LaRA: Layer-wise Representation Analyses for Detecting Data Contamination in RL Post-Training

Anonymous Authors[1]

Abstract

Reinforcement learning (RL) post-training has shown to improve reasoning in large language models (LLMs). However, there has been little exploration on the problem of data contamination in RL post-training, potentially undermining generalization and evaluation reliability of the training process itself. Existing detection methods primarily rely on output-level signals such as likelihood or entropy, which become unreliable for RL-trained models since RL shapes behavior through trajectory-level rewards rather than token likelihoods. We propose LaRA, a layer-wise representation analysis framework for detecting contamination in RL post-trained LLMs. LaRA introduces three complementary metrics, measuring perturbation sensitivity, directional collapse, and local representation rigidity under controlled perturbations. We find that contamination produces progressive geometric deviations across layers, including amplified perturbation sensitivity, stronger directional collapse, and enhanced local rigidity. Based on our findings, we also develop a contamination detection protocol that aggregates representation-level deviations across layers and metrics. Experiments on RL-trained reasoning models show that our protocol outperforms existing output-level baselines for contamination detection.

1. Introduction

Reinforcement learning (RL) has shown its effectiveness in training Large Language Models (LLMs) for complex reasoning tasks (Guo et al., 2025; Guha et al., 2025; Li et al., 2025; Hochlehnert et al., 2025). However, it also raises a critical but underexplored issue of *data contamination* in RL post-training (Tao et al., 2025; Wang et al., 2025; Wu et al., 2026), the inclusion of evaluation or benchmark samples within the RL training data. Contaminated samples can induce reward-driven overfitting and implicit memorization, undermining generalization and evaluation reliability.

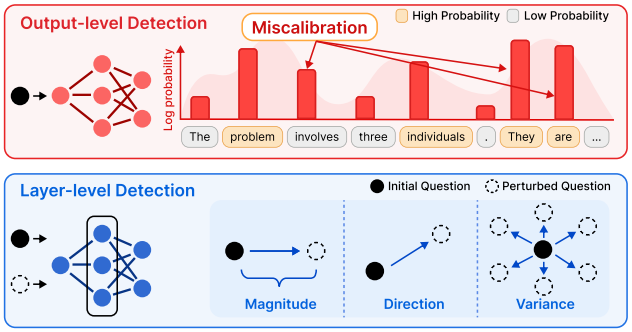

Figure 1. **Output vs. Representation-level detection.** Output-level signals are sensitive to output miscalibration, such as overconfidence of superficially plausible tokens. In contrast, representation geometry provides more robust and interpretable contamination signals.

Prior work on data contamination in LLMs has mostly focused on pre-training or supervised fine-tuning (SFT) stages (Zhang et al., 2024; Shi et al., 2023; Xie et al., 2024), where memorization is typically characterized by higher token likelihoods or lower entropy (Gonen et al., 2023). Consequently, existing approaches primarily rely on output-level signals derived from model likelihoods or generation statistics. Recent work has extended this paradigm to reasoning trajectories for detecting data contamination in RL, using entropy or behavioral divergence across generation stages as contamination signals (Tao et al., 2025).

However, such output-level statistics can be unreliable due to the poor calibration of LLM output distributions, as shown in Figure 2 (**??**). Moreover, unlike pre-training or SFT, RL optimizes expected reward over entire reasoning trajectories rather than token-wise likelihoods, making likelihood-based behavioral signals less directly aligned with the underlying training objective. These challenges motivate a shift toward representation-level analysis, where memorization can be probed directly in the model's internal geometry, bypassing the calibration issues and objective mismatch that confound output-level signals.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

We propose **LaRA**, a **La**yerwise **R**epresentation **A**nalysis framework for detecting data contamination in RL post-training. Our key hypothesis is that RL-induced memorization produces abnormal representation responses under controlled perturbations: memorized samples become overly stable to semantically equivalent variations, yet exhibit disproportionately large representation shifts when memorized information is removed. To test this, we construct structural control groups of semantically similar questions, apply consistent information masking, and analyze layer-wise representation dynamics across perturbations.

Specifically, we introduce three complementary metrics: (1) *Representation Shift Magnitude (RSM)* measures how strongly representations change when important information is removed, capturing perturbation sensitivity. (2) *Directional Collapse (DC)* measures whether representation changes collapse toward shared dominant directions, indicating reduced representational diversity. (3) *Representation Stability Index (RSI)* quantifies how invariant representations remain across semantically similar variants, capturing local rigidity under meaning-preserving perturbations. Together, these metrics characterize distinct geometric signatures of RL-induced memorization.

Across multiple RL-trained models, we empirically show that contaminated samples exhibit consistent geometric abnormalities compared to non-trained samples. In particular, contaminated samples exhibit abnormal directional collapse, higher local representational rigidity, and greater sensitivity to information removal. Furthermore, our LaRA-based contamination detection score consistently outperforms output-level baselines, suggesting that representation geometry provides a more reliable signal of RL-induced memorization.

In summary, our contributions are as follows:

- We are the first to propose a representation-level framework as well as a training and evaluation setup for detecting contamination in RL post-training via stiffness and rigidity.
- We introduce a contamination-detection protocol that consistently outperforms output-level baselines across RL-trained models, achieving up to +9.6% AUC improvement and 3.5× higher TPR@FPR=5% compared to the strongest prior output-level method.
- We provide empirical insights into how RL training affects representation geometry across layers.

2. LaRA: Layer-wise Representation Analysis to Detect RL Contamination

We frame the problem of detecting data contamination during RL post-training as Membership Inference Attack (MIA). Given an RL-trained model $\mathcal{M}$ and a candidate sample x, our goal is to determine membership $\mathcal{F}(M, x) \in \{0, 1\}$, where $\mathcal{F}(M, x) = 1$ indicates that x was a member of the training dataset and therefore indicates contamination, while 0 indicates otherwise. The central question motivating our analyses is: *do layer-wise representation signals behave differently between member and non-member samples?* To answer this, we introduce three complementary metrics.

2.1. Contamination Dataset Construction

To explore MIA in RL training setting, we construct controlled contamination benchmarks that support a two-stage analysis: (i) detecting contamination in released open-source RL checkpoints based on their known training data, and (ii) tracking how detection signals evolve under additional RL training that we perform on a controlled corpus.

Evaluation set. We construct a contamination evaluation set from the publicly open dataset of the three open-source RL-trained models (`EURUS-2-7B-PRIME` (Eurus) (Cui et al., 2025), `LIMR` (Li et al., 2025), and `Olmo-3.1-7B-RL-Zero-Math` (Olmo) (**?**)). For each model, we sample 30 Olympiad-level mathematics problems from its own RL training set as members, and pair them with 30 problems from AIME 2026 (Balunović et al., 2025) as non-members. This yields a balanced 60-sample evaluation set per model. Non-member split is shared across all three models, while the member split is model-specific.

Training set. To study how contamination signals vary during continued RL training, we re-use each model's 30 member samples as deliberate contamination targets and augment them with 970 Olympiad-level problems drawn from the RL-MIA (Tao et al., 2025) Math dataset, yielding a 1,000-sample training corpus per model. Using this data, we resume RL training on each open-source checkpoint and track how member vs. non-member signals diverge during RL post-training. Further details on datasets are provided in Appendix C.

2.2. Three Metrics for Analysis

Metric 1: Representation Shift Magnitude. To quantify how strongly a model's internal representation responds to the removal of important information, we introduce Representation Shift Magnitude (RSM). Given an original question q_0, we construct a set of semantically similar questions

$$\mathcal{Q} = \{q_0, q_1, \ldots, q_K\},$$

where K denotes the number of generated semantic neighbors excluding the original question. For each question $q_i \in \mathcal{Q}$, we apply an importance-based blanking operator BLANKIMPORTANT that removes key information spans

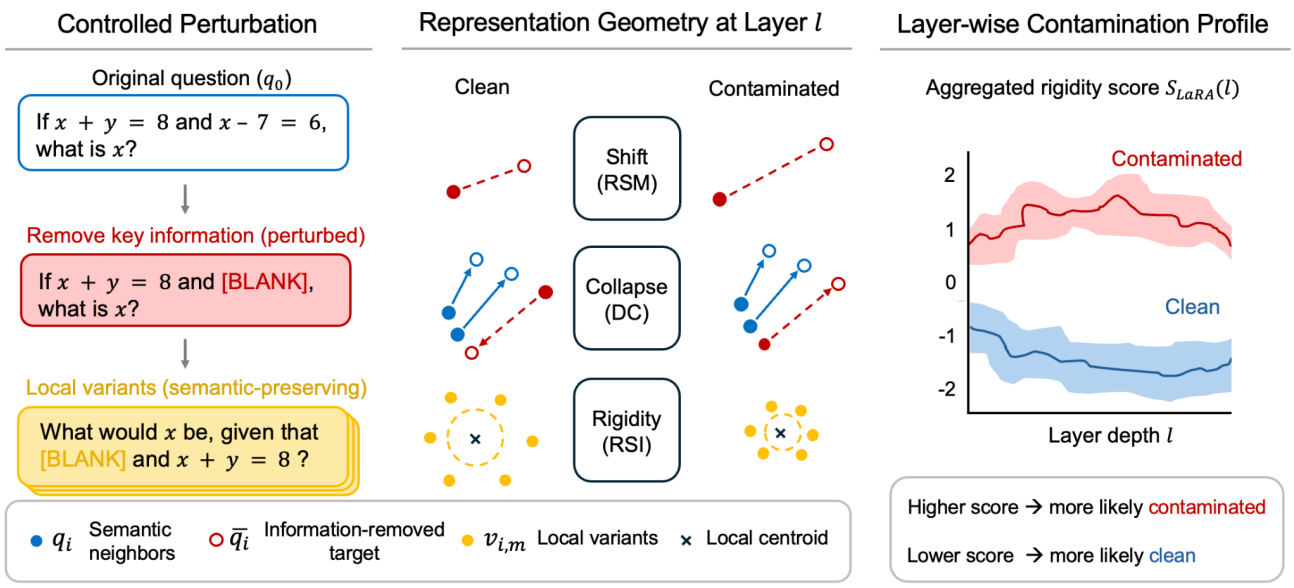

Figure 2. **Overview of LaRA, the proposed layer-wise representation geometry analysis framework.** Given an original question q_0, we generate semantically similar questions and remove shared key information to construct perturbed variants. Hidden representations are extracted across all transformer layers for original, perturbed, and paraphrased inputs. We then compute three complementary geometric metrics: Representation Shift Magnitude (RSM), Directional Collapse (DC), and Representation Stability Index (RSI), which characterize perturbation sensitivity, directional organization, and local representation variability under controlled perturbations.

while preserving the overall question structure:

$$q_i^- \leftarrow \textsc{BlankImportant}(q_i, k),$$

where k denotes the number of inserted `[BLANK]` tokens. Refer to Appendix B for details of the perturbation construction process. Let $h_\ell(\cdot)$ denote the mean-pooled hidden representation extracted from transformer layer ℓ, where $\ell \in \mathcal{L} = \{0, 1, \ldots, L-1\}$. For each ℓ, we extract hidden representations $u_i = h_\ell(q_i)$ and $w_i = h_\ell(q_i^-)$, where $u_i, w_i \in \mathbb{R}^d$ and d is the hidden representation dimension. We then compute the perturbation-induced representation shift Δ_i and define its magnitude S_i as:

$$S_i = \|\Delta_i\|_2, \quad \Delta_i = u_i - w_i$$

where $\|\cdot\|_2$ denotes the Euclidean norm. To capture how anomalously the original question responds to perturbation relative to its semantic neighbors, we standardize its shift magnitude using the mean and standard deviation of the similar-question set:

$$RSM_\ell = \frac{S_0 - \mu_S}{\sigma_S + \epsilon},$$

where

$$\mu_S = \frac{1}{K}\sum_{i=1}^{K} S_i, \; \sigma_S = \sqrt{\frac{1}{K-1}\sum_{i=1}^{K}(S_i - \mu_S)^2},$$

and $\epsilon > 0$ is a numerical stability constant. A high RSM_ℓ indicates that the original question exhibits a larger representation shift under information removal compared to semantically similar questions, suggesting stronger perturbation sensitivity from memorization and risk of contamination.

Metric 2: Directional Collapse. We introduce Directional Collapse (DC) to characterize the directional organization of perturbation-induced representation changes. We first compute the average perturbation direction across similar questions:

$$\bar{s}_\ell = \frac{1}{K}\sum_{i=1}^{K} \Delta_i,$$

where $\bar{s}_\ell \in \mathbb{R}^d$ represents the average perturbation direction shared across the semantic group. DC is then defined as:

$$DC_\ell = \frac{\Delta_0^\top \bar{s}_\ell}{(\|\Delta_0\|_2 + \epsilon)(\|\bar{s}_\ell\|_2 + \epsilon)}.$$

This quantity measures the cosine alignment between the original perturbation direction and the average perturbation direction of semantically similar questions. High DC_ℓ values indicate that perturbation responses are strongly aligned along a shared low-dimensional direction, whereas lower values indicate more distributed or heterogeneous perturbation dynamics.

Metric 3: Representation Stability Index. Finally, we measure local representation stability under semantically preserving perturbations through the Representation Stability Index (RSI). For each perturbed question q_i^-, we generate M paraphrastic variants while preserving the blank

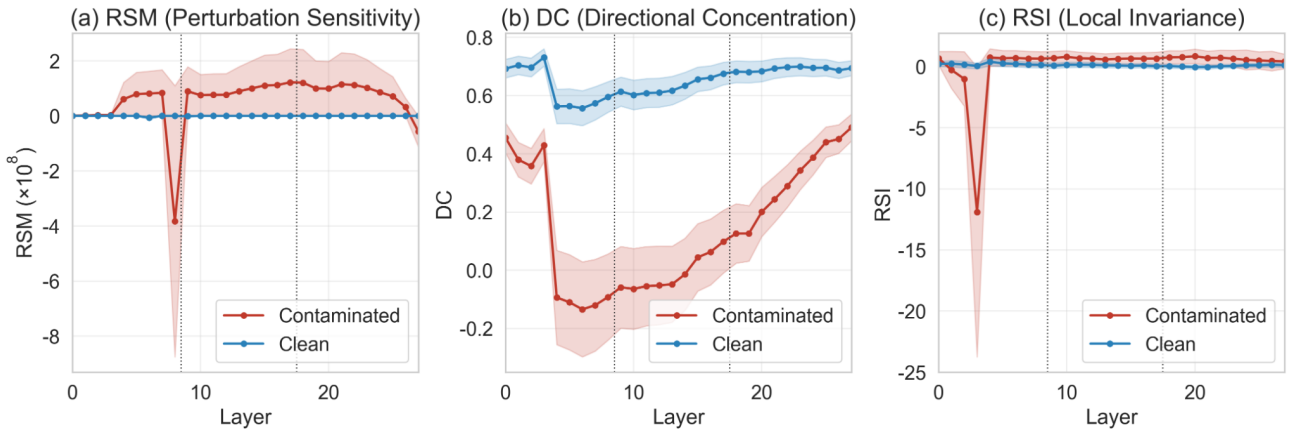

Figure 3. **Layer-wise representation geometry patterns of RL post-trained model.** We compare layer-wise representation geometry between contaminated and clean samples under input perturbations. Contaminated samples consistently exhibit deviating perturbation sensitivity (**RSM**), abnormal directional concentration dynamics (**DC**), and altered local representation variability patterns (**RSI**) across transformer layers, indicating that memorized samples form distinct and less robust internal representation structures compared to clean samples.

positions:

$$\{v_{i,1}, \ldots, v_{i,M}\} \sim \textsc{VariantGen}(q_i^-).$$

We then extract their hidden representations:

$$\phi_{i,m} = h_\ell(v_{i,m}),$$

where $\phi_{i,m} \in \mathbb{R}^d$. Next, we compute the local representation centroid:

$$\bar{\phi}_i = \frac{1}{M} \sum_{m=1}^{M} \phi_{i,m},$$

and define its average deviation:

$$R_i = \frac{1}{M} \sum_{m=1}^{M} \|\phi_{i,m} - \bar{\phi}_i\|_2.$$

We then standardize the original question's local variability relative to its semantic neighbors:

$$RSI_\ell = \frac{R_0 - \mu_R}{\sigma_R + \epsilon},$$

where

$$\mu_R = \frac{1}{K} \sum_{i=1}^{K} R_i, \ \sigma_R = \sqrt{\frac{1}{K-1} \sum_{i=1}^{K} (R_i - \mu_R)^2}.$$

A high RSI_ℓ indicates that the original question exhibits larger local representation variability relative to semantically similar questions under paraphrastic perturbations, while lower values indicate more locally stable representation behavior.

2.3. Layer-wise Analysis with Three Metrics

Figure 3 shows the representation geometry patterns measured by the three metrics in Section 2.2. Contaminated samples consistently exhibit larger perturbation-induced representation shifts (RSM) than clean samples across most layers, while clean samples remain near zero throughout depth. In particular, contaminated samples sharply deviate around layers 7–9, indicating substantially higher sensitivity to targeted information removal and stronger dependence on memorized information. DC results further show that contaminated samples exhibit distinct directional concentration dynamics compared to clean samples. RSI results show that contaminated samples exhibit lower local representation variability, particularly in early layers, indicating more rigid and invariant local representation geometry under paraphrastic perturbations. Additional results are provided in Appendix F.

3. Contamination Detection Protocol

Motivated by Section 2.3, we formulate contamination detection as a layer-aware representation anomaly detection problem. We find that contaminated samples exhibit distinct representation profiles across depth, including amplified perturbation sensitivity, abnormal directional concentration dynamics, and local variability under controlled perturbations. Consequently, contamination should be characterized through *deviation from clean geometric profiles* across multiple metrics and layers, rather than from isolated layer-wise statistics.

Step 1: Clean-reference Robust Standardization. Let $\mathcal{M} = \{\mathrm{RSM}, \mathrm{DC}, \mathrm{RSI}\}$ denote the set of representation geometry metrics, $\mathcal{L}$ denote the set of probed transformer

layers, and $m_\ell(x)$ denote the value of metric m at layer ℓ for sample x. The three metrics span several orders of magnitude in raw form, so we first apply a sign-preserving compression to tame their heavy-tailed regime while leaving values near zero unchanged:

$$\tilde{m}_\ell(x) = \mathrm{sign}\big(m_\ell(x)\big)\ \log\big(1 + |m_\ell(x)|\big). \tag{1}$$

For each (m, ℓ), we estimate the clean reference *center* and *scale* from non-contaminated validation samples $\mathcal{D}^{\mathrm{clean}}$:

$$\mu^{\mathrm{clean}}_{m,\ell} = \mathrm{median}\Big(\tilde{m}_\ell(x) : x \in \mathcal{D}^{\mathrm{clean}}\Big), \tag{2}$$

$$\sigma^{\mathrm{clean}}_{m,\ell} = 1.4826 \cdot \mathrm{MAD}\Big(\tilde{m}_\ell(x) : x \in \mathcal{D}^{\mathrm{clean}}\Big), \tag{3}$$

where the factor 1.4826 is the standard scaling to make median absolute deviation (MAD) a consistent estimator of the standard deviation under Gaussian noise (see Appendix H). The standardized geometric deviation of sample x at (m, ℓ) is then:

$$z_{m,\ell}(x) = \frac{\tilde{m}_\ell(x) - \mu^{\mathrm{clean}}_{m,\ell}}{\sigma^{\mathrm{clean}}_{m,\ell} + \epsilon}, \tag{4}$$

with ϵ a small numerical constant. This formulation preserves the relative magnitude of geometric deviations while preventing a small number of extreme contaminated samples from inflating the clean-reference scale and washing out the signal for the rest of the population.

Step 2: Metric-specific Anomaly Alignment. Our analyses show that contamination affects each metric through a different geometric mechanism. Contaminated samples tend to exhibit elevated perturbation sensitivity in RSM, abnormal directional concentration dynamics in DC, and reduced or unstable local invariance in RSI. To account for these heterogeneous behaviors, we align each metric according to its contamination-associated pattern:

$$\hat{z}_{m,\ell}(x) = \begin{cases} z_{m,\ell}(x), & m = \mathrm{RSM}, \\ z_{m,\ell}(x), & m = \mathrm{DC}, \\ -z_{m,\ell}(x), & m = \mathrm{RSI}. \end{cases} \tag{5}$$

For DC, we preserve the signed deviation because the contamination signal is directional. For RSM and RSI, the alignment similarly recovers deviations associated with contamination-related geometric behavior.

Step 3: Layer-wise Aggregation. We aggregate the aligned deviations $\hat{z}_{m,\ell}(x)$ across the metric set $\mathcal{M}$ and layer set $\mathcal{L}$ to obtain a single per-sample score, where larger values indicate stronger overall deviation from the clean geometric profile.

$$S_{\mathrm{LaRA}}(x) = \frac{1}{|\mathcal{M}|\,|\mathcal{L}|} \sum_{m \in \mathcal{M}} \sum_{\ell \in \mathcal{L}} \hat{z}_{m,\ell}(x). \tag{6}$$

Because all (m, ℓ) contributions are standardized onto the same robust z-scale before aggregation, abnormalities arising from different layers and metrics can be consistently compared and combined within $S_{\mathrm{LaRA}}(x)$.

3.1. Main Results

Table 1 shows that our proposed representation-based membership score, S_{LaRA}, consistently achieves strong and stable detection performance across different RL model families and training checkpoints. In the initial checkpoints, S_{LaRA} attains the best overall performance on LIMR with an AUC of 0.80 and TPR@FPR=5% of 0.46, substantially outperforming standard baselines such as Recall, Min-K%, and SC. We also explore combining S_{LaRA} with SC, the sota output-level detection method, to see the complementarity of the two detection regimes. Combining S_{LaRA} with SC ($\mathrm{SC}+S_{\mathrm{LaRA}}$) achieves the strongest overall performance on Eurus, reaching an AUC of 0.73 and TPR@FPR=5% of 0.31 at initialization, while also maintaining competitive performance throughout RL training. Across Eurus checkpoints, the combined score steadily improves from $(0.73, 0.31)$ to $(0.79, 0.38)$ in terms of (AUC, TPR@FPR=5%), suggesting that representation-level contamination signals become increasingly separable during RL optimization. LIMR exhibits a similar trend for S_{LaRA}, where performance remains consistently high across checkpoints, peaking at $(0.81, 0.20)$ at epoch2. Although PPL occasionally shows relatively high AUC values, their TPR@FPR=5% remains substantially lower and less stable than the proposed methods. PPL often relies on superficial token likelihood differences that can fluctuate across model families and RL stages, whereas S_{LaRA} and $\mathrm{SC} + S_{\mathrm{LaRA}}$ capture deeper geometric inconsistencies in hidden representations. Therefore, our approach leads to more reliable detection under strict low-FPR operating regimes critical for realistic settings.

3.2. Additional Analyses

Metric Ablations. Table 2 shows that combining all three components (RSM, DC, and RSI) consistently achieves the best overall performance across RL checkpoints, improving both AUC and robustness to training-stage shifts. While **DC** provides the strongest standalone discrimination signal, its performance varies more across epochs, particularly in TPR@FPR=5%, indicating reduced robustness when used alone. In contrast, **RSM** and **RSI** individually produce weaker detection performance but contribute to improving generalization under RL post-training. Removing an individual component from the full metric consistently degrades performance, showing that the final score benefits from jointly modeling perturbation sensitivity, directional representation geometry, and local invariance. Overall, the results suggest that robust contamination detection requires integrating multiple representation-level signals rather than

Method	Eurus		LIMR		OLMO	
	AUC	TPR	AUC	TPR	AUC	TPR
Recall	0.59	0.07	0.54	0.07	0.42	0.03
CDD	0.38	0.00	0.64	0.00	0.50	0.00
Min-K%	0.32	0.07	0.26	0.03	0.50	0.03
Min-K%++	0.29	0.03	0.60	0.07	0.46	0.07
PPL	0.68	0.23	0.73	0.13	0.49	0.07
SC	0.70	0.27	0.44	0.00	0.42	0.03
lightblue S_{LaRA}	0.63	0.19	**0.80**	**0.46**	**0.54**	0.04
lightblue SC + S_{LaRA}	**0.73**	**0.31**	0.47	0.04	0.50	**0.08**

(a) Initial checkpoint results across models.

Method	Eurus						LIMR					
	Init		E1		E2		Init		E1		E2	
	AUC	TPR	AUC	TPR	AUC	TPR	AUC	TPR	AUC	TPR	AUC	TPR
Recall	0.59	0.07	0.56	0.13	0.52	0.00	0.54	0.07	0.47	0.03	0.47	0.00
CDD	0.38	0.00	0.38	0.00	0.38	0.00	0.64	0.00	0.61	0.00	0.59	0.00
Min-K%	0.32	0.07	0.31	0.07	0.33	0.07	0.26	0.03	0.26	0.00	0.26	0.00
Min-K%++	0.29	0.03	0.26	0.03	0.20	0.03	0.60	0.07	0.58	0.07	0.63	0.00
PPL	0.68	0.23	**0.69**	0.23	0.67	0.17	0.73	0.13	**0.74**	0.13	0.74	0.17
SC	0.70	0.27	0.58	0.07	0.78	0.30	0.44	0.00	0.40	0.00	0.46	0.00
lightblue S_{LaRA}	0.63	0.19	0.68	0.19	0.70	0.15	**0.80**	**0.46**	0.72	**0.23**	**0.81**	**0.20**
lightblue SC + S_{LaRA}	**0.73**	**0.31**	0.65	**0.35**	**0.79**	**0.38**	0.47	0.04	0.46	0.04	0.54	0.04

(b) Results across Eurus and LIMR training checkpoints.

Table 1. **Membership inference performance across different RL checkpoints and model families.** (a) compares initial checkpoints across Eurus, LIMR, and OLMO, while (b) analyzes performance evolution during RL training for Eurus and LIMR. TPR denotes TPR@FPR=5%.

RSM	DC	RSI	Init		E1		E2	
			AUC	TPR	AUC	TPR	AUC	TPR
55	55	51	0.60	0.19	0.51	0.15	0.56	0.16
55	51	55	0.71	0.42	0.66	0.08	0.76	0.24
51	55	55	0.75	0.12	0.69	0.08	0.71	0.04
55	51	51	0.76	0.46	0.69	0.15	0.81	0.32
51	55	51	0.74	0.27	0.68	0.08	0.71	0.12
51	51	55	0.78	0.31	0.71	0.19	0.79	0.12
lightblue 51	51	51	0.80	0.46	0.72	0.23	0.81	0.20

Table 2. **Metric ablations across RL training epochs.**

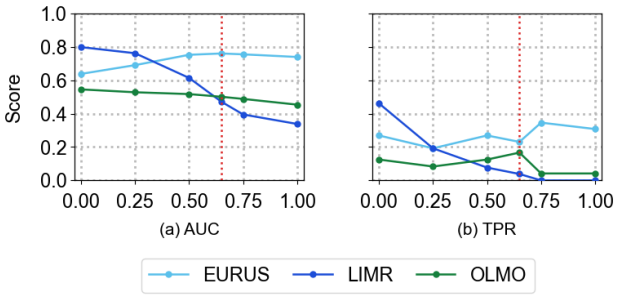

Figure 4. **Beta sweep over score mix.** Default β = 0.65.

relying on a single geometric statistic.

Beta Sweep over S_{LaRA} and SC Mix. We sweep the mixture weight β in $\text{mix} = \beta(\text{SC}) + (1-\beta)(S_{\text{LaRA}})$ on the member-detection benchmark ($\beta \in \{0, 0.25, 0.5, 0.65, 0.75, 1\}$). The optimal balance between SC and S_{LaRA} is strongly model-dependent. For Eurus, performance improves as more weight is assigned to SC, with AUC peaking at the default $\beta = 0.65$ and TPR@FPR=5% at $\beta = 0.75$. In contrast, LIMR performs best with only the S_{LaRA}, with performance degrading as β increases. For OLMO, AUC is highest at $\beta = 0$, while TPR@FPR=5% peaks at $\beta = 0.65$. Overall, these results suggest that no single mixture weight is universally optimal; however, despite not being tuned per model, the shared default $\beta = 0.65$ still achieves competitive overall performance, consistent with the main results.

Correlation with Output-level Metrics. Figure 5(a) shows the correlation between the proposed S_{LaRA} and several output-level metrics. Higher S_{LaRA} values are negatively correlated with SC ($\rho = -0.16$) and PPL ($\rho = -0.29$), while exhibiting a weak positive correlation with Min-K%++ ($\rho = 0.17$). Additionally, samples with low S_{LaRA} display substantially larger variability across all metrics, whereas high-S_{LaRA} samples are concentrated within narrower output regimes. Correlations suggest that stronger contamination-related geometric deviations are associated with increasingly confident, less reflective, and more behaviorally concentrated generations. In particular, high-S_{LaRA} samples occupy narrower output regimes characterized by reduced variability across output-level metrics.

Analysis on Number of Perturbations. The perturbation-count sweep on Eurus (Figure 5(b)) shows that S_{LaRA} remains relatively stable across different numbers of blanks. Although AUC improves from the default $k = 1$ (0.62) to a peak at $k = 3$ (0.77), TPR@FPR=5% only substantially increases at $k = 4$ (0.32), where AUC slightly declines (0.69). This indicates that increasing the number of blanks can sometimes slightly strengthen detection, but the gains depend on the evaluation metric and operating regime. Overall, the method is robust to the choice of k, and the default setting $k = 1$ already provides competitive performance without requiring additional perturbation variants.

Analysis on Perturbation Types. The perturbation analysis in Figure 6 shows that LaRA remains relatively robust across different perturbation types, with all variants achieving comparable AUC values between 0.56 and 0.69. Although `Distractor Insert.` achieves the highest AUC (0.69) and `Num Replace.` obtains the best TPR@FPR=5% (0.17), the default setting based on `Info Rem.`, the most naive method in making perturbations, still maintains reasonable detection performance without requiring semantically invasive modifications. In contrast to perturbations that directly alter numerical or semantic reasoning components, the default perturbation preserves the original problem structure more conservatively while still produc-

data_source	prompt	member	RSM	DC	RSI
numina_amc_aime	Let the set $S = \{P_1, P_2, \ldots, P_{12}\}$ consist of the twelve vertices of a regular 12-gon. A subset Q of S is called "communal" if there is a circle such that all points of Q are inside the circle, and all points of S not in Q are outside of the circle. How many communal subsets are there? (Note that the empty set is a communal subset.) Present the answer in LaTeX format: \boxed{Your answer}	1	0.151	0.423	0.310
aime26	Let triangle ABC have side lengths AB = 13, BC = 14, and CA = 15. Triangle A'B'C' is obtained by rotating triangle ABC about its circumcenter so that AC is perpendicular BC, with A' and B not on the same side of line B'C'. Find the integer closest to the area of hexagon AA'CC'BB'.Present the answer in LaTeX format: \boxed{Your answer}	0	1.191	9.765	0.370

Table 3. **Error analysis.** (1) member sample often undetected, and (2) non-member sample often wrongly detected.

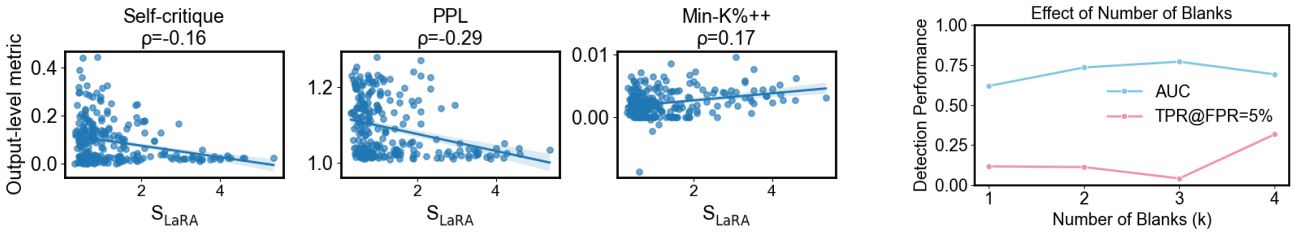

Figure 5. **(a) Correlation plot of output-level signals and S_{LaRA}.** **(b) Analysis on the # of perturbations.**

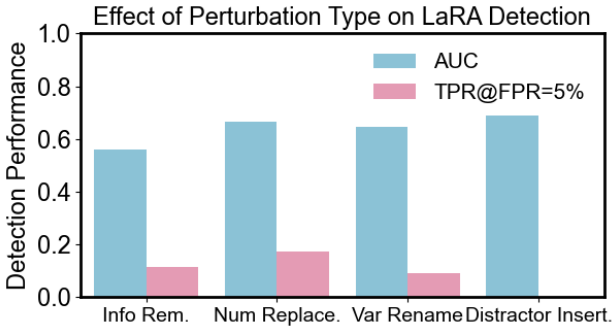

Figure 6. **Perturbation type analysis.** Results show that S_{LaRA} is robust to perturbation types.

ing distinguishable contamination signals. Overall, results suggest that LaRA does not rely on a single perturbation type and exhibits stable detection behavior across diverse perturbation strategies.

Error analysis. Failure cases presented in Table 3 suggest that S_{LaRA} misses member examples whose representation geometry remains close to the non-member manifold. The false-negative member sample exhibits uniformly low component scores, with RSM, DC, and RSI values of 0.151, 0.423, and 0.310, respectively, producing a low aggregate score of 0.295, well below the detection threshold. This indicates that the sample does not induce strong perturbation sensitivity, directional concentration, or local invariance, causing its layer-wise representation trajectory to resemble that of a clean example. Conversely, the non-member example that is falsely detected exhibits an abnormally high S_{LaRA} score, primarily driven by an unusually large DC value. Although the sample is not a member, its representation geometry deviates substantially from the typical clean distribution, leading the detector to classify it as contaminated.

4. Conclusion

We introduce LaRA, a layer-wise representation analysis framework for contamination detection in RL-trained models. Unlike prior methods, LaRA detects contamination through perturbation-induced representation geometry across layers, measured with three proposed metrics. Based on layer-wise analysis, we propose a detection framework that aggregates geometric deviations across metrics and layers. Experiments across various RL-trained models demonstrate that representation-level signals complement output-level methods and improve detection performance. Overall, our results show that contamination in RL-trained LLMs is strongly reflected in internal representation geometry, highlighting the effectiveness of representation-level auditing.

Limitations

Despite new findings and the effectiveness, our work has several limitations. First, LaRA relies on representation extraction under multiple semantic perturbations and layer-wise hidden-state analyses, which introduces additional computational overhead compared to lightweight output-level approaches. In particular, the framework requires generating perturbed variants, extracting intermediate representations across transformer layers, and aggregating multiple geometric statistics, making inference more expensive than methods operating solely on final outputs or token probabilities. Second, although the proposed framework consistently improves contamination detection performance over exist-

ing baselines, detection remains imperfect for certain challenging examples whose representation geometry closely overlaps with the clean distribution. This suggests that some memorized samples may not produce sufficiently distinctive internal signatures for reliable separation. In addition, while our analyses reveal consistent trends across models and training checkpoints, the precise relationship between RL post-training dynamics and representation-level memorization behavior remains only partially understood. Future work may explore more computationally efficient perturbation strategies, stronger representation aggregation methods, and deeper investigations into the causal mechanisms underlying memorization and contamination in reasoning models.

Broader Impact and Ethical Implications

This work contributes to improving transparency and reliability in the evaluation of RL-trained LLMs by introducing a representation-level framework for contamination detection. Stronger contamination auditing can help researchers better assess benchmark integrity, reduce hidden memorization effects, and improve the trustworthiness of reported reasoning capabilities. At the same time, contamination detection methods may potentially be adapted for membership inference or unauthorized dataset auditing, which could raise privacy or data governance concerns when applied to sensitive or proprietary data. Our experiments are conducted exclusively on publicly available open-source models and public mathematical datasets, and we do not use private or personally identifiable information. Overall, we believe representation-level auditing should be used responsibly as one component of broader efforts toward reliable, transparent, and reproducible evaluation of large language models.

Impact Statement

This work studies data contamination detection in RL post-trained large language models through layer-wise representation geometry analyses. As RL-trained reasoning models become increasingly common, contamination during post-training can undermine benchmark validity, inflate reported reasoning performance, and reduce the reliability of scientific comparisons. By introducing a representation-level framework for identifying contamination-related behaviors, our work aims to improve the transparency and trustworthiness of reasoning model evaluation.

More broadly, our findings suggest that contamination in RL-trained models is reflected not only in output behavior, but also in internal representation geometry. This may motivate future work on representation-level auditing, interpretability, and reliability analysis for post-trained LLMs. In particular, methods that characterize perturbation sensitivity and representation rigidity may help identify failure modes that are difficult to observe from outputs alone.

At the same time, representation-level analysis methods may introduce dual-use concerns. Techniques that probe internal activations could potentially be adapted for stronger membership inference attacks or for extracting information about training distributions. Although our framework is designed for aggregate contamination auditing rather than recovery of individual training samples, advances in representation analysis may nevertheless increase privacy-related risks if misused. We therefore encourage future work on safeguards, controlled evaluation protocols, and privacy-aware auditing methodologies.

Our experiments are conducted exclusively on publicly available models and datasets within a controlled research setting. We do not release methods intended to reconstruct memorized data or reveal sensitive user information. Overall, we believe this work contributes toward more reliable and rigorous evaluation of RL-trained reasoning systems while highlighting the importance of studying the broader implications of representation-level auditing techniques.

References

Balunović, M., Dekoninck, J., Petrov, I., Jovanović, N., and Vechev, M. Matharena: Evaluating llms on uncontaminated math competitions, February 2025. URL https://matharena.ai/.

Cui, G., Yuan, L., Wang, Z., Wang, H., Zhang, Y., Chen, J., Li, W., He, B., Fan, Y., Yu, T., et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025.

Dong, Y., Jiang, X., Liu, H., Jin, Z., Gu, B., Yang, M., and Li, G. Generalization or memorization: Data contamination and trustworthy evaluation for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 12039–12050, 2024.

Gonen, H., Iyer, S., Blevins, T., Smith, N. A., and Zettlemoyer, L. Demystifying prompts in language models via perplexity estimation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 10136–10148, 2023.

Guha, E., Marten, R., Keh, S., Raoof, N., Smyrnis, G., Bansal, H., Nezhurina, M., Mercat, J., Vu, T., Sprague, Z., et al. Openthoughts: Data recipes for reasoning models. *arXiv preprint arXiv:2506.04178*, 2025.

Guo, D., Yang, D., Zhang, H., Song, J., Wang, P., Zhu, Q., Xu, R., Zhang, R., Ma, S., Bi, X., et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Hampel, F. R., Ronchetti, E. M., Rousseeuw, P. J., and Stahel, W. A. *Robust Statistics: The Approach Based on Influence Functions*. Wiley, 1986.

Hochlehnert, A., Bhatnagar, H., Udandarao, V., Albanie, S., Prabhu, A., and Bethge, M. A sober look at progress in language model reasoning: Pitfalls and paths to reproducibility. *arXiv preprint arXiv:2504.07086*, 2025.

Huber, P. J. and Ronchetti, E. M. *Robust Statistics*. Wiley, 2 edition, 2009.

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J., Zhang, H., and Stoica, I. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

Li, X., Zou, H., and Liu, P. Limr: Less is more for rl scaling. *arXiv preprint arXiv:2502.11886*, 2025.

Maronna, R. A., Martin, R. D., Yohai, V. J., and Salibián-Barrera, M. *Robust Statistics: Theory and Methods (with R)*. Wiley, 2 edition, 2019.

OpenAI. Gpt-4o mini: advancing cost-efficient intelligence, 2024.

OpenRouter. Openrouter: Unified api for AI models, 2024. URL https://openrouter.ai.

Rousseeuw, P. J. and Croux, C. Alternatives to the median absolute deviation. *Journal of the American Statistical Association*, 88(424):1273–1283, 1993.

Rousseeuw, P. J. and Leroy, A. M. *Robust Regression and Outlier Detection*. Wiley, 1987.

Sheng, G., Zhang, C., Ye, Z., Wu, X., Zhang, W., Zhang, R., Peng, Y., Lin, H., and Wu, C. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

Shi, W., Ajith, A., Xia, M., Huang, Y., Liu, D., Blevins, T., Chen, D., and Zettlemoyer, L. Detecting pretraining data from large language models. *arXiv preprint arXiv:2310.16789*, 2023.

Tao, Y., Wang, T., Dong, Y., Liu, H., Zhang, K., Hu, X., and Li, G. Detecting data contamination from reinforcement learning post-training for large language models. *arXiv preprint arXiv:2510.09259*, 2025.

Wang, H., Li, H., Ko, B., and Zhang, H. On the fragility of benchmark contamination detection in reasoning models. *arXiv preprint arXiv:2510.02386*, 2025.

Wu, M., Zhang, Z., Dong, Q., Xi, Z., Zhao, J., Jin, S., Fan, X., Zhou, Y., Lv, H., Zhang, M., et al. Reasoning or memorization? unreliable results of reinforcement learning due to data contamination. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 33944–33952, 2026.

Xie, R., Wang, J., Huang, R., Zhang, M., Ge, R., Pei, J., Gong, N. Z., and Dhingra, B. 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.

Zhang, J., Sun, J., Yeats, E., Ouyang, Y., Kuo, M., Zhang, J., Yang, H. F., and Li, H. Min-k%++: Improved baseline for detecting pre-training data from large language models. *arXiv preprint arXiv:2404.02936*, 2024.

A. Algorithm

Details of the LaRA algorithm is in Appendix A.

Algorithm 1 LaRA: Per-sample Layer-wise Representation Geometry Extraction

Input: Original question q_0; similar-question generator SIMILARGEN; importance-based blanking operator BLANKIMPORTANT producing k `[BLANK]` tokens; LLM paraphrase generator VARIANTGEN that preserves `[BLANK]` positions; mean-pooled hidden-state extractor $h_\ell(\cdot)$; layer set $\mathcal{L} = \{0, 1, \ldots, L-1\}$ (every transformer layer); number of similar questions K; number of paraphrase variants M; number of `[BLANK]` tokens k; numerical floor ϵ

Output: Per-layer geometric scores $\{\text{zRSM}_\ell, \text{DC}_\ell, \text{zRSI}_\ell\}_{\ell \in \mathcal{L}}$

$\{q_1, \ldots, q_K\} \leftarrow$ SIMILARGEN(q_0) $\mathcal{Q} \leftarrow \{q_0, q_1, \ldots, q_K\}$

foreach $q_i \in \mathcal{Q}$ **do**

  $q_i^- \leftarrow$ BLANKIMPORTANT$(q_i,\ k)$ $\{v_{i,1}, \ldots, v_{i,M}\} \leftarrow$ VARIANTGEN(q_i^-)

end

foreach $\ell \in \mathcal{L}$ **do**

  `/* (1) Representation Shift Magnitude → zRSM */`

  for $i = 0$ **to** K **do**

    $u_i \leftarrow h_\ell(q_i)$ $w_i \leftarrow h_\ell(q_i^-)$ $\Delta_i \leftarrow u_i - w_i$ $S_i \leftarrow \|\Delta_i\|_2$

  end

  $\mu_S \leftarrow \frac{1}{K}\sum_{i=1}^{K} S_i$; `// similars only`

  $\sigma_S \leftarrow \sqrt{\frac{1}{K-1}\sum_{i=1}^{K}(S_i - \mu_S)^2}$; `// sample std`

  $\text{zRSM}_\ell \leftarrow \dfrac{S_0 - \mu_S}{\sigma_S + \epsilon}$

  `/* (2) Directional Collapse → DC */`

  $\bar{s}_\ell \leftarrow \frac{1}{K}\sum_{i=1}^{K} \Delta_i$; `// mean shift over similars`

  $\text{DC}_\ell \leftarrow \dfrac{\Delta_0^\top \bar{s}_\ell}{(\|\Delta_0\|_2 + \epsilon)\,(\|\bar{s}_\ell\|_2 + \epsilon)}$; `// cosine`

  `/* (3) Representation Stability Index → zRSI */`

  for $i = 0$ **to** K **do**

    for $m = 1$ **to** M **do**

      $\phi_{i,m} \leftarrow h_\ell(v_{i,m})$

    end

    $\bar{\phi}_i \leftarrow \frac{1}{M}\sum_{m=1}^{M} \phi_{i,m}$ $R_i \leftarrow \frac{1}{M}\sum_{m=1}^{M} \|\phi_{i,m} - \bar{\phi}_i\|_2$; `// mean L2 distance`

  end

  $\mu_R \leftarrow \frac{1}{K}\sum_{i=1}^{K} R_i$ $\sigma_R \leftarrow \sqrt{\frac{1}{K-1}\sum_{i=1}^{K}(R_i - \mu_R)^2}$

  $\text{zRSI}_\ell \leftarrow \dfrac{R_0 - \mu_R}{\sigma_R + \epsilon}$

end

return $\{\text{zRSM}_\ell, \text{DC}_\ell, \text{zRSI}_\ell\}_{\ell \in \mathcal{L}}$

B. Details of Generating Similar and Perturbed Questions

B.1. Prompts

To analyze representation dynamics under controlled perturbations, we use a three-stage prompt pipeline consisting of: (1) generating structurally similar questions, (2) identifying removable key information, and (3) generating paraphrased perturbation variants.

As shown in Table 4, we first generate semantically similar math problems that preserve the same reasoning structure and difficulty while modifying numerical values. This produces structurally matched control groups for representation comparison. Table 5 and 6 illustrates the prompt used to generate paraphrased variants of perturbed questions while preserving the exact position and semantic role of the `[BLANK]` placeholder. These variants enable measurement of local representation variability for computing RSI. Other prompts used to make the variants are in Table 7, 8, and 9.

B.2. Examples of the Generated Questions

We present the examples of the original and generated questions to qualitatively show their generations in Table 10. Other questions generated by other perturbation techniques are in Table 11, 12, and 13.

Settings	Content
Similar Questions	You are a math problem generator. Given an original math problem, create {num_questions} similar problems that: 1. Follow the EXACT same structure and solution method as the original 2. Use DIFFERENT numerical values (change all numbers to make the problem unique) 3. Maintain the same difficulty level 4. Have the same type of solution approach 5. Are valid, solvable problems Original Problem: {original_question} Generate {num_questions} similar problems. For each problem: - Change ALL numerical values to create unique scenarios - Keep the problem structure and mathematical concepts identical - Ensure the problem remains solvable and realistic - Make sure the new numbers create valid mathematical relationships Output ONLY a JSON array of {num_questions} similar problems, where each element is a string containing the full problem text. Do not include solutions or explanations, only the problems. Format your response as: [""Problem 1 text here...", ""Problem 2 text here..."]

Table 4. Prompt used for generating similar math questions.

Settings	Content
Perturbed Questions	You are a question editor that identifies key information to remove from math problems. Given a math problem, identify ONE key piece of information that should be removed. Describe this information in a way that can be consistently applied to similar problems. For example: - "the total number of residents/people" - "the initial quantity" - "the final result value" - "the time duration" - "the distance measurement" Original Problem: {original_question} Output ONLY a short description of what information type should be removed (e.g., "the total number of residents"). Do not include the actual value or explain why, just describe the information type in 5-10 words.

Table 5. Prompt used for identifying removable information in math questions.

Settings	Content
Perturbed Variants	You are a text rewriter that creates paraphrased versions of math problems. Given an incomplete math problem with [BLANK] placeholders, create {num_variants} paraphrased versions that: 1. Preserve the EXACT position and meaning of [BLANK] - do NOT move or change [BLANK] 2. Use different wording and phrasing while maintaining the same mathematical meaning 3. Keep the same structure and logical flow 4. Do NOT reveal what the blank should be 5. Maintain all mathematical relationships and constraints Incomplete Problem: {incomplete_question} Output ONLY a JSON array of {num_variants} strings with the paraphrased versions. Do not include explanations or notes.

Table 6. Prompt used for generating perturbed paraphrased variants.

Setting	Content
Perturbation Analysis (Variable Renaming)	`You edit math problems for robustness testing. For each problem, replace EXACTLY ONE single-letter variable (e.g. x, y, n, t) with a random common English noun (not a single letter). Example: replace "x" with "widget" everywhere that variable appears in the problem, or only once if the variable appears once --- pick one occurrence if multiple variables exist. - Do NOT use [BLANK]. - Keep all numbers and the problem structure unchanged except for that substitution. Problems: {numbered} Output ONLY a JSON array of strings with the same length and order as the input.`

Table 7. Prompt used for perturbation analysis through renaming variable.

Setting	Content
Perturbation Analysis (Number Replacement)	`You edit math problems for robustness testing. For each problem, change EXACTLY ONE numeric literal to a different numeric literal. - Do NOT use [BLANK]. - Do NOT change anything else (wording, variables, structure). - The new number must differ from the original. Problems: numbered Output ONLY a JSON array of strings with the same length and order as the input.`

Table 8. Prompt used for perturbation analysis through number replacement.

Setting	Content
Perturbation Analysis (Insert Distractor)	`You edit math problems for robustness testing. For each problem, insert EXACTLY ONE short distractor sentence (one clause) that is irrelevant or misleading but plausible-sounding. Place it naturally in the problem text. - Do NOT use [BLANK]. - Do not change the original numbers or the core question being asked. Problems: numbered Output ONLY a JSON array of strings with the same length and order as the input.`

Table 9. Prompt used for perturbation analysis through inserting distractor.

Settings	Content
Original Question	A plane contains points A and B with $AB = 1$. Point A is rotated in the plane counterclockwise through an acute angle θ around point B to point A'. Then B is rotated in the plane clockwise through angle θ around point A' to point B'. Suppose that $AB' = \frac{4}{3}$. The value of $\cos\theta$ can be written as $\frac{m}{n}$, where m and n are relatively prime positive integers. Find $m+n$. Present the answer in LaTeX format: $\boxed{\text{Your answer}}$
Similar Question	A plane contains points A and B with $AB = 2$. Point A is rotated in the plane counterclockwise through an acute angle θ around point B to point A'. Then B is rotated in the plane clockwise through angle θ around point A' to point B'. Suppose that $AB' = \frac{5}{4}$. The value of $\cos\theta$ can be written as $\frac{m}{n}$, where m and n are relatively prime positive integers. Find $m+n$.
Perturbed Question	A plane contains points A and B with $AB = [BLANK]$. Point A is rotated in the plane counterclockwise through an acute angle θ around point B to point A'. Then B is rotated in the plane clockwise through angle θ around point A' to point B'. Suppose that $AB' = \frac{4}{3}$. The value of $\cos\theta$ can be written as $\frac{m}{n}$, where m and n are relatively prime positive integers. Find $m+n$.
Perturbed Variant	In a plane, there are points A and B such that $AB = [BLANK]$. Point A is rotated counterclockwise around point B through an acute angle θ, resulting in point A'. Next, point B is rotated clockwise around point A' through the same angle θ, leading to point B'. It is given that $AB' = \frac{9}{8}$. The value of $\cos\theta$ can be expressed as $\frac{m}{n}$, where m and n are positive integers with no common factors. Determine the sum $m+n$.

Table 10. Examples of Target, Similar, and Perturbed Questions

Settings	Content
Original Question	A plane contains points A and B with $AB = 1$. Point A is rotated in the plane counterclockwise through an acute angle θ around point B to point $A^{/prime}$. Then B is rotated in the plane clockwise through angle θ around point $A^{/prime}$ to point $B^{/prime}$. Suppose that $AB^{/prime} = \frac{4}{3}$. The value of $\cos\theta$ can be written as $\frac{m}{n}$, where m and n are relatively prime positive integers. Find $m+n$. Present the answer in LaTeX format: $\boxed{\text{Your answer}}$
Similar Question	A plane contains points A and B with $AB = 2$. Point A is rotated in the plane counterclockwise through an acute angle $theta$ around point B to point A'. Then B is rotated in the plane clockwise through angle $theta$ around point A' to point B'. Suppose that $AB' = 3$. The value of cos $theta$ can be written as $fracmn$, where m and n are relatively prime positive integers. Find $m+n$. Present the answer in LaTeX format: $\boxed{\text{Your answer}}$
Perturbed Question	A plane contains points A and B with $AB = 1$. Point A is rotated in the plane counterclockwise through an acute angle θ around point B to point A'. Then B is rotated in the plane clockwise through angle θ around point A' to point B'. Suppose that $AB' = \frac{4}{3}$. The value of $\cos\theta$ can be written as $\frac{m}{n}$, where m and n are relatively prime positive integers. Find $m+n$.
Perturbed Variant	In a plane, there are points A and B such that $AB = 1$. Point A is rotated around point B counterclockwise through an acute angle θ to reach point A'. Next, point B is rotated clockwise around point A' through the same angle θ, resulting in point B'. Given that $AB' = \frac{4}{3}$, express the value of $\cos\theta$ as $\frac{m}{n}$, where m and n are coprime positive integers. Determine $m+n$.

Table 11. Examples of Target, Similar, and Perturbed Questions in Variable Renaming Perturbation.

Settings	Content
Original Question	A plane contains points A and B with $AB = 1$. Point A is rotated in the plane counterclockwise through an acute angle θ around point B to point A'. Then B is rotated in the plane clockwise through angle θ around point A' to point B'. Suppose that $AB' = \frac{4}{3}$. The value of $\cos\theta$ can be written as $\frac{m}{n}$, where m and n are relatively prime positive integers. Find $m+n$. Present the answer in LaTeX format: $\boxed{\text{Your answer}}$
Similar Question	A plane contains points A and B with $AB = 2$. Point A is rotated in the plane counterclockwise through an acute angle θ around point B to point A'. Then B is rotated in the plane clockwise through angle θ around point A' to point B'. Suppose that $AB' = \frac{5}{4}$. The value of $\cos\theta$ can be written as $\frac{m}{n}$, where m and n are relatively prime positive integers. Find $m+n$.
Perturbed Question	A plane contains points A and B with $AB = 3$. Point A is rotated in the plane counterclockwise through an acute angle θ around point B to point A'. Then B is rotated in the plane clockwise through angle θ around point A' to point B'. Suppose that $AB' = \frac{7}{3}$. The value of $\cos\theta$ can be written as $\frac{m}{n}$, where m and n are relatively prime positive integers. Find $m+n$.
Perturbed Variant	In a plane, there are points A and B such that the distance AB equals 3. Point A is rotated counterclockwise by an acute angle θ around point B to reach point A'. Next, point B is rotated clockwise by angle θ around point A' to arrive at point B'. If the distance AB' is $\frac{7}{3}$, then the value of $\cos\theta$ can be expressed as $\frac{m}{n}$, where m and n are relatively prime positive integers. Calculate $m+n$.

Table 12. Examples of Target, Similar, and Perturbed Questions in Number Replacement Perturbation.

Settings	Content
Original Question	Isosceles triangle $\triangle ABC$ has $AB = BC$. Let I be the incenter of $\triangle ABC$. The perimeters of $\triangle ABC$ and $\triangle AIC$ are in the ratio $125:6$, and all the sides of both triangles have integer lengths. Find the minimum possible value of AB. Present the answer in LaTeX format: $\boxed{\text{Your answer}}$
Similar Question	Isosceles triangle $\triangle DEF$ has $DE = EF$. Let J be the incenter of $\triangle DEF$. The perimeters of $\triangle DEF$ and $\triangle DEJ$ are in the ratio $100:5$, and all the sides of both triangles have integer lengths. Find the minimum possible value of DE.
Perturbed Question	Isosceles triangle $\triangle DEF$ has $DE = EF$. Let J be the incenter of $\triangle DEF$. The perimeters of $\triangle DEF$ and $\triangle DEJ$ are in the ratio $100:5$, and all the sides of both triangles have integer lengths, as is common in many geometric problems. Find the minimum possible value of DE.
Perturbed Variant	In isosceles triangle $\triangle DEF$, the sides satisfy $DE = EF$. Denote the incenter of $\triangle DEF$ as point J. The ratio of the perimeters of $\triangle DEF$ to $\triangle DEJ$ is $100:5$, and all sides of both triangles are integers, which is typical in geometric scenarios. Determine the smallest possible value for DE.

Table 13. Examples of target, similar, and perturbed questions in distractor insertion perturbation.

C. Details of Curated Datasets

We provide examples of the curated evaluation and training dataset in Table 14 and Table 15.

D. Implementation Details

D.1. Training Details

We fine-tune the base model using Group Relative Policy Optimization (GRPO) within the VeRL (Sheng et al., 2024) training framework. Training is conducted for 2 epochs with a learning rate of 1×10^{-6}, train batch size 128, validation batch size 512, maximum prompt length 1024, and maximum response length 4096. We enable gradient checkpointing and dynamic batch sizing during optimization, with a per-GPU token budget of 16384 tokens. For rollout generation, we use vLLM (Kwon et al., 2023) with 4 sampled responses per prompt at temperature 1.0, while validation uses temperature 0.6. We do not apply explicit KL regularization during training. All training experiments are performed on $8\times$ NVIDIA A6000 GPUs.

D.2. Inference Details

We conduct two types of evaluation: reasoning evaluation and contamination evaluation.

Reasoning Evaluation. We conduct reasoning evaluation in Section E.1 to test training robustness. Here, we generate 5 sampled responses per example and report $\mathrm{pass@5}$, where a prediction is considered correct if at least one sampled response matches the ground-truth answer after extracting the final boxed or numeric answer. In addition, we compute the mean length-normalized token log-probability of the gold answer under the model:

$$\frac{1}{T}\sum_{t=1}^{T}\log p(a_t \mid \mathrm{prompt}, a_{<t}),$$

where T denotes the answer length. Log-probabilities are computed using either a standard Transformers forward pass or vLLM-based prompt log-probability scoring.

Contamination Evaluation. Contamination evaluation follows a two-stage pipeline. First, we generate model responses together with token-level statistics such as log-probabilities and entropies. Second, we compute contamination detection scores using both output-based and representation-based methods.

The evaluated baselines include Min-K%, PPL, CDD, Recall, and Self-Critique. For representation-based methods, LaRA constructs semantically related and perturbed variants of each question using gpt-4o-mini-generated (via OpenRouter API) (OpenAI, 2024; OpenRouter, 2024) paraphrases and incomplete-question variants, from which representation-level signals such as RSI/RSM and directional collapse are derived.

We evaluate member versus non-member separability using AUROC, TPR at fixed FPR (0.05). All inference and evaluation experiments are performed on $4\times$ NVIDIA RTX 3090 GPUs.

Split	Source	Composition		Purpose
		Mem.	Non-Mem.	
Evaluation	Open-source + AIME'26	30	30	Evaluation
Training	30 members + 970 RL-MIA	30	970	RL Analysis

Table 14. **Overview of the curated datasets.**

data_source	prompt	answer	member	metadata
Contamination evaluation samples				
aime26	[{role: system, content: When tackling complex reasoning tasks, you have access to actions such as ASSESS, ADVANCE, VERIFY, SIMPLIFY, SYNTHESIZE, PIVOT, and OUTPUT.}, {role: user, content: Patrick started walking at a constant rate from school to the park. Tanya ran 2 miles per hour faster than Patrick, Jose bicycled 7 miles per hour faster than Tanya, and all three arrived at the same time. Find m+n.}]	277	0	–
numina_amc_aime	[{role: system, content: When tackling complex reasoning tasks, you have access to actions such as ASSESS, ADVANCE, VERIFY, SIMPLIFY, SYNTHESIZE, PIVOT, and OUTPUT.}, {role: user, content: The highest price is $8.50 and the lowest price is $5.50. Calculate the percent by which the highest price is more than the lowest price.}]	70%	1	–
RL training sample				
olympiads	[{role: system, content: Your task is to follow a systematic, thorough reasoning process before providing the final solution. Structure your response into two sections: Thought and Solution. In the Thought section, present your reasoning using <think>{thoughts}</think>. In the Solution section, provide the final logical answer, optionally in \boxed{} format.}, {role: user, content: Determine the largest even positive integer which cannot be expressed as the sum of two composite odd positive integers.}]	38	–	{style: "rule"}

Table 15. **Samples from the curated contamination evaluation and RL training datasets.** Each instance contains the data source, structured conversational prompt, ground-truth answer, membership label when applicable, and metadata annotations.

Model	Overall Pass@5	Pass@5 (%)		LogP(answer | prompt)	
		Member	Non-member	Member	Non-member
lightblue **Qwen2.5-Math-7B (Base)**	15.0 (9/60)	26.7 (8/30)	3.3 (1/30)	-2.08	-2.44
Eurus-2-7B-PRIME (Initial)	18.3 (11/60)	33.3 (10/30)	3.3 (1/30)	-8.54	-7.88
Eurus-2-7B-PRIME (Epoch 1)	18.3 (11/60)	36.7 (11/30)	0.0 (0/30)	-9.06	-8.50
Eurus-2-7B-PRIME (Epoch 2)	18.3 (11/60)	36.7 (11/30)	0.0 (0/30)	-8.85	-9.12
lightblue **Qwen2.5-Math-7B (Base)**	10.0 (6/60)	20.0 (6/30)	0.0 (0/30)	-2.07	-2.44
LIMR (Initial)	8.0 (5/60)	17.0 (5/30)	0.0 (0/30)	-2.48	-2.92
LIMR (Epoch 1)	13.3 (8/60)	23.3 (7/30)	3.0 (1/30)	-2.44	-2.90
LIMR (Epoch 2)	10.0 (6/60)	20.0 (6/30)	0.0 (0/30)	-2.43	-2.87

Table 16. **Comparison between the RL-trained open-source model and its base model on member and non-member samples.**

E. Validation on Training Setup

Before analyzing contamination-related representation dynamics, we first validate whether the GRPO-based RL post-training setup induces meaningful policy adaptation. Since our primary goal is to study how contamination signatures emerge after RL post-training rather than to optimize the RL algorithm itself, we focus on verifying whether the trained checkpoints exhibit non-trivial optimization and behavioral changes compared to the initial model. We use three checkpoints corresponding to different stages of RL post-training: the initial model before additional training (**epoch 0**), the checkpoint after the first training epoch (**epoch 1**), and the checkpoint after the second training epoch (**epoch 2**).

E.1. Performance Evaluation of Trained Models on Member vs. Non-member

We further evaluate whether RL post-training induces different behaviors on member and non-member samples by comparing answer accuracy and token-level confidence between the two groups.

Evaluation setup. We evaluate the RL-trained open-source model `Eurus-2-7B-PRIME` and its corresponding base model `Qwen2.5-Math-7B`. Following prior contamination analyses, we partition evaluation samples into *member* and *non-member* subsets based on whether the underlying samples originate from the training distribution used during RL post-training. For each sample, we compute: (i) **Pass@5**, which measures whether the correct answer appears among five sampled generations, and (ii) the length-normalized token-level log-probability of the generated answer conditioned on the prompt in Section D.2.

Results. Table 16 shows that the RL post-training setup induces meaningful behavioral changes across checkpoints. For `Eurus-2-7B-PRIME`, overall Pass@5 improves from 15.0% in the base model to 18.3% after RL post-training, indicating successful policy adaptation. Moreover, member samples consistently achieve substantially higher Pass@5 than non-member samples across all checkpoints, with the gap further increasing during training. In particular, member Pass@5 rises from 33.3% at initialization to 36.7% after RL training, while non-member performance drops to 0.0%.

A similar trend is observed for `LIMR`. Although overall Pass@5 does not monotonically improve across all checkpoints, the RL-trained checkpoints still consistently achieve higher performance on member samples than on non-member samples. For example, Epoch 1 improves overall Pass@5 from 10.0% to 13.3%, while maintaining a clear member–non-member performance gap (23.3% vs. 3.3%). Even in checkpoints where overall performance remains similar to the base model, member samples remain substantially easier for the trained model than non-member samples.

Overall, these results confirm that the RL post-training procedure induces non-trivial optimization and measurable behavioral changes, while also revealing a consistent preference toward member samples across checkpoints.

F. Additional Results on Figure Geometry

Figure 7 further shows that the layer-wise representation geometry patterns vary across RL-trained models while still exhibiting contamination-associated deviations. In `Eurus-2-7B-PRIME`, contaminated samples exhibit persistently elevated RSM values throughout depth with substantially larger magnitudes than clean samples, indicating strong perturbation sensitivity under information removal. Regarding DC, while the specific directional profiles vary across models and checkpoints, contaminated samples consistently exhibit deviations from the stable directional organization observed in clean samples, indicating disrupted and lower-dimensional perturbation dynamics. Exhibit comparatively lower RSI values than clean samples, suggesting induced local representation flexibility under paraphrastic perturbations.

In contrast, `Olmo-3.1-7B-RL-Zero-Math` exhibits weaker RSM separation, with both contaminated and clean samples remaining near zero across most layers. However, contamination-related deviations remain observable in DC and RSI, where contaminated samples maintain consistently higher directional concentration and smoother local variability patterns than clean samples. Overall, these results suggest that while the precise geometric behaviors vary across models, contamination consistently alters perturbation sensitivity, directional organization, and local representation variability.

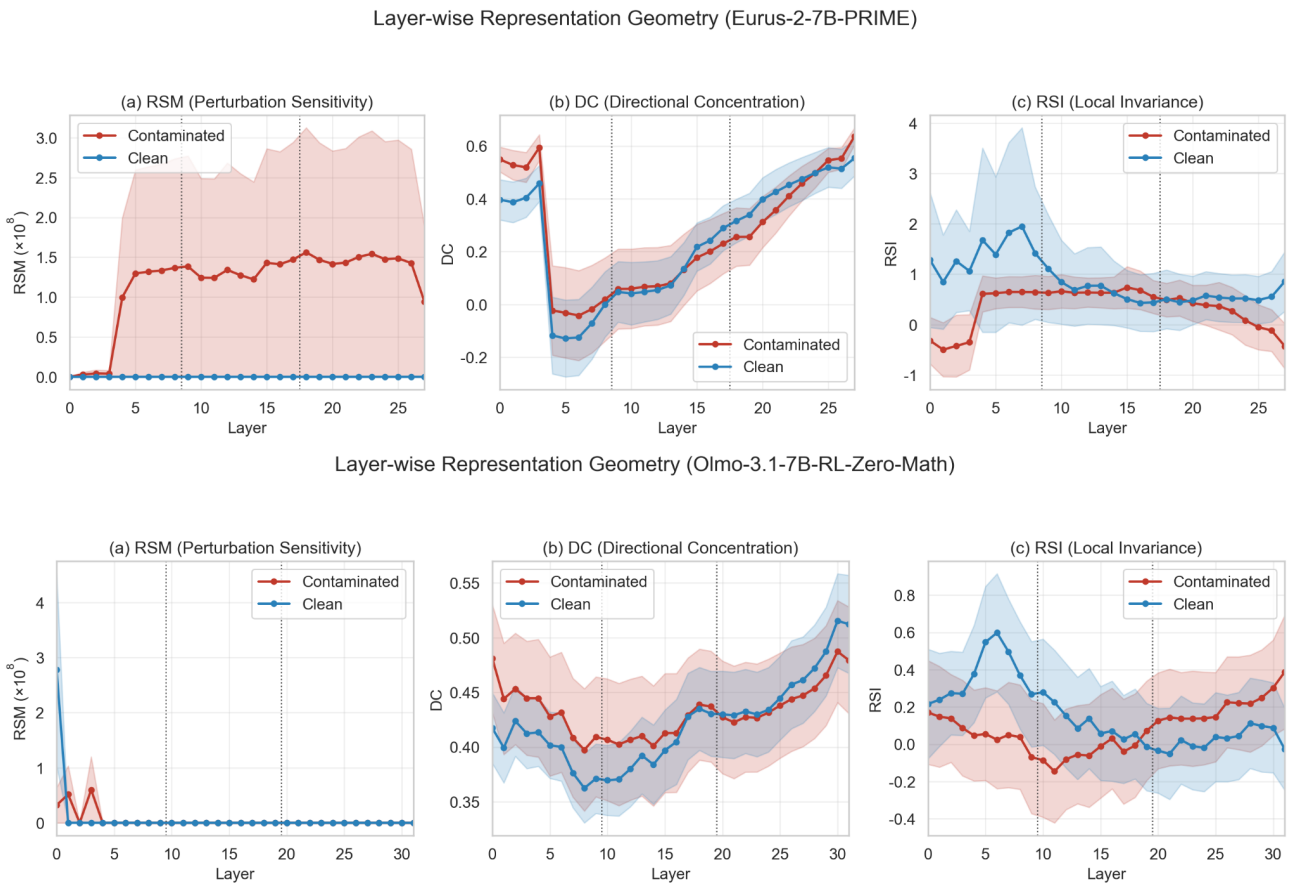

Figure 7. **Representation Geometry of Additional Models.**

G. Further Analyses

G.1. Analyses of Layer-window Trends on Different Models

Analysis on `Eurus-2-7B-Prime`. Figure 9 shows contaminated-vs-clean separation (Cohen's d) across early (0–8), mid (9–17), and late (18–27) layers over three RL checkpoints. Three consistent trends emerge. First, **RSM remains nearly unchanged across both depth and training**. All checkpoints show stable positive separation ($d \approx 0.27$–0.35), indicating that perturbation sensitivity is largely preserved throughout RL fine-tuning. Second, **DC becomes increasingly negative as RL training progresses**, especially in deeper layers. While the initial checkpoint shows weak or near-zero separation, Epoch 1 and Epoch 2 exhibit progressively larger negative effects, with the strongest separation appearing in late layers ($d \approx -0.65$ at Epoch 2). This suggests that RL training amplifies directional-collapse behavior on contaminated samples, particularly in deeper representations. Third, **RSI is strongest in early and mid layers but weakens in late layers after training**. Early-layer separation remains consistently negative across checkpoints, whereas the late-layer signal gradually diminishes and nearly disappears by Epoch 2. This indicates that local-invariance differences are primarily concentrated in shallower representations. Overall, the figure reveals a clear layer-conditioned structure: RSM is stable across training, DC is progressively amplified by RL in deeper layers, and RSI is concentrated in earlier layers. These complementary trends motivate combining all three metrics in LaRA rather than relying on a single layer-wise signal.

Analysis on `LIMR`. Figure 10 shows contaminated-vs-clean separation (Cohen's d) across early (0–8), mid (9–17), and late (18–27) layers over three RL checkpoints. Several distinct trends emerge. First, **RSM remains consistently positive across all layer groups**, indicating that contaminated samples exhibit larger representation shifts under perturbation throughout training. However, unlike the relatively stable behavior observed in `Eurus-2-7B-Prime`, the magnitude of separation decreases after RL training, particularly in deeper layers. Early and mid layers initially show strong separation ($d \approx 0.43$), but this gradually weakens by Epoch 2, especially in the late layers where the effect becomes nearly negligible. This suggests that RL training partially suppresses perturbation-sensitive geometry in LIMR. Second, **DC exhibits the strongest and most stable separation across all checkpoints**. All layer groups consistently show large negative effects ($d \approx -0.85$ to -1.25), with mid and late layers displaying the strongest separation. Moreover, RL training slightly amplifies this negative separation in Epoch 1 and Epoch 2, indicating that contaminated representations become increasingly directionally collapsed during RL optimization. Compared to the other metrics, DC provides the clearest and most persistent distinction between contaminated and clean samples across depth. Third, **RSI shows progressively stronger positive separation as RL training proceeds**, particularly in mid and late layers. While the initial checkpoint exhibits only moderate separation, Epoch 2 produces substantially larger effects ($d \approx 0.40$–0.45), especially beyond the middle layers. This indicates that RL training amplifies local-invariance differences between contaminated and clean samples, causing contaminated representations to become increasingly invariant under local perturbations. Overall, LIMR exhibits a complementary layer-wise structure distinct from `Eurus-2-7B-Prime`: RSM weakens during RL training, DC remains consistently dominant across all depths, and RSI becomes progressively amplified in deeper layers. These trends suggest that RL optimization in LIMR increasingly concentrates contaminated representations into geometrically collapsed yet locally invariant structures, motivating the joint use of RSM, DC, and RSI in LaRA for robust contamination detection.

G.2. Aggregation over Different Layer-windows.

Figure 11 compares contamination detection performance when aggregating representation statistics over different layer windows (Early, Mid, and Late) as well as over all layers jointly. We report both AUC and TPR@FPR=5% across three RL-trained models. A clear and consistent trend emerges: **the relative contamination-separation behavior remains largely stable across different layer regions**.

Across all models, the performance ordering is highly consistent regardless of which layer window is used. LIMR consistently achieves the strongest separation across every window, obtaining nearly identical AUC values (≈ 0.8) for Early, Mid, and Late layers, along with the highest TPR values throughout. EURUS exhibits moderate but stable performance across all windows, while OLMO consistently shows weaker separation. Importantly, no single layer region dominates the results; instead, contamination-related geometric signals persist throughout the network depth. The results demonstrate that contamination-related representation geometry is remarkably stable across layer depth. Rather than arising from isolated layers, the separation signal persists throughout the network, motivating the use of layer-window aggregation as a robust and model-agnostic strategy for contamination detection.

G.3. Metric Ablations on Additional Model

We also conduct an additional ablation study on another model, `Eurus-2-7B-Prime` in Table 17. The results show that S_{LaRA} is relatively robust to ablations, as shown in `LIMR` results in the main ablation studies.

G.4. Analysis on the Evolution of Representation Geometry across RL Training Stages

Figure 8 illustrates how RL training progressively reshapes the layer-wise representation geometry of contaminated samples relative to clean samples in `Eurus-2-7B-PRIME`. Across all three metrics, the separation between clean and contaminated trajectories becomes increasingly pronounced as training advances from the initial checkpoint to Epochs 1 and 2. In particular, **RSM** exhibits the strongest divergence, where contaminated samples show dramatically amplified sensitivity beginning in early-middle layers and remaining consistently elevated throughout deeper layers, while clean samples stay nearly flat across all stages. **DC** further reveals that RL training induces increasingly distinct directional structure: contaminated samples transition from negative or near-zero values in middle layers to strongly positive values in deeper layers, whereas clean samples maintain smoother and more stable trajectories. Similarly, **RSI** demonstrates that contaminated samples undergo substantial geometric instability during RL training, especially in shallow layers where sharp spikes emerge after training, while clean samples retain comparatively moderate and stable behavior. Importantly, these trends remain consistent across layer depth and training stages, suggesting that RL fine-tuning systematically amplifies hidden-state geometric irregularities associated with memorized or contaminated data rather than producing isolated layer-specific effects.

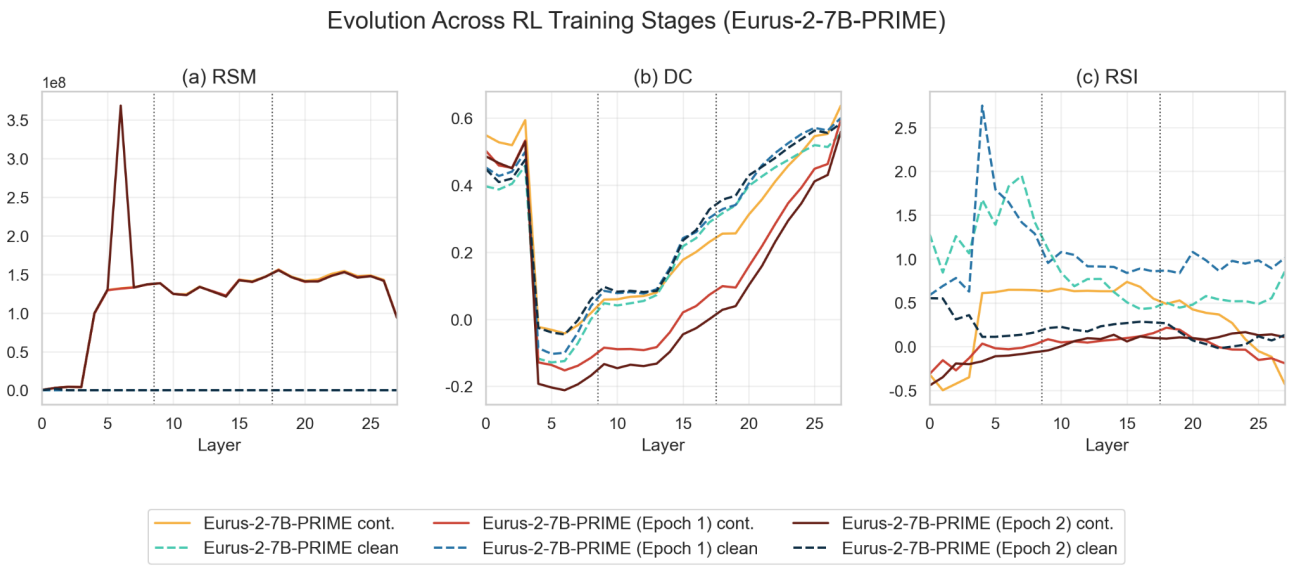

Figure 8. **Evolution of layer-wise representation geometry across RL training stages.** RL progressively amplifies geometric deviations in each metric (**RSM, DC, and RSI**) between clean and contaminated samples over epochs.

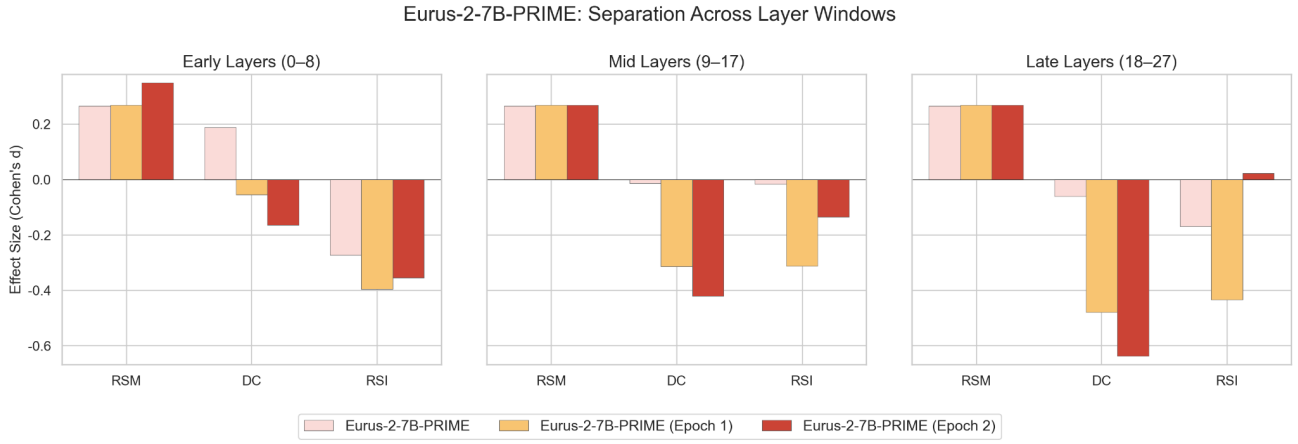

Figure 9. **Effect-size separation across early, mid, and late layer windows of `Eurus-2-7B-PRIME`.** Contaminated samples are consistently elevated on RSM but progressively lower on DC and RSI in mid-to-late layers, with the widened gap indicating that RL training amplifies a layer-selective representational signature of contamination.

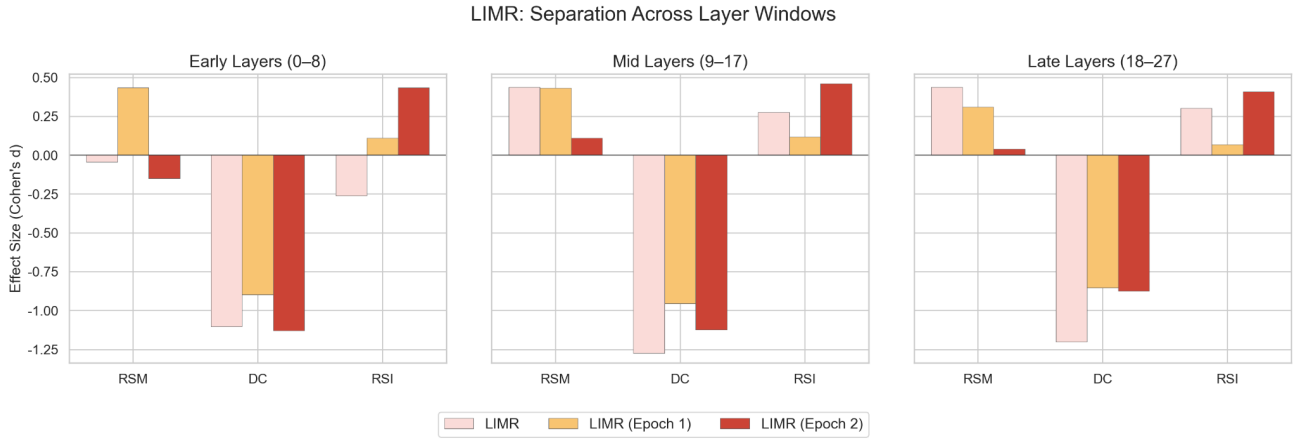

Figure 10. **Effect-size separation across early, mid, and late layer windows of `LIMR`.** Contaminated samples exhibit consistently lower DC across all layer windows, while RSI becomes increasingly positive in mid-to-late layers as RL training progresses. In contrast to Eurus-2-7B-PRIME, the separation dynamics in LIMR are dominated by strong directional concentration differences together with progressively amplified RSI gaps, indicating that RL training induces a distinct layer-dependent representation geometry signature of contamination in LIMR.

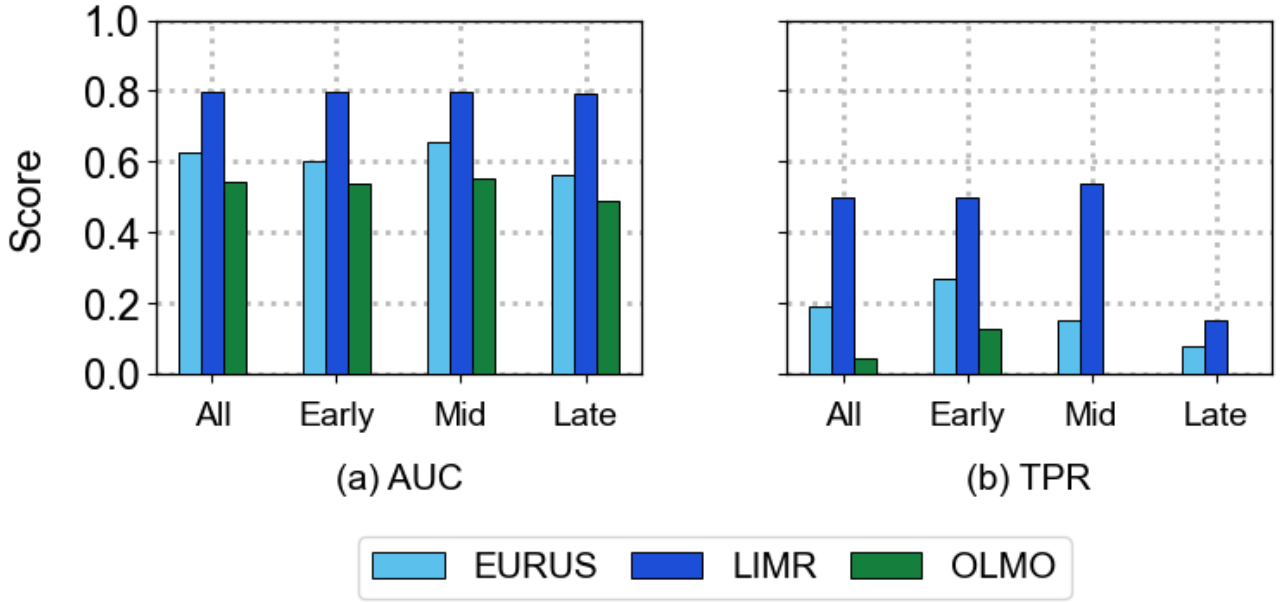

Figure 11. **Results over different layer windows on three models**

RSM	DC	RSI	Init		E1		E2	
			AUC	TPR	AUC	TPR	AUC	TPR
55	55	51	0.59	0.19	0.49	0.00	0.54	0.08
55	51	55	0.68	0.12	0.74	0.35	0.62	0.04
51	55	55	0.59	0.27	0.53	0.23	0.52	0.27
55	51	51	0.68	0.19	0.69	0.35	0.61	0.08
51	55	51	0.62	0.23	0.54	0.23	0.56	0.23
51	51	55	0.67	0.23	0.73	0.15	0.62	0.19
lightblue 51	51	51	0.68	0.19	0.70	0.15	0.63	0.19

Table 17. **Ablations on Eurus across RL epochs.**

H. Justifications of the Scaling Factor Metric in Contamination Detection Protocol

The factor $1.4826 = 1/\Phi^{-1}(0.75)$ is the standard Fisher-consistency constant used in robust statistics to make the median absolute deviation (MAD) a consistent estimator of the standard deviation under a Gaussian reference (Hampel et al., 1986; Huber & Ronchetti, 2009; Rousseeuw & Croux, 1993). Specifically, if $X \sim \mathcal{N}(\mu, \sigma^2)$, then $\operatorname{median}|X - \mu| = \sigma\,\Phi^{-1}(0.75)$, so multiplying the empirical MAD by $1/\Phi^{-1}(0.75) \approx 1.4826$ recovers σ in the large-sample limit.

Two properties of this scaling are particularly important for our protocol. First, it places the robust scale estimate on the same numerical units as the ordinary sample standard deviation, so the standardized deviations $z_{m,\ell}(x)$, the metric-specific alignments $\hat{z}_{m,\ell}(x)$, and the aggregated score $S_{\mathrm{LaRA}}(x)$ retain their interpretation as approximate Gaussian-style z-scores. Consequently, replacing the standard deviation with a robust scale estimator does not implicitly retune downstream thresholds or alter the semantic interpretation of the score.

Second, unlike the sample standard deviation, which has a 0% breakdown point and can be driven arbitrarily large by a single extreme outlier, the MAD achieves a 50% breakdown point and therefore remains stable even when the clean-reference pool $\mathcal{D}^{\mathrm{clean}}$ contains a substantial fraction of atypical samples (Hampel et al., 1986; Rousseeuw & Leroy, 1987). This is particularly relevant for representation-geometry metrics, whose raw distributions are often heavy-tailed even after signed-$\log(1 + |\cdot|)$ compression. The resulting robustness-efficiency trade-off is well established in the robust statistics literature: while the MAD is less asymptotically efficient than the sample standard deviation under perfectly Gaussian noise, it provides substantially improved stability under contamination and heavy-tailed deviations (Huber & Ronchetti, 2009; Maronna et al., 2019).

I. Metrics and Baselines

I.1. Metrics

ROC-AUC measures the model's ability to distinguish between member and non-member samples across all possible decision thresholds. It captures the overall separability of the two classes and is threshold-independent, making it a robust indicator of detection quality. **TPR@FPR=5%** reports the true positive rate (i.e., correctly identified members) when the false positive rate (i.e., non-members incorrectly flagged as members) is fixed at 5%. This reflects performance in the low false-positive regime, which is critical in contamination detection where incorrectly labeling clean samples as contaminated is costly. These metrics provide a comprehensive evaluation of global discrimination ability (ROC-AUC) and practical operating performance under strict error constraints (TPR@FPR=5%).

I.2. Baselines

We use six baselines to compare against our contamination detection protocol. (1) Recall (Xie et al., 2024), which probes memorization by measuring the model's ability to regenerate ground-truth answers under controlled prompting; (2) CDD (Dong et al., 2024), which detects contamination via discrepancies in model predictions under input or prompt perturbations, based on the intuition that memorized samples are less sensitive to such changes; (3) Min-K% Prob (Shi et al., 2023), a likelihood-based metric that averages the log-probability over the lowest-probability tokens in a sequence, assuming memorized samples exhibit fewer low-confidence tokens; (4) Min-K%++ (Zhang et al., 2024), which extends Min-K% with improved normalization and calibration for greater robustness across settings; (5) PPL (Gonen et al., 2023), which measures sequence-level likelihood via perplexity, where unusually low values indicate potential memorization; and (6) Self-Critique (Tao et al., 2025), which leverages the model's own reflective reasoning to assess contamination based on the confidence and consistency of its self-evaluation.

J. Usage of AI Assistants

In preparing this work, we used AI-based writing assistants to improve sentence structure, correct grammatical errors, and enhance overall readability. These tools were employed solely for language refinement and did not contribute to the development of technical content, research methodology, or experimental analysis. All scientific ideas, results, and conclusions presented in the paper were conceived and authored entirely by the researchers. Use of AI assistance was restricted to editorial purposes and did not affect the originality or intellectual contributions of the work.