THE SCAR OF RL: MEMBERSHIP INFERENCE ATTACK FOR REASONING MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning (RL) is a critical post-training stage of frontier reasoning models. However, the prompts used during RL training are rarely disclosed, making it difficult to determine whether benchmark questions used to evaluate post-trained models were included during RL training, and potentially biasing downstream evaluations. Existing membership inference attacks, developed primarily for pretraining and supervised fine-tuning, typically exploit the reduced loss of an observed training sequence. However, this principle does not transfer directly to RL where there are no observed sequences. Instead, RL optimizes for rewards given to sampled outputs and reshapes the distribution over model-generated solution trajectories. We show that this change leaves a multi-dimensional signature in the model's own outputs as prompts seen during RL tend to induce higher confidence in sampled continuations, more similar solution trajectories and final answers, more saturated verifier outcomes, and less variation across independent rollouts. However, the magnitude of these effects varies substantially across model–dataset pairs, making any single statistic unreliable. In response, we introduce SCAR, which defines 15 features capturing complementary signatures of RL exposure, and combines them through label-free multi-axis aggregation, avoiding reliance on any single signature. Across eleven unsupervised aggregation rules, combining the four axes (Self-sharpening, Collapse, Answer saturation, and Rollout-sharpening) consistently improves macro-average performance over individual signatures. Our strongest aggregation reaches 0.764 mean ROC-AUC, compared with 0.673 for the strongest single-statistic baseline.

1 INTRODUCTION

Reinforcement Learning with Verifiable Rewards (RLVR) is now a crucial stage in training reasoning models. In RLVR, a policy samples candidate solutions, an automatic verifier scores them, and the policy is updated to more likely generate higher-reward outputs (Shao et al., 2024; Guo et al., 2025; Lambert et al., 2024). The prompt sets used at this stage are often assembled from competition archives, scraped problem banks, and synthetic mixtures, but are rarely disclosed. This opacity creates the possibility that reasoning benchmark prompt sets may be used during RLVR training stages. Consequently, measured performance gaps between competing models could be contaminated by exposure to even a small number of evaluation questions during training. To make sound decisions based on these evaluations, practitioners therefore need a way to assess whether benchmark questions were included in a model's RL training data, rather than inadvertently favoring systems that are overfit to a particular benchmark.

Membership inference is natural for this *auditing* problem, but does not transfer cleanly to RL. Existing membership inference attacks for pretraining and supervised fine-tuning assume access to a candidate sequence whose membership is being tested, and exploit how the model scores that sequence (Yeom et al., 2018; Shi et al., 2024; Zhang et al., 2025; Mattern et al., 2023; Carlini et al., 2022). In RL, however, the auditor typically only knows the prompt as training uses model-generated responses sampled during optimization. Because these are generally unavailable after training, there is no fixed target sequence on which to apply conventional likelihood-based membership inference attacks. Instead, exposure to a prompt must be inferred from how RL has altered the model's distribution over *any* possible solution trajectory, rather than a single target.

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.

Recent RL-specific membership attacks suggest that training exposure can be detected from properties of a model's outputs (Zhang et al., 2026; Tao et al., 2026). We find, however, that no statistic reliably captures this effect across models and training settings. Instead, RL exposure leaves a multi-dimensional signature. Observed prompts may differ in the probabilities assigned to sampled continuations, the similarity of generated solution trajectories and answers, the saturation of outcomes, and the variation across rollouts. However, the magnitude of these effects varies substantially across model–dataset pairs, and a statistic that separates members for one dataset can be uninformative for another. A robust detector must therefore combine complementary signals rather than be based on a single statistical signature.

In this work, we introduce SCAR (Self-sharpening, Collapse, Answer saturation, and Rollout-sharpening), a label-free framework for RL membership inference. Our contributions are threefold. First, we characterize RL membership as a multi-dimensional change in the rollout distribution, and identify four complementary signatures of exposure. Second, we show through controlled training experiments that these signatures separate seen from unseen prompts progressively during RL, but that no individual signature is uniformly informative across checkpoints. Finally, we introduce SCAR, a membership-label-free aggregation framework whose performance is robust across a broad family of unsupervised fusion rules where our strongest aggregation reaches 0.764 mean ROC-AUC, compared with 0.673 for the strongest single-statistic baseline.

2 Related work

Detecting RL training data. Recent work has begun studying membership specifically at the reinforcement-learning stage. Self-Critique (Tao et al., 2026) detects exposure through changes in token-entropy profiles under self-revision. Min-kNN (Zhang et al., 2026) instead samples multiple completions and measures their nearest-neighbour edit distances, motivated by the structural convergence of reasoning trajectories after RL. Both approaches depart from conventional sequence-level membership inference by examining properties of model-generated outputs.

Membership inference for language models. Classical membership inference for language models identifies training examples using shadow models (Shokri et al., 2017), loss thresholds (Yeom et al., 2018), likelihood-ratio calibration (Carlini et al., 2022), neighbourhood comparison (Mattern et al., 2023), or lower-tail token likelihoods such as Min-K% and Min-K%++ (Shi et al., 2024; Zhang et al., 2025). ReCaLL similarly calibrates membership using relative conditional likelihoods (Xie et al., 2024). These methods generally assume access to a candidate sequence whose likelihood can be evaluated. Other work improves robustness by aggregating evidence at the document or dataset level (Meeus et al., 2024; Maini et al., 2024). Large-scale evaluations nevertheless show that membership signals can become weak once distribution shift between members and non-members is carefully controlled (Duan et al., 2024).

Benchmark contamination and RL dynamics. A related literature detects benchmark contamination without necessarily identifying the training stage, using memorized example orderings (Oren et al., 2024), benchmark reconstruction (Golchin & Surdeanu, 2024; Sainz et al., 2023), confidence under matched perturbations (Zhang et al., 2024), and lexical or semantic overlap (Xu et al., 2024; Yang et al., 2023). Separately, analyses of reasoning-oriented RL show that optimization can concentrate the policy distribution, reduce entropy and response diversity (Cui et al., 2025; Jin et al., 2026), alter pass@k behaviour (Yue et al., 2025), and place disproportionate updates on uncertain tokens (Wang et al., 2025). Together, these results suggest that RL can leave observable changes in the distribution of generated solutions.

3 Problem Setting and RL Membership Signatures

3.1 RL Membership Inference

Let π_{θ_0} denote a base policy and $\mathcal{D}_{\mathrm{RL}}$ the set of prompts used for reinforcement learning, producing an audited policy π_θ. An auditor is given a pool of n candidate questions $\mathcal{Q} = \{q_1, \ldots, q_n\}$, where each question has an unobserved membership label $y_q = \mathbf{1}[q \in \mathcal{D}_{\mathrm{RL}}]$. The goal is to construct a score function $f(\pi_\theta, \mathcal{Q}, q) \in \mathbb{R}$ that ranks questions by evidence of RL exposure, with larger

values indicating greater evidence of membership in $\mathcal{D}_{\mathrm{RL}}$. We assume the auditor can sample responses from π_θ and obtain the log-probability of each sampled token, but has no access to model parameters, gradients, optimizer state, or RL training data. For tasks with verifiable answers, SCAR additionally uses the task verifier to construct the Answer saturation axis from the model's extracted final answers.

3.2 From Likelihood Training to Reward Optimization

The distinction between RL and earlier training stages stems from their optimization objectives. Both pretraining and supervised fine-tuning (SFT) are likelihood-based: pretraining increases the likelihood of observed sequences, while SFT increases the likelihood of a fixed target response a conditioned on a prompt q,

$$\mathcal{L}_{\mathrm{PT}} = -\mathbb{E}_{x\sim\mathcal{D}_{\mathrm{PT}}}\left[\sum_{t=1}^{|x|}\log \pi_\theta(x_t \mid x_{<t})\right], \qquad \mathcal{L}_{\mathrm{SFT}} = -\mathbb{E}_{(q,a)\sim\mathcal{D}_{\mathrm{SFT}}}\left[\log \pi_\theta(a \mid q)\right], \tag{1}$$

where $\mathcal{D}_{\mathrm{PT}}$ and $\mathcal{D}_{\mathrm{SFT}}$ are training distributions. In both settings, the optimization objective is defined on an observed target sequence and favors assigning higher likelihood to that sequence. RL changes this structure. Rather than fitting a fixed response, the policy generates its own rollouts and is updated according to their rewards:

$$\max_\theta \;\; \mathbb{E}_{q\sim\mathcal{D}_{\mathrm{RL}}}\mathbb{E}_{u\sim\pi_\theta(\cdot|q)}\left[R(u,q)\right]. \tag{2}$$

Consequently, exposure to a prompt during RL can affect the broader distribution of generated solutions, rather than the likelihood of a particular target response. This motivates examining multiple properties of the rollout distribution when characterizing the effect of RL exposure.

3.3 Signatures of RL Exposure

The central question of our work is which signatures are detectable after a prompt has been used during RL. Unlike pretraining membership, there is no known training response whose likelihood can simply be evaluated. We therefore study how exposure changes the model's distribution over newly sampled solutions, and find that RL exposure leaves several distinct signatures in these rollouts, rather than a single universal membership statistic. We organize them into four categories: Self-sharpening, Collapse, Answer saturation, and Rollout-sharpening across rollouts.

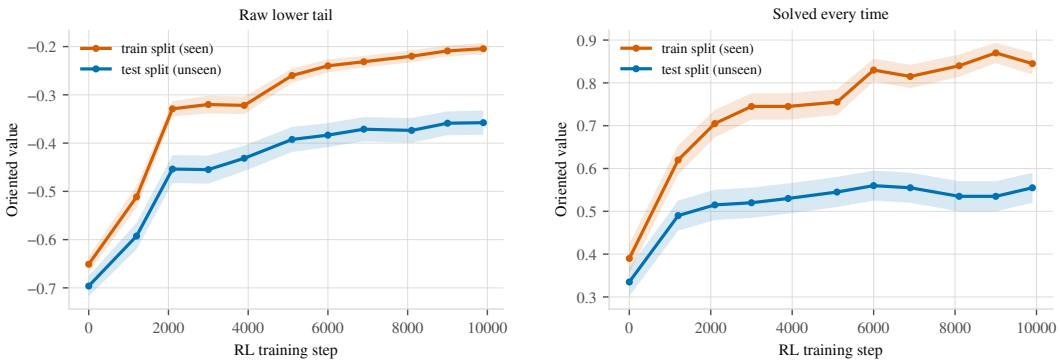
3.4 Four fingerprint axes

RL exposure can reshape the conditional distribution of solutions along several dimensions: increasing confidence in sampled continuations, concentrating on fewer solution paths, saturating outcomes, and reducing variation across independent generations. We capture these effects with 15 features organized into four axes. The features and their orientations are fixed across all models and benchmarks, with larger values indicating stronger evidence of exposure. Complete notation and feature definitions are given in Appendix A.

S: Self-sharpening within a rollout. GRPO (Shao et al., 2024) updates the policy toward sampled completions with positive group-relative advantage. For intuition, consider the unclipped policy-gradient form of GRPO:

$$\mathcal{J}_{\mathrm{GRPO}} = \mathbb{E}_{q\sim\mathcal{D}_{\mathrm{RL}}}\mathbb{E}_{u_1,\ldots,u_G\sim\pi_\theta(\cdot|q)}\left[\frac{1}{G}\sum_{i=1}^{G}\hat{A}_i\frac{1}{T_i}\sum_{t=1}^{T_i}\log \pi_\theta(x_t^i \mid q, x_{<t}^i)\right], \qquad \hat{A}_i = \frac{R_i - \bar{R}}{\mathrm{sd}(R)}. \tag{3}$$

Tokens in positive-advantage rollouts will be pushed toward higher probability, while tokens in negative-advantage rollouts are pushed down. Exposure to a prompt can therefore sharpen the probability profile of rewarded continuations, particularly around uncertain decision points (Wang et al., 2025). We capture this signal using the lower tail $\mathrm{mink}_{20}(L_i)$, the fraction of tokens below τ_{low}, the median token log-probability, the mean log-probability over the final quarter, and the negative interquartile range. Each statistic is computed within a rollout and averaged over the N rollouts.



(a) Self-sharpening: lower-tail token likelihood.

(b) Answer saturation: solved every rollout.

Figure 1: Examples of feature dynamics during GRPO training on GSM8K. Seen prompts increasingly separate from unseen prompts in both token-level confidence and outcome saturation.

C: Trajectory collapse. RLVR can reduce policy entropy and response diversity, concentrating probability mass on fewer reasoning trajectories (Jin et al., 2026; Jang et al., 2026; Zhang et al., 2026). This observation motivates the hypothesis that repeated exposure to a prompt makes its rollout distribution more concentrated around a small set of similar solution paths. We measure this effect at complementary granularities using the closest-pair normalized edit distance, cross-rollout token entropy at aligned positions, and the fraction of rollouts within distance τ_{dist} of the medoid (the rollout whose text is most similar to others on average). Together, these features capture local repetition and broader concentration in the rollout distribution.

A: Answer saturation. Because RL directly optimizes a verifier reward, exposure should concentrate the distribution of final outcomes. Under independent rollouts, $\Pr(\text{solved at least once}) = 1-(1-p_q)^N$ and $\Pr(\text{solved every time}) = p_q^N$, distinguishing occasional success from saturation. Moreover, Mahrooghi et al. (2026) shows that under binary rewards the policy-gradient magnitude contains the factor $\sqrt{p_q(1-p_q)}$, so the learning signal vanishes as $p_q \to 0$ or $p_q \to 1$. Finally, motivated by self-consistency (Wang et al., 2022), we measure the mass on the correct modal answer and the margin $\hat{p}_{\text{ans}}(a_{(1)}) - \hat{p}_{\text{ans}}(a_{(2)})$, capturing concentration of the final-answer distribution beyond correctness alone.

R: Rollout-sharpening across rollouts. While S measures confidence *within* a rollout, R asks whether these confidence statistics are consistent *across* independent generations. RLVR has been observed to narrow the reasoning distribution and reduce response diversity (Yue et al., 2025; Jin et al., 2026). Let z_i denote a rollout-level summary, such as the mean log-probability $\bar{L}_i$ or near-certain token fraction v_i. These observations motivate testing whether RL exposure also reduces cross-rollout variation in such summaries. We therefore use the negative max–median gap $-(\max_i \bar{L}_i - \text{median}_i\, \bar{L}_i)$, which measures how far the most confident rollout stands above the typical rollout, and the negative MAD $-\,\text{median}_i\, |v_i - \text{median}_j\, v_j|$, a robust measure of variation in near-certain token rates. Larger values indicate a smaller upper confidence gap and lower median variability, respectively.

These axes capture complementary manifestations of the same training event. Existing detectors effectively read individual axes of this collection, but their effectiveness changes substantially across checkpoints and benchmarks (Table 3). More generally, the axis carrying the strongest membership signal depends on the audited model and is not known in advance, motivating combining evidence across all four axes rather than relying on any single signature.

3.4.1 Evidence from training dynamics.

To examine how the proposed signatures emerge during training, we train OLMo2-1B (OLMo et al., 2024) on the GSM8K training set (Cobbe et al., 2021) with GRPO and evaluate all 15 features (Section A.2) at 11 checkpoints. As training progresses, seen prompts develop higher token confidence, more concentrated solution paths, more reliable answers, and greater consistency across rollouts

than unseen prompts. Figure 1 illustrates this evolution for two representative features, showing increasing separation in lower-tail token likelihood and answer saturation. This controlled trajectory is important because a statistic changing during RL is not sufficient for membership inference: general training progress can affect both member and non-member prompts. The membership-relevant quantity is the differential change between the two populations. Figure 1 shows that both groups can change during optimization, but the gap between them grows with continued training exposure. Appendix A.2 reports the full training dynamics of all features.

4 Label-Free Multi-Axis Aggregation

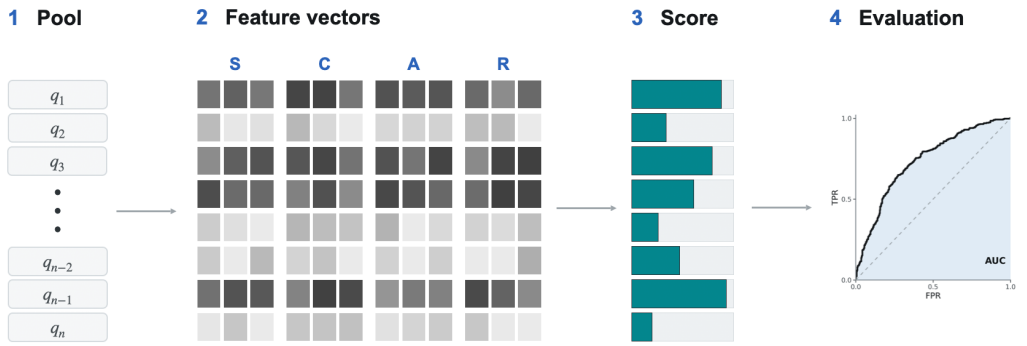


Figure 2: **Overview of the audit pipeline.** From a pool with unobserved membership labels, we draw N rollouts per question, extract features along four axes (S, C, A, R), and aggregate them into a membership score without supervision. Labels are used only for evaluation.

The four axes provide a common representation on which many label-free aggregation rules can operate. We therefore treat fusion as a modular design choice rather than defining SCAR through a single estimator. Figure 2 gives a high-level overview of the full audit pipeline. In our experiments, we evaluate eleven such rules and we use Self-Calibrated Axis Fusion (SCAR-Cal) as the default because it obtains the strongest macro-average performance.

4.1 Self-Calibrated Axis Fusion

Self-Calibrated Axis Fusion combines the four axes using only information from the unlabeled audit pool. It first constructs balanced axis scores, then estimates the reliability of each axis, and finally applies a covariance-aware refinement. Importantly, this is one aggregation rule over the shared four-axis representation; alternative aggregation rules are evaluated directly in Figure 3a.

4.1.1 Balanced axis construction

We first reduce the 15 features to four balanced axis scores. Let $\Phi_f(q)$ denote a feature, with larger values indicating stronger evidence of membership, and let $\mathrm{norm}(\cdot)$ standardize across the audit pool. For axis $a \in \mathcal{A} = \{S, C, A, R\}$ with feature set $\mathcal{F}_a$, we define $\mathbf{A}_{\cdot a} = \mathrm{norm}\Big(\frac{1}{|\mathcal{F}_a|}\sum_{f\in\mathcal{F}_a}\mathrm{norm}(\Phi_f)\Big)$, for each $a \in \mathcal{A}$. Thus, $\mathbf{A} \in \mathbb{R}^{n\times 4}$ gives each signature equal representation regardless of its number or scale of features. Further details are given in Appendix B.

4.1.2 Label-free reliability weighting

Because the axes vary in informativeness across models, we estimate their reliability directly from the unlabeled audit pool. We combine two signals: agreement with the remaining axes and separation into two latent groups. Let w_a^{agr} and w_a^{sep} denote these quantities. The initial weights and membership score are $w_a^{(0)} = w_a^{\mathrm{agr}} w_a^{\mathrm{sep}}$, and $s^{(0)} = \mathbf{A}w^{(0)}$. Axes therefore receive larger weight when they both agree with the other signatures and exhibit clear structure within the pool. The two reliability measures are defined in Appendix B.

4.1.3 TAIL REFINEMENT

The seed score weights axes individually but does not account for their dependence. We therefore use its highest- and lowest-scoring $\rho = 0.30$ tails as pseudo-seen and pseudo-unseen groups, estimate their mean difference δ and within-group covariance $\widehat{\Sigma}$, and apply a regularized Fisher-style refinement:

$$\Sigma_\lambda = (1-\lambda)\widehat{\Sigma} + \lambda \operatorname{diag}(\widehat{\Sigma}), \qquad w^{(1)} = \left(\Sigma_\lambda + \varepsilon \mathbf{I}\right)^{-1}\delta, \qquad s^{(1)} = \mathbf{A}w^{(1)}. \tag{4}$$

Appendix B gives the full construction and its connection to Fisher's linear discriminant.

5 EXPERIMENTS

Benchmarks and audit protocol. We evaluate eight model–dataset pairs: four public RL-trained checkpoints and four controlled GRPO runs with known membership. The public models are JustRL-DeepSeek-1.5B (He et al., 2025; Guo et al., 2025), DAPO-Qwen-32B (Yu et al., 2025; Yang et al., 2024a), Qwen-2.5-32B-SimpleRL-Zoo (Zeng et al., 2025), and Open-Reasoner-Zero-7B (Hu et al., 2025), spanning 1.5B–32B parameters. The controlled runs consist of OLMo2-1B-Instruct (OLMo et al., 2024) on GSM8K, Qwen2.5-Math-7B (Yang et al., 2024b) on AIME 2025 and 2026, and DeepSeek-Math-7B-Instruct (Shao et al., 2024) on AIME 2025. For the AIME experiments, we split the competition problems into train and held-out halves and augment the training half with NuminaMath (Li et al., 2024), reducing distribution-shift confounds between members and non-members (Duan et al., 2024). At audit time, for each question, we use $N = 16$ rollouts and apply the same SCAR-Cal configuration throughout, with $\lambda = 0.1$, $\rho = 0.30$, and one refinement step. Full details are given in Appendix C and D.

Baselines and metrics. We compare against five standard membership attacks. **PPL** (Yeom et al., 2018) uses mean sampled token log-probability, while **Min-K%** (Shi et al., 2024) averages the lowest $k\%$ of sampled token log-probabilities. **Min-K%++** (Zhang et al., 2025) additionally requires the full next-token distribution and thus assumes a stronger access model. Since there is no known target response whose likelihood can be evaluated when evaluating RL membership, we compute these statistics on the model's generated completions rather than on the prompt tokens. **Min-kNN** (Zhang et al., 2026) measures nearest-neighbour similarity across rollouts, and **Self-Critique (SC)** (Tao et al., 2026) compares greedy and self-revised entropy profiles. We report ROC-AUC and TPR at 5% and 10% FPR.

6 RESULTS

6.1 RANKING ACCURACY

Table 1 reports ROC-AUC across the pipelines, confirming that no single baseline is reliable across settings. Min-kNN is strong on ORZ and SimpleRL, but falls close to random on DS-Math/AIME25, while PPL and Min-K% exhibit the opposite pattern. SCAR-Cal is the only method that stays clear of chance in every setting, with a worst-case AUC of .687 compared with .523 for Min-kNN, the strongest baseline by mean AUC, across both public RL checkpoints and controlled GRPO runs. On some pipelines, SCAR-Cal matches rather than exceeds the best single statistic, but which statistic is most informative cannot be known without membership labels. By combining complementary signatures without such labels, SCAR-Cal provides more reliable performance across settings.

To test whether this separation reflects pre-existing differences rather than RL exposure, we additionally evaluate each controlled run on its matched pre-RL policy π_{θ_0} using identical candidate pools, labels, rollout budgets, and sampling settings. As shown in Table 1, SCAR-Cal remains near chance on all four base policies but increases substantially after RL. We report this control only for the controlled runs, where the matched pre-RL checkpoint is available; the public RL checkpoints do not provide a sufficiently matched pre-RL policy to cleanly isolate the audited RL stage.

Uncertainty corresponds to ± 1 bootstrap standard deviation over 4000 paired resamples of each audited question pool and captures question-sampling variability within each pipeline, not variation across checkpoints. The full axis-level base-policy results are reported in Appendix F.6.

Table 1: ROC-AUC across eight model–dataset pipelines. Values are point estimates with ± 1 bootstrap standard deviation over 4000 paired resamples where available. Bold marks the best method per column. † denotes baselines that require access to the full next-token distribution. SCAR-Cal (base) reports the base-policy control on π_{θ_0} for the four controlled runs.

Method	JustRL	DAPO	SimpleRL	ORZ	OLMo/GSM8K	Qwen/AIME25	Qwen/AIME26	DS-Math/AIME25	Mean
PPL	.667 ±.022	.716 ±.020	.628 ±.023	.681 ±.022	.617 ±.028	.656 ±.109	.684 ±.102	.569 ±.116	.652 ±.026
Min-K%	.680 ±.022	.729 ±.020	.635 ±.022	.683 ±.022	.617 ±.028	.656 ±.109	.689 ±.102	.569 ±.114	.657 ±.026
Min-kNN	.633 ±.023	.738 ±.020	.741 ±.020	.817 ±.018	.637 ±.028	.574 ±.115	.724 ±.094	.523 ±.118	.673 ±.028
Self-Critique†	.521 ±.018	.558 ±.023	.518 ±.017	.553 ±.023	.511 ±.019	.518 ±.073	.609 ±.085	.677 ±.097	.558 ±.019
Min-K%++†	.633 ±.023	.671 ±.021	.615 ±.023	.646 ±.022	.570 ±.029	.538 ±.117	.653 ±.102	.482 ±.113	.601 ±.026
SCAR-Cal	**.711** ±.021	**.809** ±.018	**.743** ±.020	**.830** ±.017	**.691** ±.027	**.687** ±.106	**.787** ±.087	**.851** ±.075	**.764** ±.021
SCAR-Cal (base)	–	–	–	–	.553 ±.029	.523 ±.118	.533 ±.111	.492 ±.116	–

Table 2: TPR at low-FPR operating points. Values are point estimates with ± 1 bootstrap standard deviation over 4000 paired resamples where available. Bold marks the best method per column. † denotes baselines that require access to the full next-token distribution.

FPR	Method	JustRL	DAPO	SimpleRL	ORZ	OLMo/GSM8K	Qwen/AIME25	Qwen/AIME26	DS-Math/AIME25	Mean
5%	PPL	**.180** ±.027	.167 ±.035	.083 ±.028	.100 ±.049	.110 ±.035	.154 ±.123	.000 ±.187	.308 ±.132	.138
	Min-K%	**.180** ±.026	**.187** ±.037	.093 ±.032	.100 ±.052	**.115** ±.033	.154 ±.123	.000 ±.189	.308 ±.132	.142
	Min-kNN	.130 ±.031	.183 ±.055	**.240** ±.048	**.240** ±.053	.090 ±.028	.077 ±.177	.067 ±.202	.000 ±.128	.128
	Self-Critique†	.037 ±.014	.070 ±.015	.103 ±.017	.063 ±.022	.055 ±.024	.077 ±.145	.000 ±.105	.154 ±.135	.070
	Min-K%++†	.140 ±.023	.147 ±.046	.097 ±.027	.110 ±.029	.105 ±.033	.077 ±.090	.067 ±.123	.154 ±.104	.112
	SCAR-Cal	.160 ±.043	.160 ±.064	.137 ±.052	.203 ±.042	.095 ±.029	**.308** ±.164	**.333** ±.142	**.462** ±.171	**.232**
10%	PPL	.247 ±.038	.317 ±.054	.190 ±.039	.280 ±.049	.160 ±.040	.231 ±.154	**.333** ±.215	.308 ±.133	.258
	Min-K%	.253 ±.036	.330 ±.051	.213 ±.040	.297 ±.049	.170 ±.041	.231 ±.154	**.333** ±.220	.308 ±.133	.267
	Min-kNN	.220 ±.037	**.387** ±.068	.357 ±.062	**.480** ±.088	.125 ±.028	.385 ±.183	**.333** ±.229	.154 ±.186	.305
	Self-Critique†	.077 ±.018	.093 ±.021	.137 ±.024	.170 ±.040	.100 ±.030	.231 ±.164	.067 ±.154	.154 ±.176	.128
	Min-K%++†	.190 ±.032	.277 ±.041	.203 ±.041	.223 ±.049	**.175** ±.041	.077 ±.119	.067 ±.185	.154 ±.115	.171
	SCAR-Cal	**.310** ±.053	.327 ±.069	**.393** ±.076	.427 ±.104	**.175** ±.058	**.462** ±.167	**.333** ±.184	**.615** ±.176	**.380**

6.2 BEHAVIOUR AT LOW FALSE-POSITIVE RATES

ROC-AUC summarizes performance across all thresholds, but practical auditing requires reliable detection at low false-positive rates. Table 2 reports TPR at 5% and 10% FPR. The gains are especially pronounced on the controlled AIME runs, where single-statistic baselines are often unstable or near zero. This mirrors the ROC-AUC results: no single baseline dominates across pipelines, whereas multi-axis aggregation provides more consistent low-FPR detection.

7 ANALYSIS AND ABLATIONS

7.1 AXIS INFORMATIVENESS VARIES ACROSS MODEL-DATASET PAIRS

Table 3 evaluates each axis independently and shows that no single signature is uniformly reliable across checkpoints. The strongest axis changes across pipelines: self-sharpening dominates in some settings, trajectory collapse or answer saturation in others, and rollout-sharpening is strongest elsewhere. At the feature level, Figure 13 shows the same: no individual feature is consistently strong across benchmarks, and the best mean ROC-AUC of any single feature is only about 0.68.

Table 3: ROC-AUC of each axis in isolation compared with full SCAR-Cal. Bold marks the best-performing **individual axis** in each column and the overall performance of SCAR-Cal.

Axis	JustRL	DAPO	SimpleRL	ORZ	OLMo/GSM8K	Qwen/AIME25	Qwen/AIME26	DS-Math/AIME25	Mean
S (Self-sharpening)	.686	.758	.694	.725	.622	.641	.622	**.764**	**.689**
C (Trajectory collapse)	**.689**	.735	**.725**	.800	**.666**	.574	.769	.482	.680
A (Answer saturation)	.528	**.763**	.710	**.808**	.653	.646	**.798**	.587	.687
R (Rollout-sharpening)	.615	.731	.609	.698	.650	**.682**	.711	.672	.671
SCAR-Cal (All four)	.711	.809	.743	.830	.691	.687	.787	.851	**.764**

To test whether each axis contributes beyond the remaining signatures, we repeat SCAR-Cal after removing one axis at a time while leaving the aggregation procedure unchanged. Table 4 shows

that removing any axis decreases mean ROC-AUC relative to the full model. Together with the single-axis results in Table 3, this shows that the gain from SCAR-Cal is not driven by one dominant signature and that each axis contributes complementary information on average.

Table 4: Leave-one-axis-out ablation of SCAR-Cal. Mean ROC-AUC is computed across the eight model–dataset pipelines; uncertainty follows the same bootstrap protocol as Table 1. Δ AUC is relative to the full 15-feature SCAR-Cal.

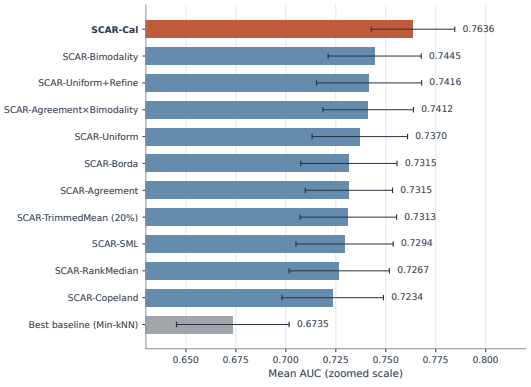
Removed axis	Remaining features	Mean AUC	Δ AUC
None (full SCAR-Cal)	15	**.7636** $\pm$.0209	–
S (Self-sharpening)	10	.7098 $\pm$.0274	-.0538
C (Trajectory collapse)	12	.7486 $\pm$.0220	-.0150
A (Answer saturation)	10	.7246 $\pm$.0221	-.0390
R (Rollout-sharpening)	13	.7364 $\pm$.0255	-.0272

7.2 Ablation Study

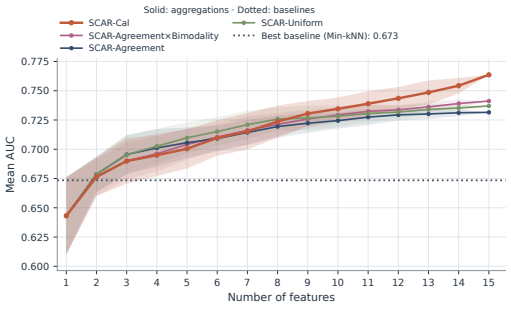
Having established that an ensemble of multiple features achieves the best membership inference performance, we next analyze how much performance depends on the particular rule used to combine them, as well as how many such features are needed for membership inference.

We first compare SCAR-Cal with ten additional unsupervised aggregation rules applied to the same four axes, spanning uniform averaging, rank-based fusion, separation-driven weighting, and spectral weighting (Parisi et al., 2014) (Figure 3a). Our results show that all eleven aggregation strategies exceed both the strongest individual axis and the strongest single-statistic baseline in macro-average ROC-AUC across the eight pipelines, suggesting that the main contribution of SCAR comes from representing RL exposure along multiple complementary dimensions, and that this benefit is relatively robust to the specific aggregation rule used. In particular, SCAR-Uniform reaches .737 mean AUC without reliability weighting or covariance refinement, compared with .673 for Min-kNN, the best baseline (SCAR-Cal increases this further to .764).

To test SCAR's robustness to identified feature sets, we also evaluate SCAR-Cal over 30 random feature addition orders (Figure 3b). We observe that performance improves steadily as features are added to the aggregation. With only three randomly selected features, the average performance already exceeds the strongest single-metric baseline, highlighting the importance of combining multiple weak, complementary signals compared to depending on any single statistic.



(a) Comparison of eleven unsupervised aggregation strategies over the same four axes.



(b) Mean ROC-AUC across eight benchmarks as features are added in 30 random orders; shading shows ± 1 standard deviation.

Figure 3: Robustness of multi-signal aggregation. Left: performance across different aggregation rules. Right: performance as the number of features increases. Per-pipeline numerical values are reported in Appendix F.5.

7.3 Robustness to Prompt Formulations

Up until now, we have assumed a setting where explicit prompts from an existing benchmark are tested for membership inference in the unknown training set of an RLVR pipeline. However, this setting implicitly assumes that benchmark prompts may only contaminate RLVR training when directly integrated into the set of training prompts. In practice, very similar prompts to benchmark instantiations may also contaminate training sets, undermining any conclusions about generalization. In this experiment, we investigate whether our method can detect when similar prompts from a variety of transformations are used for RLVR training.

Specifically, we re-write the prompts from OLMo/GSM8K using Gemini 3.5 Flash (Google DeepMind, 2026) to generate one- and two-word edits, light, medium, and heavy paraphrases, and cover-story rewrites with fixed or resampled numbers (check Appendix E for more details). Each re-write of a prompt is sampled three times to provide a diverse set of contamination prompts. After excluding unfaithful rewrites, we retain 166 GSM8K questions valid across all conditions and re-train OLMo with prompts included in the training set. We observe that when testing these prompts for membership inference, SCAR-Cal remains informative under small lexical changes, achieving mean ROC-AUC 0.628 for one-word edits, but degrades under stronger reformulations, reaching 0.507 under heavy paraphrasing and 0.499 for cover-story rewrites with resampled numbers. Single-statistic baselines likewise approach chance under strong rewrites, indicating that the rollout statistics exploited by current methods are not invariant to substantial changes in prompt form. These results motivate more semantically invariant membership signatures.

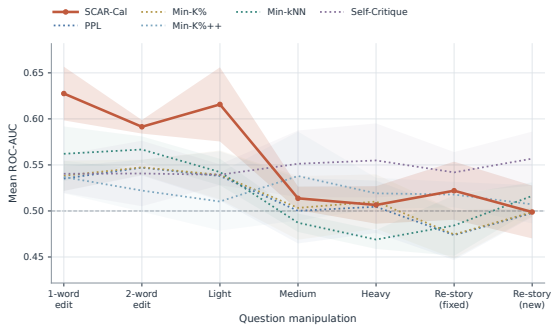


Figure 4: Robustness to Gemini prompt rewriting on OLMo/GSM8K. Curves show mean full-pool ROC-AUC across three rewrite runs, with shading indicating ± 1 standard deviation across runs.

7.4 Further Analyses

Due to the page limit, we defer additional supporting analyses to Appendix F. *Hyperparameter sensitivity* (Table 8) shows that SCAR-Cal does not depend on careful tuning, with a single fixed configuration performing close to oracle selection. *Pool robustness* (Figure 9) shows that performance remains stable across a wide range of contamination rates and improves as the audit pool grows, indicating robustness to unknown membership prevalence. Finally, *rollout-budget scaling* (Table 9) shows that additional rollouts substantially strengthen detection.

8 Conclusion

We study RL-stage membership inference for reasoning models and show that training exposure leaves a heterogeneous signature across the rollout distribution. SCAR captures this through four complementary axes and membership-label-free aggregation, improving robustness over individual signatures across eight RL pipelines. Controlled training dynamics further show that the member–non-member separation emerges during RL. These results suggest that reliable RL auditing benefits from combining complementary rollout signatures rather than relying on a single statistic. Limitations are discussed in Appendix H.

AI USE STATEMENT

We used generative AI tools to assist with drafting, editing, and improving the clarity of the manuscript. Gemini 3.5 Flash was used to generate the prompt reformulations evaluated in our paraphrasing robustness experiment, as described in Section 7.3. Apart from these explicitly reported prompt transformations, no AI-generated examples, labels, or evaluation data were used in the experiments. All methodological choices, experimental design, analysis, and interpretation were carried out and verified by the authors.

REPRODUCIBILITY STATEMENT

We provide complete feature definitions, aggregation details, training hyperparameters, dataset construction procedures, and prompt-rewriting protocols in the appendices. All experiments use fixed feature definitions and aggregation settings across model–dataset pairs to support direct reproduction of the reported results.

REFERENCES

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *2022 IEEE symposium on security and privacy (SP)*, pp. 1897–1914. IEEE, 2022.

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

Jacob Cohen. *Statistical power analysis for the behavioral sciences*. routledge, 2013.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, et al. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*, 2025.

Michael Duan, Anshuman Suri, Niloofar Mireshghallah, Sewon Min, Weijia Shi, Luke Zettlemoyer, Yulia Tsvetkov, Yejin Choi, David Evans, and Hannaneh Hajishirzi. Do membership inference attacks work on large language models? *arXiv preprint arXiv:2402.07841*, 2024.

Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7 (2):179–188, 1936.

Shahriar Golchin and Mihai Surdeanu. Time travel in llms: Tracing data contamination in large language models. In *International Conference on Learning Representations*, volume 2024, pp. 43008–43029, 2024.

Google DeepMind. Gemini 3.5 flash model card. `https://deepmind.google/models/model-cards/gemini-3-5-flash/`, may 2026. Accessed: 2026-09-24.

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, Zekai Qu, Zeyuan Liu, Yinghao Chen, Yuxin Zuo, Cheng Qian, Kaiyan Zhang, Weize Chen, Chaojun Xiao, Ganqu Cui, et al. Justrl: Scaling a 1.5 b llm with a simple rl recipe. *arXiv preprint arXiv:2512.16649*, 2025.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *ArXiv*, abs/2503.24290, 2025. URL `https://api.semanticscholar.org/CorpusID:277468189`.

Jaeeun Jang, Hansle Lee, and Sangmin Kim. A few bad apples spoil the bunch: Preventing global entropy collapse driven by a small set of tokens in llm reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 13134–13154, 2026.

Renren Jin, Pengzhi Gao, Yuqi Ren, Zhuowen Han, Tongxuan Zhang, Wuwei Huang, Wei Liu, Jian Luan, and Deyi Xiong. Revisiting entropy in reinforcement learning for large reasoning models. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 25300–25322, 2026.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Olivier Ledoit and Michael Wolf. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, 88(2):365–411, 2004. ISSN 0047-259X. doi: https://doi.org/10.1016/S0047-259X(03)00096-4. URL https://www.sciencedirect.com/science/article/pii/S0047259X03000964.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, et al. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13(9):9, 2024.

Ilia Mahrooghi, Aryo Lotfi, and Emmanuel Abbe. Goldilocks rl: Tuning task difficulty to escape sparse rewards for reasoning. *arXiv preprint arXiv:2602.14868*, 2026.

Pratyush Maini, Hengrui Jia, Nicolas Papernot, and Adam Dziedzic. Llm dataset inference: Did you train on my dataset? *Advances in Neural Information Processing Systems*, 37:124069–124092, 2024.

Justus Mattern, Fatemehsadat Mireshghallah, Zhijing Jin, Bernhard Schölkopf, Mrinmaya Sachan, and Taylor Berg-Kirkpatrick. Membership inference attacks against language models via neighbourhood comparison. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 11330–11343, 2023.

Matthieu Meeus, Shubham Jain, Marek Rei, and Yves-Alexandre de Montjoye. Did the neurons read your book? document-level membership inference for large language models. In *33rd USENIX Security Symposium (USENIX Security 24)*, pp. 2369–2385, 2024.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, et al. 2 olmo 2 furious. *arXiv preprint arXiv:2501.00656*, 2024.

Yonatan Oren, Nicole Meister, Niladri Chatterji, Faisal Ladhak, and Tatsunori Hashimoto. Proving test set contamination in black-box language models. In *International Conference on Learning Representations*, volume 2024, pp. 16354–16372, 2024.

Fabio Parisi, Francesco Strino, Boaz Nadler, and Yuval Kluger. Ranking and combining multiple predictors without labeled data. *Proceedings of the National Academy of Sciences*, 111(4):1253–1258, 2014.

Oscar Sainz, Jon Campos, Iker García-Ferrero, Julen Etxaniz, Oier Lopez de Lacalle, and Eneko Agirre. Nlp evaluation in trouble: On the need to measure llm data contamination for each benchmark. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 10776–10787, 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.

Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. Detecting pretraining data from large language models. In *International Conference on Learning Representations*, volume 2024, pp. 51826–51843, 2024.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pp. 3–18. IEEE, 2017.

Yongding Tao, Tian Wang, Yihong Dong, Huanyu Liu, Kechi Zhang, Hu XiaoLong, and Ge Li. Detecting data contamination from reinforcement learning post-training for large language models. In *International Conference on Learning Representations*, volume 2026, pp. 104568–104590, 2026.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. TRL: Transformers Reinforcement Learning, 2020. URL https://github.com/huggingface/trl.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhen-Ru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bo-Wen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *ArXiv*, abs/2506.01939, 2025. URL https://api.semanticscholar.org/CorpusID:279119146.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Roy Xie, Junlin Wang, Ruomin Huang, Minxing Zhang, Rong Ge, Jian Pei, Neil Zhenqiang Gong, and Bhuwan Dhingra. Recall: Membership inference via relative conditional log-likelihoods. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 8671–8689, 2024.

Cheng Xu, Shuhao Guan, Derek Greene, M Kechadi, et al. Benchmark data contamination of large language models: A survey. *arXiv preprint arXiv:2406.04244*, 2024.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runxin Xu, Tianhao Li, Tianyu Liu, Wang Fan, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xinyuan Li, Yang Fan, Yang Tan, Yu Wan, Yuze Gao, Yuqiang Xie, Zekun Wang, Zengmao Wang, Zhihao Xiang, Zhitang Li, Zhiyuan Gao, Zhiyuan Xu, and Zick Ye. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024a. URL https://arxiv.org/abs/2412.15115.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024b.

Shuo Yang, Wei-Lin Chiang, Lianmin Zheng, Joseph E Gonzalez, and Ion Stoica. Rethinking benchmark and contamination for language models with rephrased samples. *arXiv preprint arXiv:2311.04850*, 2023.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*, pp. 268–282. IEEE, 2018.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tian-Tian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Honglin Yu, Weinan Dai, Yuxuan Song, Xiang Wei, Haodong Zhou, Jingjing Liu, Wei Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yong-Xu Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale. *ArXiv*, abs/2503.14476, 2025. URL https://api.semanticscholar.org/CorpusID:277104124.

Yang Yue, Zhiqin Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *ArXiv*, abs/2504.13837, 2025. URL https://api.semanticscholar.org/CorpusID:277940134.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *arXiv preprint arXiv:2503.18892*, 2025.

Hongbo Zhang, Yue Yang, Jianhao Yan, Guangsheng Bao, and Yue Zhang. Detecting rlvr training data via structural convergence of reasoning. *arXiv preprint arXiv:2602.11792*, 2026.

Huixuan Zhang, Yun Lin, and Xiaojun Wan. Pacost: Paired confidence significance testing for benchmark contamination detection in large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 1794–1809, 2024.

Jingyang Zhang, Jingwei Sun, Eric Yeats, Yang Ouyang, Martin Kuo, Jianyi Zhang, Hao Yang, and Hai Li. Min-k%++: Improved baseline for pre-training data detection from large language models. In *International Conference on Learning Representations*, volume 2025, pp. 64845–64862, 2025.

A THE 15 FEATURES IN FULL

A.1 NOTATION

For each candidate question q, we draw N independent rollouts $u^1, \ldots, u^N \sim \pi_\theta(\cdot \mid q)$. Rollout i contains tokens $x_1^i, \ldots, x_{T_i}^i$ with sampled log-probabilities $\ell_t^i = \log \pi_\theta(x_t^i \mid q, x_{<t}^i)$, decoded response r_i, and verifier outcome $c_i = \mathbf{1}[g(u_i) \equiv a_q^\star]$, where function g extracts the final answer of u_i. We write $L_i = (\ell_t^i)_{t=1}^{T_i}$, $\bar{L}_i$ for its mean, and $\hat{p}_q = N^{-1} \sum_i c_i$ for the rollout pass rate.

For a sequence $v = (v_1, \ldots, v_T)$ with ordered entries $v_{(1)} \le \cdots \le v_{(T)}$, we define $\mathrm{mink}_k(v)$ following Shi et al. (2024) as $\mathrm{mink}_k(v) = \frac{1}{m} \sum_{j=1}^{m} v_{(j)}$, where $m = \left\lceil \frac{kT}{100} \right\rceil$, i.e., the mean of the lowest $k\%$ of entries in v. We further let $d_{ij} \in [0, 1]$ denote the normalized Levenshtein distance between rollout texts r_i and r_j. Table 5 defines the 15 features. We use the fixed hyperparameters $\tau_{\mathrm{low}} = -2$ for low-probability tokens, $\tau_{\mathrm{high}} = -0.01$ for near-certain tokens, and $\tau_{\mathrm{dist}} = 0.30$ for the representative-text cluster radius. We further define $T_{\min} = \min_i T_i$, $h_i = \max(1, \lceil T_i/4 \rceil)$, and let Q_p denote the pth percentile. The representative rollout is the medoid $m = \arg\min_i \sum_{j \neq i} d_{ij}$. Token entropy $\hat{H}_t$ is computed from the empirical distribution of sampled token IDs across rollouts at position t. Final answers are extracted from the full responses, with $\hat{p}_{\mathrm{ans}}$ denoting their empirical distribution. We define $a_{\mathrm{mode}} = \arg\max_a \hat{p}_{\mathrm{ans}}(a)$ as the modal answer, and let $a_{(1)}$ and $a_{(2)}$ denote the most and second-most frequent answers, respectively. Finally, $v_i = \frac{1}{T_i} \sum_t \mathbf{1}[\ell_t^i > \tau_{\mathrm{high}}]$ denotes the fraction of near-certain tokens in rollout i.

Table 5: The 15 features used by SCAR.

Axis	Statistic	Definition
S	Raw lower tail	$\frac{1}{N} \sum_i \mathrm{mink}_{20}(L_i)$
	Low-likelihood token rate	$-\frac{1}{N} \sum_i \frac{1}{T_i} \sum_t \mathbf{1}[\ell_t^i < \tau_{\mathrm{low}}]$
	Median token likelihood	$\frac{1}{N} \sum_i \mathrm{median}(L_i)$
	Final-quarter token likelihood	$\frac{1}{N} \sum_i \frac{1}{h_i} \sum_{t=T_i-h_i+1}^{T_i} \ell_t^i$
	Within-rollout likelihood IQR	$-\frac{1}{N} \sum_i \big(Q_{75}(L_i) - Q_{25}(L_i)\big)$
C	Closest-pair similarity	$-\min_{i \neq j} d_{ij}$
	Aligned-token entropy	$-\frac{1}{T_{\min}} \sum_{t=1}^{T_{\min}} \hat{H}_t$
	Representative-text cluster share	$\frac{1}{N-1} \sum_{j \neq m} \mathbf{1}[d_{mj} < \tau_{\mathrm{dist}}]$
A	Pass rate	$\hat{p}$
	Solved at least once	$\mathbf{1}[\hat{p} > 0]$
	Solved every time	$\mathbf{1}[\hat{p} = 1]$
	Correct modal-answer share	$\hat{p}_{\mathrm{ans}}(a_{\mathrm{mode}})\, \mathbf{1}[a_{\mathrm{mode}} \text{ is correct}]$
	Modal-answer separation	$\hat{p}_{\mathrm{ans}}(a_{(1)}) - \hat{p}_{\mathrm{ans}}(a_{(2)})$
R	Mean-likelihood max–median gap	$-\big(\max_i \bar{L}_i - \mathrm{median}_i\, \bar{L}_i\big)$
	Near-certain token-rate MAD	$-\,\mathrm{median}_i \lvert v_i - \mathrm{median}_j\, v_j \rvert$

A.2 FEATURE DYNAMICS THROUGHOUT GSM8K TRAINING

Experiment and interpretation. We follow OLMo-2-0425-1B-Instruct (OLMo et al., 2024) during GRPO training on GSM8K at steps $0, 1200, 2100, 3000, 3900, 5100, 6000, 6900, 8100, 9000, 9900$. Each checkpoint is evaluated on 200 questions from the seen training split and 200 from the unseen test split, using $N = 16$ rollouts per question. Step 0 is the matched untuned policy. Figures 5 to 8 show all 15 features in Table 5, using the same definitions and names: five S, three C, five A, and two R statistics.

We use GSM8K for this training-dynamics analysis because its dense training split provides enough repeatedly exposed prompts to estimate smooth checkpoint-level trends. In the AIME runs, only 15 competition problems are embedded in a much larger NuminaMath training mixture, so checkpoint-level estimates for the benchmark subset are considerably noisier.

Curves show split means and ± 1 standard error across questions. A change shared by both splits can reflect general improvement; separation between their curves is the membership-relevant observation. The bands describe uncertainty in each mean, not variation across independent training runs or a hypothesis test.

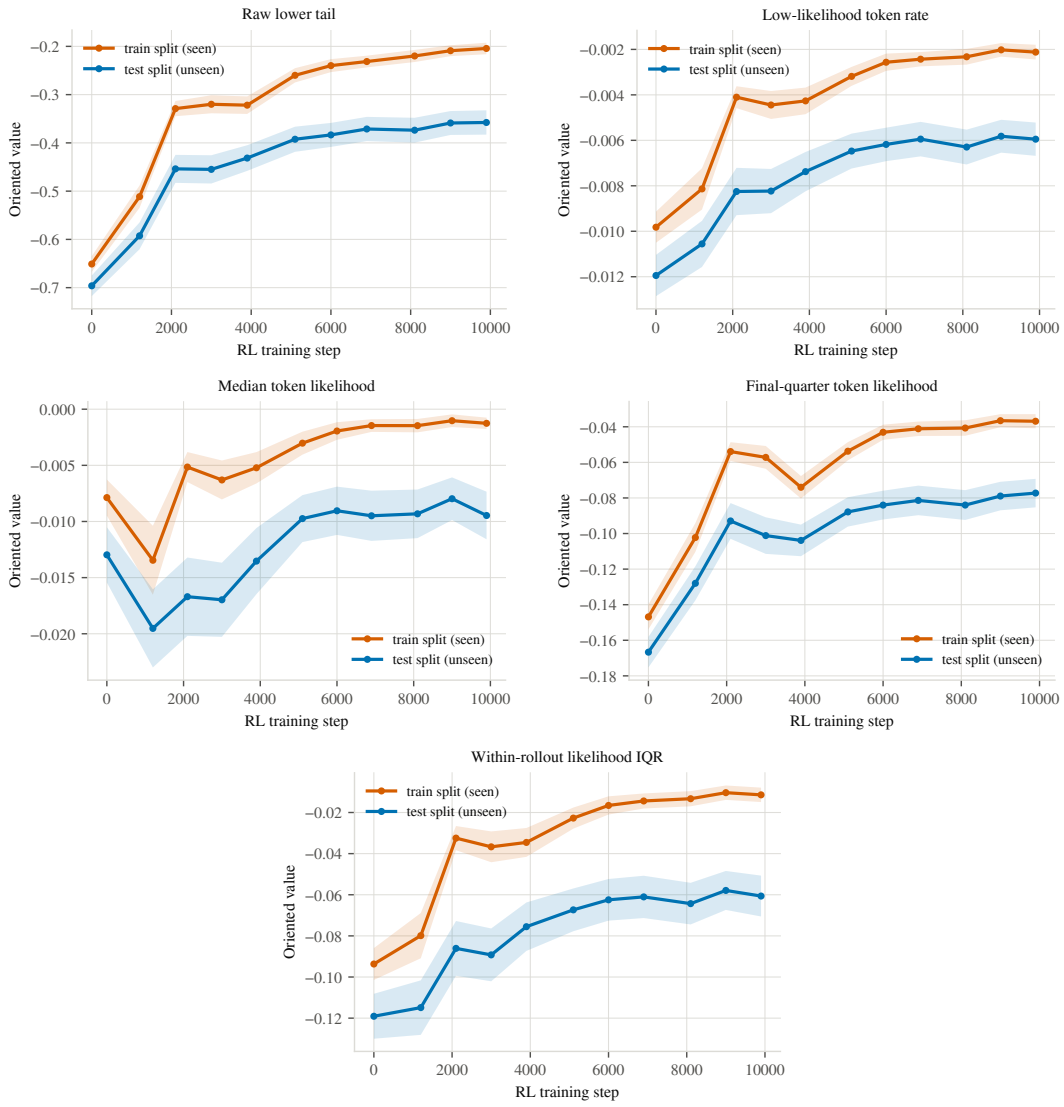


Figure 5: **Self-sharpening throughout GSM8K training.** The five panels use the exact S statistics in Table 5. Orange denotes seen prompts and blue unseen prompts; bands show ± 1 standard error across questions with $N = 16$ rollouts each. The lower tail, low-likelihood token rate, median, final quarter, and likelihood IQR describe distinct parts of the confidence profile.

Confidence develops differently on seen prompts. Figure 5 compares the lower tail, frequency of unlikely tokens, typical likelihood, likelihood near completion, and the spread of the central half of token likelihoods. These measurements distinguish increased confidence from a reduction in uncertain decisions or dispersion. At the final checkpoint, the low-likelihood token rate is 0.00212 on seen prompts versus 0.00595 on unseen prompts. Raw lower-tail likelihood is -0.204 versus -0.358. Their different summaries of the same likelihood profile explain why retaining both a level and an uncertainty rate can be useful; they need not rank every question identically.

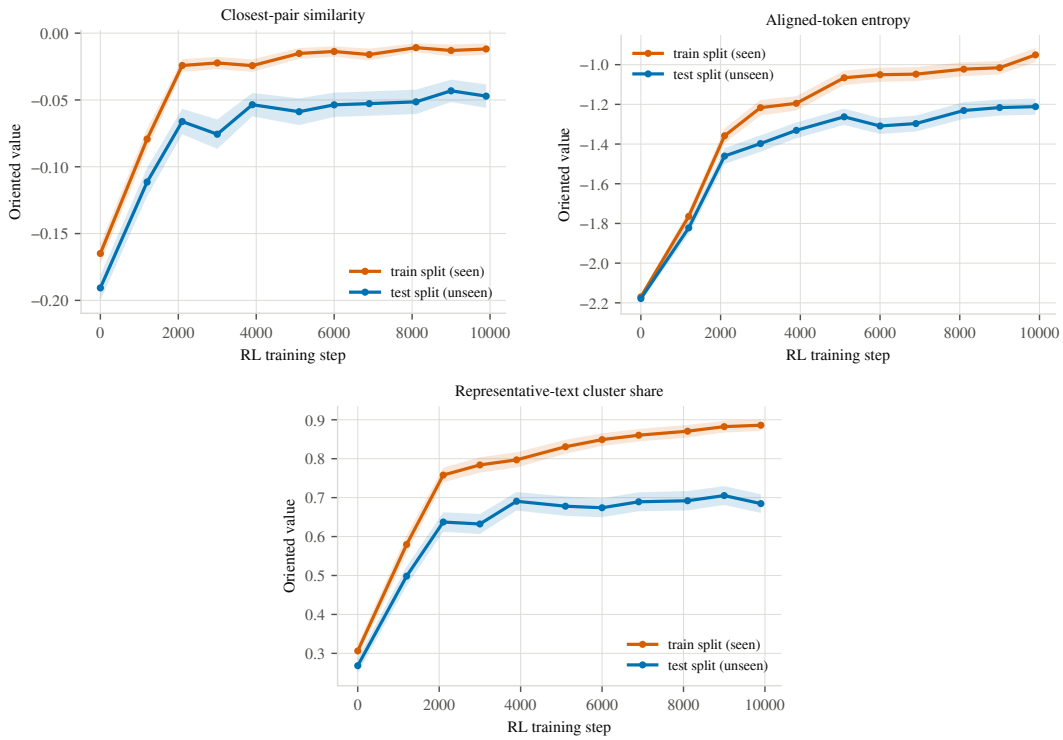


Figure 6: **Trajectory collapse throughout GSM8K training.** The three panels use the C statistics in Table 5; means and bands follow Figure 5. Closest-pair similarity measures the nearest repetition, Aligned-token entropy measures branching at token positions, and Representative-text cluster share measures the fraction within distance 0.30 of the medoid. Together they distinguish one repeated path from broader concentration of the rollout group.

Concentration extends beyond one repeated solution. Figure 6 asks whether trajectories contain a close pair, whether token choices branch at aligned positions, and whether many rollouts lie close to the group's representative text. At step 9900, closest-pair distance is 0.012 on seen prompts and 0.047 on unseen prompts; the representative-text cluster share is 0.886 versus 0.685. The latter uses the same radius 0.30 as Table 5. A close pair alone can reflect an isolated repetition, whereas cluster share measures the extent of concentration across the group. Aligned-token entropy supplies a further view of branching based on token identities rather than model probabilities.

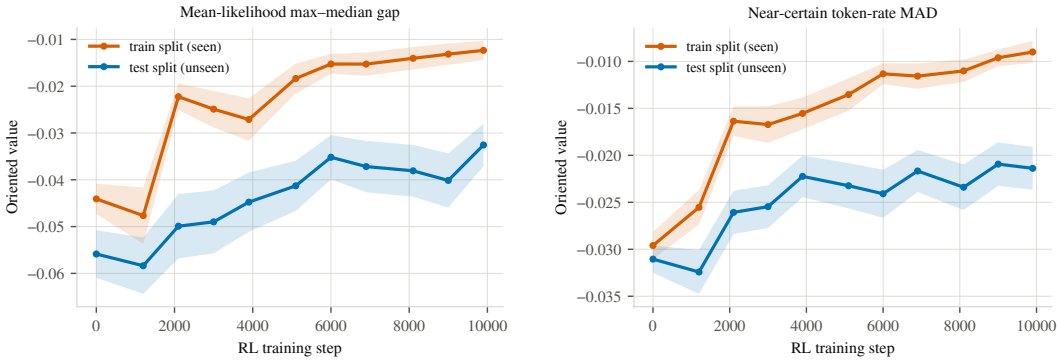


Figure 7: **Across-rollout consistency throughout GSM8K training.** The two panels use the R statistics in Table 5; means and bands follow Figure 5. Mean-likelihood max–median gap compares the most confident and typical rollout, while Near-certain token-rate MAD measures dispersion around the median near-certain token rate. Both retain the negative orientation specified in the feature table, so higher values indicate more consistent generations.

Confidence also becomes more consistent across rollouts. Figure 7 measures the gap between the most confident and median rollout and the dispersion of near-certain token rates. The mean-likelihood max–median gap falls from 0.044 to 0.012 on seen prompts, versus 0.056 to 0.033 on unseen prompts. The near-certain token-rate MAD ends at 0.009 versus 0.021. Both statistics use variation across generations, rather than the absolute confidence within a single rollout, motivating R as a separate feature family.

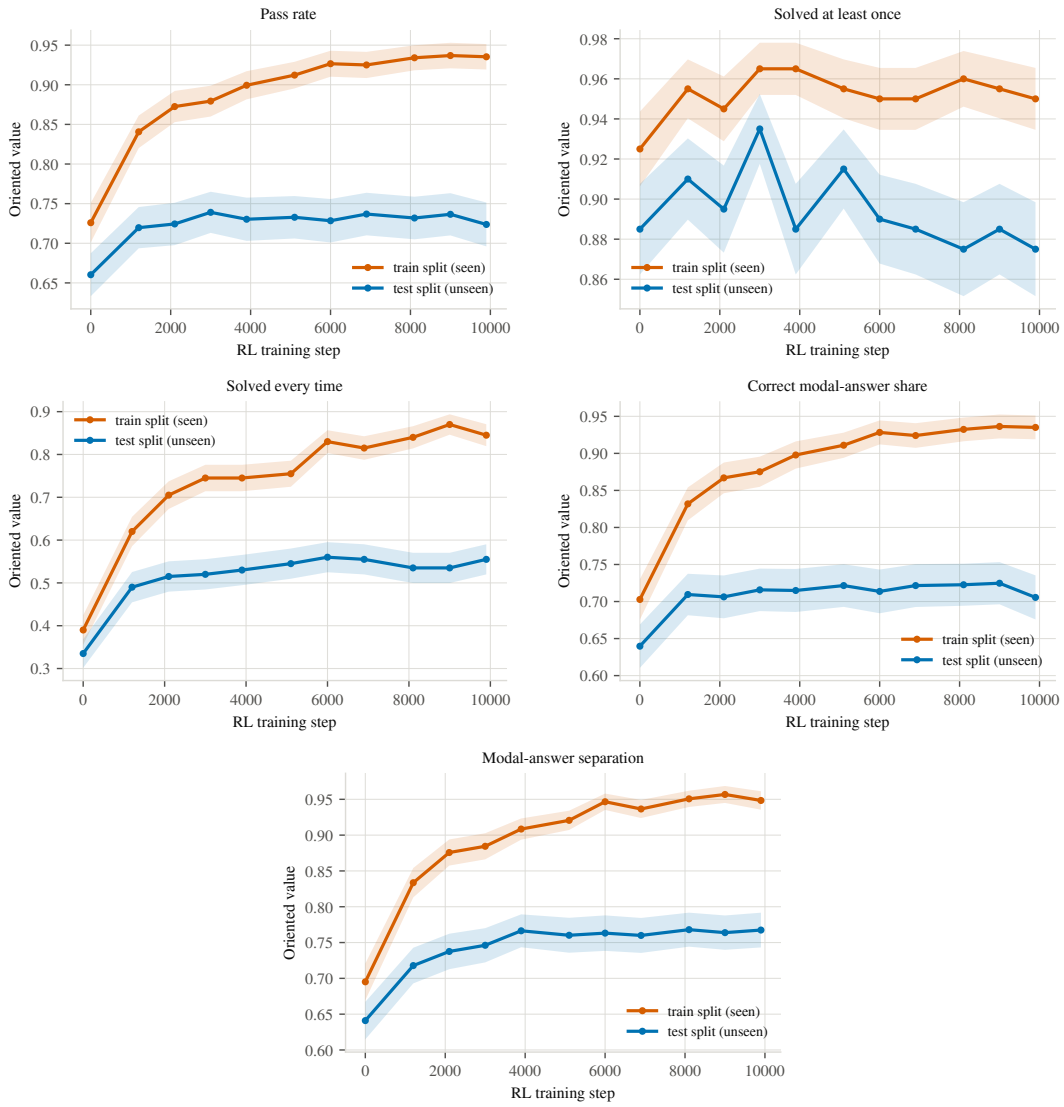


Figure 8: **Answer saturation throughout GSM8K training.** The five panels use the A statistics in Table 5; means and bands follow Figure 5. Average success, any success, complete success, correct modal-answer share, and modal-answer separation capture different aspects of the final-answer distribution. These measurements distinguish occasional correct solutions from concentration on a correct answer, and complement the path-based features in Figure 6.

Reliable answers complement similar solution paths. In Figure 8, pass rate rises from 0.726 to 0.935 on seen prompts, compared with 0.660 to 0.724 on unseen prompts. Solved at least once and Solved every time distinguish occasional success from complete correctness at the fixed rollout budget. Correct modal-answer share measures the mass converged on a correct answer, while Modal-answer separation measures how far the most frequent answer leads its runner-up. The final correct modal-answer share is 0.935 versus 0.706. Different paths can reach the same answer, and a dominant answer can be wrong; this motivates keeping both outcome and path-based measurements.

B AGGREGATION DETAILS

This section provides the full construction underlying Sections 4.1.1 to 4.1.3.

Balanced axes. The inner standardization in Section 4.1.1 places features within an axis on a common scale, while the outer standardization places the resulting axes on a common scale before aggregation. Averaging within each axis first is important because the four signatures contain different numbers of features; otherwise, an axis with more measurements would receive greater weight by construction.

Agreement weight. If several axes capture different manifestations of the same membership signal, an informative axis should tend to agree with the consensus of the others. For axis a, define the leave-one-out consensus $\bar{\mathbf{A}}^{(-a)} = (K-1)^{-1} \sum_{a' \neq a} \mathbf{A}_{\cdot a'}$. We set

$$w_a^{\mathrm{agr}} = \max\Big(0,\, \mathrm{corr}\Big(\mathbf{A}_{\cdot a}, \bar{\mathbf{A}}^{(-a)}\Big)\Big). \tag{5}$$

Negative correlations are clipped to zero so that an axis that moves against the consensus is ignored rather than reversed.

Mixture-separation weight. Agreement may miss an informative axis that captures a distinct signature, so we also measure whether its scores separate into two latent groups. For each axis we fit $\pi_1 \mathcal{N}(\mu_{a,1}, \sigma_{a,1}^2) + \pi_2 \mathcal{N}(\mu_{a,2}, \sigma_{a,2}^2)$ and define

$$w_a^{\mathrm{sep}} = \frac{|\mu_{a,1} - \mu_{a,2}|}{\sqrt{(\sigma_{a,1}^2 + \sigma_{a,2}^2)/2}}. \tag{6}$$

This is a label-free analogue of Cohen's d (Cohen, 2013). Multiplying w_a^{agr} and w_a^{sep} yields the seed weights.

Covariance-aware refinement. Under a shared-covariance Gaussian model, Fisher's linear discriminant uses

$$w^\star = \Sigma^{-1}(\mu_1 - \mu_0), \tag{7}$$

which discounts redundant directions and exploits joint information across axes (Fisher, 1936). Since membership labels are unavailable, we estimate the required quantities from the confident tails of $s^{(0)}$. Let Q_ρ denote its empirical ρ-quantile and define $\mathcal{U} = \{q : s_q^{(0)} \leq Q_\rho\}$ and $\mathcal{S} = \{q : s_q^{(0)} \geq Q_{1-\rho}\}$. We estimate

$$\delta = \bar{\mathbf{A}}_{\mathcal{S}} - \bar{\mathbf{A}}_{\mathcal{U}}, \tag{8}$$

$$\widehat{\Sigma} = \frac{1}{|\mathcal{U}| + |\mathcal{S}| - 1} \sum_{\mathcal{C} \in \{\mathcal{U}, \mathcal{S}\}} \sum_{q \in \mathcal{C}} \big(\mathbf{A}_{q\cdot} - \bar{\mathbf{A}}_{\mathcal{C}}\big)\big(\mathbf{A}_{q\cdot} - \bar{\mathbf{A}}_{\mathcal{C}}\big)^\top. \tag{9}$$

Here δ captures separation between the two tails, while $\widehat{\Sigma}$ captures within-group covariance. Because directly inverting $\widehat{\Sigma}$ can amplify estimation noise, we shrink it toward its diagonal following Ledoit & Wolf (2004), yielding the final refinement in Equation (4). We use $\lambda = 0.1$ and $\varepsilon = 10^{-6}$ and apply the refinement once.

C GRPO TRAINING HYPERPARAMETERS

Table 6 summarizes the hyperparameters used for the four controlled GRPO runs. All runs use full-parameter training with a learning rate of 1×10^{-5}, zero KL and entropy coefficients, and one inner iteration. Parameters not listed in the table use the corresponding TRL (von Werra et al., 2020) defaults.

Table 6: GRPO hyperparameters for the four controlled runs. "Effective batch" is per-device batch $\times$ gradient accumulation $\times$ 8 GPUs. Training temperature is also used for audit rollouts.

	OLMo2-1B-Instruct GSM8K	Qwen2.5-Math-7B AIME 2025	Qwen2.5-Math-7B AIME 2026	DeepSeek-Math-7B-Instruct AIME 2025
Learning rate	1×10^{-5}	1×10^{-5}	1×10^{-5}	1×10^{-5}
KL coefficient β	0	0	0	0
Entropy coefficient	0	0	0	0
Inner iterations	1	1	1	1
Group size G	16	8	8	8
Per-device batch	4	1	1	1
Grad. accumulation	8	16	16	16
Effective batch	256	128	128	128
Max completion tokens	2048	4096	4096	3072
Training temperature	0.6	1.0	1.0	1.0

D DATASET CONSTRUCTION

OLMo2-1B-Instruct on GSM8K. For training, we use the GSM8K training split (Cobbe et al., 2021). Since OLMo is trained through an open pipeline, we can verify that its pretraining and instruction-tuning data do not include the GSM8K test split. For the membership audit, we sample 200 seen questions from the training split and 200 unseen questions from the official test split.

Qwen2.5-Math-7B on AIME 2025, Qwen2.5-Math-7B on AIME 2026, and DeepSeek-Math-7B-Instruct on AIME 2025. For each AIME experiment, we split the 30 competition problems evenly into training and held-out sets. AIME 2025 is used for Qwen2.5-Math-7B on AIME 2025 and DeepSeek-Math-7B-Instruct on AIME 2025, while Qwen2.5-Math-7B on AIME 2026 uses AIME 2026. Because 15 competition problems are too few for RL training, we augment the training split with 30 000 NuminaMath (Li et al., 2024) problems. For AIME 2025, we additionally exclude four competition problems that have highly similar counterparts in NuminaMath, to reduce the risk of pre-existing overlap and ensure a cleaner evaluation of RL-stage contamination.

Public checkpoints. For JustRL-DeepSeek-1.5B on DAPO-Math, DAPO-Qwen-32B on DAPO-Math, Qwen-2.5-32B-SimpleRL-Zoo on the released math mixture, and Open-Reasoner-Zero-7B on the released math mixture, we can't control the training process. We therefore derive membership labels from the released RL prompt sets like Zhang et al. (2026). Each audit pool contains 300 seen and 300 unseen prompts.

E PROMPT-REWRITING PROCEDURE

We use Gemini 3.5 Flash (Google DeepMind, 2026) to construct controlled variants of GSM8K prompts at increasing levels of semantic-preserving transformation. For all paraphrase conditions, the model is instructed to preserve the mathematical problem exactly: every numerical value must remain unchanged, named entities and objects must be preserved, the question being asked must remain the same, and the correct answer must not change. The model is further instructed not to add or remove information, solve the problem, or provide any explanation beyond the rewritten prompt.

For *light* paraphrasing, we ask the model to make primarily lexical substitutions, replacing a limited fraction of content words with synonyms while preserving sentence order, clause structure, and grammatical voice. For *medium* paraphrasing, we additionally permit clauses to be reordered within sentences and allow changes between active and passive voice, while keeping the same number of sentences. For *heavy* paraphrasing, we ask for a substantially different verbal realization of the same problem: sentences may be merged or split, the passage may be reorganized, and the register may be changed, while all numerical values, entities, mathematical relations, and the answer remain fixed. The one- and two-word conditions are more conservative versions in which only a minimal number of lexical changes are introduced.

We also construct *cover-story* rewrites designed to remove the original surface scenario while preserving its underlying computation. To do so, we first ask the model to extract the problem into a structured representation containing its numerical parameters, their semantic roles, the relevant entities, and an executable computation that reproduces the original answer from those parameters alone. We then provide this representation to a second rewrite step and ask the model to instantiate the same computation in an entirely different domain, with new people, objects, activities, and narrative context. In the *fixed-number* condition, all original numerical values are retained and assigned the same computational roles in the new story. In the *resampled-number* condition, the original values are replaced by a new set of values while the computational structure is kept unchanged; the model is explicitly instructed not to reuse the original numbers and to ensure that the new quantities remain plausible for the rewritten setting.

For every transformation condition, we sample three independent rewrites to introduce variation in the surface realization. Outputs that fail to preserve the intended mathematical semantics or otherwise violate the transformation constraints are excluded from evaluation.

F Other Ablation and Robustness Analysis

F.1 Component ablations

Table 7 isolates the contribution of the two aggregation components in SCAR-Cal: label-free reliability weighting and covariance-aware refinement. Removing either component reduces overall performance, and removing both causes a further drop. Importantly, even the simplest variant without weighting or refinement still outperforms the strongest baseline, showing that much of the gain already comes from combining the four complementary RL signatures. The additional weighting and refinement steps then provide further improvements, with the full SCAR-Cal achieving the strongest overall performance.

Table 7: Component ablations of SCAR-Cal. The baseline with the highest mean AUC (Min-kNN) is shown for reference.

Variant	JustRL	DAPO	SimpleRL	ORZ	OLMo/GSM8K	Qwen/AIME25	Qwen/AIME26	DS-Math/AIME25	Mean
SCAR-Cal (full)	.711	.809	.743	.830	.691	.687	.787	.851	**.764**
SCAR-Agreement × Bimodality (− refinement)	.714	.805	.747	.822	.681	.672	.720	.769	.741
SCAR-Uniform+Refine (− weights)	.714	.805	.748	.830	.691	.672	.791	.682	.742
SCAR-Uniform (− both)	.713	.793	.748	.815	.681	.656	.751	.738	.737
Best baseline (Min-kNN)	.633	.738	.741	.817	.637	.574	.724	.523	.673

F.2 Hyperparameter Sensitivity

SCAR-Cal uses the same aggregation hyperparameters, $(\rho, \lambda) = (0.30, 0.1)$, across all pipelines. We evaluate whether performance depends on this choice using two additional protocols: (i) leave-one-pipeline-out (LOPO) selection, where hyperparameters are chosen on the other seven pipelines, and (ii) per-benchmark oracle selection, where the best configuration is selected directly on the target pipeline. The oracle is not a deployable setting and only measures the available tuning headroom.

Table 8: Sensitivity to aggregation hyperparameters.

Pipeline	Fixed	LOPO	Oracle
JustRL / DAPO-Math	.711	.7106	.7118
DAPO / DAPO-Math	.809	.8094	.8102
SimpleRL / MathMix	.743	.7429	.7461
ORZ / MathMix	.830	.8295	.8303
OLMo / GSM8K	.691	.6923	.6927
Qwen / AIME-2025	.687	.6923	.7026
Qwen / AIME-2026	.787	.7867	.7911
DeepSeek / AIME-2025	.851	.6359	.8513
Mean	**.764**	**.737**	**.767**

Oracle tuning improves mean AUC only from 0.7636 to 0.7670, showing that SCAR-Cal is not sensitive to benchmark-specific hyperparameter optimization. LOPO selection achieves 0.7374, indicating that hyperparameters optimized on other RL pipelines do not transfer better than the fixed configuration. This is consistent with the substantial variation in informative membership signatures across pipelines. We therefore use one fixed configuration throughout and let SCAR-Cal adapt to each unlabeled audit pool.

F.3 Robustness to Pool Size and Contamination Prevalence

A practical concern for any transductive audit is that real-world benchmarks cannot guarantee balanced class distributions, nor does the auditor know contamination prevalence beforehand. Evaluating sensitivity across diverse sample regimes is therefore essential. Figure 9 analyzes detection performance across varying member fractions (15%–85%) and pool sizes (n) on both OLMo/GSM8K (GSM8K) and DS-Math/AIME25 (AIME 2025).

Across both benchmarks, SCAR-Cal demonstrates strong prevalence invariance, retaining its performance edge across extreme class imbalances (15% to 85% contamination ratios) without requiring prior knowledge of the base rate. Moreover, while SCAR-Cal is competitive even in minimal data regimes (e.g., n=8 on AIME and n=25 on GSM8K), its detection margin steadily widens as n scales, since larger pools yield more reliable covariance estimates for feature weighting and refinement. Thus, SCAR-Cal operates effectively across arbitrary pool configurations, remaining robust to severe imbalance while scaling efficiently with audit size.

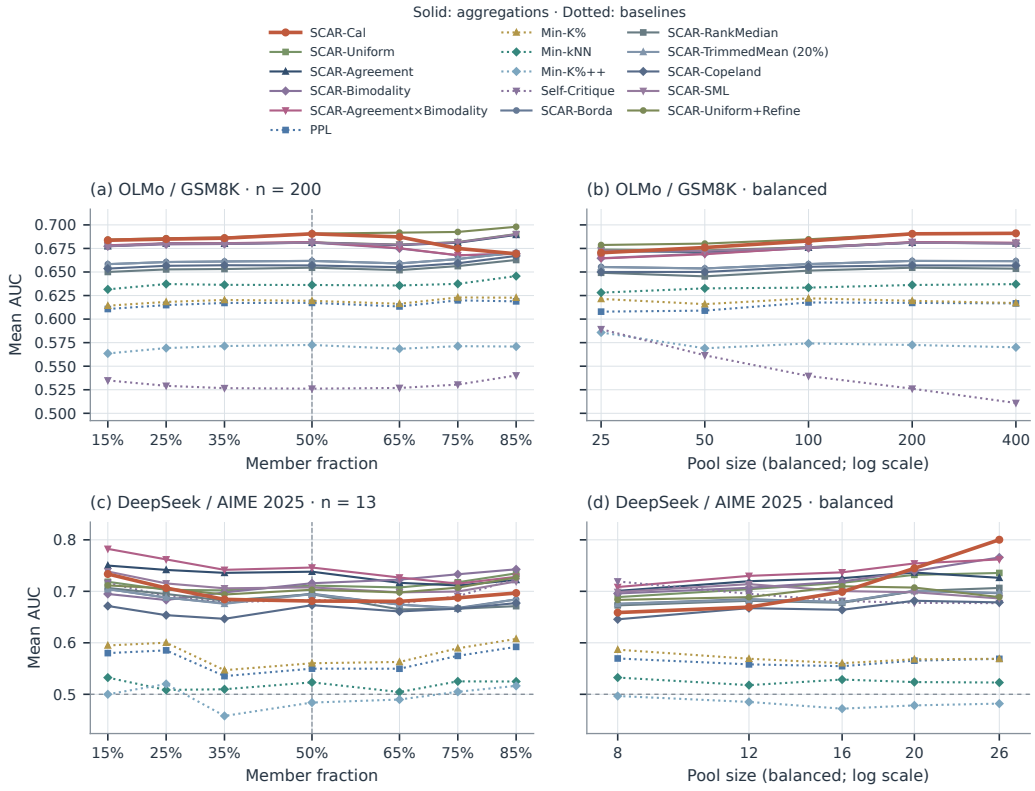


Figure 9: Robustness to contamination prevalence (left) and pool size under balanced splits (right) across OLMo/GSM8K (top) and DS-Math/AIME25 (bottom). SCAR-Cal consistently outperforms single-metric baselines across all member fractions (15%–85%) and scales monotonically with larger pool sizes. Curves show means over 150 sampled pools per setting; all aggregation variants and per-question baselines share each panel.

F.4 Impact of the Rollout Budget (N)

The primary operational parameter under the auditor's control is the number of sampled rollouts, N. Table 9 evaluates detection performance as N scales from 1 to 16. SCAR-Cal improves overall from 0.6739 at $N{=}1$ to 0.7636 at $N{=}16$, consistently outperforming the best baseline at every budget. The scaling trend directly supports our core formulation. At $N{=}1$, cross-rollout statistics, specifically collapse (C) and rollout sharpening (R), are undefined or constant, restricting informative inputs to within-self-sharpening (S) and Answer signals (A). The most substantial performance jump occurs between $N{=}8$ and $N{=}16$, as additional samples stabilize estimates of trajectory agreement and inter-rollout dispersion. This underscores that multi-rollout fingerprints provide crucial signal that single-sample detectors fundamentally cannot capture.

Table 9: Mean ROC-AUC across pipelines as a function of the rollout budget N. SCAR-Cal carries ± 1 bootstrap standard deviation over 600 paired resamples of the audited questions. The best-baseline column is a post-hoc maximum over a candidate set rather than a single estimator, so no interval is quoted for it.

N	SCAR-Cal	Best Baseline
1	$0.6739 \pm .025$	0.6574
2	$0.6952 \pm .028$	0.6574
4	$0.6902 \pm .027$	0.6574
8	$0.7225 \pm .026$	0.6631
16	$\mathbf{0.7636} \pm .021$	0.6735

F.5 Per-Pipeline Aggregation and Feature-Addition Results

Table 10 gives the values underlying the aggregation comparison in Figure 3a, separately for every model–dataset pipeline. Tables 11 to 14 give the per-pipeline values for every feature count in Figure 3b. All results use the same fixed feature definitions and aggregation parameters as those figures. Bootstrap uncertainty for aggregation rules and variation across random feature orders are reported separately in the table captions.

Table 10: Per-pipeline ROC-AUC for the aggregation rules in Figure 3a. Each cell shows the full-pool estimate above its ± 1 bootstrap SD (4,000 paired question resamples). Bold marks the largest estimate in each column. The mean and its SD use the same eight-pipeline averaging as the plot; Min-kNN is the fixed baseline shown there.

Aggregation	JustRL	DAPO	SimpleRL	ORZ	OLMo GSM8K	Qwen AIME25	Qwen AIME26	DS-Math AIME25	Mean
SCAR-Cal	.7107 ±.021	**.8094** ±.018	.7429 ±.020	.8296 ±.017	.6911 ±.027	**.6872** ±.106	.7867 ±.087	**.8513** ±.075	**.7636** ±.021
SCAR-Bimodality	.7136 ±.021	.8069 ±.018	.7475 ±.020	.8242 ±.017	.6803 ±.027	.6564 ±.111	.7422 ±.092	.7846 ±.088	.7445 ±.023
SCAR-Uniform+Refine	.7144 ±.021	.8048 ±.018	.7478 ±.020	**.8301** ±.017	**.6911** ±.027	.6718 ±.110	**.7911** ±.086	.6821 ±.105	.7416 ±.026
SCAR-Agreement×Bimodality	.7137 ±.021	.8053 ±.018	.7469 ±.020	.8216 ±.018	.6811 ±.027	.6718 ±.109	.7200 ±.096	.7692 ±.092	.7412 ±.023
SCAR-Uniform	.7128 ±.021	.7934 ±.018	**.7480** ±.020	.8150 ±.018	.6807 ±.027	.6564 ±.110	.7511 ±.090	.7385 ±.097	.7370 ±.024
SCAR-Borda	.7193 ±.021	.7971 ±.018	.7293 ±.021	.8165 ±.018	.6615 ±.028	.6564 ±.113	.7800 ±.088	.6923 ±.104	.7315 ±.024
SCAR-Agreement	.7155 ±.021	.7910 ±.018	.7464 ±.020	.8119 ±.018	.6804 ±.027	.6513 ±.111	.7378 ±.093	.7179 ±.098	.7315 ±.022
SCAR-TrimmedMean (20%)	.7192 ±.021	.7971 ±.018	.7293 ±.021	.8165 ±.018	.6615 ±.028	.6564 ±.113	.7778 ±.088	.6923 ±.104	.7313 ±.024
SCAR-SML	.7156 ±.021	.7923 ±.018	.7475 ±.020	.8139 ±.018	.6804 ±.027	.6564 ±.110	.7467 ±.091	.6821 ±.104	.7294 ±.024
SCAR-RankMedian	**.7220** ±.021	.7848 ±.019	.7226 ±.021	.8117 ±.018	.6535 ±.028	.6564 ±.109	.7600 ±.092	.7026 ±.103	.7267 ±.025
SCAR-Copeland	.7052 ±.021	.7915 ±.018	.7264 ±.021	.8120 ±.018	.6568 ±.028	.6538 ±.108	.7622 ±.091	.6795 ±.103	.7234 ±.025
Best baseline (Min-kNN)	.6333 ±.023	.7383 ±.020	.7405 ±.020	.8166 ±.018	.6371 ±.028	.5744 ±.115	.7244 ±.094	.5231 ±.118	.6735 ±.028

Table 11: SCAR-Cal: per-pipeline feature-addition values underlying Figure 3b. Each cell shows mean AUC above its ± 1 SD across 30 random feature orders. This SD measures feature-order variation, not bootstrap uncertainty. The final column gives the plotted mean across eight pipelines and its across-order SD. The constant Min-kNN reference is given in Table 10.

Features	JustRL	DAPO	SimpleRL	ORZ	OLMo GSM8K	Qwen AIME25	Qwen AIME26	DS-Math AIME25	Mean
1	.6050 ±.083	.6896 ±.081	.6625 ±.052	.7207 ±.056	.6226 ±.038	.6048 ±.066	.6511 ±.073	.5887 ±.098	.6431 ±.032
2	.6614 ±.056	.7465 ±.028	.6983 ±.040	.7629 ±.042	.6468 ±.026	.6041 ±.067	.6696 ±.060	.6232 ±.087	.6766 ±.016
3	.6775 ±.038	.7626 ±.025	.7218 ±.022	.7907 ±.035	.6535 ±.022	.6058 ±.058	.6786 ±.064	.6286 ±.093	.6899 ±.019
4	.6868 ±.018	.7729 ±.023	.7213 ±.020	.7983 ±.025	.6605 ±.018	.5956 ±.052	.6855 ±.063	.6383 ±.094	.6949 ±.018
5	.6896 ±.015	.7773 ±.020	.7268 ±.019	.8053 ±.026	.6643 ±.018	.6115 ±.045	.6984 ±.067	.6303 ±.105	.7004 ±.017
6	.6955 ±.009	.7824 ±.017	.7273 ±.019	.8060 ±.030	.6694 ±.017	.6251 ±.040	.7181 ±.053	.6537 ±.091	.7097 ±.015
7	.6969 ±.008	.7854 ±.016	.7304 ±.015	.8098 ±.024	.6732 ±.020	.6373 ±.037	.7293 ±.041	.6603 ±.102	.7153 ±.015
8	.7003 ±.006	.7909 ±.015	.7350 ±.015	.8153 ±.021	.6787 ±.017	.6426 ±.036	.7455 ±.031	.6814 ±.087	.7237 ±.013
9	.7020 ±.006	.7927 ±.015	.7368 ±.014	.8156 ±.020	.6839 ±.009	.6528 ±.033	.7523 ±.028	.7075 ±.069	.7305 ±.010
10	.7032 ±.005	.7958 ±.013	.7394 ±.011	.8186 ±.015	.6849 ±.009	.6526 ±.033	.7535 ±.026	.7282 ±.067	.7345 ±.010
11	.7041 ±.004	.7991 ±.012	.7421 ±.009	.8241 ±.010	.6860 ±.007	.6511 ±.031	.7567 ±.022	.7480 ±.070	.7389 ±.011
12	.7062 ±.004	.8015 ±.009	.7428 ±.009	.8264 ±.007	.6865 ±.006	.6591 ±.029	.7661 ±.019	.7595 ±.073	.7435 ±.009
13	.7081 ±.003	.8033 ±.007	.7434 ±.008	.8272 ±.006	.6880 ±.003	.6663 ±.027	.7754 ±.016	.7768 ±.069	.7486 ±.010
14	.7090 ±.003	.8073 ±.005	.7430 ±.005	.8283 ±.004	.6897 ±.003	.6682 ±.022	.7759 ±.013	.8130 ±.054	.7543 ±.007
15	.7107 ±.000	.8094 ±.000	.7429 ±.000	.8296 ±.000	.6911 ±.000	.6872 ±.000	.7867 ±.000	.8513 ±.000	.7636 ±.000

Table 12: SCAR-Agreement×Bimodality: per-pipeline feature-addition values underlying Figure 3b. Each cell shows mean AUC above its ±1 SD across 30 random feature orders. This SD measures feature-order variation, not bootstrap uncertainty. The final column gives the plotted mean across eight pipelines and its across-order SD. The constant Min-kNN reference is given in Table 10.

Features	JustRL	DAPO	SimpleRL	ORZ	OLMo GSM8K	Qwen AIME25	Qwen AIME26	DS-Math AIME25	Mean
1	.6050 ±.083	.6896 ±.081	.6625 ±.052	.7207 ±.056	.6226 ±.038	.6048 ±.066	.6511 ±.073	.5887 ±.098	.6431 ±.032
2	.6613 ±.056	.7471 ±.028	.6987 ±.045	.7612 ±.043	.6465 ±.023	.6082 ±.066	.6606 ±.059	.6232 ±.083	.6759 ±.015
3	.6775 ±.038	.7635 ±.026	.7244 ±.024	.7895 ±.037	.6544 ±.021	.6166 ±.053	.6706 ±.067	.6240 ±.079	.6900 ±.015
4	.6881 ±.018	.7751 ±.022	.7266 ±.018	.8000 ±.026	.6625 ±.017	.6080 ±.048	.6772 ±.062	.6315 ±.093	.6961 ±.015
5	.6910 ±.016	.7804 ±.018	.7312 ±.017	.8072 ±.022	.6650 ±.015	.6210 ±.043	.6935 ±.068	.6439 ±.094	.7042 ±.013
6	.6967 ±.010	.7846 ±.016	.7315 ±.016	.8075 ±.026	.6685 ±.013	.6304 ±.036	.6967 ±.058	.6672 ±.082	.7104 ±.011
7	.6983 ±.010	.7881 ±.015	.7353 ±.015	.8106 ±.024	.6724 ±.013	.6381 ±.032	.7030 ±.047	.6832 ±.079	.7161 ±.012
8	.7014 ±.007	.7916 ±.015	.7404 ±.014	.8147 ±.021	.6769 ±.009	.6426 ±.027	.7068 ±.040	.6962 ±.066	.7213 ±.010
9	.7030 ±.008	.7928 ±.015	.7412 ±.013	.8135 ±.019	.6780 ±.007	.6487 ±.026	.7172 ±.036	.7122 ±.044	.7258 ±.008
10	.7046 ±.007	.7951 ±.014	.7412 ±.012	.8134 ±.017	.6793 ±.006	.6491 ±.026	.7197 ±.036	.7325 ±.032	.7293 ±.007
11	.7056 ±.006	.7976 ±.012	.7452 ±.008	.8182 ±.012	.6803 ±.005	.6509 ±.024	.7215 ±.029	.7395 ±.033	.7324 ±.006
12	.7080 ±.005	.7991 ±.010	.7458 ±.006	.8187 ±.008	.6803 ±.004	.6533 ±.021	.7204 ±.020	.7436 ±.032	.7337 ±.004
13	.7106 ±.004	.7997 ±.009	.7460 ±.005	.8190 ±.007	.6810 ±.002	.6590 ±.021	.7237 ±.016	.7506 ±.026	.7362 ±.004
14	.7118 ±.004	.8037 ±.006	.7460 ±.004	.8196 ±.005	.6811 ±.002	.6668 ±.016	.7199 ±.011	.7634 ±.018	.7390 ±.004
15	.7137 ±.000	.8053 ±.000	.7469 ±.000	.8216 ±.000	.6811 ±.000	.6718 ±.000	.7200 ±.000	.7692 ±.000	.7412 ±.000

Table 13: SCAR-Agreement: per-pipeline feature-addition values underlying Figure 3b. Each cell shows mean AUC above its ± 1 SD across 30 random feature orders. This SD measures feature-order variation, not bootstrap uncertainty. The final column gives the plotted mean across eight pipelines and its across-order SD. The constant Min-kNN reference is given in Table 10.

Features	JustRL	DAPO	SimpleRL	ORZ	OLMo GSM8K	Qwen AIME25	Qwen AIME26	DS-Math AIME25	Mean
1	.6050 ±.083	.6896 ±.081	.6625 ±.052	.7207 ±.056	.6226 ±.038	.6048 ±.066	.6511 ±.073	.5887 ±.098	.6431 ±.032
2	.6593 ±.055	.7491 ±.027	.7017 ±.041	.7636 ±.039	.6469 ±.022	.6123 ±.062	.6690 ±.057	.6272 ±.080	.6786 ±.015
3	.6746 ±.038	.7651 ±.023	.7263 ±.024	.7905 ±.032	.6546 ±.021	.6157 ±.052	.6858 ±.058	.6512 ±.079	.6955 ±.016
4	.6871 ±.017	.7742 ±.018	.7282 ±.020	.7995 ±.022	.6629 ±.017	.6072 ±.044	.6920 ±.054	.6581 ±.097	.7011 ±.016
5	.6897 ±.014	.7792 ±.015	.7317 ±.017	.8051 ±.020	.6656 ±.016	.6185 ±.036	.6963 ±.056	.6586 ±.097	.7056 ±.013
6	.6849 ±.047	.7811 ±.011	.7318 ±.016	.8024 ±.022	.6687 ±.013	.6297 ±.036	.7071 ±.046	.6672 ±.082	.7091 ±.011
7	.6871 ±.047	.7840 ±.011	.7364 ±.016	.8052 ±.022	.6722 ±.013	.6371 ±.031	.7132 ±.038	.6798 ±.080	.7144 ±.011
8	.7006 ±.010	.7854 ±.009	.7397 ±.014	.8079 ±.019	.6766 ±.009	.6410 ±.029	.7215 ±.027	.6825 ±.067	.7194 ±.009
9	.7022 ±.009	.7865 ±.008	.7406 ±.013	.8065 ±.017	.6776 ±.008	.6462 ±.023	.7261 ±.027	.6926 ±.049	.7223 ±.008
10	.7036 ±.008	.7869 ±.007	.7402 ±.011	.8052 ±.014	.6788 ±.006	.6494 ±.021	.7233 ±.025	.7082 ±.036	.7244 ±.007
11	.7048 ±.007	.7885 ±.006	.7438 ±.006	.8091 ±.010	.6800 ±.005	.6532 ±.021	.7250 ±.023	.7152 ±.036	.7275 ±.007
12	.7078 ±.007	.7896 ±.005	.7442 ±.005	.8097 ±.007	.6799 ±.004	.6530 ±.015	.7273 ±.014	.7232 ±.030	.7294 ±.005
13	.7110 ±.006	.7896 ±.004	.7449 ±.004	.8102 ±.005	.6805 ±.002	.6540 ±.016	.7268 ±.011	.7241 ±.028	.7302 ±.004
14	.7128 ±.005	.7908 ±.003	.7453 ±.003	.8103 ±.004	.6803 ±.002	.6533 ±.013	.7295 ±.013	.7270 ±.021	.7312 ±.003
15	.7155 ±.000	.7910 ±.000	.7464 ±.000	.8119 ±.000	.6804 ±.000	.6513 ±.000	.7378 ±.000	.7179 ±.000	.7315 ±.000

Table 14: SCAR-Uniform: per-pipeline feature-addition values underlying Figure 3b. Each cell shows mean AUC above its ± 1 SD across 30 random feature orders. This SD measures feature-order variation, not bootstrap uncertainty. The final column gives the plotted mean across eight pipelines and its across-order SD. The constant Min-kNN reference is given in Table 10.

Features	JustRL	DAPO	SimpleRL	ORZ	OLMo GSM8K	Qwen AIME25	Qwen AIME26	DS-Math AIME25	Mean
1	.6050 ±.083	.6896 ±.081	.6625 ±.052	.7207 ±.056	.6226 ±.038	.6048 ±.066	.6511 ±.073	.5887 ±.098	.6431 ±.032
2	.6593 ±.055	.7491 ±.027	.7017 ±.041	.7636 ±.039	.6469 ±.022	.6123 ±.062	.6690 ±.057	.6272 ±.080	.6786 ±.015
3	.6744 ±.038	.7651 ±.023	.7270 ±.024	.7905 ±.032	.6546 ±.021	.6150 ±.051	.6863 ±.057	.6498 ±.078	.6953 ±.016
4	.6856 ±.017	.7755 ±.018	.7338 ±.016	.7994 ±.022	.6629 ±.016	.6068 ±.047	.6984 ±.052	.6583 ±.082	.7026 ±.015
5	.6889 ±.015	.7800 ±.015	.7379 ±.013	.8047 ±.019	.6657 ±.015	.6152 ±.041	.7061 ±.052	.6798 ±.074	.7098 ±.013
6	.6954 ±.011	.7827 ±.010	.7366 ±.012	.8043 ±.020	.6693 ±.013	.6238 ±.034	.7150 ±.047	.6935 ±.083	.7151 ±.012
7	.6981 ±.011	.7853 ±.010	.7409 ±.013	.8065 ±.020	.6728 ±.013	.6328 ±.030	.7234 ±.041	.7077 ±.071	.7209 ±.012
8	.7010 ±.009	.7874 ±.008	.7433 ±.013	.8093 ±.019	.6771 ±.009	.6378 ±.026	.7339 ±.030	.7168 ±.056	.7258 ±.010
9	.7028 ±.008	.7886 ±.008	.7436 ±.011	.8084 ±.016	.6780 ±.007	.6426 ±.021	.7341 ±.027	.7159 ±.060	.7267 ±.011
10	.7036 ±.007	.7886 ±.007	.7420 ±.010	.8072 ±.014	.6793 ±.006	.6462 ±.019	.7390 ±.026	.7202 ±.053	.7283 ±.009
11	.7046 ±.007	.7901 ±.005	.7451 ±.006	.8107 ±.010	.6804 ±.005	.6463 ±.016	.7416 ±.020	.7256 ±.041	.7306 ±.007
12	.7071 ±.006	.7913 ±.004	.7452 ±.005	.8119 ±.007	.6803 ±.004	.6491 ±.015	.7439 ±.016	.7258 ±.033	.7318 ±.005
13	.7093 ±.005	.7919 ±.004	.7462 ±.004	.8129 ±.006	.6807 ±.002	.6530 ±.012	.7486 ±.007	.7294 ±.025	.7340 ±.005
14	.7110 ±.004	.7931 ±.002	.7466 ±.003	.8133 ±.004	.6807 ±.002	.6538 ±.010	.7511 ±.006	.7325 ±.014	.7353 ±.003
15	.7128 ±.000	.7934 ±.000	.7480 ±.000	.8150 ±.000	.6807 ±.000	.6564 ±.000	.7511 ±.000	.7385 ±.000	.7370 ±.000

F.6 THE AGGREGATE MEMBERSHIP SIGNAL EMERGES AFTER RL

We additionally decompose the pre-RL signal into the four individual axes. For each controlled run, we repeat the audit on the matched base policy π_{θ_0} using the same candidate pool, membership labels, rollout budget, and sampling settings as for the audited RL checkpoint.

As shown in Table 15, aggregate SCAR-Cal performance is near chance before RL and increases substantially after training. Individual axes can, however, exhibit pre-existing separation, particularly on the small AIME pools. The control should therefore not be interpreted as showing that every component of SCAR is created by RL. Rather, it shows that the aggregate membership signature exploited by SCAR-Cal is weak before RL, while the axis-level breakdown clarifies which pre-existing differences are already present.

Table 15: Base-policy control. ROC-AUC on π_{θ_0}, with the audited SCAR-Cal score shown for reference.

	Base policy					Audited
Pipeline	S	C	A	R	SCAR-Cal	SCAR-Cal
OLMo / GSM8K	.524 $\pm$.029	.546 $\pm$.029	.547 $\pm$.029	.528 $\pm$.029	.553 $\pm$.029	.691
Qwen / AIME25	.395 $\pm$.111	.610 $\pm$.113	.592 $\pm$.109	.559 $\pm$.118	.523 $\pm$.118	.687
Qwen / AIME26	.462 $\pm$.111	.658 $\pm$.104	.609 $\pm$.102	.547 $\pm$.110	.533 $\pm$.111	.787
DS-Math / AIME25	.487 $\pm$.115	.446 $\pm$.116	.362 $\pm$.088	.482 $\pm$.115	.492 $\pm$.116	.851

G ADDITIONAL RESULTS AND FIGURES

This section provides additional visualizations supporting the main results and ablations.

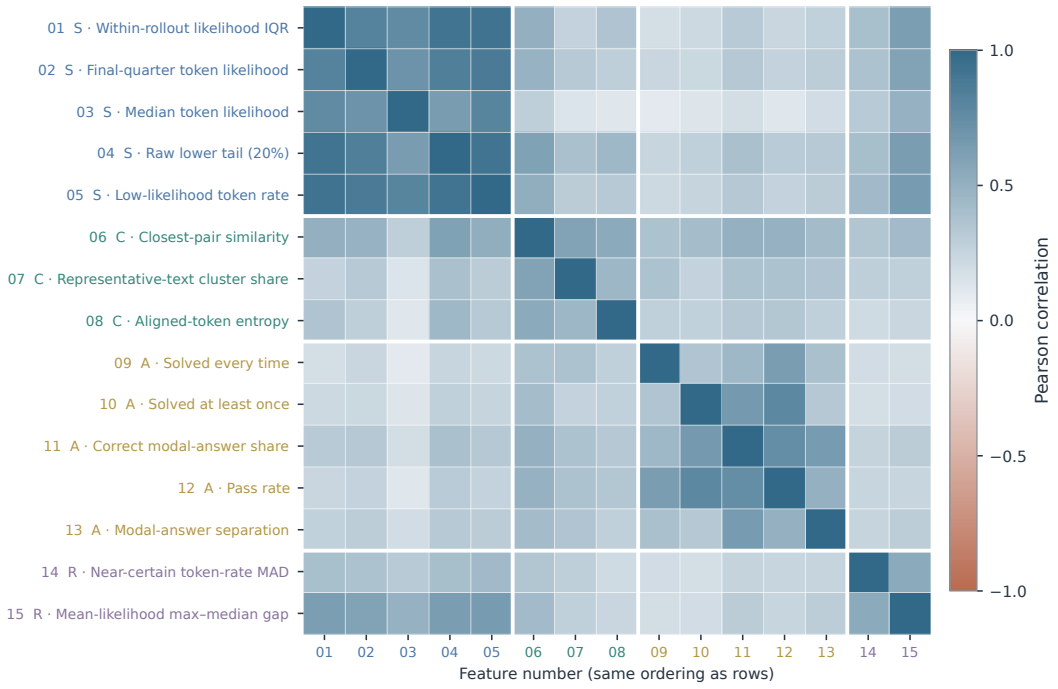


Figure 10: Feature correlation across multiple experiments

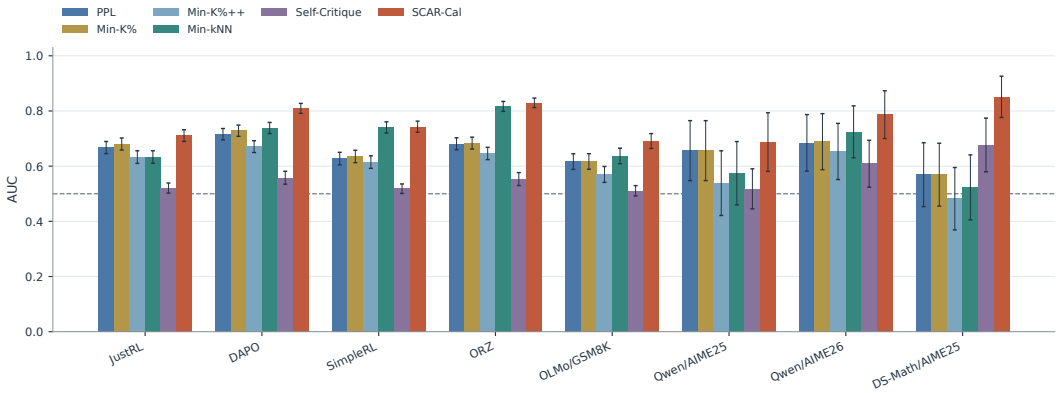
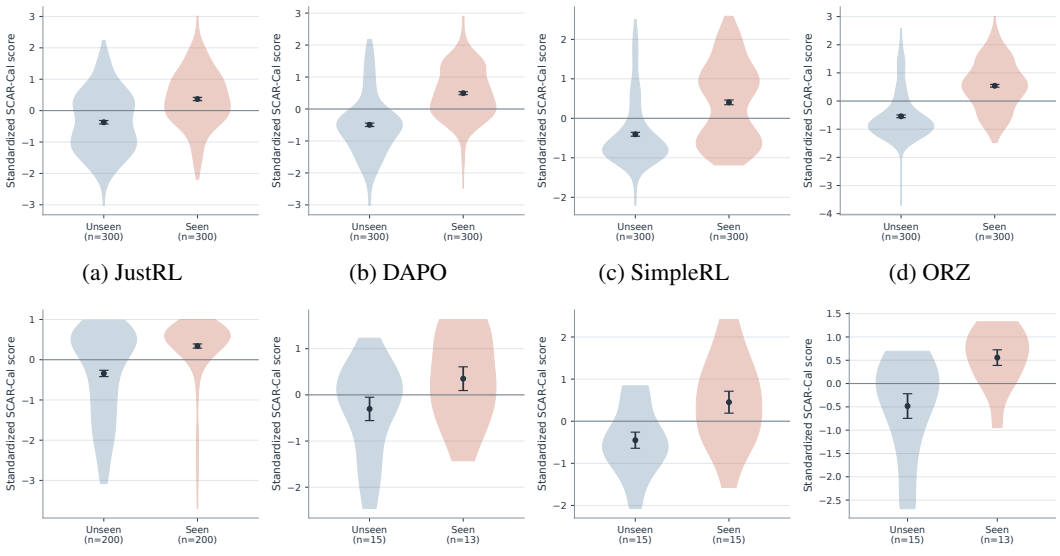


Figure 11: ROC-AUC by method across the eight model–dataset pipelines. Baseline performance varies substantially across settings, whereas SCAR-Cal is more consistent.



(a) JustRL (b) DAPO (c) SimpleRL (d) ORZ

(e) OLMo/GSM8K (f) Qwen/AIME25 (g) Qwen/AIME26 (h) DS-Math/AIME25

Figure 12: SCAR-Cal score distributions for seen and unseen prompts across all eight pipelines. Violins show the full distributions, and points indicate class means with ± 1 bootstrap standard deviation.



Figure 13: ROC-AUC of each individual feature across the eight pipelines, with the mean shown at right. Feature informativeness varies substantially across models and datasets, motivating aggregation across complementary signals.

H LIMITATIONS

SCAR is transductive: its normalization and label-free aggregation operate over a candidate pool and therefore assume that the pool contains meaningful score variation. Performance can deteriorate when the pool is extremely small or nearly single-class. SCAR also requires multiple rollouts per prompt and, in its full form, sampled-token log-probabilities and a task verifier. Finally, prompt reformulation remains an important weakness: on GSM8K, performance approaches chance under heavy paraphrasing and cover-story rewrites with resampled numbers. Our experiments focus on verifiable mathematical reasoning tasks, so it remains open whether the same four signatures transfer unchanged to other RL domains.