: a reliability classifier $f_{\psi_1^t}^{\text{rel}}$ and a supervision-value classifier $f_{\psi_2^t}^{\text{cag}}$. Together, they partition $\mathcal{B}^t$ into three disjoint subsets: reliable pseudo-labeled samples $\mathcal{S}_{\text{unsup}}^t$, unreliable high-value samples selected for annotation $\mathcal{S}_{\text{sup}}^t$, and unreliable low-value samples discarded from policy optimization

$\mathcal{S}^t_{\text{drop}}$. In Stage-I, the reliability classifier outputs $c_i^{(1),t} = f^{\text{rel}}_{\psi_1^t}(\phi_i^t) \in [0,1]$, interpreted as the probability that the pseudo-label $\hat{y}_i^t$ is equal to the ground truth label y_i^t. Prompts with the highest p_2 reliability scores are assigned to $\mathcal{S}^t_{\text{unsup}}$, and the rest are passed to Stage-II. In Stage-II, CARE estimates the supervision value only within this predicted unreliable region. Given a wrong majority-vote pseudo-label, the CAG is determined by the known majority cluster and the number of correct non-majority responses. Thus, instead of regressing CAG directly or predicting the ground-truth answer identity, $f^{\text{cag}}_{\psi_2^t}$ predicts a distribution over this latent count:

$$\mathbf{d}_i^t = f^{\text{cag}}_{\psi_2^t}(\phi_i^t), \qquad \hat{s}_i^t = \sum_{k \in \mathcal{K}_i^t} d_{i,k}^t \, s_i^t(k), \tag{6}$$

where $d_{i,k}^t$ is the predicted probability that exactly k non-majority responses are correct, $\mathcal{K}_i^t$ is the admissible set of such counts, and $s_i^t(k)$ is the corresponding class-wise CAG. Among the Stage-II candidates, prompts with the highest expected CAG scores under the annotation budget p are assigned to $\mathcal{S}^t_{\text{sup}}$, while the rest are assigned to $\mathcal{S}^t_{\text{drop}}$.

5.1.2 Online Training Procedure

After partitioning, CARE updates the policy using $\mathcal{S} * \text{sup}^t \cup \mathcal{S} * \text{unsup}^t$; samples in $\mathcal{S}^t_{\text{drop}}$ are excluded. For annotated samples, rewards and advantages are computed with ground-truth labels. For reliable pseudo-labeled samples, CARE downweights the pseudo-label advantage by the Stage-I reliability score:

$$A_{i,g}^{\text{mix},t} = \begin{cases} A_{i,g}^{\star,t}, & i \in \mathcal{S}^t_{\text{sup}} \text{ (ground-truth label } y_i^t), \\ c_i^{(1),t} \tilde{A}_{i,g}^t, & i \in \mathcal{S}^t_{\text{unsup}} \text{ (pseudo-label } \hat{y}_i^t). \end{cases} \tag{7}$$

CARE then optimizes Eq. (2) using $A_{i,g}^{\text{mix},t}$. After the policy update, newly annotated samples are added to the labeled buffer and provide online supervision for both classifiers, yielding the next-round parameters ψ_1^{t+1} and ψ_2^{t+1}.

5.2 Detail Implementations

Cascaded Classifier Representation. The cascaded classifiers use lightweight rollout-derived representations that can be obtained with negligible additional cost. Following prior observations that hidden states, response length, and answer consistency are correlated with response correctness (Zhang et al., 2025a; Azaria & Mitchell, 2023; Su et al., 2025; Manakul et al., 2023), we construct $\phi_i^t = E(x_i^t, \{o_{i,g}^t\}_{g=1}^G)$ from two types of signals: prompt-level statistics summarizing the rollout set and response-level statistics describing individual responses. Specifically, we use the final-layer hidden states of the last token, the valid rollout ratio, the distribution of answer clusters, and normalized response lengths. These signals are directly available from standard rollout statistics or forward-pass hidden states, and therefore introduce negligible additional overhead. Detailed feature definitions are provided in Appendix C.2.

Cascaded Classifiers Network Design. To keep the acquisition classifiers lightweight, we implement both stages with MLPs trained using cross-entropy loss. To improve classifier learning, we also use an auxiliary response-level head to predict each response's correctness:

$$\mathcal{L}_{\text{stage-1}} = \mathcal{L}_{\text{classifier}} + \lambda_{\text{aux}} \mathcal{L}_{\text{aux}}, \tag{8}$$

where $\mathcal{L}_{\text{classifier}}$ is the main classification loss and $\mathcal{L}_{\text{aux}}$ is the auxiliary response-level correctness loss. We further use a training buffer and class-balanced sampling to improve sample efficiency and robustness. More details are provided in Appendix C.2.

6 Theoretical Analysis

In this section, we theoretically analyze the proposed CAG metric by connecting it to the deviation of the optimization direction. For a prompt x and its sampled responses $\{o_i\}_{i=1}^G$, let g_x denote the GRPO gradient computed with ground-truth rewards in equation 2, and let $\hat{g}_x$ denote the gradient computed

with pseudo rewards. We measure their directional alignment by $\cos\theta_x = \langle g_x, \hat{g}_x\rangle/(\|g_x\|_2\|\hat{g}_x\|_2)$. For simplicity, we consider the strict on-policy GRPO setting, where the gradient is determined by the group-normalized advantage vector. Specifically, for an advantage vector $A = (A_1, \ldots, A_G)$, the corresponding gradient can be written as $g(A) = \frac{1}{G}\sum_{i=1}^{G} A_i s_i$, where $s_i = \nabla_\theta \sum_{j=1}^{|o_i|} \frac{1}{|o_i|}\pi_\theta(o_{i,j} \mid x, o_{i,<j})$ is the score vector of response o_i. Let $S = [s_1, \ldots, s_G]$ be the score matrix and define $K_x = S^\top S$. Lemma A.1, stated and proved in Appendix A.1, shows that gradient alignment equals the cosine similarity between the transformed advantage vectors $K_x^{1/2}A$ and $K_x^{1/2}\hat{A}$. The next theorem directly connects the proposed CAG metric, i.e., the advantage deviation, to the alignment of the induced optimization direction.

Theorem 6.1 (CAG upper-bounds gradient alignment)**.** *Let $\mathcal{H} = \{v \in \mathbb{R}^G : \mathbf{1}^\top v = 0\}$ be the centered subspace. Assume $K_x \succ 0$ and define $\lambda_{\max}(K_x|_{\mathcal{H}}) = \max_{\substack{v\in\mathcal{H}\\ v\neq 0}} \frac{v^\top K_x v}{v^\top v}$, $\lambda_{\min}(K_x|_{\mathcal{H}}) = \min_{\substack{v\in\mathcal{H}\\ v\neq 0}} \frac{v^\top K_x v}{v^\top v}$, and $\kappa(x) = \frac{\lambda_{\max}(K_x|_{\mathcal{H}})}{\lambda_{\min}(K_x|_{\mathcal{H}})}$. Then, for any nonzero CAG score $d_x = \|A - \hat{A}\|_2 > 0$, the gradient alignment satisfies*

$$\cos\theta_x \le 1 - \frac{2}{4\kappa(x)\frac{G}{d_x^2} - (\kappa(x) - 1)}. \tag{9}$$

Remark. (Proof deferred to Appendix A.2.) The theorem provides theoretical insight for CAG being a meaningful indicator of pseudo-label quality: a larger advantage deviation d_x yields a smaller guaranteed upper bound on the gradient alignment, implying that pseudo labels may introduce a more misaligned optimization signal. In particular, the upper bound becomes negative when $d_x^2 > 4\kappa(x)G/(\kappa(x)+1)$, which guarantees $\cos\theta_x < 0$ and hence an obtuse angle between the ground-truth and pseudo-label gradients. The bound also shows that this relationship depends on the geometry of the sampled responses through the condition number $\kappa(x)$. When $\kappa(x)$ is large, the same advantage deviation can have a less predictable effect on the gradient direction due to anisotropic score-vector geometry. If $\lambda_{\min}(K_x|_{\mathcal{H}}) = 0$, $\kappa(x)$ is undefined and the theorem does not apply; as $\lambda_{\min}(K_x|_{\mathcal{H}})$ approaches zero, $\kappa(x)$ diverges and the upper bound becomes vacuous. Notably, when $\kappa(x)$ takes the minimum value, i.e., $\kappa(x) = 1$, the bound becomes tight and reduces to the exact identity $\cos\theta_x = 1 - \frac{d_x^2}{2G}$. Appendix D.3 empirically examines this non-degeneracy condition.

Proposition 6.2 (CAG characterizes one-step corrective benefit)**.** *Assume that $J_x^\star$ has a bounded Hessian in a neighborhood of θ, the ground-truth and pseudo-reward groups are non-degenerate and exactly advantage-normalized, and the response-level NTK is locally isotropic on the centered subspace, $K_x|_{\mathcal{H}} = \kappa_{\text{iso}} I$ for $\kappa_{\text{iso}} > 0$. Under the strict on-policy update, for sufficiently small η,*

$$V_{\text{corr}}(x) = \frac{\eta\kappa_{\text{iso}}}{2G^2}\,\mathrm{CAG}(x)^2 + \mathcal{O}(\eta^2). \tag{10}$$

Remark. (Proof in Appendix A.3.) Under local isotropy, Proposition A.2 shows that CAG ranks samples by corrective benefit to first order. Thus, CAG measures *corrective supervision value* rather than sample difficulty alone.

7 Experiment

7.1 Experiment Setup

Baselines. We compare four types of baselines: (1) **Unsupervised training**: TTRL; (2) **Semi-supervised RLVR**: Random uniformly selects samples for annotation, while TraPO (Yang et al., 2025c) uses the same random selection and treats the rest as unlabeled; (3) **Active learning**: Entropy uses the answer-cluster distribution, Prob uses average response probability, and Oracle uses the ground-truth CAG score; and (4) **Fully supervised training**: GT uses ground-truth labels for all samples. For Random and active-learning methods, selected samples receive ground-truth annotations and the rest use majority-voting labels. All methods are trained with GRPO (Shao et al., 2024).

Implementation and Evaluation. Following URLVR (He et al., 2026), we use DAPO-17k (Yu et al., 2025a) for training and optimize all methods for two epochs with Qwen3-Base (Yang et al.,

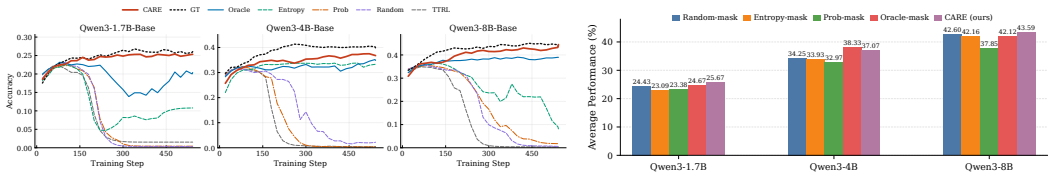

Figure 3: **(Left)** Comparison of training curves among baseline methods on math tasks. **(Right)** Comparison of baseline training performance on math tasks after removing unsupervised samples.

2025a) (1.7B, 4B, and 8B) and Phi4-mini-instruct (Abdin et al., 2024), under a labeling budget of $p = 20\%$. We evaluate on out-of-domain mathematical reasoning benchmarks: MATH500 (Hendrycks et al., 2021), AIME24, AIME25, AMC23, Hmmt Feb 25, and Olympiad (He et al., 2024). We use temperature 0.6 and average@8, increasing to average@32 for the smaller AIME, AMC, and Hmmt sets for stability. Correctness is measured against reference answers. Further details and results are in Appendices C.2, D.11, and D.12.

7.2 Main Result

CARE improves training effectiveness and stabilizes training. As shown in Table 1, CARE consistently outperforms TTRL, Random, TraPO, and the active-learning baselines Entropy, Prob, and Oracle on average. Figure 3 (left) shows that most methods suffer from training collapse under erroneous labels. Oracle mitigates this by annotating samples most misleading to the training signal, supporting the CAG metric, but requires ground-truth labels to compute CAG. CARE instead predicts CAG with classifiers and selects high-confidence samples for unsupervised training, stabilizing training and approaching the GT upper bound while outperforming other baselines.

Proper use of unsupervised samples has the potential to improve training. To disentangle the effect of training collapse on active learning methods, we further evaluate Random, Prob, Entropy, and Oracle without their unsupervised samples, training them only on the annotated samples selected by each method. As shown in Figure 3 (right), these variants consistently outperform their counterparts trained with additional unsupervised samples, suggesting that excessive erroneous pseudo-labeled samples may even hinder training. In contrast, CARE achieves better performance than most baselines by leveraging high-confidence unsupervised samples, demonstrating both the value of properly utilizing unsupervised data and the effectiveness of CARE.

CARE incurs little computational overhead. To assess computational overhead, we compare per-step training time across active learning methods using Qwen3-4B-Base. As shown in Figure 4 (Left), CARE has rollout and policy optimization costs comparable to other methods because classifier features are extracted during rollout and the classifier is lightweight. Dropping samples filtered by the second-stage classifier further reduces overhead by 21.03%. Appendix D.1 reports total training time, where CARE remains relatively efficient even without sample dropping for fair comparison.

7.3 Ablation Study

Validation of component design. We conduct ablations for each CARE component, with 4B results in Figure 4(Middle) and others in Appendix D.9. We compare classification-based CAG prediction with regression (*CARE-mse*), remove pseudo-labeled samples (*w/o ps*), replace the second-stage classifier with random selection of $p\%$ remaining samples (*w/o stage2*), and ablate the auxiliary and class-balancing losses (*w/o aux*, *w/o cb*). *CARE-mse* yields worse CAG prediction and downstream performance, supporting classification-based prediction. Both *w/o ps* and *w/o stage2* consistently degrade performance, confirming the value of low-risk pseudo-labeled samples and second-stage selection. While *w/o aux* and *w/o cb* cause only minor changes, combining both losses improves average performance. We further apply CARE's filtering and downweighting mechanisms to Entropy and Prob in Appendix D.5 to isolate their effects.

Analysis of Annotation Budget Ratio p. We compare CARE with the three strongest baselines on Qwen3-1.7B-Base model under varying annotation budget ratios p in Figure 4(Right). CARE consistently performs best across all budgets. As p increases, performance improves: CARE matches

Table 1: Math-task performance (%) of Qwen3-Base models. Vanilla and GT are excluded from ranking. **Bold** and underline denote the best and second-best results.

Methods	AIME24	AIME25	MATH500	AMC23	Hmmt25	Olympiad	Average
Qwen3-1.7B-Base							
Vanilla	0.25	0.24	40.45	25.62	0.00	18.38	14.89
GT	8.02	4.69	67.65	47.73	1.35	30.70	26.69
TTRL	6.88	4.17	63.65	36.33	0.10	27.99	23.18
Random	6.15	5.10	62.83	35.70	0.31	27.08	22.86
TraPO	8.02	5.10	65.25	37.10	**0.83**	28.96	24.21
Entropy	5.52	4.79	62.02	35.47	0.10	27.28	22.53
Prob	5.94	3.44	62.73	33.98	0.10	27.19	22.23
Oracle	5.73	4.58	62.88	36.02	0.10	27.27	22.76
CARE (ours)	**9.27**	**5.83**	**67.50**	**40.08**	0.73	**30.62**	**25.67**
Qwen3-4B-Base							
Vanilla	8.02	3.44	31.17	22.73	0.52	22.46	14.72
GT	22.18	20.62	83.78	60.62	8.44	51.17	41.14
TTRL	11.35	7.92	76.23	53.52	1.15	42.01	32.03
Random	10.42	8.12	76.20	55.23	0.42	42.54	32.15
TraPO	**15.31**	10.83	78.32	56.64	1.67	45.48	34.71
Entropy	11.67	7.71	76.42	51.09	1.15	41.20	31.54
Prob	11.98	10.42	76.48	54.45	0.83	42.29	32.74
Oracle	12.60	9.05	79.60	60.08	3.02	44.95	34.95
CARE (ours)	14.79	**16.04**	**80.18**	**61.02**	**4.27**	**46.12**	**37.07**
Qwen3-8B-Base							
Vanilla	10.83	7.29	55.40	43.13	1.46	34.23	25.39
GT	24.79	19.06	87.08	70.31	10.63	55.70	44.59
TTRL	16.15	13.96	80.45	60.16	2.92	45.47	36.52
Random	14.58	11.67	79.70	58.91	2.19	45.71	35.46
TraPO	23.44	19.17	84.22	**67.42**	7.81	53.31	42.56
Entropy	16.15	13.02	80.67	61.09	4.06	46.28	36.88
Prob	15.52	12.92	80.12	60.47	1.98	45.91	36.15
Oracle	18.96	15.83	82.17	66.41	5.62	48.06	39.51
CARE (ours)	**24.79**	**20.83**	**85.60**	65.23	**10.73**	**54.34**	**43.59**

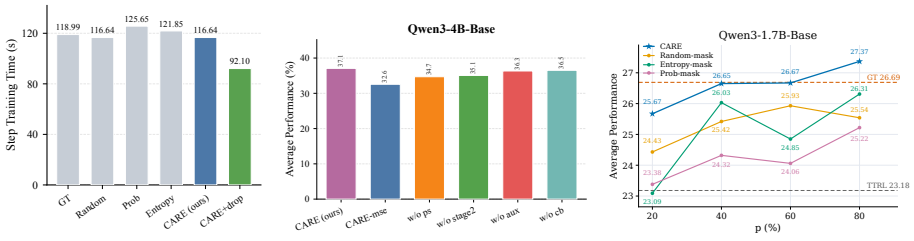

Figure 4: **(Left)** Comparison of step training time across different baselines on math tasks. **(Middle)** Ablation study of CARE on math tasks. **(Right)** Hyperparameter analysis of p on math tasks.

GT with only 40% of annotations and surpasses GT at 80%. These results show that CARE remains effective under limited annotation budgets and maintains consistent advantages over baselines.

8 Conclusion

In this paper, we propose RLAVR, a new setting that actively acquires ground-truth annotations for a subset of samples under a limited annotation budget and integrates the verified labels with pseudo-labels for training, thereby achieving competitive performance with high sample efficiency. We investigate the underlying causes of training collapse in current unsupervised RLVR pipelines and introduce a novel metric, CAG, to quantify the annotation value of individual samples, further substantiating its validity through theoretical analysis. Building upon these insights, we propose CARE, a framework that leverages a cascaded classifier network to predict CAG, perform label selection, and conduct RLAVR training. This mechanism simultaneously stabilizes the training dynamics and enhances overall performance. Extensive empirical evaluations across diverse domains, model scales, and model families unequivocally demonstrate the efficacy of CARE.

AI USE STATEMENT

We used LLM tools to polish parts of this manuscript and to help run some experiments that we designed. We checked the experimental procedures and results ourselves and take responsibility for the paper's content and claims.

REFERENCES

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024.

Shivam Agarwal, Zimin Zhang, Lifan Yuan, Jiawei Han, and Hao Peng. The unreasonable effectiveness of entropy minimization in llm reasoning. *arXiv preprint arXiv:2505.15134*, 2025.

Charu C Aggarwal, Xiangnan Kong, Quanquan Gu, Jiawei Han, and Philip S Yu. Active learning: A survey. In *Data classification*, pp. 599–634. Chapman and Hall/CRC, 2014.

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, 2023.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Markus Bayer, Justin Lutz, and Christian Reuter. Activellm: Large language model-based active learning for textual few-shot scenarios. *Transactions of the Association for Computational Linguistics*, 14:1–22, 2026.

Wenbin Cai, Ya Zhang, and Jun Zhou. Maximizing expected model change for active learning in regression. In *2013 IEEE 13th international conference on data mining*, pp. 51–60. IEEE, 2013.

Yang Chen, Zhuolin Yang, Zihan Liu, Chankyu Lee, Peng Xu, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. Acereason-nemotron: Advancing math and code reasoning through reinforcement learning. *arXiv preprint arXiv:2505.16400*, 2025.

Keyu Duan, Zichen Liu, Xin Mao, Tianyu Pang, Changyu Chen, Qiguang Chen, Michael Qizhe Shieh, and Longxu Dou. Efficient process reward model training via active learning. *arXiv preprint arXiv:2504.10559*, 2025.

Xuening Feng, Zhaohui Jiang, Timo Kaufmann, Puchen Xu, Eyke Hüllermeier, Paul Weng, and Yifei Zhu. Duo: Diverse, uncertain, on-policy query generation and selection for reinforcement learning from human feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 16604–16612, 2025.

Yonatan Geifman and Ran El-Yaniv. Deep active learning over the long tail. *arXiv preprint arXiv:1711.00941*, 2017.

Mukesh Ghimire, Aosong Feng, Liwen You, Youzhi Luo, Fang Liu, and Xuan Zhu. Prism: A unified framework for post-training llms without verifiable rewards. *arXiv preprint arXiv:2601.04700*, 2026.

Xinyu Guan, Li Lyna Zhang, Yifei Liu, Ning Shang, Youran Sun, Yi Zhu, Fan Yang, and Mao Yang. Rstar-math: Small llms can master math reasoning with self-evolved deep thinking. *arXiv preprint arXiv:2501.04519*, 2025.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Bingxiang He, Yuxin Zuo, Zeyuan Liu, Shangziqi Zhao, Zixuan Fu, Junlin Yang, Cheng Qian, Kaiyan Zhang, Yuchen Fan, Ganqu Cui, et al. How far can unsupervised rlvr scale llm training? *arXiv preprint arXiv:2603.08660*, 2026.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, 2024.

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. *arXiv preprint arXiv:2103.03874*, 2021.

Kaixuan Ji, Jiafan He, and Quanquan Gu. Reinforcement learning from human feedback with active queries. *arXiv preprint arXiv:2402.09401*, 2024.

Seho Kee, Enrique Del Castillo, and George Runger. Query-by-committee improvement with diversity and density in batch active learning. *Information Sciences*, 454:401–418, 2018.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph 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*, pp. 611–626, 2023.

Ruotong Liao, Nikolai Röhrich, Xiaohan Wang, Yuhui Zhang, Yasaman Samadzadeh, Volker Tresp, and Serena Yeung-Levy. Tool verification for test-time reinforcement learning. *arXiv preprint arXiv:2603.02203*, 2026.

Zihan Lin, Xiaohan Wang, Jie Cao, Jiajun Chai, Guojun Yin, Wei Lin, and Ran He. Rest: Reshaping token-level policy gradients for tool-use large language models. *arXiv preprint arXiv:2509.21826*, 2025a.

Zihan Lin, Xiaohan Wang, Hexiong Yang, Jiajun Chai, Jie Cao, Guojun Yin, Wei Lin, and Ran He. Awpo: Enhancing tool-use of large language models through adaptive integration of reasoning rewards. *arXiv preprint arXiv:2512.19126*, 2025b.

Pangpang Liu, Chengchun Shi, and Will Wei Sun. Dual active learning for reinforcement learning from human feedback. *arXiv preprint arXiv:2410.02504*, 2024.

Xiaodong Lu, Xiaohan Wang, Jiajun Chai, Guojun Yin, Wei Lin, Zhijun Chen, Yu Luo, Fuzhen Zhuang, Yikun Ban, and Deqing Wang. Contextual rollout bandits for reinforcement learning with verifiable rewards. *arXiv preprint arXiv:2602.08499*, 2026.

Tianyang Luo, Tao Feng, Zhigang Hua, Yan Xie, Shuang Yang, Ge Liu, and Jiaxuan You. Memreward: Graph-based experience memory for llm reward prediction with limited labels. *arXiv preprint arXiv:2603.19310*, 2026.

Potsawee Manakul, Adian Liusie, and Mark Gales. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 9004–9017, 2023.

Viraj Mehta, Vikramjeet Das, Ojash Neopane, Yijia Dai, Ilija Bogunovic, Jeff Schneider, and Willie Neiswanger. Sample efficient reinforcement learning from human feedback via active exploration. 2023.

Stephen Mussmann, Julia Reisler, Daniel Tsai, Ehsan Mousavi, Shayne O'Brien, and Moises Goldszmidt. Active learning with expected error reduction. *arXiv preprint arXiv:2211.09283*, 2022.

Teng Pan, Yuchen Yan, Zixuan Wang, Ruiqing Zhang, Gaiyang Han, Wanqi Zhang, Weiming Lu, Jun Xiao, and Yongliang Shen. Coverrl: Breaking the consensus trap in label-free reasoning via generator-verifier co-evolution. *arXiv preprint arXiv:2603.17775*, 2026.

Mihir Prabhudesai, Lili Chen, Alex Ippoliti, Katerina Fragkiadaki, Hao Liu, and Deepak Pathak. Maximizing confidence alone improves reasoning. *arXiv preprint arXiv:2505.22660*, 2025.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

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

Chuyi Tan, Peiwen Yuan, Xinglin Wang, Yiwei Li, Shaoxiong Feng, Yueqi Zhang, Jiayi Shi, Ji Zhang, Boyuan Pan, Yao Hu, et al. Diagnosing and mitigating system bias in self-rewarding rl. *arXiv preprint arXiv:2510.08977*, 2025.

Dan Wang and Yi Shang. A new active labeling method for deep learning. In *2014 International joint conference on neural networks (IJCNN)*, pp. 112–119. IEEE, 2014.

Peng-Yuan Wang, Tian-Shuo Liu, Chenyang Wang, Ziniu Li, Yidi Wang, Shu Yan, Chengxing Jia, Xu-Hui Liu, Xinwei Chen, Jiacheng Xu, et al. A survey on large language models for mathematical reasoning. *ACM Computing Surveys*, 58(8):1–35, 2026.

Tingting Wang, Xufeng Zhao, Qiujian Lv, Bo Hu, and Degang Sun. Density weighted diversity based query strategy for active learning. In *2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pp. 156–161. IEEE, 2021.

Weiqin Wang, Yile Wang, Kehao Chen, and Hui Huang. Beyond majority voting: Towards fine-grained and more reliable reward signal for test-time reinforcement learning. *arXiv preprint arXiv:2512.15146*, 2025.

Zhiping Xiao, Yusheng Zhao, Qixin Zhang, Jiaye Xie, Wanjia Zhao, Weizhi Zhang, Xiao Luo, Philip Yu, and Ming Zhang. Sample lottery: Unsupervised discovery of critical instances for llm reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 8919–8949, 2026.

Chulin Xie, Yangsibo Huang, Chiyuan Zhang, Da Yu, Xinyun Chen, Bill Yuchen Lin, Bo Li, Badih Ghazi, and Ravi Kumar. On memorization of large language models in logical reasoning. In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pp. 2742–2785, 2025.

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

Dayu Yang, Tianyang Liu, Daoan Zhang, Antoine Simoulin, Xiaoyi Liu, Yuwei Cao, Zhaopu Teng, Xin Qian, Grey Yang, Jiebo Luo, et al. Code to think, think to code: A survey on code-enhanced reasoning and reasoning-driven code intelligence in llms. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 2586–2616, 2025b.

Shenzhi Yang, Guangcheng Zhu, Xing Zheng, Yingfan MA, Zhongqi Chen, Bowen Song, Weiqiang Wang, Junbo Zhao, Gang Chen, and Haobo Wang. Trapo: A semi-supervised reinforcement learning framework for boosting llm reasoning. *arXiv preprint arXiv:2512.13106*, 2025c.

Hao Yi, Yulan Hu, Xin Li, Sheng Ouyang, Lizhong Ding, and Yong Liu. Learn more with less: Uncertainty consistency guided query selection for rlvr. *arXiv preprint arXiv:2601.22595*, 2026.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025a.

Zhaoning Yu, Will Su, Leitian Tao, Haozhu Wang, Aashu Singh, Hanchao Yu, Jianyu Wang, Hongyang Gao, Weizhe Yuan, Jason Weston, et al. Restrain: From spurious votes to signals–self-driven rl with self-penalization. *arXiv preprint arXiv:2510.02172*, 2025b.

Anqi Zhang, Yulin Chen, Jane Pan, Chen Zhao, Aurojit Panda, Jinyang Li, and He He. Reasoning models know when they're right: Probing hidden states for self-verification. *arXiv preprint arXiv:2504.05419*, 2025a.

Kongcheng Zhang, Qi Yao, Shunyu Liu, Yingjie Wang, Baisheng Lai, Jieping Ye, Mingli Song, and Dacheng Tao. Consistent paths lead to truth: Self-rewarding reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.08745*, 2025b.

Qingyang Zhang, Haitao Wu, Changqing Zhang, Peilin Zhao, and Yatao Bian. Right question is already half the answer: Fully unsupervised llm reasoning incentivization. *arXiv preprint arXiv:2504.05812*, 2025c.

Zizhuo Zhang, Jianing Zhu, Xinmu Ge, Zihua Zhao, Zhanke Zhou, Xuan Li, Xiao Feng, Jiangchao Yao, and Bo Han. Co-rewarding: Stable self-supervised rl for eliciting reasoning in large language models. *arXiv preprint arXiv:2508.00410*, 2025d.

Xuandong Zhao, Zhewei Kang, Aosong Feng, Sergey Levine, and Dawn Song. Learning to reason without external rewards. *arXiv preprint arXiv:2505.19590*, 2025.

Yujun Zhou, Zhenwen Liang, Haolin Liu, Wenhao Yu, Kishan Panaganti, Linfeng Song, Dian Yu, Xiangliang Zhang, Haitao Mi, and Dong Yu. Evolving language models without labels: Majority drives selection, novelty promotes variation. *arXiv preprint arXiv:2509.15194*, 2025.

Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, Yuchen Zhang, Xinwei Long, Ermo Hua, et al. Ttrl: Test-time reinforcement learning. *arXiv preprint arXiv:2504.16084*, 2025.

A Proofs

A.1 Gradient Cosine as a Transformed Advantage Cosine

Lemma A.1 (Gradient cosine as a transformed advantage cosine)**.** *Let $S = [s_1, \dots, s_G]$ be the score matrix and define $K_x = S^\top S$. Let A and $\hat{A}$ be the advantage vectors induced by ground-truth and pseudo rewards, respectively. If $K_x \succ 0$, then*

$$\cos\theta_x = \frac{\langle g_x, \hat{g}_x \rangle}{\|g_x\|_2 \|\hat{g}_x\|_2} = \frac{\left\langle K_x^{1/2} A, K_x^{1/2} \hat{A} \right\rangle}{\|K_x^{1/2} A\|_2 \|K_x^{1/2} \hat{A}\|_2}. \tag{11}$$

Remark. The lemma shows that gradient alignment can be viewed as the cosine similarity between the transformed advantage vectors $K_x^{1/2} A$ and $K_x^{1/2} \hat{A}$. The matrix $K_x = S^\top S$ is the empirical NTK matrix over sampled responses, reflecting the intrinsic similarity between responses. The positive definiteness assumption $K_x \succ 0$ is mild since G is much smaller than the parameter dimension, making exact linear dependence among the score vectors unlikely. We further empirically verify that this condition holds in our experiments. Hence, the deviation between A and $\hat{A}$ directly affects the induced optimization direction.

Proof. Under the strict on-policy setting, the gradients induced by the ground-truth and pseudo-reward advantages can be written as

$$g_x = \frac{1}{G} S A, \qquad \hat{g}_x = \frac{1}{G} S \hat{A}. \tag{12}$$

Therefore,

$$\langle g_x, \hat{g}_x \rangle = \frac{1}{G^2} A^\top S^\top S \hat{A} = \frac{1}{G^2} A^\top K_x \hat{A}. \tag{13}$$

Similarly, their norms satisfy

$$\|g_x\|_2 = \frac{1}{G}\sqrt{A^\top K_x A}, \qquad \|\hat{g}_x\|_2 = \frac{1}{G}\sqrt{\hat{A}^\top K_x \hat{A}}. \tag{14}$$

Substituting these identities into the definition of cosine similarity gives

$$\cos\theta_x = \frac{A^\top K_x \hat{A}}{\sqrt{A^\top K_x A}\sqrt{\hat{A}^\top K_x \hat{A}}}. \tag{15}$$

Since $K_x \succ 0$, its unique positive definite square root $K_x^{1/2}$ exists. Hence,

$$A^\top K_x \hat{A} = \left\langle K_x^{1/2} A, K_x^{1/2} \hat{A} \right\rangle, \tag{16}$$

and

$$A^\top K_x A = \|K_x^{1/2} A\|_2^2, \qquad \hat{A}^\top K_x \hat{A} = \|K_x^{1/2} \hat{A}\|_2^2. \tag{17}$$

Thus,

$$\cos\theta_x = \frac{\left\langle K_x^{1/2} A, K_x^{1/2} \hat{A} \right\rangle}{\|K_x^{1/2} A\|_2 \|K_x^{1/2} \hat{A}\|_2}. \tag{18}$$

This completes the proof. □

A.2 Proof Theorem 6.1

Proof. Since A and $\hat{A}$ are group-normalized and centered, we have $A, \hat{A} \in \mathcal{H}$ and

$$\|A\|_2^2 = \|\hat{A}\|_2^2 = G. \tag{19}$$

Let

$$u = \frac{A}{\sqrt{G}}, \qquad v = \frac{\hat{A}}{\sqrt{G}}. \tag{20}$$

Then $u, v \in \mathcal{H}$ and $\|u\|_2 = \|v\|_2 = 1$. Define

$$\gamma = u^\top v. \tag{21}$$

Since $d_x = \|A - \hat{A}\|_2$, we have

$$d_x^2 = \|A - \hat{A}\|_2^2 = \|A\|_2^2 + \|\hat{A}\|_2^2 - 2A^\top \hat{A} = 2G(1 - \gamma). \tag{22}$$

Therefore,

$$\gamma = 1 - \frac{d_x^2}{2G}. \tag{23}$$

By Lemma A.1, the gradient alignment is

$$\cos\theta_x = \frac{u^\top K_x v}{\sqrt{u^\top K_x u}\sqrt{v^\top K_x v}}. \tag{24}$$

Since multiplying K_x by a positive scalar does not change the above cosine, we normalize K_x on $\mathcal{H}$ so that its eigenvalues lie in $[1, \kappa(x)]$.

If $\gamma = -1$, then $v = -u$, hence $\cos\theta_x = -1$. In this case $d_x^2 = 4G$, and the claimed bound also equals -1. Therefore, it remains to consider $-1 < \gamma < 1$.

Let

$$c = \sqrt{\frac{1+\gamma}{2}}, \qquad h = \sqrt{\frac{1-\gamma}{2}}. \tag{25}$$

Define

$$p = \frac{u+v}{2c}, \qquad q = \frac{u-v}{2h}. \tag{26}$$

Then $p, q \in \mathcal{H}$ are orthonormal, and

$$u = cp + hq, \qquad v = cp - hq. \tag{27}$$

It suffices to consider the two-dimensional subspace spanned by p and q. Let

$$Q = [p, q] \in \mathbb{R}^{G\times 2}, \qquad B = Q^\top K_x Q \in \mathbb{R}^{2\times 2} \tag{28}$$

be the compression of K_x onto this subspace. Since B is symmetric positive definite, let its eigenvalues be $\mu \geq \nu > 0$ and define $\ell = \mu/\nu$. By the Rayleigh quotient characterization, we have $1 \leq \ell \leq \kappa(x)$. Since multiplying B by a positive scalar does not change the cosine, we can normalize $\nu = 1$. Therefore, up to a rotation, this compression can be written as

$$B = R_\phi \begin{pmatrix} \ell & 0 \\ 0 & 1 \end{pmatrix} R_\phi^\top, \qquad R_\phi = \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix}. \tag{29}$$

Let $z = \cos 2\phi$. A direct calculation gives

$$F_\ell(z) = \frac{u^\top K_x v}{\sqrt{u^\top K_x u}\sqrt{v^\top K_x v}} = \frac{\gamma + \frac{\ell-1}{2}(\gamma + z)}{\sqrt{1 + (\ell-1)(1+\gamma z) + \frac{(\ell-1)^2}{4}(\gamma+z)^2}}. \tag{30}$$

Moreover,

$$\frac{\partial F_\ell(z)}{\partial z} = \frac{\ell(\ell-1)(1-\gamma^2)}{2\left[1 + (\ell-1)(1+\gamma z) + \frac{(\ell-1)^2}{4}(\gamma+z)^2\right]^{3/2}} \geq 0. \tag{31}$$

Thus, for fixed ℓ, the maximum is attained at $z = 1$. This gives

$$F_\ell(z) \leq F_\ell(1) = \frac{\ell(1+\gamma) - (1-\gamma)}{\ell(1+\gamma) + (1-\gamma)}. \tag{32}$$

Since

$$\frac{\partial}{\partial \ell} F_\ell(1) = \frac{2(1-\gamma^2)}{[\ell(1+\gamma) + (1-\gamma)]^2} \geq 0, \tag{33}$$

the maximum over all admissible ℓ is attained at $\ell = \kappa(x)$. Therefore,

$$\cos\theta_x \leq \frac{\kappa(x)(1+\gamma) - (1-\gamma)}{\kappa(x)(1+\gamma) + (1-\gamma)}. \tag{34}$$

Substituting $\gamma = 1 - d_x^2/(2G)$ yields

$$\cos\theta_x \leq \frac{4\kappa(x)G - (\kappa(x)+1)d_x^2}{4\kappa(x)G - (\kappa(x)-1)d_x^2}. \tag{35}$$

Equivalently,

$$\cos\theta_x \leq 1 - \frac{2d_x^2}{4\kappa(x)G - (\kappa(x)-1)d_x^2} = 1 - \frac{2}{4\kappa(x)\frac{G}{d_x^2} - (\kappa(x)-1)}. \tag{36}$$

This completes the proof. □

A.3 CAG and One-Step Corrective Benefit

We connect CAG to the objective-level benefit of replacing a pseudo-label-induced update with its ground-truth counterpart. Let $J_x^\star$ denote the ground-truth prompt-level objective and define the one-step corrective benefit as

$$V_{\mathrm{corr}}(x) = J_x^\star(\theta + \eta g_x) - J_x^\star(\theta + \eta \hat{g}_x), \tag{37}$$

where $\eta > 0$ is the step size.

Proposition A.2 (CAG characterizes one-step corrective benefit)**.** *Assume that $J_x^\star$ has a bounded Hessian in a neighborhood of θ, the ground-truth and pseudo-reward groups are non-degenerate and exactly advantage-normalized, and the response-level NTK is locally isotropic on the centered subspace, $K_x|_{\mathcal{H}} = \kappa_{\mathrm{iso}} I$ for some $\kappa_{\mathrm{iso}} > 0$. Under the strict on-policy update, for sufficiently small η,*

$$V_{\mathrm{corr}}(x) = \frac{\eta \kappa_{\mathrm{iso}}}{2G^2} \mathrm{CAG}(x)^2 + \mathcal{O}(\eta^2). \tag{38}$$

Remark. Under the local isotropy condition, ranking samples by CAG is equivalent, to first order, to ranking them by the gain from correcting the pseudo-label-induced update. Thus, CAG measures corrective supervision value rather than sample difficulty alone; the preceding theorem describes how this connection is distorted in the anisotropic case.

Proof. Under the strict on-policy setting, $g_x = \nabla_\theta J_x^\star(\theta)$. A Taylor expansion of the two terms in Eq. (37) gives

$$V_{\mathrm{corr}}(x) = \eta \langle g_x, g_x - \hat{g}_x \rangle + \mathcal{O}(\eta^2). \tag{39}$$

Exact group normalization for the two non-degenerate reward groups gives $\|A\|_2^2 = \|\hat{A}\|_2^2 = G$. Because $A, \hat{A} \in \mathcal{H}$ and $K_x|_{\mathcal{H}} = \kappa_{\mathrm{iso}} I$, we have

$$\|g_x\|_2^2 = \|\hat{g}_x\|_2^2 = \frac{\kappa_{\mathrm{iso}}}{G}, \qquad \|g_x - \hat{g}_x\|_2^2 = \frac{\kappa_{\mathrm{iso}}}{G^2} \|A - \hat{A}\|_2^2. \tag{40}$$

The equal gradient norms imply

$$\langle g_x, g_x - \hat{g}_x \rangle = \frac{1}{2} \|g_x - \hat{g}_x\|_2^2. \tag{41}$$

Finally, using $\mathrm{CAG}(x) = \|A - \hat{A}\|_2$ yields

$$V_{\mathrm{corr}}(x) = \frac{\eta \kappa_{\mathrm{iso}}}{2G^2} \mathrm{CAG}(x)^2 + \mathcal{O}(\eta^2), \tag{42}$$

which completes the proof. □

B Theoretical and Empirical Justification of the CARE Cascade

B.1 Exact Cascaded Structure of CAG

For the closed-form derivation below, we consider exact group normalization with $\epsilon = 0$. Consider a rollout group of size G. We select one of the largest answer clusters as the majority-vote cluster, with ties resolved by a fixed deterministic rule. We assume that correctness is constant within each answer cluster and that all correct responses, when present, belong to a single answer cluster. Let $V \in \{1, \ldots, G\}$ be the size of the selected cluster, and let $S \in \{0, 1\}$ indicate whether its answer is correct. When $S = 0$, let C denote the number of responses belonging to the correct answer cluster, with $C = 0$ if no correct response exists. Since the selected cluster is incorrect when $S = 0$, the selected and correct clusters are disjoint, and thus $0 \le C \le G - V$.

The CAG target Y has the following exact structure:

$$Y = g_G(V, S, C) = \begin{cases} 0, & S = 1, \\ 0, & S = 0,\ C = 0,\ V = G, \\ \sqrt{G}, & S = 0,\ C = 0,\ 1 \le V < G, \\ \sqrt{2G\left(1 + \sqrt{\dfrac{VC}{(G-V)(G-C)}}\right)}, & S = 0,\ 1 \le C \le G - V. \end{cases} \tag{43}$$

Proof. For a binary reward vector $z \in \{0, 1\}^G$, let $m = \sum_{i=1}^G z_i$ denote the number of positive entries. Its exact group-normalized advantage vector is

$$\mathcal{A}(z) = \begin{cases} \dfrac{Gz - m\mathbf{1}}{\sqrt{m(G-m)}}, & 0 < m < G, \\ \mathbf{0}, & m \in \{0, G\}. \end{cases} \tag{44}$$

For $0 < m < G$, it follows that

$$\|\mathcal{A}(z)\|_2^2 = G. \tag{45}$$

Let r and $\widetilde{r}$ be the ground-truth and majority-vote reward vectors, respectively, and define

$$A = \mathcal{A}(r), \qquad \widetilde{A} = \mathcal{A}(\widetilde{r}). \tag{46}$$

By definition, the CAG target is

$$Y = \left\| A - \widetilde{A} \right\|_2. \tag{47}$$

When $S = 1$, the selected majority-vote cluster is the correct cluster, so $r = \widetilde{r}$ and hence $A = \widetilde{A}$, yielding $Y = 0$.

When $S = 0$ and $C = 0$, no response is correct, so $r = \mathbf{0}$ and $A = \mathbf{0}$. If $V < G$, the pseudo-reward vector is nonconstant and Eq. (45) gives

$$Y = \left\| \widetilde{A} \right\|_2 = \sqrt{G}. \tag{48}$$

If $V = G$, then $\widetilde{r} = \mathbf{1}$ is also constant, so $A = \widetilde{A} = \mathbf{0}$ and therefore $Y = 0$.

Finally, suppose that $S = 0$ and $C > 0$. The positive entries of r and $\widetilde{r}$ correspond to the correct and selected incorrect clusters, respectively, and their supports are disjoint. Applying Eq. (44) gives

$$\begin{aligned} \left\langle A, \widetilde{A} \right\rangle &= \frac{(Gr - C\mathbf{1})^\top (G\widetilde{r} - V\mathbf{1})}{\sqrt{C(G-C)V(G-V)}} \\ &= -G\sqrt{\frac{VC}{(G-V)(G-C)}}. \end{aligned} \tag{49}$$

Moreover,

$$\|A\|_2^2 = \|\widetilde{A}\|_2^2 = G. \tag{50}$$

Therefore,

$$\begin{aligned} Y^2 &= \left\| A - \widetilde{A} \right\|_2^2 \\ &= \|A\|_2^2 + \|\widetilde{A}\|_2^2 - 2\left\langle A, \widetilde{A} \right\rangle \\ &= 2G\left(1 + \sqrt{\frac{VC}{(G-V)(G-C)}}\right), \end{aligned} \tag{51}$$

which proves Eq. (43). □

This exact structure directly motivates CARE's cascaded design. Given the observed answer-cluster structure and majority-cluster size V, Stage I predicts whether the selected majority answer is correct, while Stage II predicts the correct-cluster count only on the incorrect-majority branch. This factorization also enables supervision sharing across different values of V: samples with the same residual count C may correspond to different final CAG classes, so a flat predictor separates them, whereas Stage II pools them under the same count label. Moreover, Stage II only predicts admissible counts determined by the observed clusters, with repeated cluster sizes mapped to the same class. In the Qwen3-1.7B-Base rollout data with $G = 8$, Stage II has only 2.27 feasible classes per example on average and at most three, compared with the 15 global classes predicted by Unified. Stage I also determines whether a pseudo-label is retained for policy optimization, making its separate modeling consistent with its operational role in CARE.

B.2 Matched-Capacity Offline Evidence

We conduct a controlled offline comparison between CARE and Unified on 34,688 rollout groups collected from Qwen3-1.7B-Base checkpoints, using prompt-disjoint training, validation, and test splits of 24,284, 5,202, and 5,202 groups, respectively. Both methods use the same rollout-derived representations, data splits, losses, and optimization settings, with nearly identical predictor capacity:

CARE contains 775,475 trainable parameters, whereas Unified contains 775,459. Unified directly predicts one of the 15 global CAG classes, while CARE follows the cascaded structure in Eq. (43). As shown in Table 2, CARE consistently achieves lower NLL, better high-CAG ranking, and more reliable high-confidence zero-CAG identification across four random seeds. These preliminary controlled results show that the cascaded formulation is both consistent with the conditional structure of the CAG target and empirically more effective than direct flat classification, motivating its use in CARE.

Table 2: Matched-capacity offline comparison on 34,688 Qwen3-1.7B-Base rollout groups. Results are averaged over four random seeds.

Metric	**CARE**	**Unified**
Train NLL	**0.788**	0.973
Validation NLL	**0.833**	1.176
Test NLL	**0.840**	1.199
Top-10% Recall	**0.213**	0.152
NDCG@20%	**0.641**	0.603
Precision at $p(Y = 0) \geq 0.95$	**99.40%**	91.71%
Recall at $p(Y = 0) \geq 0.95$	**16.25%**	4.71%

C Experimental Details

C.1 Datasets for Main Experiments

We summarize the number of samples in the training and evaluation datasets in Table 3.

Table 3: Statistics of the training and test datasets used by all methods.

Dataset	**Description**	**# Train / Test**
DAPO-17k	Math Reasoning	17398 / -
AIME24	Math Reasoning	- / 30
AIME25	Math Reasoning	- / 30
MATH500	Math Reasoning	- / 500
AMC23	Math Reasoning	- / 40
Hmmt25	Math Reasoning	- / 30
Olympiad	Math Reasoning	- / 674
K&K train	Logic Reasoning	4500 / -
K&K 3ppl	Logic Reasoning	- / 100
K&K 4ppl	Logic Reasoning	- / 100
K&K 5ppl	Logic Reasoning	- / 100
K&K 6ppl	Logic Reasoning	- / 100
K&K 7ppl	Logic Reasoning	- / 100
K&K 8ppl	Logic Reasoning	- / 100

C.2 Implementation Details

Framework Implementation. Our framework is implemented on top of VeRL (Sheng et al., 2024) and URLVR (He et al., 2026). All experiments were conducted on 8×H20 GPUs with 141GB of memory, using Python 3.10 and PyTorch 2.6. All models are optimized with the AdamW optimizer ($\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay 0.01) and accelerated via vLLM (Kwon et al., 2023). We use a batch size of 64, a mini-batch size of 32, a maximum prompt length of 1024 tokens, a maximum response length of 4000 tokens, a learning rate of (1×10^{-6}), 8 GRPO rollouts, and a training rollout sampling temperature of 1.0. We train for 2 epochs on the DAPO-17k dataset (Yu et al., 2025a) and 3 epochs on the K&K dataset (Xie et al., 2025). The KL loss is not used. For Phi4-mini-instruct, we set both the batch size and mini-batch size to 64 to stabilize GRPO training and ensure on-policy

Table 4: Classifier features used by the acquisition classifier.

Representation	Feature	Definition
Global representation ϕ_i^{global}	Prompt representation	$h_i^x = \text{Enc}(x_i)_{[-1]} \in \mathbb{R}^d$
	Valid-rollout ratio	$f_i^{\text{valid}} = \frac{1}{n}\sum_{j=1}^{n} \mathbf{1}[o_{i,j} \neq \texttt{None}]$
	Cluster distribution	$f_i^{\text{cluster}} = \text{Sort}([p_{i,1}, \ldots, p_{i,C}])$, where $p_{i,c} = \frac{n_{i,c}}{\sum_{c'=1}^{C} n_{i,c'}}$
Response-level representation Φ_i^{ans}	Response representation	$h_{i,g}^{\text{ans}} = \text{Enc}(x_i, o_{i,g})_{[-1]} \in \mathbb{R}^d$
	Normalized length	$\ell_{i,g}^{\text{norm}} = \lvert o_{i,g} \rvert / L_{\max}$
	Response-level feature	$\phi_{i,g}^{\text{ans}} = [h_{i,g}^{\text{ans}}; \ell_{i,g}^{\text{norm}}]$
	Stacked response features	$\Phi_i^{\text{ans}} = [\phi_{i,1}^{\text{ans}}; \ldots; \phi_{i,n}^{\text{ans}}]$

Table 5: Class-1 selection ratio p_2 used in the first-stage selection for different models and tasks.

Task	Qwen3-1.7B-Base	Qwen3-4B-Base	Qwen3-8B-Base	Phi4-mini-instruct
Math	0.10	0.25	0.25	0.25
K&K	0.00	0.25	0.25	0.15

updates. For K&K tasks, we find that GRPO training is unstable without regularization; therefore, we use the KL loss with its coefficient set to 0.01.

CARE and Baseline Implementation Details. For the Entropy method, we first cluster the answer distribution and then compute the response entropy, selecting samples with the highest entropy for ground-truth annotation. For the Prob method, we compute the average probability of the response tokens and select samples with the lowest probability for ground-truth annotation. For CARE, the cold-start warm-up is set to 20 steps. As shown in Table 4, the classifiers adopt a two-stage MLP architecture with separate encoders for prompt- and response-level features. Across all experiments, we set the prompt encoder hidden dimension to 128, the prediction-head hidden dimension to 256, and the response encoder hidden/output dimensions to 64/512, with ReLU activations. The first-stage classifier includes a rollout-level auxiliary head with loss weight 1.5. The second-stage classifier outputs (G+1) logits for count prediction, where (G=8) is the number of rollouts per prompt; invalid count classes are masked before loss computation. We optimize the classifiers with AdamW using a learning rate of 1e-4. To stabilize online classifier learning, we maintain separate FIFO replay buffers for the two stages, each with a capacity of 2048, and mix 16 historical samples into each classifier update. For the second-stage classifier, historical samples are drawn using count-label stratification to mitigate class imbalance. In addition, we enable class-balanced losses, with class weights computed from batch-level class frequencies using power 0.5 and clipped to ([0.25,4.0]). For first-stage selection, the class-1 selection ratio p_2 for different models on math and K&K tasks is shown in Table 5. In practice, this ratio can be adjusted based on the model capability and the dataset difficulty.

D More Experimental Results

D.1 Comparison of Computational Cost

To compare the training overhead of different methods, we evaluate their total training cost using Qwen3-4B-Base as the backbone model on an 8×H20 GPU setup with 141 GB of total memory, as summarized in Table 6. For a fair comparison, during policy optimization, we do not discard the samples that CARE would otherwise exclude from training; instead, we set their advantages to zero. GT is trained on the full dataset and exhibits a substantial increase in average output length, resulting in a longer training time. Compared with the Random baseline, CARE incurs only an 8.25% increase in training time while achieving substantial performance gains. This modest overhead stems from the lightweight design of its classifier network, which consists of only a small MLP. In addition, the total training time could be further reduced by excluding samples discarded after the second-stage classifier selection from policy optimization.

Table 6: Total training time (hours) for various methods on math tasks with Qwen3-4B-Base.

Method	Training
GT	28.70
Random-mask	20.48
Prob-mask	20.43
Entropy-mask	22.13
Oracle-mask	21.51
CARE (ours)	22.17

D.2 Sensitivity to Rollout Sampling Hyperparameters

Our default training rollout temperature and rollout number follow the standard settings used in the TTRL line of work and were not separately tuned for CARE. We vary the training rollout temperature over $T \in \{0.8, 1.0, 1.2\}$ and the number of rollouts over $G \in \{4, 8, 16\}$ on Qwen3-1.7B-Base, while keeping the evaluation configuration fixed. As shown in Table 7, temperature changes have only a minor effect. Increasing G generally improves performance because it provides more response-level training samples and a more reliable group-based advantage estimate. Even with $G = 4$, CARE remains above Random-mask.

Table 7: Sensitivity of CARE to the training rollout temperature T and rollout number G on math tasks using Qwen3-1.7B-Base. Random-mask is included as a reference.

Setting	T	G	Average
Random-mask	–	8	24.43
CARE (default)	1.0	8	25.67
CARE	0.8	8	25.68
CARE	1.2	8	25.38
CARE	1.0	4	24.70
CARE	1.0	16	**26.29**

D.3 Empirical Check of NTK Non-Degeneracy

We empirically examine the positive-definiteness condition used in Theorem 6.1. We compute the empirical NTK from response-level gradients restricted to the final Transformer block for 48 rollout groups sampled from three CARE and three TTRL checkpoints, with eight groups per checkpoint and eight responses per group. As shown in Table 8, none of the examined matrices is numerically rank-deficient, and the smallest observed eigenvalue ratio is 9.87×10^{-3}. Moreover, the full-model NTK is the sum of the final-block Gram matrix and positive-semidefinite contributions from the remaining parameter blocks; hence, positive definiteness of the final-block matrix also implies positive definiteness of the corresponding full-model NTK.

Table 8: Numerical non-degeneracy of final-block empirical NTK matrices. Each checkpoint contains eight rollout groups.

Method	Step	Rank-deficient groups	Min. $\lambda_{\min}/\lambda_{\max}$
CARE	40	0/8	1.42×10^{-2}
CARE	160	0/8	9.87×10^{-3}
CARE	360	0/8	1.22×10^{-1}
TTRL	40	0/8	1.18×10^{-2}
TTRL	240	0/8	8.50×10^{-2}
TTRL	480	0/8	1.33×10^{-1}

D.4 Comparison with CONST

We also compare CARE with the sample-selection method CONST (Xiao et al., 2026), which performs sample selection in an offline preprocessing stage. To adapt it to RLAVR, we apply its selection procedure to the unlabeled training pool, annotate the selected 20% of samples, and repeat each selected sample five times so that CONST and CARE use the same number of training instances and optimization steps. As shown in Table 9, CARE consistently outperforms CONST, demonstrating the additional value of leveraging pseudo-labeled samples beyond informative sample selection. CONST requires 40 offline generation calls per candidate before the 8 online GRPO rollouts, whereas CARE directly reuses the 8 online rollouts for both acquisition and training, resulting in approximately $6\times$ fewer generation calls.

Table 9: Comparison with CONST on math tasks using Qwen3-1.7B-Base under the same 20% annotation budget.

Method	AIME24	AIME25	MATH500	AMC23	Hmmt25	Olympiad	Average
CONST	7.19	5.21	65.18	37.81	**1.67**	28.71	24.29
CARE (ours)	**9.27**	**5.83**	**67.50**	**40.08**	0.73	**30.62**	**25.67**

D.5 Controlled Comparison of Sample-Processing Mechanisms

To examine whether CARE's improvements arise merely from its pseudo-label filtering and confidence-based downweighting mechanisms, we construct two controlled variants, Entropy-FD and Prob-FD, which retain CARE's sample-utilization strategy while replacing its acquisition criterion. Entropy-FD and Prob-FD annotate the most uncertain prompts according to response entropy and average response probability, respectively. Among the remaining prompts, they retain the same number of pseudo-labeled samples as CARE, selecting the lowest-entropy samples for Entropy-FD and the highest-probability samples for Prob-FD. Their pseudo advantages are downweighted using the corresponding normalized confidence scores, while all other prompts are discarded. Both variants use the same 20% annotation budget as CARE. As shown in Table 10, CARE achieves the best performance across all three model scales, validating the effectiveness of CAG as an acquisition criterion.

Table 10: Average performance (%) of Entropy-FD and Prob-FD under CARE's filtering and downweighting mechanisms on math tasks.

Backbone	Entropy-FD	Prob-FD	CARE
Qwen3-1.7B-Base	23.03	24.77	**25.67**
Qwen3-4B-Base	33.82	33.17	**37.07**
Qwen3-8B-Base	39.09	38.59	**43.59**

D.6 Additional Uncertainty-Based Active Learning Baselines

We further compare CARE with two uncertainty estimation methods commonly used in active learning: Monte Carlo Dropout (MC-Dropout) and predictor ensembles. Both baselines use the same rollout-derived representations and MLP architecture as CARE and predict whether a majority-voted pseudo-label is correct. MC-Dropout estimates uncertainty using predictive variance across stochastic forward passes with dropout enabled, while the ensemble baseline uses disagreement among independently initialized predictors. Samples with the highest uncertainty are selected for annotation. All methods use the same annotation budget and training protocol. As shown in Table 11, CARE achieves higher average performance than both alternatives at every model scale.

Table 11: Average performance (%) of additional uncertainty-based active learning baselines on math tasks.

Backbone	MC-Dropout	Ensemble	CARE
Qwen3-1.7B-Base	15.32	23.11	**25.67**
Qwen3-4B-Base	32.38	32.34	**37.07**
Qwen3-8B-Base	35.60	36.00	**43.59**

D.7 Additional Collapse Analysis Experiments

For TTRL, we generate training labels from majority-voted responses. For the TTRL-mask variant, we compare the voted answers with the ground-truth labels and mask samples with incorrectly voted labels by assigning them a sample weight of zero before training. All other hyperparameters are kept the same as in the main experiments. In addition, we provide results on the Phi4-mini-instruct model in Figure 5 (Left), which exhibit trends consistent with those observed on the Qwen3 model. We also evaluate on K&K tasks, with results for the Qwen3 and Phi4-mini-instruct models shown in Figure 6 and Figure 7 (Left), respectively. The conclusions are consistent with those on math tasks.

D.8 Additional Preliminary Experimental Results

Due to the definition of the CAG metric, correctly pseudo-labeled samples have a CAG value of zero. As a result, the Oracle strategy, which selects samples based on CAG, is more likely to acquire labels from incorrectly pseudo-labeled samples. To ensure a fair comparison, for the Random, Entropy, and Prob strategies, we also select labels only from incorrectly pseudo-labeled samples according to their respective selection criteria. The results on Phi4-mini-instruct for math and K&K tasks without masking pseudo-labeled samples are shown in Figure 5 (Middle) and Figure 7 (Middle), respectively. The results on Qwen3 for K&K tasks are shown in Figure 8 (Left). Across different models and tasks, we observe consistent conclusions: the Oracle strategy substantially alleviates collapse, validating the effectiveness of the CAG metric for mitigating collapse in unsupervised training.

We further compared the effectiveness of different methods under the condition that only samples with ground-truth labels are used, with other samples masked out. Figure 9 and Figure 5 (Right) report the results on Qwen3 and Phi4-mini-instruct on math datasets, respectively. Figure 8 (Right) and Figure 7 (Right) report the results on Qwen3 and Phi4-mini-instruct on K&K datasets, respectively. In the absence of KL constraints, the Phi model shows occasional instability, and both Oracle-decay and GRPO tend to collapse during training, but the optimal performance of Oracle-decay remains markedly superior to that of other approaches. The results show a generally consistent trend: Oracle-mask performs favorably in most cases, providing empirical support for using CAG as a criterion for supervision value selection. We also experimented with using the CAG metric to decay the advantage:

$$\bar{A}_{i,g} = A^{\star}_{i,g} \times \exp(-100 \cdot s_i) \tag{52}$$

denoted as Oracle-decay. With this large decay coefficient, the advantage of incorrect samples approaches 0, while for unsupervised samples, where the voting is correct, CAG is 0, and the advantage scaling factor is 1, thus integrating correct unsupervised samples into training. This approach achieves performance comparable to GT in most models, further demonstrating the value of properly utilizing unsupervised samples.

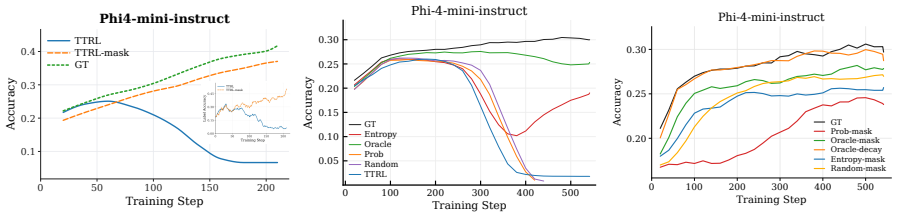

Figure 5: **(Left)** Training dynamics of Phi4-mini-instruct on math tasks. Comparison of label acquisition strategies on math-task pseudo-labeled samples without masking **(Middle)** and with masking **(Right)**.



Figure 6: Training dynamics across three Qwen3 models on K&K tasks.

D.9 More Ablation Study Results

We provide ablation study results for the Qwen3-1.7B-Base and Qwen3-8B-Base models on math tasks, as shown in Figure 10 (left and middle).

D.10 Additional Analysis of p_2 on More Models

Figure 11 shows the effect of p_2 across different models. For Qwen3-1.7B-Base, the first-stage classifier has lower accuracy, likely due to its smaller hidden dimension and less stable training. A larger p_2 therefore introduces more erroneous pseudo-labeled samples, while $p_2 = 0.1$ achieves a better accuracy–performance trade-off. For other models, classifier accuracy remains relatively stable across p_2 values, as their larger hidden dimensions and more stable training facilitate classifier learning. Increasing p_2 introduces more unsupervised training samples and improves performance, with $p_2 = 0.25$ yielding the best results.

D.11 Main Experimental Results on Math Tasks Using the Phi4 Model

To further examine the effectiveness of CARE across different model families on math tasks, we conduct experiments on Phi4-mini-instruct, following the same hyperparameter settings as those described in Appendix C.2. The experimental results on math tasks are shown in Table 12 and Figure 10 (right). CARE consistently outperforms other data selection methods in both settings, i.e., with and without pseudo-labeled samples, and even achieves performance comparable to GT. These results further demonstrate the effectiveness of CARE on math tasks.

D.12 Main Experimental Results on K&K Tasks

To further verify the effectiveness of CARE on other tasks, we evaluate it in the logical reasoning domain by conducting experiments on the K&K dataset (Xie et al., 2025), following the same hyperparameter settings as those described in Appendix C.2. We use the 3–7 ppl split for training and the 3–8 ppl split for testing. The dataset statistics are shown in Table 3. The experimental results of the Qwen3-Base and Phi4-mini-instruct models on the K&K dataset are shown in Table 13 and Figure 12. CARE consistently outperforms other data selection methods in both settings, i.e., with and without pseudo-labeled samples, validating the effectiveness of CARE on K&K tasks.

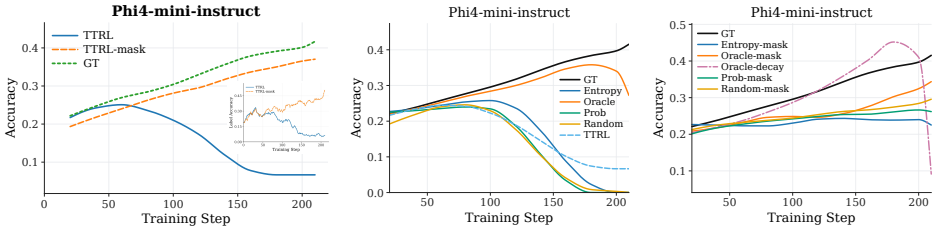

Figure 7: **(Left)** Training dynamics of Phi4-mini-instruct on K&K tasks. Comparison of label acquisition strategies on K&K pseudo-labeled samples without masking **(Middle)** and with masking **(Right)**.

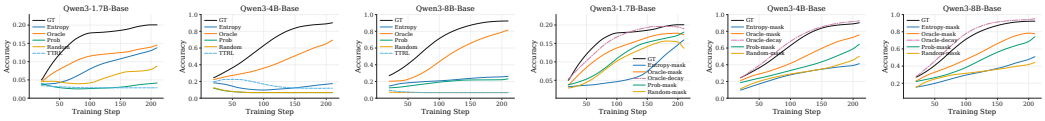

Figure 8: Comparison of label acquisition strategies on K&K pseudo-labeled samples using the Qwen3 model without masking **(Left)** and with masking **(Right)**.

E Pseudocode of Class-wise CAG Score Calculation

We present the pseudocode of Class-wise CAG Score Calculation in Algorithm 1.

F Pseudocode of CARE

We present the pseudocode of CARE in Algorithm 2.

G Limitations

Our work has several limitations. Due to limited computational resources, our experiments do not cover larger-scale models, with the largest evaluated model being 8B. In addition, the current evaluation mainly focuses on text-only mathematical and logical reasoning tasks, leaving broader task domains and multimodal scenarios unexplored. Moreover, some hyperparameters in CARE still rely on manual tuning. In future work, we plan to validate CARE on larger models and more diverse RLAVR scenarios, including code generation, tool use, agentic tasks, and multimodal tasks. For tasks with partially correct outputs, such as code that passes only a subset of test cases, CARE could use graded verifier rewards (e.g., test-pass rates) rather than binary correctness to compute pseudo-label- and ground-truth-induced advantages. Structured feedback such as execution-error categories could further be treated as outcome classes for reliability estimation, while label acquisition remains guided by the predicted corrective advantage. We also aim to make the parameter selection process adaptive to task characteristics and model capabilities.

H Broader Impacts

This work introduces the new settings RLAVR, where acquiring ground-truth labels is expensive. By improving the use of limited supervision, CARE may help improve data efficiency and reduce annotation costs in domains where labels are difficult to obtain. The potential negative impact is limited, as our work focuses on methodological improvements rather than specific downstream applications. However, as with other advances in LLMs, improved training methods may indirectly enhance generative capabilities, which could be misused for generating misleading or harmful content. These risks are general to the field and not specific to our approach.

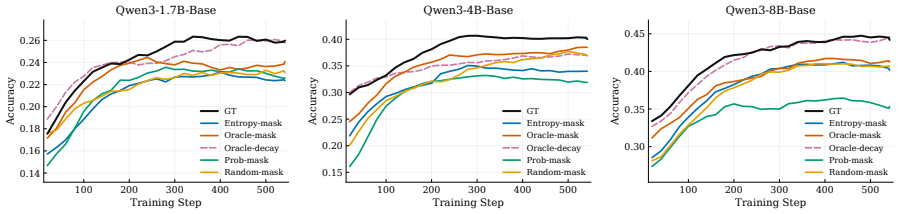

Figure 9: Comparison of label acquisition strategies across three Qwen3 model scales on math tasks.

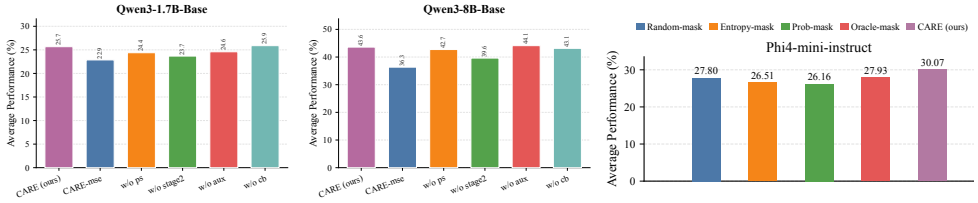

Figure 10: **(Left and Middle)** CARE component ablation on Qwen3-1.7B-Base and Qwen3-8B-Base on math tasks. **(Right)** Comparison of training performance without unsupervised samples across baseline methods on Phi4-mini-instruct on math tasks.

I Case Study

We present an example of the outputs generated by Qwen3-4B-Base trained with CARE on DAPO-17k training set in Figure 13, where the model correctly answers the question. We further provide a CAG calculation example in Figure 14.

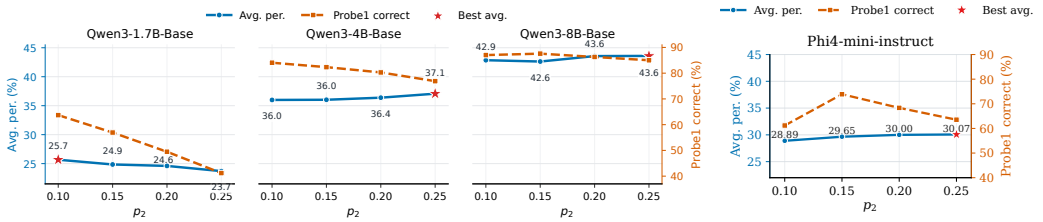

Figure 11: Hyperparameter analysis of p_2 on Qwen3-Base (left) and Phi4 (right) models on math tasks.

Table 12: Performance (%) of Phi4-mini-instruct models on math tasks. Vanilla and GT are reported as reference settings and are excluded from best/second-best ranking. The **bold** and underline indicate the best and second-best results among the remaining methods, respectively.

Methods	AIME24	AIME25	MATH500	AMC23	Hmmt25	Olympiad	Average
Phi4-mini-instruct							
Vanilla	6.46	3.33	39.93	19.30	0.52	24.11	15.61
GT	11.56	9.58	73.25	47.26	0.63	38.07	30.06
TTRL	8.44	<u>6.67</u>	66.30	42.34	0.63	33.55	26.32
Random	7.71	5.83	67.70	41.64	0.73	33.77	26.23
Entropy	8.65	5.83	63.95	40.63	0.73	33.10	25.48
Prob	8.02	4.90	67.83	42.34	<u>1.25</u>	33.14	26.25
Oracle	<u>10.93</u>	5.94	<u>69.73</u>	<u>42.97</u>	0.94	<u>35.12</u>	<u>27.60</u>
CARE (ours)	**12.92**	**7.81**	**73.77**	**47.66**	**1.35**	**36.90**	**30.07**

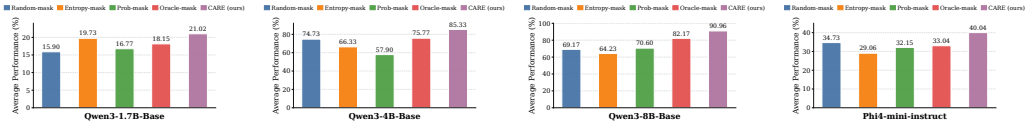

Figure 12: Comparison of training performance among baselines without unsupervised samples on K&K tasks using Qwen3-Base and Phi4-mini-instruct models.

Algorithm 1 Class-wise CAG Score Calculation

Require: Voted-cluster size m_i, admissible class $k \in \mathcal{K}_i$, number of rollouts G
Ensure: Class-wise CAG score $s_i(k)$

1: Construct the pseudo reward induced by majority voting:

$$\tilde{\mathbf{r}}_i = [\underbrace{1, \ldots, 1}_{m_i}, \underbrace{0, \ldots, 0}_{G-m_i}]$$

2: Construct the candidate ground-truth reward under class k:

$$\mathbf{r}_i^{(k)} = [\underbrace{0, \ldots, 0}_{m_i}, \underbrace{1, \ldots, 1}_{k}, \underbrace{0, \ldots, 0}_{G-m_i-k}]$$

3: Compute the pseudo-label advantage $\tilde{\mathbf{A}}_i$ from $\tilde{\mathbf{r}}_i$ using Eq. (1)
4: Compute the candidate ground-truth advantage $\mathbf{A}_i^{(k)}$ from $\mathbf{r}_i^{(k)}$ using Eq. (1)
5: Compute the class-wise CAG score $s_i(k)$ using Eq. (4)
6: **return** $s_i(k)$

Table 13: Performance (%) of Qwen3-Base and Phi4-mini-instruct models on K&K tasks. Vanilla and GT are reported as reference settings and are excluded from best/second-best ranking. The **bold** and underline indicate the best and second-best results among the remaining methods, respectively.

Methods	3ppl	4ppl	5ppl	6ppl	7ppl	8ppl	Average
Qwen3-1.7B-Base							
Vanilla	2.13	1.50	0.37	0.75	0.12	0.00	0.81
GT	40.25	30.13	19.00	20.87	6.75	8.12	20.85
TTRL	10.63	4.87	1.88	1.75	1.25	1.25	3.60
Random	18.25	10.00	5.00	4.25	0.88	0.00	6.40
Entropy	13.12	2.88	5.12	2.75	0.63	0.00	4.08
Prob	11.63	2.88	5.12	2.75	0.63	0.00	3.60
Oracle	27.38	20.87	15.75	15.25	4.00	5.00	14.71
CARE (ours)	**41.37**	**30.50**	**17.75**	**18.63**	**6.50**	**11.34**	**21.02**
Qwen3-4B-Base							
Vanilla	9.25	7.63	4.63	3.00	1.75	0.63	4.48
GT	99.37	98.12	95.25	88.38	85.12	74.75	90.17
TTRL	50.25	35.37	28.13	21.88	12.87	9.25	26.29
Random	56.63	43.00	28.75	24.38	13.88	10.88	29.58
Entropy	28.25	16.75	14.12	10.50	3.00	4.13	12.79
Prob	28.37	18.00	11.87	11.12	2.62	4.87	12.81
Oracle	89.00	87.25	74.87	63.38	59.25	54.37	71.42
CARE (ours)	**97.75**	**96.25**	**89.50**	**83.13**	**76.75**	**68.62**	**85.33**
Qwen3-8B-Base							
Vanilla	25.25	17.50	13.88	8.75	5.50	3.62	12.42
GT	99.00	97.87	97.00	89.00	88.75	84.62	92.71
TTRL	24.88	10.37	6.75	8.12	2.00	4.00	9.35
Random	25.75	12.13	8.25	10.63	2.37	4.13	10.54
Entropy	46.00	41.00	32.62	32.00	14.00	13.00	29.77
Prob	41.12	30.38	11.20	14.25	5.00	4.00	17.79
Oracle	95.00	93.50	86.75	78.50	74.25	69.63	82.94
CARE (ours)	**99.37**	**98.00**	**93.25**	**91.00**	**86.75**	**77.38**	**90.96**
Phi4-mini-instruct							
Vanilla	21.88	16.00	8.00	6.37	3.00	1.75	9.50
GT	64.75	54.12	41.75	38.12	27.88	21.00	41.27
TTRL	46.88	33.63	21.63	20.37	13.50	10.13	24.35
Random	45.50	31.87	19.00	18.50	13.12	9.13	22.85
Entropy	48.25	35.00	26.25	23.25	14.25	10.88	26.31
Prob	47.37	33.88	24.25	20.50	13.38	12.25	25.27
Oracle	54.25	44.50	32.88	33.50	23.50	17.25	34.31
CARE (ours)	**64.50**	**54.62**	**37.38**	**36.00**	**25.50**	**22.25**	**40.04**

Algorithm 2 CARE: Correction-Aware Reliability Estimation for RLAVR

Input: initial policy $\pi_{\theta_{\text{init}}}$; training prompts D_{train}; verifier; annotation ratio p, selection ratio of unsupervised samples p_2; warmup steps T_{warm}; GRPO steps μ; replay buffer $\mathcal{R}$; first-stage classifier g_{ψ_1}; second-stage classifier g_{ψ_2}
Output: policy model π_θ
Initialize policy $\pi_\theta \leftarrow \pi_{\theta_{\text{init}}}$
Initialize replay buffer $\mathcal{R} \leftarrow \emptyset$
for iteration $= 1, \ldots, I$ **do**
  Set reference model $\pi_{\text{ref}} \leftarrow \pi_\theta$
  for step $= 1, \ldots, M$ **do**
    Sample a batch of prompts D_b from D_{train}
    Update old policy $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta$
    Sample G rollouts $\mathcal{B}_i = \{o_{i,1}, \ldots, o_{i,G}\} \sim \pi_{\theta_{\text{old}}}(\cdot \mid x_i)$ for each $x_i \in D_b$
    Compute classifier representation $\phi_i = [\phi_i^{\text{global}}; \Phi_i^{\text{ans}}]$ for each $x_i \in D_b$
    if step $\leq T_{\text{warm}}$ **then**
      Set $\mathcal{S}_{\text{err}} \leftarrow D_b$ and $\mathcal{S}_{\text{unsup}} \leftarrow \emptyset$
    else
      Use g_{ψ_1} to predict the vote reliability score $c_i^{(1)}$ for each $x_i \in D_b$
      Select the top-$p_2\%$ prompts with the largest $c_i^{(1)}$ as $\mathcal{S}_{\text{unsup}}$
      Let $\mathcal{S}_{\text{err}} \leftarrow D_b \setminus \mathcal{S}_{\text{unsup}}$
    end if
    Compute CAG scores $\{\hat{s}_i\}_{x_i \in \mathcal{S}_{\text{err}}}$ using g_{ψ_2} and Algorithm 1
    Select the top-$\lfloor p|\mathcal{D}_b| \rfloor$ prompts from $\mathcal{S}_{\text{err}}$ with the largest $\hat{s}_i$ as $\mathcal{S}_{\text{sup}}$
    Acquire ground-truth labels for prompts in $\mathcal{S}_{\text{sup}}$
    Compute ground-truth rewards and advantages $\{A^\star_{i,g}\}_{j=1}^n$ for each $x_i \in \mathcal{S}_{\text{sup}}$
    Compute pseudo rewards and pseudo advantages $\{\tilde{A}_{i,g}\}_{j=1}^n$ for each $x_i \in \mathcal{S}_{\text{unsup}}$
    Use the reliability-weighted pseudo advantages $\{\bar{A}_{i,g}\}_{j=1}^n$ for each $x_i \in \mathcal{S}_{\text{unsup}}$
    Construct mixed advantages $A^{\text{mix}}_{i,g}$ using $A^\star_{i,g}$ for $\mathcal{S}_{\text{sup}}$ and $\bar{A}_{i,g}$ for $\mathcal{S}_{\text{unsup}}$
    for GRPO update $= 1, \ldots, \mu$ **do**
      Update π_θ by optimizing the GRPO objective in Eq. (2) with $A^{\text{mix}}_{i,g}$
    end for
    Add newly labeled samples from $\mathcal{S}_{\text{sup}}$ to the replay buffer $\mathcal{R}$
    Update g_{ψ_1} and g_{ψ_2} using current labeled samples and replayed samples from $\mathcal{R}$
  end for
end for
return π_θ

CARE Qwen3-4B-Base Output Case

Question:
In Perfectville, the streets are all 20 feet wide and the blocks they enclose are all squares of side length 400 feet, as shown. Sarah runs around the block on the 400-foot side of the street, while Sam runs on the opposite side of the street. How many more feet than Sarah does Sam run for every lap around the block?\n\n\\[asy]\ndraw((0,0)--(10,0));\ndraw((1,1)--(9,1));\ndraw((0,0)--(0,10));\ndraw((0,10)--(10,10));
\ndraw((10,10)--(10,0));\ndraw((9,1)--(9,9));\ndraw((9,9)--(1,9));\ndraw((1,9)--(1,1));\nfill((1,1)--(1,9)--(9,9)--(9,1)--cycle,gray);\ndraw((9,10.5)--(9,11));
\ndraw((10,10.5)--(10,11));\ndraw((9,10.75)--(10,10.75));\nlabel(\"20'\",(9.5,11),N);\ndraw((10.5,9)--(11,9));\ndraw((10.5,10)--(11,10));\ndraw((10.75,9)--(10.75,10));\nlabel(\"20'\",(11,9.5),E);\nlabel(\"400'\",(5,9),S);\nlabel(\"400'\",(9,5),W);\n[/asy]

Model Answer:
To determine how many more feet than Sarah does Sam run for every lap around the block, we need to calculate the perimeter of the paths that Sarah and Sam run and then find the difference between these perimeters.

First, let's calculate the perimeter of the path that Sarah runs. Sarah runs around the inner block, which is a square with a side length of 400 feet. The perimeter \\(P_{\\text{Sarah}} \\) of a square is given by:
\[
P_{\\text{Sarah}} = 4 \\times \\text{side length} = 4 \\times 400 = 1600 \\text{ feet}
\\]
Next, let's calculate the perimeter of the path that Sam runs. Sam runs around the outer block, which is a square with a side length that is 20 feet wider on each side than the inner block. Therefore, the side length of the outer square is:
\\[
400 + 2 \\times 20 = 400 + 40 = 440 \\text{ feet}
\\]
The perimeter \\(P_{\\text{Sam}} \\) of the outer square is:
\\[
P_{\\text{Sam}} = 4 \\times \\text{side length} = 4 \\times 440 = 1760 \\text{ feet}
\\]
Now, we need to find the difference in the perimeters of the paths that Sam and Sarah run:
\\[
P_{\\text{Sam}} - P_{\\text{Sarah}} = 1760 - 1600 = 160 \\text{ feet}
\\]\
Thus, Sam runs \\(\\boxed{160}\\) more feet than Sarah for every lap around the block. ✓

Figure 13: Example 1 of the CARE method using the Qwen3-4B-Base model.

CAG Calculation Example

Question:
Carl, James, Saif, and Ted play several games of two-player For The Win on the Art of Problem Solving website. If, among these games, Carl wins 5 and loses 0, James wins 4 and loses 2, Saif wins 1 and loses 6, and Ted wins 4, how many games does Ted lose?

Model Boxed Answer:
1. 3
2. 4
3. None
4. 4
5. 10
6. None
7. 6
8. 3

Pseudo Laebl: ✗
vote answer: 3
r_pseudo= [1 0 0 0 0 0 0 1]
A_pseudo= [1.7320508076, -0.5773502692, -0.5773502692, -0.5773502692, -0.5773502692, -0.5773502692, -0.5773502692, 1.7320508076]

Ground-truth Label： ✓
true Answer: 6
r_true= [0 0 0 0 0 0 1 0]
A_true =[-0.3779644730, -0.3779644730, -0.3779644730, -0.3779644730, -0.3779644730, -0.3779644730, 2.6457513111, -0.3779644730]

CAG： $||A_pseudo - A_true||_2$=4.4149163349

Figure 14: CAG calculation example.