LEARNING TO DETECT TRAINING DATA VIA ACTIVE RECONSTRUCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Detecting LLM training data is generally framed as a membership inference attack (MIA) problem. However, most MIAs operate passively on fixed model weights, using log-likelihoods or text generations. In this work, we introduce **Active Data Reconstruction Attack** (ADRA), a family of MIAs that actively induces a model to reconstruct a given text through training. We hypothesize that training data are *more reconstructible* than non-members, and the difference in their reconstructibility can be exploited for membership inference. Motivated by findings that reinforcement learning (RL) sharpens behaviors already encoded in weights, we leverage on-policy RL to actively elicit data reconstruction by finetuning a policy initialized from the target model. To effectively use RL for MIA, we design reconstruction metrics and contrastive rewards. The resulting algorithms, ADRA and its adaptive variant ADRA+, improve both reconstruction and detection given a pool of candidate data. Across pre-training, post-training, and distillation, our best method improves over the strongest available loss-based baseline by 11.2 AUROC points on average across 14 evaluation settings, and the strongest of all evaluated baselines by 6.8 points. As a highlight, ADRA+ improves over Min-K%++ by 18.8 points on BookMIA and 7.6 points on AIME.

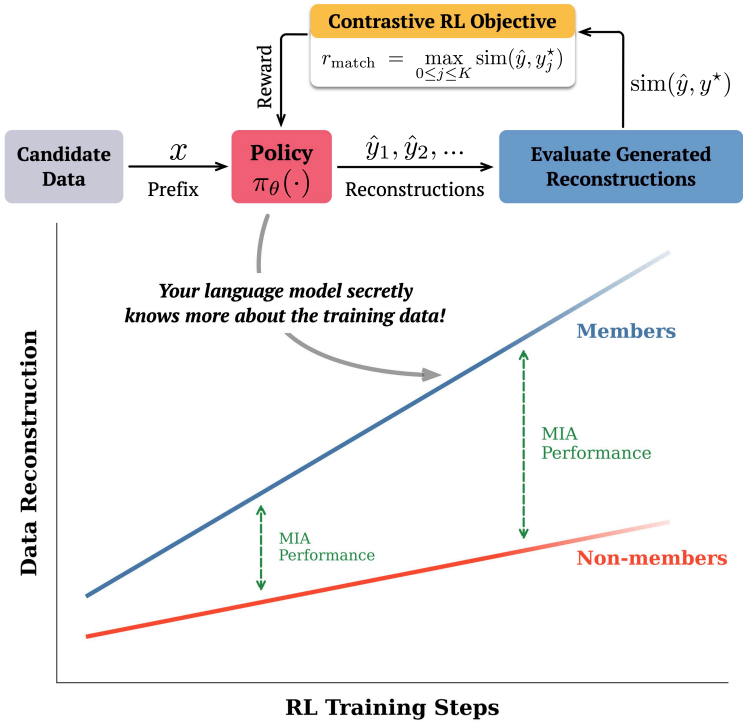


Figure 1: **Active Data Reconstruction Attack.** Language model generates reconstructions from a candidate prefix and is rewarded via a contrastive objective. Members become easier to reconstruct than non-members over RL training, improving MIA performance.

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.

1 INTRODUCTION

Modern language models are known to memorize and regurgitate training data, raising concerns about copyright, privacy, and data contamination (Carlini et al., 2021; Ahmed et al., 2026). Membership inference attacks (MIAs) are methods that detect whether a given input was present in the training data (Shokri et al., 2017), and have shown some success when applied to LLMs (Shi et al., 2023).

Most prior MIAs query the target model without modifying its weights, *passively* extracting signals exposed through the output layer—such as loss (Shokri et al., 2017; Yeom et al., 2018; Carlini et al., 2021), log-probabilities (Shi et al., 2023; Zhang et al., 2024b), or text generations (Hallinan et al., 2025). We hypothesize that model weights encode *latent membership signal* that passive methods fail to reliably expose. If such a signal exists, it could be *actively* elicited through model weight updates.

We introduce **Active Data Reconstruction Attack (ADRA)**, a family of MIAs that uses Reinforcement Learning (RL) with reconstruction rewards to elicit latent membership signals. Because model weights encode traces of training data, RL training exploits the difference in member and non-member reconstructibility for detection. Our approach is motivated by recent findings that RL sharpens behaviors already encoded in weights (Yue et al., 2025; Wang et al., 2025) while transferring minimal new information (Schulman & Lab, 2025), suggesting RL is well-suited to surface latent reconstruction signal.

Figure 1 illustrates our attack: given a candidate prefix, the model generates continuations (*reconstructions*) and is rewarded for producing generations similar to the candidate suffix. To effectively leverage RL for MIA, we design reconstruction metrics and contrastive rewards that match generations against a pool containing the true suffix and negative distractors. This yields two algorithms: ADRA, which rewards the best match among the pool, and ADRA+, which adaptively adjusts how often the true suffix should be included during matching based on a prior derived from loss-based scores.

Empirically, we run comprehensive experiments spanning pre-training, post-training, and distillation settings across 6 open-weight LLMs. To better match the latest models' knowledge cutoffs and heavier post-training, we construct 6 new MIA datasets and evaluate on 2 established benchmarks. We find that ADRA and ADRA+ outperform prior MIAs on most evaluated rows, with an average improvement of 11.2 AUROC points over the strongest available loss-based baseline across 14 dataset/variant rows (6.8 over the strongest of all evaluated baselines, including our proposed baseline N-Sampling). For pre-training data detection, on the challenging WikiMIA_{2024} Hard benchmark—where most baselines hover below random—ADRA+ reaches 60.6% AUROC, outperforming Min-K%++ by 10%. For post-training, under controlled contamination on AIME, ADRA+ achieves 85.9% AUROC, improving over N-Sampling by 13.2% and Min-K%++ by 7.6%. For distillation, on prefix-matched distillation traces, ADRA reaches 98.4% under DeepSeek-R1 distillation.[1] Ablations confirm that RL-based optimization, reconstruction-reward design, and the contrastive objective are critical to these gains.

Our results suggest that model weights encode more about training data than fixed outputs reveal. RL training helps surface this latent information, enabling stronger reconstruction and membership inference. Code, models, and datasets will be released.

2 BACKGROUND

Membership inference (Shokri et al., 2017) aims to determine if a datapoint x was present in the training data of model θ. We focus on the specific problem of detecting training data from LLMs, which is known to be difficult (Duan et al., 2024).

Notation. Let p_θ be the target LLM with parameters θ, trained on a dataset $\mathcal{D}_{train}$. Given a candidate document $x \in \mathcal{D}_{cand}$, write $x = (c, y^\star)$ where c is the observed prefix and $y^\star$ is the held-out suffix. Membership inference is the binary classification problem of determining whether x was drawn from $\mathcal{D}_{train}$ or not. Define a scoring function $f(x;\theta)$ that maps each candidate to a scalar membership score; MIAs predict "member" iff $f(x;\theta) \geq \epsilon$ for a threshold ϵ.

[1] This is a paired trace-discrimination setting in which two candidate suffixes share a prompt, not arbitrary unpaired MIA.

Loss-based Methods. Most prior MIAs score a candidate by the model's token-level log-probabilities on x, with the intuition that models are likely to have a lower loss on sequences that have been seen during training:

- *Loss* (Yeom et al., 2018): $f(x) = \mathcal{L}(x;\theta)$.
- *R-Loss* (Carlini et al., 2021): calibrate by a reference model θ_{ref}, $f = \mathcal{L}(x;\theta) - \mathcal{L}(x;\theta_{\text{ref}})$.
- *Zlib* (Carlini et al., 2021): normalize by the zlib compression of x, $f = \mathcal{L}(x;\theta)/\text{zlib}(x)$.
- *Min-K%* (Shi et al., 2023): average over the $k\%$ lowest-likelihood tokens, $f = \frac{1}{|k(x)|}\sum_{i\in\min\ k(x)} -\log p(x_i \mid x_{<i};\theta)$.
- *Min-K%++* (Zhang et al., 2024b): z-score each next-token log-prob by the vocab mean/std under θ, then average the $k\%$ smallest: $s_i = \frac{\log p(x_i|x_{<i};\theta) - \mu_{x_{<i}}}{\sigma_{x_{<i}}}$, $f = \frac{1}{|k(x)|}\sum_{i\in\min\ k(x)} s_i$.

Reconstruction-based Methods. Hallinan et al. (2025) propose sampling completions and scoring membership via n-gram metrics—effectively a reconstruction-based MIA. Formally, given a fixed prompt q and prefix c, we sample N completions $\{\hat{y}\}_{i=1}^{N}$ from p_θ with nucleus sampling:

$$\{\hat{y}^{(i)}\}_{i=1}^{N} \sim p_\theta(\cdot \mid q, c). \tag{1}$$

We score each candidate x by comparing each completion $\hat{y}^{(i)}$ to the ground truth suffix $y^\star$ using a similarity function $\text{sim}(\hat{y}^{(i)}, y^\star)$:

$$f(x) \;=\; \frac{1}{N}\sum_{i=1}^{N} \text{sim}\big(\hat{y}^{(i)}, y^\star\big). \tag{2}$$

3 Active Data Reconstruction Attack

The above-mentioned MIAs operate on fixed model weights, querying the target model's log-probabilities or generations. They assume that passive querying extracts all available membership signal. Conversely, we ask if training data leaves a *latent membership signal* in a model's parameters that is not captured by existing MIA methods. If such latent signal exists, we can design an *Active MIA* that actively elicits the latent signal rather than measuring it passively. Thus, we propose Active Data Reconstruction Attack (ADRA).

Model access. ADRA targets open-weight models or managed APIs supporting on-policy RL with custom rewards.[2] On our evaluated open-weight targets, ADRA and likelihood-based MIAs share the same model-availability boundary. *Note that closed-source model APIs provide neither the teacher-forced candidate likelihoods required by the likelihood-based MIAs nor ADRA's update capability.* We detail these assumptions in Section E.3.

3.1 RL-induced Data Reconstruction

Recent work shows that RL sharpens behaviors already encoded in weights (Yue et al., 2025; Wang et al., 2025) while transferring minimal new information (Schulman & Lab, 2025), suggesting RL could be well-suited to elicit latent membership signals. We leverage this insight by explicitly rewarding the model for reconstructing candidate data.

Concretely, we fine-tune a policy initialized from the target model p_θ with on-policy RL, reusing the similarity function from Equation (2) as the reward:

$$r^{(i)} \leftarrow \text{sim}(\hat{y}^{(i)}, y^\star), \quad \forall i \in \{1, ..., N\}. \tag{3}$$

For candidate x, the policy generates completions $\hat{y}$ given prefix (q, c) and receives a scalar reward $r(\hat{y}, x) = \text{sim}(\hat{y},\ y^\star)$. We update the policy by maximizing the expected return:

$$J_x(\theta) \;=\; \mathbb{E}_{\hat{y}\sim p_\theta(\cdot|q,c)}\big[r(\hat{y}, x)\big]. \tag{4}$$

We estimate Equation (4) with a Monte Carlo average over the N samples above. In practice we apply Group Relative Policy Optimization (GRPO) (Shao et al., 2024b), which optimizes a clipped/KL-regularized surrogate of Equation (4). After T on-policy RL updates, we obtain θ_T. To evaluate x, we resample M completions $\hat{y}^{(1:M)} \sim p_{\theta_T}(\cdot \mid c)$ and re-aggregate their similarities into a membership score. One shared policy is trained per dataset and seed over the unlabeled candidate pool, so the attack is transductive by design and never uses membership labels (Section C).

[2] Recent open-weight releases—Kimi K3, Qwen3.8, and Inkling—and API training services such as Tinker support these two assumptions (Qwen Team, 2026; Kimi Team, 2026; Thinking Machines Lab, 2026; 2025).

Algorithm 1 ACTIVE DATA RECONSTRUCTION ATTACK (one shared policy per dataset/seed).

Input: target LM p_θ, unlabeled candidate pool $\mathcal{D}_{cand} = \{x_j = (c_j, y_j^\star)\}_{j=1}^{J}$, (optional) prompt q, threshold ϵ, RL steps T, candidate minibatch size B, samples per candidate N, reward mode MODE $\in$ {MATCH, ADAPT}, similarity $\mathrm{sim}(\cdot,\cdot)$.

```
Initialize θ_0 ← θ
for t = 0 to T − 1 do
    Sample candidate minibatch B_t ⊂ D_cand, |B_t| = B
    for each x_j = (c_j, y_j^⋆) ∈ B_t do
        Sample rollouts ŷ_j^(1:N) ~ p_{θ_t}(· | q, c_j)
        Compute rewards r_j^(i) ← r_MODE(ŷ_j^(i), x_j)
    end for
    Update shared policy θ_{t+1} using all minibatch rewards
end for
for each x_j ∈ D_cand do
    Evaluate S(x_j) = (1/N) Σ_{i=1}^{N} sim(ŷ_j^(i), y_j^⋆) using new samples from p_{θ_T}(· | c_j)
end for
```

Output: membership scores $\{S(x_j)\}_{j=1}^{J}$, reconstructions, and decisions $\mathbb{I}\{S(x_j) \geq \epsilon\}$.

3.2 DATA RECONSTRUCTION METRICS

We instantiate reconstruction rewards with lexical similarity metrics $\mathrm{sim}(\hat{y}, y^\star)$ that measure the fraction of the reference suffix $y^\star$ reconstructed by a model's completion $\hat{y}$. Since RL algorithms suffer from sparse rewards (Sutton & Barto, 2018), we propose a variety of surrogate dense rewards that yield sufficient learning signals. All functions $\mathrm{sim}_\star$ share the property that if $y = y^\star$, $\mathrm{sim}_*(y, y^\star) = 1$.

Token Set Similarity (Morris et al., 2025b; Shao et al., 2024a). $\mathrm{sim}_{\text{tok}}(\hat{y}, y^\star) = \frac{|T(\hat{y}) \cap T(y^\star)|}{|T(y^\star)|}$, where $T(\cdot)$ denotes the set of unique tokens. This metric measures the fraction of unique reference tokens in $y^\star$ that appear in $\hat{y}$, emphasizing vocabulary recall and penalizing repetition via set overlap.

Longest Common Subsequence (Saggion et al., 2002; Lin, 2004). $\mathrm{sim}_{\text{lcs}}(\hat{y}, y^\star) = \frac{\mathrm{LCS}(\hat{y}, y^\star)}{|y^\star|}$. This rewards correct content in the correct order.

N-gram Set Coverage (Hallinan et al., 2025). $\mathrm{sim}_{\text{ng}}(\hat{y}, y^\star) = \frac{|\mathcal{N}_{[L_{\min}, L_{\max}]}(\hat{y}) \cap \mathcal{N}_{[L_{\min}, L_{\max}]}(y^\star)|}{|\mathcal{N}_{[L_{\min}, L_{\max}]}(y^\star)|}$, where $\mathcal{N}_{[L_{\min}, L_{\max}]}(\cdot)$ denotes the set of contiguous n-grams with $n \in [L_{\min}, L_{\max}]$. Treating n-grams as a set discourages reward hacking via repetition. Following previous works, we fix $L_{\min} = 3$ and set $L_{\max} = \mathrm{LCS}(\hat{y}, y^\star)$.

3.3 CONTRASTIVE REWARDS

Directly maximizing Equation (3) provides strong suffix supervision, obscuring whether post-update separation reflects an asymmetry at initialization or attack-time learning. To constrain supervision, we use a *contrastive* reward: each rollout $\hat{y}$ is matched against a pool of reference suffixes $\{g_j\}_{j=0}^{K}$ containing the true suffix $g_0 = y^\star$ and K distractors sampled from other candidates, with the best match as reward.

Contrastive rewards provide relative supervision over the true suffix and negative distractors, reducing direct supervision toward the true suffix compared with Equation (3). We propose two contrastive reward variants that form our methods: matching (ADRA) and adaptive matching (ADRA+).

Matching (ADRA). For each candidate $x = (c, y^\star)$, set $g_0 := y^\star$ and sample K additional distractor suffixes $\{g_j\}_{j=1}^{K}$ i.i.d. from $\mathcal{D}_{cand}$. Given a rollout $\hat{y} \sim p_\theta(\cdot \mid q, c)$, we compute similarities

$$\mathrm{sim}(\hat{y}, g_j) \in [0, 1], \qquad j = 0, \ldots, K,$$

using the reconstruction metrics from Section 3.2. The *matching* reward then scores $\hat{y}$ against all suffixes and takes the maximum:

$$r_{\text{match}}(\hat{y}, x) \;=\; \max_{0 \leq j \leq K} \mathrm{sim}(\hat{y}, g_j). \tag{5}$$

Because member data are seen during the model's training, their rollouts are more likely to match g_0 during matching $\max_{0 \leq j \leq K} \mathrm{sim}(\hat{y}, g_j)$. We term this method ADRA.

Task	Dataset	Dataset Cutoff	Model	Model Cutoff	Max Tokens
Pre-training	BookMIA	2023	Llama2-7B	2022*	1024
	WikiMIA$_{2024}$ Hard	2024	Qwen2-7B	2023*	1024
	Dolma3♠ (Olmo3 Mix)	2025	Olmo3-7B	2024	2048
Post-training	AIME♠	2024	Tulu2-7B	2023	4096
	Olympia Math♠	2025	Tulu2-7B	2023	4096
	Tulu3 Mix♠	2024	Tulu3-8B	2024	1024
Distillation	S1 / S1.1♠	2025	Qwen2.5-7B	2024*	2048

Table 1: **Datasets, models, cutoffs, and maximum sequence lengths for pre-training, post-training, and distillation settings.** Newly constructed MIA datasets are marked with ♠. Cutoff dates marked with * are estimated.

Adaptive Matching (ADRA+). Matching can be further sharpened by incorporating a membership prior $p(x) \in [0,1]$[3], derived from the base model's log-probabilities (e.g., Min-$K\%$ or Min-$K\%^{++}$ scores). Prior scores are calibrated using scores from the unlabeled candidate pool, without membership labels; see Section C.4. For simplicity, we treat all loss-based variants as $p(x)$ for notation.

For each x we have the same pool of references $\{g_j\}_{j=0}^{K}$ and scores $\{\mathrm{sim}_j\}_{j=0}^{K}$ and define

$$c_{\text{union}} := \max_{0 \le j \le K} \mathrm{sim}(\hat{y}, g_j), \qquad c_{\text{distract}} := \max_{1 \le j \le K} \mathrm{sim}(\hat{y}, g_j).$$

The adaptive reward r_{adapt} draws a switch $z \sim \mathrm{Bernoulli}(p(x))$ and returns c_{union} if $z = 1$ (true suffix + distractors) or c_{distract} if $z = 0$ (distractors only). In practice, for a fixed $(\hat{y}, x)$, we compute the expected reward:

$$\mathbb{E}_z[r_{\text{adapt}}(\hat{y}, x)] = p(x)\, c_{\text{union}} + (1 - p(x))\, c_{\text{distract.}} \tag{6}$$

Intuitively, when $p(x)$ is high (likely member), we often include g_0, making it easier for RL to reinforce reconstruction of the true suffix. When $p(x)$ is low, we often match only against distractors, receiving weaker and noisier supervision. We term this method ADRA+.

For both variants, the underlying RL algorithm (GRPO) is unchanged: we replace $r(y, x)$ in (4) with either r_{match} or r_{adapt}. Because our methods perform on-policy updates, they are more compute-intensive than passive MIAs. However, this attack-time model-update compute introduces a compute-to-performance scaling: detection generally improves with more RL steps and remains strong under strict budgets. We quantify this compute scaling in Section D.

4 Experimental Setup

As shown in Table 1, we evaluate membership inference on LLMs spanning three training stages: pre-training, post-training, and distillation. See Section A for full details.

Pre-training. Following Hallinan et al. (2025), we assess pre-training data detection on *BookMIA* and *WikiMIA$_{2024}$-Hard*. BookMIA (Shi et al., 2023) contains 512-word snippets sampled from 100 books, and non-members come from books after 2023. WikiMIA$_{2024}$-Hard (Hallinan et al., 2025) pairs Wikipedia article versions across date cutoffs and minimizes temporal distribution shift between members and non-members. We use Llama2-7B and Qwen2-7B—their knowledge cutoff aligns well with the dataset cutoffs. In addition, we evaluate the fully-open language model Olmo3 (Olmo et al., 2025) on its pre-training mix Dolma3.

Post-training. Post-training has become a growing focus of LLM development (Guo et al., 2025). We study two settings: (i) *Controlled contamination:* we simulate test-set contamination by SFT'ing on member examples from AIME and Olympia Math. (ii) *Post-training mixtures:* we evaluate membership on the open Tulu3 post-training mix on the Tulu3 model across domains (Lambert et al., 2024).

Distillation. Finally, we study data detection under model distillation. Following Muennighoff et al. (2025), we construct reasoning traces from Deepseek-R1 and Gemini-2.0-Flash.

[3]We define $p(x)$ such that larger values indicate higher membership likelihood.

		Active MIAs		Passive MIAs					
Model	**Type**	**ADRA+**	**ADRA**	**N-Sampling**	**Loss**	**R-Loss**	**Zlib**	**Min-K%**	**Min-K%++**
BookMIA									
Llama2-7B	Orig.	**78.4**	**76.2**	73.6	60.6	N/A	50.4	63.3	59.6
Llama2-7B	Para.	**74.5**	**73.0**	68.7	57.2	N/A	40.6	61.0	53.7
WikiMIA$_{2024}$ Hard									
Qwen2-7B	Orig.	**60.6**	**59.1**	54.3	45.7	N/A	44.5	45.4	50.6
Qwen2-7B	Para.	**56.6**	**58.4**	55.8	47.4	N/A	47.3	48.3	48.5
Dolma3 arXiv									
Olmo3-7B-Instruct	Orig.	**92.4**	**91.8**	81.5	71.1	N/A	42.3	64.4	37.4
Olmo3-7B-Instruct	Para.	**90.3**	**84.7**	70.4	69.4	N/A	43.9	57.5	29.9

Table 2: **Pre-training MIA results** on BookMIA, WikiMIA$_{2024}$ Hard, and Dolma3 arXiv. **Bold** denotes the top-2 performance for each dataset. Our active MIA methods (ADRA and ADRA+) consistently outperform passive MIAs. R-Loss is not applicable for these targets (see Section E). See Table 18 for low FPR results.

Model	**Type**	**Method**	**Budget** (N)	**Lexical Jaccard** (Best / Avg)	**Lexical LCS** (Best / Avg)	**Lexical Coverage** (Best / Avg)	**Embedding Cosine** (Best / Avg)
BookMIA							
Llama2-7B	Orig.	ADRA+	32	21.9 / **18.5**	62 / **54**	5.7 / **2.4**	89.0 / 83.9
		ADRA	32	18.9 / 15.9	56 / 48	4.2 / 2.0	88.6 / **84.4**
		N-Sampling	32	19.9 / 15.6	57 / 46	4.9 / 1.5	89.6 / 82.7
WikiMIA$_{2024}$ Hard							
Qwen2-7B	Orig.	ADRA+	32	20.1 / 14.1	51 / 33	16.1 / 6.0	88.4 / 82.3
		ADRA	32	17.3 / **14.8**	34 / 27	7.0 / 4.3	87.0 / **84.3**
		N-Sampling	32	16.6 / 11.4	52 / **42**	16.7 / **7.6**	87.9 / 81.6
Dolma3 arXiv							
Olmo3-7B-Instruct	Orig.	ADRA+	32	15.2 / 12.6	69 / **60**	9.5 / **5.0**	87.5 / **84.3**
		ADRA	32	15.9 / **13.0**	69 / 58	9.7 / 4.9	87.4 / 83.9
		N-Sampling	32	15.7 / 11.9	59 / 35	8.1 / 2.4	87.6 / 83.4

Table 3: **Pre-training member data reconstruction** for original verbatim setting. **Bold** denotes the best average performance for each dataset. ADRA+ and ADRA outperform N-Sampling on nearly all metrics; on WikiMIA$_{2024}$ Hard, N-Sampling retains higher LCS and n-gram coverage. See Section F for paraphrased setting results.

4.1 LLM-MIA Dataset Construction

Most MIA datasets target pre-training and use cutoffs around 2023 (Shi et al., 2023). To match frontier LLMs with later cutoffs and heavy post-training, we develop 6 new datasets.

Dolma3 / Tulu3 Mix: Fully open models enable controlled membership evaluation. We construct pre-training and post-training MIA datasets from the Dolma3 pre-training mix (Olmo et al., 2025) and Tulu3 SFT mix Lambert et al. (2024). For Dolma3, we choose arXiv documents as members, and sample non-members from arXiv papers beyond the mix cutoff. For Tulu3, we focus on Aya (Singh et al., 2024b) and WildChat (Zhao et al., 2024); members are the subset included in the Tulu3 mix while non-members are excluded from the mix.

AIME / Olympia Math: We construct controlled-contamination math MIA datasets by defining members as SFT data and non-members as held-out data. For AIME, we collect problems from 2021–2025, using 2021–2024 as members and 2025 as non-members. For Olympia Math (AI-MO, 2025), we split problems into members and non-members randomly.

S1 / S1.1 Distillation: Following Muennighoff et al. (2025), we use reasoning traces from two teacher models, Gemini-2.0-Flash (S1) and DeepSeek-R1 (S1.1). We treat either group as members depending on the experiment.

Overall, we evaluate on 8 MIA datasets. In addition to the verbatim (original) setting, we evaluate the *paraphrased* setting by paraphrasing the datasets with Gemini-2.5-Flash. See Section B for dataset construction details. These designs provide complementary controls: temporal cutoffs establish high-confidence non-members when training data are hidden, while WikiMIA$_{2024}$-Hard, randomized Olympia Math inclusion, and same-source Tulu3 subsets reduce temporal or source shifts.

4.2 MIA Baselines & Evaluation

Following Duan et al. (2024), we consider five loss-based baselines, each described in Section 2. We also include the reconstruction-based baseline of Hallinan et al. (2025): given a prefix, sample N continuations and score each by n-gram overlap metrics to the ground truth suffix. We generalize

		ACTIVE MIAS		PASSIVE MIAS					
Model	**Type**	**ADRA+**	**ADRA**	**N-Sampling**	**Loss**	**R-Loss**	**Zlib**	**Min-K%**	**Min-K%++**
Olympia Math									
Tulu2-7B	Orig.	**72.4**	67.8	59.5	53.7	**70.3**	55.6	53.2	60.2
Tulu2-7B	Para.	**68.5**	**69.4**	58.2	49.6	60.1	54.9	51.1	52.9
AIME									
Tulu2-7B	Orig.	**85.9**	**85.6**	72.7	71.1	67.8	61.7	75.8	78.3
Tulu2-7B	Para.	**86.2**	**82.2**	75.4	66.8	58.4	60.2	72.1	73.2
Tulu3 Mix									
Tulu3-8B	Aya	**69.7**	**68.9**	61.3	62.0	68.0	56.4	61.6	56.7
Tulu3-8B	Wildchat	**67.5**	**66.8**	56.9	51.8	66.2	46.7	52.8	64.3

Table 4: **Post-training MIA results** on AIME, Olympia Math, and Tulu3 Mix. Our active MIAs outperform calibration-free passive methods and match R-Loss. See Table 20 for low FPR results.

Model	**Type**	**Method**	**Budget** (N)	**Lexical Jaccard** (Best / Avg)	**Lexical LCS** (Best / Avg)	**Lexical Coverage** (Best / Avg)	**Embedding Cosine** (Best / Avg)
Olympia Math							
Tulu2-7B	Orig.	ADRA+	32	24.1 / 20.7	68 / 59	15.3 / **10.9**	91.5 / 88.8
		ADRA	32	25.6 / **23.4**	79 / **68**	12.3 / 10.2	92.8 / **91.3**
		N-Sampling	32	22.4 / 15.5	60 / 34	11.7 / 5.0	92.2 / 87.3
AIME							
Tulu2-7B	Orig.	ADRA+	32	22.3 / 18.1	55 / 39	17.2 / 11.9	91.9 / 88.9
		ADRA	32	21.7 / **18.5**	58 / **43**	15.5 / **12.5**	91.8 / **89.0**
		N-Sampling	32	19.9 / 12.9	50 / 27	14.7 / 5.7	91.8 / 85.1
Tulu3 Mix							
Tulu3-8B	Aya	ADRA+	32	44.3 / 43.9	14 / 14	47.8 / 47.4	89.2 / 88.9
		ADRA	32	51.6 / **51.2**	16 / **16**	53.0 / **52.5**	91.5 / **91.1**
		N-Sampling	32	41.6 / 26.5	13 / 10	40.0 / 23.9	90.4 / 84.9
Tulu3-8B	Wildchat	ADRA+	32	41.4 / **40.3**	63 / **61**	42.6 / **40.4**	92.0 / **91.4**
		ADRA	32	37.5 / 34.9	54 / 49	36.7 / 31.7	91.7 / 90.2
		N-Sampling	32	35.8 / 28.1	58 / 48	33.7 / 24.0	92.9 / 89.5

Table 5: **Post-training member data reconstruction** for original verbatim setting. Our methods outperform N-Sampling across lexical and semantic metrics. See Table 19 for paraphrased results.

the method of Hallinan et al. (2025) to N-SAMPLING and evaluate under our lexical metrics TRIO (Details in Section C.2). For each candidate, the final scalar membership score is Average-of-N (AoN).

We report **AUROC** for MIA performance, following (Shi et al., 2023; Duan et al., 2024; Hallinan et al., 2025), and additionally report **TPR@1%FPR** (Carlini et al., 2022). All main results are means over five random seeds. We evaluate data reconstruction performance using the same lexical and embedding similarity metrics. For evaluation details see Section E.

5 RESULTS

Pre-training Table 2 reports AUROC (%) for pre-training MIAs, comparing ADRA/ADRA+ to N-Sampling and 4 loss-based baselines. Across original and paraphrased settings, an active method achieves the highest AUROC on every dataset. In the original setting, ADRA+ outperforms the strongest loss-based MIA by 15.1, 10.0, and 21.3 percentage points and N-Sampling by 4.8, 6.3, and 10.9 points on BookMIA, WikiMIA$_{2024}$ Hard, and Dolma3 arXiv, respectively. At the 1% FPR threshold (Table 18), an active method yields the highest TPR on every dataset (up to 30.5%), while passive baselines frequently fall below 1% TPR.

Table 3 reports member reconstruction in the original setting. ADRA and ADRA+ improve reconstruction on average, most clearly in lexical Jaccard and embedding cosine; on WikiMIA$_{2024}$ Hard they raise Jaccard and embedding metrics but reduce LCS and n-gram coverage, a trade-off caused by reward dominance. Paraphrased results follow the same trends (Section F).

Post-training Table 4 reports AUROC (%) under controlled contamination, where Tulu2-7B is finetuned on held-out member examples from AIME and Olympia Math at a 10% contamination rate, and under a post-training mixture, where Tulu3-8B is evaluated on Aya and WildChat from the Tulu3 Mix. R-Loss uses backbone-matched references (Section A).

On Olympia Math and AIME, an active method achieves the best AUROC in both the original and paraphrased settings, improving over N-Sampling by 12.9 and 13.2 points in the original setting, with

ACTIVE MIA	PASSIVE MIAS					
ADRA	**N-Sampling**	**Loss**	**R-Loss**	**ZLib**	**Min-K%**	**Min-K%++**
S1.1 Distillation: Deepseek-R1						
98.4	90.4	55.1	70.0	52.6	55.7	70.6
S1 Distillation: Gemini-2.0-flash						
85.2	80.8	69.8	79.3	74.3	71.9	81.7

Table 6: **Distillation MIA results.** ADRA outperforms all passive MIAs on both distillation datasets (S1.1 and S1).

Method	**Lex. Jaccard** (Best / Avg)	**Lex. LCS** (Best / Avg)	**Lex. Coverage** (Best / Avg)	**Emb. Cosine** (Best / Avg)
S1.1 Distillation: Deepseek-R1				
ADRA	45.4 / **44.5**	145 / **143**	48.6 / **47.7**	97.6 / **97.5**
N-Sampling	32.7 / 25.5	111 / 82	34.9 / 22.3	97.4 / 95.6
S1 Distillation: Gemini-2.0-Flash				
ADRA	31.0 / **27.4**	117 / **106**	32.8 / **28.3**	96.4 / **95.6**
N-Sampling	27.8 / 21.2	111 / 80	27.4 / 17.5	96.2 / 93.7

Table 7: **Distillation member data reconstruction.** ADRA achieves significantly better reconstruction across all metrics.

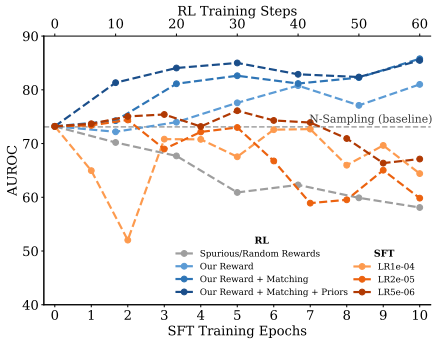


Figure 2: **Optimization controls on AIME**. Spurious-reward RL and SFT fail to improve over the initial AUROC, while reconstruction-aligned rewards improve detection; matching yields further gains.

similar gains under paraphrasing. On Tulu3 SFT Mix, ADRA+ slightly exceeds R-Loss on Aya (69.7 vs. 68.0) and WildChat (67.5 vs. 66.2) and outperforms the remaining baselines without requiring a calibration model. At 1% FPR (Table 20), the active methods again achieve the highest TPR on Olympia Math and AIME.

Reconstruction gains over N-Sampling are more pronounced than in pre-training across all metrics (Table 5): on Olympia Math, average Jaccard improves by 7.9, LCS by 34, coverage by 5.9, and embedding cosine by 4.0. Post-training data therefore appear more extractable, consistent with stronger memorization late in the training pipeline.

Distillation Table 6 reports AUROC (%) in a distillation setting where a student (Qwen2.5-7B-Instruct) is trained for one epoch on member reasoning traces produced by a teacher (Deepseek-R1 or Gemini-2.0-Flash), following Muennighoff et al. (2025). Because ADRA+ is not directly applicable to this setting (see Section C.3), we use ADRA only.[4]

ADRA reaches 98.4% AUROC under DeepSeek-R1 distillation and 85.2% under Gemini distillation, improving over the strongest loss-based baseline Min-K%++ by 27.8 and 3.5 points and over N-Sampling by 8.0 and 4.4 points. At 1% FPR (Table 21) it reaches 73.4% TPR under DeepSeek-R1, far exceeding all baselines, and reconstruction improves over N-Sampling by 19.0 average Jaccard and 25.4 coverage (Table 7).

6 Ablations & Analysis

Optimization Controls. A key alternative hypothesis is that any training / fine-tuning would also improve AUROC. To test whether gains arise merely from optimization, we compare SFT and spurious-reward RL (Shao et al., 2025) on AIME. Under matched RL settings (matched FLOPs), random rewards reduce AUROC to 58.1 over 60 steps, whereas reward-only RL, ADRA, and ADRA+ reach 81.3, 85.8, and 85.5; SFT across three learning rates also fails to improve with more compute. Thus additional optimization alone is insufficient: gains require reconstruction-aligned rewards, with matching further improving detection (Figure 2).

Disentangling Membership from Attack-Time Learning. Repeating the attack on Olympia Math from the Tulu2-7B checkpoint before contamination leaves every method near chance (ADRA+ 52.8, ADRA 53.1, N-Sampling 53.5). The 12.9-point advantage over N-Sampling on the contaminated target therefore requires prior exposure in the target weights, not attack-time training alone (Section J).

Strict Compute Budget. Although our method incurs additional attack-time model-update cost, it remains effective under strict RL compute budgets. On BookMIA and Dolma3 arXiv, both variants outperform all evaluated passive methods in AUROC after one RL step; by 10 steps, they are within 0.6–1.9 points of their 30-step performance. Linear prorations of measured 30-step runtimes give average 1- and 10-step costs of $0.68 and $6.82 on 8 H200s, respectively (Section D).

[4]Each pair shares a prompt and differs only in which teacher generated the suffix, exactly one of which trained the student. This pairing is fixed by the s1 distillation-data construction.

ADRA+'s Prior. We investigate how dependent ADRA+ is on prior accuracy. Without a prior, ADRA reaches 91.8 AUROC on Dolma3 arXiv. Across ADRA+ variants with prior AUROC from 37.4 to 71.1, final AUROC remains 87.7–92.4; *even below-chance priors outperform all passive baselines*, showing that the prior is a refinement rather than the primary signal (Table 13).

Model Size. We scale to Tulu2-13B on AIME (Table 22). Results mirror Tulu2-7B, with ADRA and ADRA+ outperforming existing MIAs.

Reward Type. We compare our lexical reconstruction rewards with two model-based alternatives under non-contrastive RL: embedding (Qwen3-8B-Embedding) and LLM-as-judge (Qwen3-32B). As shown in Table 23, both model-based variants improve over baseline but achieve lower performance than lexical rewards. Lexical rewards are directly verifiable, while model-based rewards are more prone to reward hacking.

7 Related Work

Membership Inference Attacks (MIA). Given a target model and candidate data, MIA determines whether that data is in the model's training set (Shokri et al., 2017). In language modeling, most prior work uses the model's log likelihoods on the candidate sequence as a proxy for membership. Yeom et al. (2018) directly threshold loss, while Carlini et al. (2021) calibrate the target's loss with a reference model or zlib compression. Shi et al. (2023) propose Min-$K\%$ to score only the lowest $k\%$ log-probabilities, and Zhang et al. (2024b) normalize Min-$K\%$ by the expected token log-probabilities over the vocabulary.

A smaller line of work leverages model generations, including methods that exploit verbatim-vs-paraphrase distinctions (Duarte et al., 2024) or poison training data to facilitate extraction (Tramèr et al., 2022). Closest to ours, Hallinan et al. (2025) show that n-gram metrics over sampled continuations approach loss-based MIA performance, yet these methods remain passive: they elicit one-shot outputs under fixed prompts and score them post hoc. In contrast, our approach to MIA actively incentivizes LLMs to reconstruct their own training data by maximizing a reconstruction-aligned reward.

Training Data Detection in LLMs. Reliably detecting training data is a key challenge: training corpora span trillions of tokens and underlying datasets are often undisclosed (Touvron et al., 2023; Balloccu et al., 2024). Maini et al. (2024); Duan et al. (2024); Das et al. (2025) have shown that the efficacy of MIAs is confounded by the distribution shifts between member and non-members. Rigorously proving data membership requires a low false-positive rate (Carlini et al., 2022; Zhang et al., 2024a), which is difficult to evaluate on closed-data models. Dataset inference and test-set contamination are discussed in Section K. Sainz et al. (2023); Singh et al. (2024a); Liu et al. (2025a) argue that data membership for LLMs should be defined behaviorally rather than by exact match. Our work explores both *strict* membership and *soft* membership, which counts semantically or functionally equivalent variants such as paraphrases as members. Under this formulation, false positives are less concerning as semantically equivalent data can constitute effective contamination.

Training Data Extraction and Memorization. Modern language models are known to memorize and regurgitate training data, which includes harmful and copyrighted information (Carlini et al., 2021; Biderman et al., 2023; Nasr et al., 2023; Zhang et al., 2023; Cooper & Grimmelmann, 2025). Pretraining dataset size often far exceeds model size, making data extraction difficult (Morris et al., 2025a). Models are also often intentionally aligned to make extraction more difficult after the alignment stage. Even with these theoretical and empirical safeguards in place, models still exhibit memorization of data after finetuning and post-training (Barbero et al., 2025; Ahmed et al., 2026). Data extraction commonly works by *discoverable extraction*: splitting a sample into a prefix and a suffix, prompting the LLM with the prefix, and evaluating similarity between sampled generations and the suffix (Carlini et al., 2021; Hayes et al., 2025). Our work improves extraction by fine-tuning the target model with on-policy RL, actively eliciting memorization through weight updates.

8 Conclusion

We propose Active Data Reconstruction Attack (ADRA) that leverages RL for data detection and reconstruction. ADRA is the first RL-based active MIA, improving over passive MIAs across different stages of model training. Section L scopes the remaining discussions and limitations.

AI USE STATEMENT

In this work, we used generative AI tools to create or modify styles of scientific figures, check grammar, draft parts (table, prompt template, etc) of the appendix, and improve the paper's readability. We did not use generative AI for other tasks requiring disclosure, and the remaining required tasks were not applicable. The authors reviewed and revised all AI-assisted output and take responsibility for the final content.

ETHICS STATEMENT

This paper investigates whether training data can be detected and reconstructed from language model weights through membership inference and active reconstruction methods. Our goal is to understand the privacy risks inherent in current language models across their lifecycle, from pre-training through post-training. The purpose of this work is to evaluate the limits of existing privacy protections and to motivate the development of more robust defenses against unintended data leakage.

We acknowledge that publishing improved attack methods carries dual-use risks. However, we believe that transparently characterizing these vulnerabilities is necessary for the community to develop adequate defenses, and that withholding such findings would leave a false sense of security. All experiments in this work use publicly available models and established benchmarks; no private user data was accessed or compromised in the course of this research.

REPRODUCIBILITY STATEMENT

We will release model checkpoints, code, and data associated with this paper post-review to allow reproduction of our evaluations. We have attempted to provide significant detail in our work on our setup and data construction. Where possible, we reuse existing open-source standards baselines for evaluation.

REFERENCES

Ahmed Ahmed, A Feder Cooper, Sanmi Koyejo, and Percy Liang. Extracting books from production language models. *arXiv preprint arXiv:2601.02671*, 2026.

AI-MO. Ai-mo olympiad reference dataset. Hugging Face Dataset, 2025. https://huggingface.co/datasets/AI-MO/olympiads-ref.

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen McAleer, Albert Q Jiang, Jia Deng, Stella Biderman, and Sean Welleck. Llemma: An open language model for mathematics. *arXiv preprint arXiv:2310.10631*, 2023.

Simone Balloccu, Patrícia Schmidtová, Mateusz Lango, and Ondřej Dušek. Leak, cheat, repeat: Data contamination and evaluation malpractices in closed-source llms. *arXiv preprint arXiv:2402.03927*, 2024.

Federico Barbero, Xiangming Gu, Christopher A. Choquette-Choo, Chawin Sitawarin, Matthew Jagielski, Itay Yona, Petar Veličković, Ilia Shumailov, and Jamie Hayes. Extracting alignment data in open models, 2025. URL https://arxiv.org/abs/2510.18554.

Stella Biderman, USVSN Sai Prashanth, Lintang Sutawika, Hailey Schoelkopf, Quentin Anthony, Shivanshu Purohit, and Edward Raff. Emergent and predictable memorization in large language models, 2023. URL https://arxiv.org/abs/2304.11158.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX security symposium (USENIX Security 21)*, pp. 2633–2650, 2021.

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.

A. Feder Cooper and James Grimmelmann. The files are in the computer: On copyright, memorization, and generative ai, 2025. URL https://arxiv.org/abs/2404.12590.

Debeshee Das, Jie Zhang, and Florian Trantèr. Blind baselines beat membership inference attacks for foundation models. In *2025 IEEE Security and Privacy Workshops (SPW)*, pp. 118–125. IEEE, 2025.

Jasper Dekoninck, Mark Niklas Müller, Maximilian Baader, Marc Fischer, and Martin Vechev. Evading data contamination detection for language models is (too) easy. *arXiv preprint arXiv:2402.02823*, 2024.

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.

André V Duarte, Xuandong Zhao, Arlindo L Oliveira, and Lei Li. De-cop: Detecting copyrighted content in language models training data. *arXiv preprint arXiv:2402.09910*, 2024.

Yanai Elazar, Akshita Bhagia, Ian Magnusson, Abhilasha Ravichander, Dustin Schwenk, Alane Suhr, Pete Walsh, Dirk Groeneveld, Luca Soldaini, Sameer Singh, et al. What's in my big data? *arXiv preprint arXiv:2310.20707*, 2023.

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

Skyler Hallinan, Jaehun Jung, Melanie Sclar, Ximing Lu, Abhilasha Ravichander, Sahana Ramnath, Yejin Choi, Sai Praneeth Karimireddy, Niloofar Mireshghallah, and Xiang Ren. The surprising effectiveness of membership inference with simple n-gram coverage. *arXiv preprint arXiv:2508.09603*, 2025.

Jamie Hayes, Marika Swanberg, Harsh Chaudhari, Itay Yona, Ilia Shumailov, Milad Nasr, Christopher A Choquette-Choo, Katherine Lee, and A Feder Cooper. Measuring memorization in language models via probabilistic extraction. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 9266–9291, 2025.

Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A Smith, Iz Beltagy, et al. Camels in a changing climate: Enhancing lm adaptation with tulu 2. *arXiv preprint arXiv:2311.10702*, 2023.

Alon Jacovi, Avi Caciularu, Omer Goldman, and Yoav Goldberg. Stop uploading test data in plain text: Practical strategies for mitigating data contamination by evaluation benchmarks. *arXiv preprint arXiv:2305.10160*, 2023.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Kimi Team. Kimi k3: Open frontier intelligence. Technical report, 2026. URL https://github.com/MoonshotAI/Kimi-K3/blob/main/k3_tech_report.pdf.

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.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pp. 74–81, 2004.

Ken Ziyu Liu, Christopher A Choquette-Choo, Matthew Jagielski, Peter Kairouz, Sanmi Koyejo, Percy Liang, and Nicolas Papernot. Language models may verbatim complete text they were not explicitly trained on. *arXiv preprint arXiv:2503.17514*, 2025a.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025b.

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.

John X. Morris, Chawin Sitawarin, Chuan Guo, Narine Kokhlikyan, G. Edward Suh, Alexander M. Rush, Kamalika Chaudhuri, and Saeed Mahloujifar. How much do language models memorize?, 2025a. URL https://arxiv.org/abs/2505.24832.

John X Morris, Junjie Oscar Yin, Woojeong Kim, Vitaly Shmatikov, and Alexander M Rush. Approximating language model training data from weights. *arXiv preprint arXiv:2506.15553*, 2025b.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 20286–20332, 2025.

Milad Nasr, Nicholas Carlini, Jonathan Hayase, Matthew Jagielski, A. Feder Cooper, Daphne Ippolito, Christopher A. Choquette-Choo, Eric Wallace, Florian Tramèr, and Katherine Lee. Scalable extraction of training data from (production) language models, 2023. URL https://arxiv.org/abs/2311.17035.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, et al. Olmo 3. *arXiv preprint arXiv:2512.13961*, 2025.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. *Proceedings of machine learning and systems*, 5:606–624, 2023.

Penghui Qi, Zichen Liu, Xiangxin Zhou, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Defeating the training-inference mismatch via fp16. *arXiv preprint arXiv:2510.26788*, 2025.

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork, August 2026. URL https://qwen.ai/blog?id=qwen3.8.

Horacio Saggion, Dragomir Radev, Simone Teufel, and Wai Lam. Meta-evaluation of summaries in a cross-lingual environment using content-based metrics. In *COLING 2002: The 19th International Conference on Computational Linguistics*, 2002.

Oscar Sainz, Jon Ander 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. *arXiv preprint arXiv:2310.18018*, 2023.

John Schulman and Thinking Machines Lab. Lora without regret. *Thinking Machines Lab: Connectionism*, 2025. doi: 10.64434/tml.20250929. https://thinkingmachines.ai/blog/lora/.

Rulin Shao, Jacqueline He, Akari Asai, Weijia Shi, Tim Dettmers, Sewon Min, Luke Zettlemoyer, and Pang Wei W Koh. Scaling retrieval-based language models with a trillion-token datastore. *Advances in Neural Information Processing Systems*, 37:91260–91299, 2024a.

Rulin Shao, Shuyue Stella Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, Simon Shaolei Du, Nathan Lambert, Sewon Min, Ranjay Krishna, Yulia Tsvetkov, Hannaneh Hajishirzi, Pang Wei Koh, and Luke Zettlemoyer. Spurious rewards: Rethinking training signals in rlvr. *arXiv preprint arXiv:2506.10947*, 2025.

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*, 2024b.

Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. Detecting pretraining data from large language models. *arXiv preprint arXiv:2310.16789*, 2023.

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.

Aaditya K Singh, Muhammed Yusuf Kocyigit, Andrew Poulton, David Esiobu, Maria Lomeli, Gergely Szilvasy, and Dieuwke Hupkes. Evaluation data contamination in llms: how do we measure it and (when) does it matter? *arXiv preprint arXiv:2411.03923*, 2024a.

Shivalika Singh, Freddie Vargus, Daniel D'souza, Börje F Karlsson, Abinaya Mahendiran, Wei-Yin Ko, Herumb Shandilya, Jay Patel, Deividas Mataciunas, Laura O'Mahony, et al. Aya dataset: An open-access collection for multilingual instruction tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11521–11567, 2024b.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, second edition, 2018.

Michael Tänzer, Sebastian Ruder, and Marek Rei. Memorisation versus generalisation in pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7564–7578, 2022.

Thinking Machines Lab. Announcing tinker. Product announcement, 2025. URL https://thinkingmachines.ai/news/announcing-tinker/.

Thinking Machines Lab. Inkling model card. Model card, 2026. URL https://thinkingmachines.ai/model-card/inkling/.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Florian Tramèr, Reza Shokri, Ayrton San Joaquin, Hoang Le, Matthew Jagielski, Sanghyun Hong, and Nicholas Carlini. Truth serum: Poisoning machine learning models to reveal their secrets, 2022. URL https://arxiv.org/abs/2204.00032.

Yiping Wang, Qing Yang, Zhiyuan Zeng, Liliang Ren, Liyuan Liu, Baolin Peng, Hao Cheng, Xuehai He, Kuan Wang, Jianfeng Gao, et al. Reinforcement learning for reasoning in large language models with one training example. *arXiv preprint arXiv:2504.20571*, 2025.

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, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Yang Yue, Zhiqi 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 preprint arXiv:2504.13837*, 2025.

Chiyuan Zhang, Daphne Ippolito, Katherine Lee, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. Counterfactual memorization in neural language models, 2023. URL https://arxiv.org/abs/2112.12938.

Hengxiang Zhang, Songxin Zhang, Bingyi Jing, and Hongxin Wei. Fine-tuning can help detect pretraining data from large language models. In *International Conference on Learning Representations*, 2025.

Jie Zhang, Debeshee Das, Gautam Kamath, and Florian Tramèr. Membership inference attacks cannot prove that a model was trained on your data. *arXiv preprint arXiv:2409.19798*, 2024a.

Jingyang Zhang, Jingwei Sun, Eric Yeats, Yang Ouyang, Martin Kuo, Jianyi Zhang, Hao Frank Yang, and Hai Li. Min-k%++: Improved baseline for detecting pre-training data from large language models. *arXiv preprint arXiv:2404.02936*, 2024b.

Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. Wildchat: 1m chatgpt interaction logs in the wild. *arXiv preprint arXiv:2405.01470*, 2024.

Table 8: Prompts used for pre-training experiments across datasets. The {prefix} placeholder indicates where the input text is inserted.

Dataset	Prompt Template
BookMIA	You will receive a prefix from a passage and be asked to complete it based on the text of a famous work. Provide only the continuation for the last given prefix without any extra commentary, formatting, or additional text. Complete the prefix: {prefix}
WikiMIA$_{2024}$ Hard	Continue the generation as closely to verbatim as possible. {prefix}
Dolma3 arXiv	Continue the generation as closely to verbatim as possible. {prefix}

Table 9: Supervised fine-tuning hyperparameters for controlled contamination setting.

Dataset	Learning rate	Eff. batch size	LR Schedule	Precision	Epochs	Warmup ratio	Max seq. len.
AIME	2e−5	64	Linear	BF16	1	0.00	4096
Olympia Math	5e−6	128	Constant				

A Experimental Details

We provide implementation details for replicating our experimental results, including prefix construction and hyperparameters for each training stage. All trainings are done in a single node consisting of 8 H200s.

A.1 Pre-training

Following Hallinan et al. (2025), we use approximately 50% of the candidate data as the prefix and reconstruct the remaining held-out suffix. We limit generation length to half the maximum token count of the dataset. We list the prompts we use before the candidate prefix in Table 8.

A.2 Post-training

Controlled contamination. To simulate test-set contamination, we fine-tune on member examples from AIME and Olympia Math. We leverage the natural problem-solution structure in both math datasets by including the full problem plus 25% of the solution as the prefix. To simulate practical contamination scenarios, we also mix member examples with other SFT data at a fixed 10% contamination rate using the Tulu2 SFT mix (Ivison et al., 2023). Table 9 details our fine-tuning hyperparameters.

Post-training mixtures. We leverage dataset structure to determine the amount of prefix. Both Aya and WildChat are instruction-tuning datasets containing user requests and assistant responses, and our curated MIA datasets only contain single-turn conversations. We include the full user request plus 25% of the assistant response as the prefix.

R-Loss reference models. For R-Loss, we calibrate with backbone-matched references: Llama2-7B for Tulu2-7B and Llama3-8B for Tulu3-8B.

A.3 Distillation

S1 and S1.1 consist primarily of problem-solution pairs. Since problems and solutions have comparable lengths, we include only the problem as the prefix. Following Muennighoff et al. (2025), we perform model distillation on Qwen2.5-7B-Instruct by fine-tuning on the entire S1 or S1.1 dataset without additional data mixing. Table 10 details our fine-tuning hyperparameters. Qualitatively, we find that Qwen2.5-7B-Instruct output resembles that of Deepseek-R1 more closely, so a smaller

Table 10: Supervised fine-tuning hyperparameters for model distillation setting.

Dataset	Learning rate	Eff. batch size	LR Schedule	Precision	Epochs	Warmup ratio	Max seq. len.
S1 - Gemini-2.0-Flash	2e−5	128	Linear	BF16	1	0.00	8192
S1.1 - Deepseek-R1	5e−6						

Paraphrasing Prompt Template

You will be given an ORIGINAL block that may contain natural-language, math, code, or other texts. Your job is to rephrase the natural-language content while mostly PRESERVING NON-NATURAL-LANGUAGE CONTENT.

1. Please ensure the paraphrased output have similar length to the original text (±15%).
2. When encountering mathematical formulas, if necessary, you can replace variable names. For example, you can replace 'x' with 'y' or 'a'.
3. No extra commentary: Do not add explanations, apologies, or any other text besides the required output.
4. Output directly the paraphrased output after "`rewrite_output:`".

`ORIGINAL: {input}`

Figure 3: Prompts used to paraphrase datasets. The `{input}` placeholder indicates where the input text is inserted.

learning rate is used relative to distilling from Gemini-2.0-Flash. For R-Loss, the reference model is Qwen2.5-7B-Instruct before distillation.

B MIA Dataset Construction

We provide details on our MIA dataset curation process. Due to computational constraints, we sample subsets from each constructed dataset and average results across five random seeds.

In addition to the original (verbatim) setting, we also evaluate a paraphrased setting where member and non-member examples are rewritten while preserving their semantic content. Specifically, we use the Gemini-2.5-Flash-v3.0 model API via LiteLLM to generate paraphrased instances. For experiments evaluating existing models, we use paraphrased examples as the evaluation set. For experiments involving fine-tuning, we use paraphrased examples as input to the training set and original examples as the evaluation set. We list the prompts used for each dataset in Figure 3.

Membership certainty and distribution matching. Testing whether members are more reconstructible than non-members requires knowing, with high confidence, which candidates the target was trained on. For most LLMs the training corpus is undisclosed, but when a model or training-corpus cutoff is documented, text first published after the cutoff cannot have been trained on. Temporal benchmarks such as BookMIA and WikiMIA build on this property to provide reliable non-member ground truth without access to the training data (Shi et al., 2023; Hallinan et al., 2025), which is what makes our hypothesis testable on real pre-trained models. Temporal construction secures label certainty rather than identically distributed member and non-member cohorts (Duan et al., 2024; Maini et al., 2024; Das et al., 2025; Zhang et al., 2024a), so we pair it with designs that reduce distribution differences to varying degrees: matched article versions, randomized inclusion, and same-source splits (see *Complementary controls across datasets* below).

Complementary controls across datasets. Each benchmark design controls a different source of variation. WikiMIA$_{2024}$-Hard pairs old and post-cutoff versions of the same article, holding the article fixed (Hallinan et al., 2025). Olympia Math randomly assigns examples to inclusion or holdout before controlled SFT, following randomized inclusion designs (Zhang et al., 2023; Carlini et al., 2022), so members and non-members are identically distributed by construction. Tulu3 compares included and excluded examples within the same Aya or WildChat source dataset, and the distillation study holds the prompt fixed while the two candidate traces come from different teachers. An active method

outperforms every evaluated baseline under the matched, randomized, and same-source designs, which argues against an explanation based on any single provenance cue.

B.1 DOLMA3 PRE-TRAINING MIX

For the pre-training setting, we collect arXiv documents from the Dolma3 pre-training data mix. These documents originate from the Proof-Pile-2 dataset (Azerbayev et al., 2023) and have a cutoff date of April 2023. We sub-sample 3,000 documents to form our member set. To construct non-members, we collect 1,000 arXiv papers released after 2025 and verify using OlmoTrace that they are not present in the pre-training data. For each seed, we sample 512 examples.

B.2 TULU3 SFT MIX

For the post-training mix setting, we collect two datasets from the Tulu3 SFT Mix: Aya and WildChat. Aya is a multilingual instruction-tuning dataset containing human-annotated prompt-completion pairs (Singh et al., 2024b), while WildChat contains real-world user interactions with ChatGPT (Zhao et al., 2024). For both datasets, members are sampled from data included in the Tulu3 SFT Mix, and non-members are data from the original datasets that were excluded from the mix. To control for length variation between members and non-members, we limit both user and assistant turns to a maximum of 1,024 tokens. For each seed, we sample 512 examples.

B.3 AIME / OLYMPIA MATH

For the controlled-contamination setting, we collect challenging math datasets: AIME and Olympia Math. For AIME, we collect problems and solutions from 2021–2025, using 2021–2024 as members and 2025 as non-members. For Olympia Math (AI-MO, 2025), because the dataset was released in 2025, we split problems into members and non-members using random partitions. For each seed, we sample 512 Olympia Math examples.

B.4 S1 / S1.1

For the distillation setting, we follow Muennighoff et al. (2025) and collect reasoning trace datasets from two teacher models: Gemini-2.0-Flash (S1) and DeepSeek-R1 (S1.1). We do not mix additional data, so the distilled traces constitute the entire training data. Depending on the experiment, we treat either group as members while the other serves as non-members. For each seed, we sample 512 examples.

C ACTIVE DATA RECONSTRUCTION ATTACK DETAILS

Below we provide more details on implementation of our method Active Data Reconstruction Attck.

C.1 GRPO HYPERPARAMTERS

We implement our method ADRA using VERL and the GRPO training hyperparameters are listed in Table 11.

C.2 RECONSTRUCTION METRICS DETAILS

We propose three lexical reconstruction metrics that our policy attempts to maximize, as detailed in Section 3.2. In practice, we experiment with two aggregated variants: (1) TRIO, which averages token set similarity, longest common subsequence, and N-gram set coverage; and (2) N-GRAM, which uses N-gram set coverage alone. TRIO gives the strongest overall results, so all reported reconstruction-based MIA results use TRIO. In the reconstruction tables, columns labeled Lexical LCS report the raw, unnormalized longest common subsequence length in words; the normalized LCS ratio defined in Section 3.2 is used only as a training reward. Because the reward compares rollouts with each candidate's true suffix, our reconstruction results measure discoverable extraction of candidates the

Table 11: GRPO hyperparameters used with `verl`. Only core settings are shown; batch size, rollouts, sequence lengths, and training epochs vary by dataset.

Hyperparameters	**Setting**
Advantage estimator	GRPO
Learning rate (actor)	$5e-05$
Normalize advantages (GRPO)	True
Use KL *loss* (not reward penalty)	True
KL loss coef.	0.005
Entropy coefficient	0
Generation temperature	1.0
Generation top-p	0.95
Generation top-k	50
LoRA rank / α	64 / 128
Critic warmup	0

Table 12: Dataset-specific hyperparameters for ADRA training.

Dataset	**Batch size**	**Rollouts**	**Max prompt / response tokens**	**Steps**	**Reported Reward**	**Distractors K**	**Length penalty**
BookMIA	64	32	1028 / 1028	30	TRIO	7	1.5
WikiMIA$_{2024}$ Hard	64	32	1028 / 1028	30	TRIO	7	1.5
Dolma3 arXiv	64	32	1028 / 1028	30	TRIO	7	1.5
AIME	64	32	2048 / 2048	100	TRIO	7	1.5
OlympiaMath	64	32	2048 / 2048	100	TRIO	7	1.5
Tulu3 Aya / Wildchat	64	32	1024 / 512	100	TRIO	7	1.5
S1 / S1.1	64	16	1028 / 4096	100	TRIO	N/A	1.5

auditor already holds, that is, how much attack-time RL increases their reconstructibility, rather than extraction of text unknown to the attacker.

To constrain generation length and prevent reward hacking through excessive output, we apply a length penalty to both metrics. The penalty operates on a threshold-based ratio system: candidate outputs are penalized if their word count falls outside the range $[\ell_{\text{ref}}/\tau, \ell_{\text{ref}} \times \tau]$, where ℓ_{ref} is the reference length and τ is the threshold parameter. We set $\tau = 1.50$ for both TRIO and N-GRAM, allowing candidates to be as short as 67% or as long as 150% of the reference length without penalty. Outside this range, a linear penalty proportional to the length deviation is applied, multiplying the base similarity score by $\min(1.0, \min(\ell_{\text{cand}}/\ell_{\text{ref}}, \ell_{\text{ref}}/\ell_{\text{cand}}))$.

We provide dataset-specific hyperparameters for these metrics in Table 12.

Shared-policy batching. Each dataset/seed attack run initializes one policy from the target model and trains it jointly over the full unlabeled candidate pool using candidate minibatches of size 64. Policy updates accumulate across minibatches; the model is not reset per candidate or per minibatch. After the final update, the same shared policy scores every candidate in the pool. Candidate texts and distractors are therefore used transductively, but membership labels are never provided during training, scoring, or prior calibration.

C.3 CONTRASTIVE REWARD DETAILS

We use $K = 7$ distractors, so each generation is scored against 8 candidate suffixes: 1 ground truth plus 7 distractors. Distractors are sampled i.i.d. from the evaluation set.

For WikiMIA$_{2024}$ Hard, which contains paired Wikipedia articles from different date cutoffs (one member, one non-member), we include the paired article as one distractor in addition to $K - 1$ randomly sampled distractors.

Paired distillation-trace inference. Each pair has one shared prompt and two teacher-generated candidate suffixes, exactly one of which was used to train the student. This pairing is inherited from the distillation-data construction and fixed before ADRA is applied; it is not an attack-specific pairing choice. ADRA receives both suffixes symmetrically and is never told which is the member; its reward takes the larger reconstruction score over the pair and therefore does not encode a membership label or privilege the member suffix. Access to both candidate texts is part of the MIA input. Holding the prompt fixed removes prefix variation, so post-update separation measures which trace the distilled

student reconstructs more readily. Accordingly, the 98.4% result is scoped to paired distillation-trace discrimination rather than arbitrary unpaired MIA.

Practical motivation and evaluation scope. Reasoning-model distillation trains student models on teacher-generated reasoning traces, as in the S1 setup (Muennighoff et al., 2025). This creates a concrete data-provenance audit: given two known teacher traces for the same prompt, determine which trace was used to train the student. The resulting paired provenance protocol isolates this question by holding the prompt fixed and varying only the candidate suffix. We report 98.4% AUROC specifically for this paired audit. We neither claim nor estimate the same performance under arbitrary unpaired MIA; the result and its comparisons should therefore be interpreted within the paired distillation-trace protocol rather than as evidence about the unpaired setting.

C.4 ADRA+ Details

ADRA+ augments the base reconstruction reward with a loss-based membership prior. Let s_i denote the raw loss-based MIA score for the i-th candidate, and let $\mathbf{s} = [s_1, \ldots, s_M]$ contain scores from the full unlabeled candidate pool. Calibration uses these scores but never membership labels.

We apply joint min-max normalization: $w_i = (s_i - \min(\mathbf{s}))/(\max(\mathbf{s}) - \min(\mathbf{s}))$. When the score range is near-degenerate ($\max(\mathbf{s}) \approx \min(\mathbf{s})$), normalization is numerically unstable, so all weights fall back to a uniform value of 0.5, effectively disabling the prior. Since lower loss-based scores indicate higher membership likelihood, we invert the normalized weights to obtain the membership prior: $p_i = 1 - w_i \in [0, 1]$. We found this simple calibration to be stable across all experimental settings.

Method / Prior	Prior AUROC	Final AUROC	Δ vs. ADRA
ADRA (no prior)	–	91.8	–
ADRA+ / Loss	71.1	92.4	+0.6
ADRA+ / Min-K%	64.4	91.2	−0.6
ADRA+ / Zlib	42.3	90.0	−1.8
ADRA+ / Min-K%++	37.4	87.7	−4.1

Table 13: **Robustness to passive membership priors** on Dolma3 arXiv. ADRA+ remains strong across priors ranging from informative to below chance. Weaker priors can underperform ADRA, indicating that the prior is a quality-dependent refinement rather than the primary source of performance.

D Compute Budget, Runtime, & FLOPs

Because our method leverages on-policy RL training to improve data detection accuracy, it incurs attack-time model-update compute. We quantify this cost using an idealized floating-point operations (FLOPs) model, measured runtimes, and strict-budget experiments, then discuss the compute-to-accuracy scaling that our active MIA introduces.

D.1 FLOPs Comparison

Cost model. Let C_{fwd} denote the FLOPs required for a single forward pass on a candidate sequence of length L through a model with P parameters. We adopt the following standard approximations:

1. Forward & Backward pass cost. A forward pass through a Transformer is dominated by the linear projections in each layer, costing approximately $2PL$ FLOPs (Kaplan et al., 2020). We denote this cost as C_{fwd}. Following previous works, we estimate the backward pass cost as $2C_{\text{fwd}}$.
2. Autoregressive generation cost. Generating L tokens autoregressively requires L sequential forward steps. With the standard key–value (KV) cache (Pope et al., 2023), each step after the initial prefill processes only a single token, costing approximately $2P$ FLOPs. The total generation cost is therefore $\sum_{i=1}^{L} 2P \approx 2PL = C_{\text{fwd}}$. [5]

[5] This approximation neglects the quadratically growing attention cost per step, which is subdominant to the linear projection cost for typical sequence lengths and model sizes used in our experiments (Kaplan et al., 2020).

Per-method cost breakdown. Using these approximations, we analyze the FLOPs cost of each MIA method summarized in Table 14:

- *Loss-based scores* (PPL, Zlib, Min-K%, Min-K%++): These methods compute a single loss over the candidate sequence, requiring one forward pass: C_{fwd}.
- *Reference-based scores* (R-Loss): In addition to evaluating the candidate model, R-Loss requires an extra forward pass through a separate reference model for calibration, totaling $2C_{\text{fwd}}$.
- *N-Sampling*: This method generates N completions from a prompted model and aggregates their statistics. Generation costs $N \cdot C_{\text{fwd}}$ in total.
- *Ours (GRPO, T RL steps, training-rollout budget N, evaluation budget M)*: Each of the T RL steps consists of (i) generating N rollouts ($N \cdot C_{\text{fwd}}$) and (ii) one GRPO policy update, approximated as a forward and backward pass over the rollouts ($3N \cdot C_{\text{fwd}}$). After T steps, evaluation uses M final samples ($M \cdot C_{\text{fwd}}$). The idealized model-compute proxy is:

$$\underbrace{TNC_{\text{fwd}}}_{\text{generation}} + \underbrace{3TNC_{\text{fwd}}}_{\text{GRPO updates}} + \underbrace{MC_{\text{fwd}}}_{\text{final eval}} = (4TN + M)C_{\text{fwd}}. \tag{7}$$

  When $M = N$, this reduces to $(4T + 1)NC_{\text{fwd}}$. This proxy omits implementation overhead and should not be interpreted as measured end-to-end FLOPs.

Method	**MIA Type**	**FLOPs Cost**
Loss / PPL / Zlib / Min-K% / Min-K%++	Passive	C_{fwd}
R-Loss	Passive	$2 \cdot C_{\text{fwd}}$
N-Sampling (budget N)	Passive	$N \cdot C_{\text{fwd}}$
Ours (GRPO, T RL steps)	Active	$(4TN + M)C_{\text{fwd}}$

Table 14: Idealized FLOPs comparison across MIA methods. C_{fwd} denotes the proxy cost of one forward pass or generation. ADRA adds an attack-time model-update budget T.

D.2 Performance under strict compute budget.

The reported 30-step pre-training runs took 48 minutes on BookMIA and 70 minutes on Dolma3 arXiv using 8 H200 GPUs. Each timing covers one complete shared-policy dataset/seed attack run over 512 candidates; it is not a per-candidate runtime. Panel (b) of Table 15 linearly prorates their 59-minute average to shorter RL budgets; these estimates exclude fixed overhead. Dollar costs use GMI Cloud's advertised starting price of $2.60 per H200-hour (https://www.gmicloud.ai/en/pricing, accessed May 27, 2026).

D.3 Compute-Matched Comparisons

The reported N-Sampling baseline uses $N = 32$. Across all evaluated datasets, we find that increasing its budget to $N \in \{64, 128, 256, 512\}$ does not meaningfully improve AUROC (all within one point), showing that its performance is effectively saturated at the reported budget. Thus, matching or exceeding the active method's model-compute proxy through additional decoding does not enable N-Sampling to convert the larger inference budget into improved detection performance (Section D).

For reference, one-step ADRA with $N = 32$ has an idealized cost of 160 forward equivalents. N-Sampling at $N = 128$ is within 25% of this proxy, while $N = 256$ and $N = 512$ use $1.6\times$ and $3.2\times$ as much model compute, respectively. Thus, under near-matched and larger model-compute budgets, in contrast to ADRA, N-Sampling cannot convert additional sampling compute into improved detection performance.

D.4 Scaling training compute for detection accuracy.

A critical distinction of our active approach is that detection accuracy scales with attack-time model-update compute. Among the evaluated methods, only ADRA and ADRA+ expose this model-update axis: increasing the RL budget generally improves AUROC over the tested range (Table 15).

(a) Strict-budget AUROC (%)

Dataset	Method	N-Sampling	RL Steps			
			1	5	10	30
BookMIA	ADRA	73.6	74.7	76.1	75.6	76.2
	ADRA+		75.8	77.2	77.6	78.4
Dolma3 arXiv	ADRA	81.5	84.2	89.1	89.9	91.8
	ADRA+		84.9	87.8	90.7	92.4

(b) Average prorated runtime and cost

RL Steps	Wallclock (8×H200)	H200-hours	USD
1	2.0 min	0.26	$0.68
5	9.8 min	1.31	$3.41
10	19.7 min	2.62	$6.82
30	59 min	7.87	$20.45

Table 15: **Performance under strict RL compute budgets.** Panel (a) reports AUROC as the number of attack-time model-update steps varies; N-Sampling is the evaluated passive generation baseline. Panel (b) averages runtime across BookMIA and Dolma3 arXiv and estimates cost per shared-policy dataset/seed run over 512 candidates from the measured 30-step runs. These are not per-candidate costs.

Dataset / setting	Seeds	Max. AUROC SD	95% CI half-width
BookMIA	5	0.9	1.12
WikiMIA_{2024} Hard	5	1.2	1.49
Dolma3 arXiv	5	1.1	1.37
AIME	5	1.5	1.86
Olympia Math	5	0.8	0.99
Tulu3 Mix (Aya)	5	1.1	1.37
Tulu3 Mix (WildChat)	5	0.9	1.12
S1 (Gemini-2.0-Flash)	5	1.7	2.11
S1.1 (DeepSeek-R1)	5	1.9	2.36

Table 16: **Dataset-level AUROC uncertainty across five seeds.** The standard deviation (SD) column reports the maximum AUROC SD across the reported methods and variants for each setting. For five independent seeds, the two-sided 95% Student-t confidence interval for a mean is $\bar{x} \pm t_{0.975,4}s/\sqrt{5} = \bar{x} \pm 1.242s$; the final column reports this half-width in AUROC points.

Detection accuracy in MIA setting outweighs compute significantly. For training data detection and MIA in LLMs, the primary objective is detection accuracy and low false positives (Carlini et al., 2022). Higher-stakes settings such as data auditing and copyright compliance may prioritize detection quality over computational efficiency. We therefore consider ADRA practically feasible for targeted, high-stakes audits of open-weight models: a one-step run averages about two minutes on 8 H200s at a prorated cost below $1 and already outperforms all evaluated baselines in AUROC on both strict-budget datasets.

Comparison scope. Consistent with prior MIA evaluations (Carlini et al., 2021; Shi et al., 2023; Hallinan et al., 2025), we compare end-to-end attacks under their method-specific operating configurations rather than imposing equal FLOPs across algorithms that require different operations. Our objective is detection effectiveness under the stated threat model, while compute is quantified separately, following previous literature.

E Evaluation Details

Repeated runs and uncertainty. All main experiments use five random seeds. We report means across seeds; the dataset-level standard deviation of AUROC is at most 1.9 percentage points across all reported experiments. Table 16 reports the maximum per-dataset standard deviations and corresponding 95% confidence-interval half-widths.

Baseline availability. R-Loss requires a reference model that did not train on the candidate corpus, which exists for the six post-training rows and the two distillation rows but not for the pre-training

targets (Tables 2, 4 and 6). Restricting the comparison to the eight rows where R-Loss is reported, the better of ADRA and ADRA+ exceeds R-Loss by 11.8 AUROC points on average and the strongest available loss-based baseline by 8.3 points. The reported advantage therefore does not depend on excluding reference-calibrated rows, although the margin is smaller on those rows than on the pre-training ones. Including N-Sampling, our generalization of Hallinan et al. (2025) and the strongest baseline in 8 of the 14 settings, the better of ADRA and ADRA+ exceeds the strongest evaluated baseline by 6.8 AUROC points on average and by at least 1.3 points in every setting; it exceeds N-Sampling itself in every setting, by 9.3 points on average.

E.1 N-Sampling Baseline Evaluation

For the N-Sampling baseline, we generate N independent rollouts per input with top-$p = 0.95$ and top-$k = 50$. **To ensure a fair decoding comparison, we evaluate N-Sampling at both $\tau = 0.7$ and $\tau = 1.0$ and report the better result for each setting.** This gives N-Sampling the best chance to obtain its optimal performance. The $\tau = 0.7$ condition directly matches ADRA's final evaluation temperature, while $\tau = 1.0$ matches the exploration configuration and per-step rollout budget used during RL training (Table 12).

E.2 ADRA Evaluation

After RL training, we evaluate the policy with N rollouts per input at temperature $\tau = 0.7$, top-$p = 0.95$, and top-$k = 50$, favoring exploitation after the policy has learned to reconstruct (Sutton & Barto, 2018). Because N-Sampling is evaluated at both $\tau = 0.7$ and $\tau = 1.0$, its reported result includes a temperature-matched comparison and cannot be disadvantaged by the choice of either decoding temperature.

E.3 Model Access Assumptions

ADRA requires either (1) *open-weights access* to the target model or (2) a managed API that supports on-policy RL with custom rewards (e.g., the Thinking Machines Tinker Training API (Thinking Machines Lab, 2025)). Likelihood-based MIAs such as Loss and Min-K% require teacher-forced per-token likelihoods for an arbitrary candidate sequence. Some serving APIs return log probabilities for tokens generated in the response, but these response-token scores do not provide likelihoods for user-supplied candidate tokens.

Implementing Loss, R-Loss, Zlib, Min-K%, or Min-K%++ therefore requires open weights or a dedicated candidate-scoring interface. **In our evaluated open-weight setting, ADRA's threat model is not more restrictive than that of likelihood-based passive MIAs: both operate on the same downloaded weights and have access to the same candidate text.** ADRA changes the attack procedure and compute budget, not the model access or information available to the attacker. Conversely, a compatible managed RL API can support ADRA without exposing raw weights, while ADRA+ additionally requires access to the likelihood-derived prior in Section C.4. N-Sampling remains applicable to text-only generation APIs. We therefore consider ADRA practically grounded for open-weight and compatible managed-RL settings, but not for inference-only APIs that expose neither candidate likelihoods nor fine-tuning.

F Additional Results: Pre-training Reconstruction

F.1 Reconstruction

Table 17 presents reconstruction quality for paraphrased members across pre-training datasets. Results closely mirror the verbatim setting (Table 3). ADRA and ADRA+ consistently outperform N-Sampling on average across lexical Jaccard and embedding cosine metrics. On BookMIA, ADRA+ achieves the strongest average performance (13.5 Jaccard, 44 LCS, 1.0 coverage), while ADRA attains the best embedding cosine (84.3). On WikiMIA_{2024} Hard, ADRA reaches the best average Jaccard (13.4) and embedding cosine (84.1), though N-Sampling achieves higher LCS (37) and coverage (5.3)—consistent with the metric trade-off observed in the verbatim setting where RL optimization favors reward-aligned metrics. On Olmo3 Training Mix, ADRA attains the best

Table 17: **Pre-training member data reconstruction** for paraphrased setting. **Bold** denotes the best average performance for each dataset. ADRA+ and ADRA outperform N-Sampling on nearly all metrics; on WikiMIA$_{2024}$ Hard, N-Sampling retains higher LCS and n-gram coverage. See Table 3 for original verbatim setting results.

Model	Type	Method	Budget (N)	Lexical Jaccard (Best / Avg)	Lexical LCS (Best / Avg)	Lexical Coverage (Best / Avg)	Embedding Cosine (Best / Avg)
BookMIA							
Llama2-7B	Para.	ADRA+	32	15.8 / **13.5**	51 / **44**	3.0 / **1.0**	88.2 / 84.2
		ADRA	32	14.7 / 13.0	47 / 42	2.2 / 0.9	87.3 / **84.3**
		N-Sampling	32	15.5 / 12.3	48 / 39	2.5 / 0.5	88.2 / 81.5
WikiMIA2024-Hard							
Qwen2-7B	Para.	ADRA+	32	16.4 / 13.1	33 / 25	8.1 / 4.0	86.5 / 82.0
		ADRA	32	17.3 / **13.4**	40 / 27	9.5 / 4.2	88.1 / **84.1**
		N-Sampling	32	14.3 / 9.9	47 / **37**	12.8 / **5.3**	87.2 / 80.0
Olmo3 Training Mix							
Olmo3-7B-Instruct	Para.	ADRA+	32	14.3 / 11.8	56 / **48**	6.3 / 2.9	86.3 / 83.1
		ADRA	32	14.6 / **12.0**	56 / 47	6.4 / **3.0**	86.5 / **83.2**
		N-Sampling	32	13.8 / 10.2	40 / 25	6.2 / 1.6	86.8 / 82.5

Table 18: **Pre-training membership inference at low false-positive rates (TPR@1%FPR). Bold** denotes the best performance for each dataset–setting pair. ADRA+ and ADRA maintain meaningful detection rates at the stringent 1% FPR threshold, while most passive baselines collapse to near-zero. A dash (–) indicates the method was not applicable.

Model	Type	ADRA+	ADRA	N-Sampling	Loss	R-Loss	Zlib	Min-K%	Min-K%++
BookMIA									
Llama2-7B	Orig.	**29.8**	14.3	9.6	15.7	–	15.4	12.6	7.9
	Para.	**23.4**	15.6	12.9	9.4	–	0.2	6.3	0.3
WikiMIA2024-Hard									
Qwen2-7B	Orig.	7.2	**9.4**	1.3	1.9	–	1.7	1.8	0.3
	Para.	**4.7**	2.9	1.6	0.4	–	0.1	0.9	0.6
Dolma3 arXiv									
Olmo3-7B-Instruct	Orig.	**30.5**	15.6	7.8	12.5	–	0.3	3.1	1.5
	Para.	**21.2**	15.6	14.9	7.8	–	0.1	12.5	0.4

average Jaccard (12.0), coverage (3.0), and embedding cosine (83.2), while ADRA+ achieves the highest LCS (48). Overall, results suggest that RL improves training data extraction despite semantic perturbations.

F.2 Low-FPR

Table 18 reports TPR@1%FPR for pre-training membership inference. At 1% threshold, passive baselines largely collapse: methods such as Zlib and Min-K%++ frequently fall below 1% TPR. In contrast, ADRA+ and ADRA maintain meaningful detection rates across all three datasets, with ADRA+ achieving the highest TPR in five of six settings (up to 30.5% on Dolma3 arXiv). The gap is especially pronounced under paraphrasing, where most loss-based baselines fall below 1% TPR while ADRA+ retains up to 23.4% TPR. Results suggest that active MIAs consistently improve TPR at the low FPR threshold, yielding non-trivial detection rates across all settings where many passive baselines are near chance.

G Additional Results: Post-training

G.1 Reconstruction

Table 19 presents reconstruction quality for paraphrased members in post-training settings. Results closely mirror the verbatim setting (Table 5) with better extraction compared to pre-training. ADRA and ADRA+ consistently outperform N-Sampling across all metrics on both controlled contamination datasets. On Olympia Math, ADRA achieves the best average Jaccard (21.4), LCS (61), and embedding cosine (89.6), while ADRA+ attains the highest coverage (11.4). On AIME, ADRA+ excels in average Jaccard (18.8) and coverage (12.7) while ADRA excels in average LCS (42) and embedding cosine (89.0). The improvements over N-Sampling remain substantial: on AIME, ADRA+ improves average Jaccard by 6.7 and coverage by 7.9, while ADRA improves average LCS

Table 19: **Post-training member data reconstruction** for paraphrased setting. ADRA+ and ADRA consistently outperform N-Sampling across both lexical and semantic metrics. See Table 5 for original verbatim setting results.

Model	Type	Method	Budget (N)	Lexical Jaccard (Best / Avg)	Lexical LCS (Best / Avg)	Lexical Coverage (Best / Avg)	Embedding Cosine (Best / Avg)
Olympia Math							
Tulu2-7B	Para.	ADRA+	32	23.8 / 21.0	67 / 60	16.0 / **11.4**	91.7 / 89.3
		ADRA	32	24.1 / **21.4**	68 / **61**	12.7 / 8.5	92.1 / **89.6**
		N-Sampling	32	22.8 / 15.5	60 / 34	11.9 / 5.1	92.4 / 87.3
AIME							
Tulu2-7B	Para.	ADRA+	32	22.7 / **18.8**	54 / 40	17.5 / **12.7**	92.2 / 88.8
		ADRA	32	21.1 / 17.9	56 / **42**	15.4 / 11.7	91.8 / **89.0**
		N-Sampling	32	18.4 / 12.1	48 / 27	13.6 / 4.8	89.2 / 83.3

Table 20: **Post-training membership inference at low false-positive rates (TPR@1%FPR). Bold** denotes the best performance for each dataset–setting pair.

Model	Type	ADRA+	ADRA	N-Sampling	Loss	R-Loss	Zlib	Min-K%	Min-K%++
Olympia Math									
Tulu2-7B	Orig.	**25.0**	21.9	6.3	2.9	21.9	0.1	3.1	12.5
	Para.	**22.7**	12.5	3.0	3.4	21.9	0.2	4.4	6.3
AIME									
Tulu2-7B	Orig.	30.7	**38.8**	18.8	23.5	24.7	6.6	29.9	25.1
	Para.	**37.5**	26.0	6.6	25.1	11.5	6.3	28.2	31.3
Tulu3 Mix									
Tulu3-8B	Aya	2.3	5.9	4.7	**7.8**	4.7	3.3	**7.8**	4.7
	Wildchat	14.5	9.4	10.9	0.2	**18.8**	0.1	1.4	0.4

by 15 and embedding cosine by 5.7. Like in pre-training, results confirm that RL improves training data extraction despite semantic perturbations.

G.2 Low-FPR

Table 20 reports the corresponding post-training results. On Olympia Math, ADRA+ achieves the highest TPR in both settings (25.0% and 22.7%), while N-Sampling and most passive baselines drop sharply. On AIME, ADRA and ADRA+ lead in the original and paraphrased settings, respectively, though passive baselines such as Loss and Min-K% also retain moderate detection rates—likely reflecting the stronger memorization signal in mathematical competition data. On Tulu3 Mix, where the fine-tuning data is more diverse and less memorized, all methods yield lower TPR and no single approach dominates. Overall, active methods lead at 1% FPR on both controlled-contamination datasets, but not on Tulu3 Mix.

H Additional Results: Distillation

H.1 Low-FPR

Table 21 reports TPR@1%FPR for distillation membership inference. ADRA is the only applicable active variant and achieves the highest TPR in both settings, reaching 73.4% on Deepseek-R1 and 31.3% on Gemini 2.0-Flash, substantially outperforming all baselines. Across all three training stages, an active method achieves the highest TPR@1%FPR in 12 of 14 settings; the exceptions are Aya and WildChat in Tulu3 Mix, where passive baselines lead.

I Ablation Details

All ablations average results over three random seeds.

Optimization controls. We compare ADRA against two alternative uses of attack-time optimization. The SFT control trains directly on the candidate mixture at learning rates 10^{-4}, 2×10^{-5}, and 5×10^{-6}. Following Shao et al. (2025), the spurious-reward control samples an independent reward

Table 21: **Distillation membership inference at low false-positive rates (TPR@1%FPR). Bold** denotes the best performance for each dataset–setting pair. A dash (–) indicates the method was not applicable.

Model	Type	ADRA+	ADRA	N-Sampling	Loss	R-Loss	Zlib	Min-K%	Min-K%++
Deepseek-R1									
Qwen2.5	Orig.	–	**73.4**	10.2	0.3	0.7	6.3	22.7	30.5
Gemini 2.0-Flash									
Qwen2.5	Para.	–	**31.3**	0.5	1.9	1.5	10.2	0.2	10.2

Active MIAs		Passive MIAs					
ADRA+	**ADRA**	**N-Sampling**	**Loss**	**R-Loss**	**ZLib**	**Min-K%**	**Min-K%++**
89.5	**84.4**	72.5	70.7	63.3	60.6	77.7	82.8

Table 22: **Tulu2-13B MIA results** on AIME.

Method	Lexical	Embedding	LLM-as-Judge	No-Training
AUROC	**81.3**	76.9	78.8	73.1

Table 23: **Reward type comparison** on AIME, under non-contrastive RL. Lexical rewards outperform generative-based alternatives. See Section I for details.

from $\mathcal{U}[0, 1]$ for each rollout while keeping the target model, candidate set, GRPO algorithm, and 60-step budget identical to the ADRA arm of this ablation.

Tables 22 and 23 report the model-size and reward-type ablations discussed in Section 6.

Model-Based Reward Implementation. We implement model-based rewards by hosting dedicated embedding and vLLM inference servers, which the *verl* trainer queries via FastAPI. Specifically, we use Qwen3-8B-Embedding as the embedding model and Qwen3-32B as the LLM-as-judge.

For embedding-based rewards, the model embeds both the candidate and ground-truth suffix and returns their cosine similarity as the reward signal. For LLM-as-judge rewards, the model is prompted to assess similarity between the candidate and ground-truth based on a structured prompt and returns a numerical score. The prompt used for LLM-as-judge can be found in Figure 4.

Observations. In preliminary experiments, we find that model-based rewards are particularly prone to reward hacking, with different target models exhibiting distinct hacking behaviors. Our ablations suggest that model-based reward RL training can improve reconstruction performance, but its efficacy depends on both prompt design and model choice. We leave further exploration of model-based rewards for future work.

J Disentangling Membership from Attack-Time Learning

Section 6 establishes that optimization alone does not produce membership separation: under a matched budget, spurious rewards reduce AUROC. A second question is whether the separation requires the target model to have trained on the candidates at all, or whether it can arise from attack-time training on the scored suffixes.

Uncontaminated-target control. We repeat the full attack on Olympia Math with a single change: the policy is initialized from the Tulu2-7B checkpoint *before* contamination fine-tuning, which never saw the designated member examples. The candidate pool, prefix fraction, reconstruction rewards, distractor construction, rollout budget, and number of RL steps are identical to the contaminated run reported in Table 4. Table 24 reports the result: every method falls near chance, and the active methods no longer exceed N-Sampling. ADRA+ reaches 52.8 and ADRA 53.1 against N-Sampling's 53.5, so the $+12.9$-point advantage observed on the contaminated target becomes -0.7.

Every component of the attack that operates on the candidates is unchanged in this control: each scored suffix still supplies reward, distractors are still drawn from the same pool, and one shared policy is still trained transductively over it. Attack-time suffix supervision is therefore insufficient on its own to produce the separation reported in Table 4; that separation requires the target to have trained on the member examples. Together with the reward controls of Section 6, this indicates that reconstruction-aligned rewards and prior exposure in the target weights are jointly necessary.

Why Olympia Math. Olympia Math assigns examples to inclusion or holdout at random before contaminated fine-tuning, so on the uncontaminated checkpoint the two groups differ by nothing except the label, and any separation would indicate that the attack manufactures signal on its own.

LLM-as-Judge Prompt Template

TASK DESCRIPTION
Rate two solutions to the same math problem (one reference, one candidate) for similarity. The final answer in each solution is enclosed in "\boxed{}". Return a real-valued score between 0 and 1 with exactly 3 decimals.

EVALUATION CRITERIA (consider all, equally)

1. **Mathematical correctness** — Is the candidate mathematically sound as in the reference?
2. **Solution approach** — Does the candidate use methods/reasoning similar to the reference?
3. **Final answer** — Does the candidate arrive at the same \boxed{} answer as the reference?
4. **Overall clarity** — Are the candidate's reasoning and steps as consistent and logically sound as the reference?
5. **Surface & semantic resemblance** — Do the two solutions look very similar in wording, symbols, step order, and intermediate expressions (i.e., high lexical/token overlap and near-paraphrase semantics)? Penalize added filler or unnecessary rephrasing.

INPUTS

- Reference solution: {`REFERENCE_SOLUTION`}
- Candidate solution: {`CANDIDATE_SOLUTION`}

OUTPUT FORMAT (must follow exactly)
Output ONLY one line:
`REWARD: <real value between 0 and 1 with 3 decimals>`

Figure 4: Prompts used for LLM-as-Judge reward model evaluation.

	Active MIAs		Passive MIAs					
Target	**ADRA+**	**ADRA**	**N-Sampling**	**Loss**	**R-Loss**	**Zlib**	**Min-K%**	**Min-K%++**
Contaminated (Table 4)	**72.4**	67.8	59.5	53.7	70.3	55.6	53.2	60.2
Uncontaminated	52.8	53.1	53.5	47.7	–	50.3	48.1	43.8

Table 24: **Uncontaminated-target control** on Olympia Math, original setting. The uncontaminated row repeats the identical attack from the Tulu2-7B checkpoint before contamination fine-tuning. Every method falls near chance, and the active methods no longer exceed N-Sampling: the $+12.9$-point advantage of ADRA+ over N-Sampling on the contaminated target becomes -0.7. A dash indicates the method was not reported for this target.

The measured spread of 43.8 to 53.5 across all methods is consistent with chance. We do not run this control on AIME: the AIME member years overlap Tulu2-7B's pre-training window (Table 1), so its pre-contamination checkpoint may already have seen some member problems and is not an uncontaminated target for that dataset.

Scope. This control is available where we construct membership ourselves and can hold out a matched checkpoint. For the pre-training benchmarks no such counterfactual target exists, so the mechanism evidence there rests on the reward and optimization controls rather than on a counterfactual initialization.

K Additional Related Work

Beyond the attacks discussed in Section 7, FSD fine-tunes the target model itself on a small set of known non-members from the same domain, such as text published after the model's release, and scores candidates by how much a likelihood-based score shifts (Zhang et al., 2025). Because FSD requires a labeled set of same-domain non-members, disjoint from the candidates it scores, it operates under a stronger threat model than ours: ADRA receives only the unlabeled candidate pool. We therefore compare against baselines that, like ADRA, use no membership labels, and do not include FSD. FSD updates weights only to calibrate likelihood scores; ADRA instead optimizes the target

with on-policy RL to reconstruct the candidates themselves and scores generated reconstructions rather than likelihoods.

Instead of inferring membership for individual data, Maini et al. (2024) propose dataset inference for LLMs to detect datasets used for training. In parallel, Jacovi et al. (2023); Elazar et al. (2023) find test-set contamination to be a growing concern in evaluation: paraphrases, translations, and synthetic restatements can inflate benchmark scores while evading detection (Yang et al., 2023; Dekoninck et al., 2024).

L Limitations & Discussions

There are limitations to our method and analysis; we detail them below.

Cost. ADRA is much more compute-intensive than current MIAs because it requires on-policy RL training. Performing MIA on large-scale data may be prohibitively expensive. Further, while we find that RL on average improves MIA and reconstruction over N-sampling, the efficacy of *scaling RL training* varies across datasets.

Memorization & Generalization. Empirically, our evidence shows that reconstruction updates to a target-initialized policy increase member–non-member separation. However, it is difficult to measure precisely how much our RL formulation elicits verbatim memorization and generalization, and the balance may vary by dataset. Precisely disentangling memorization and generalization has been an ongoing challenge (Tänzer et al., 2022; Morris et al., 2025a), and we hope future work can more carefully examine these effects at scale.

RL Algorithms. We use vanilla GRPO (Shao et al., 2024b) throughout the paper. Recent work (Liu et al., 2025b; Yu et al., 2025; Qi et al., 2025) has identified failure modes of GRPO, such as training collapse and instability, and proposed several remedies. These improvements could be readily incorporated into ADRA to further boost reconstruction and MIA performance. We leave exploration of better RL algorithms to future work.

M Reconstruction Examples

We show qualitative reconstruction examples across all three training stages: pre-training (Figure 5, Olmo3 Mix arXiv), post-training (Figures 6 and 7, AIME), and distillation (Figure 8, S1.1 Deepseek-R1).

Ground Truth

The novelty of our approach is in probabilistically modelling the HT as a continuous function of frequency, rather than using multiple independent response models (psychometric functions) for a discrete set of standard frequencies. This is by endowing the threshold curve with a Gaussian process (GP) prior \citep{rasmussen_gaussian_2006}, which is why we refer to our method as 'GP-PTA'. If the responses are binary (audible or non-audible), the full model can be interpreted as a two-dimensional binary classifier that is specified by a GP in the frequency dimension and by a psychometric (sigmoid) function in the intensity dimension. It provides uncertainty bands on the resulting threshold estimate, which enables fundamental and objective stopping criteria. Moreover, it allows one to find the optimal next stimulus based on the data processed so far, and stimuli at any frequency can be used. These properties reduce the number of trials required to achieve the desired accuracy. Minimizing the amount or required trials is important to reduce the cognitive burden on the (often elderly) patients. [...] A probabilistic response model is specified to capture uncertainty in the patient's responses [...] A generative probabilistic model is derived by combining the response model with a GP. The GP provides a natural way to take prior knowledge about the threshold curve into account, and relaxes the piecewise linear assumption for the threshold curve to a more natural smoothness assumption.

Reconstruction

However, these approaches typically assume that the hearing thresholds at different frequencies are independently distributed [...] Recent work using Gaussian process (GP) models \citep{zhang2017gp, garrido2019gp, rahman2020gp} has begun to address this by modeling the HT curve as a Gaussian process, which naturally captures correlations between adjacent frequency points. These models are capable of providing not only point estimates for the threshold at each frequency but also uncertainty estimates, which can be used to determine an objective stopping criterion [...] GP models can be regularized with prior information about typical HT curves [...] In this work, we propose a simplified yet effective framework for estimating the pure tone hearing threshold curve using a structured Gaussian process. Our approach models the entire threshold curve as a function of frequency using a covariance function that explicitly encodes smoothness and periodicity constraints [...] This structure allows the model to extrapolate between measured points and to exploit the inherent correlation between thresholds at neighboring frequencies, thus reducing the number of required measurements. We also incorporate a prior based on standardized audiograms [...] A method to use the output of the model for adaptive data collection, including a stopping criterion based on the uncertainty of the estimated thresholds.

Figure 5: Qualitative example from Olmo3 Mix arXiv: Reconstruction of a GP-based audiometry paper. Highlighted phrases show semantic and lexical overlap in core technical concepts—*Gaussian process (GP)*, *threshold curve*, *function of frequency*, *uncertainty estimates/bands*, *stopping criterion/criteria*, *smoothness*, and *prior knowledge/information*—suggesting that the reconstruction recovers the paper's key methodological vocabulary and conceptual structure.

Ground Truth

$d \cdot e \cdot f =$ $(a \cdot b \cdot c) \cdot (a \cdot b \cdot c - 1) \geq 6! = 720$, so $a \cdot b \cdot c \geq 28$. It follows that $a \cdot b \cdot c$ and $d \cdot e \cdot f$ are consecutive composites with prime factors no other than $2, 3, 5$, and 7. The smallest values for $a \cdot b \cdot c$ and $d \cdot e \cdot f$ are 36 and 35, respectively. So, we have $\{a, b, c\} = \{2, 3, 6\}, \{d, e, f\} = \{1, 5, 7\}$, and $\{g, h, i\} = \{4, 8, 9\}$, from which $\frac{a \cdot b \cdot c - d \cdot e \cdot f}{g \cdot h \cdot i} = \frac{1}{288}$. If we do not minimize the numerator, then $a \cdot b \cdot c - d \cdot e \cdot f > 1$. Note that $\frac{a \cdot b \cdot c - d \cdot e \cdot f}{g \cdot h \cdot i} \geq \frac{2}{7 \cdot 8 \cdot 9} > \frac{1}{288}$.Together, we conclude that the minimum possible positive value of $\frac{a \cdot b \cdot c - d \cdot e \cdot f}{g \cdot h \cdot i}$ is $\frac{1}{288}$. Therefore, the answer is $1 + 288 = \boxed{289}$.

Reconstruction

$d \cdot e \cdot f = 1$. Therefore, the minimum possible positive value of $\frac{a \cdot b \cdot c - d \cdot e \cdot f}{g \cdot h \cdot i}$ is $\frac{1}{7 \cdot 8 \cdot 9}$.
Similarly, to minimize a positive fraction, we minimize its numerator and maximize its denominator. It is clear that $\frac{a \cdot b \cdot c - d \cdot e \cdot f}{g \cdot h \cdot i} \geq \frac{1}{7 \cdot 8 \cdot 9}$. If we minimize the numerator, then $a \cdot b \cdot c - d \cdot e \cdot f = 1$. Note that $a \cdot b \cdot c \cdot d \cdot e \cdot f = 1$. Therefore, the minimum possible positive value of $\frac{a \cdot b \cdot c - d \cdot e \cdot f}{g \cdot h \cdot i}$ is $\frac{1}{7 \cdot 8 \cdot 9}$.
Therefore, the sum of the relatively prime positive integers is $1 + 1 = 2$.
Therefore, we need to find the minimum possible positive value of $\frac{a \cdot b \cdot c - d \cdot e \cdot f}{g \cdot h \cdot i}$. In this case, the minimum possible positive value is $\frac{1}{7 \cdot 8 \cdot 9}$.
Therefore, the answer is $\frac{1}{7 \cdot 8 \cdot 9}$.
Therefore, the answer is 2.

Figure 6: Qualitative Example 1 from AIME. The reconstruction captures core mathematical reasoning patterns and expressions from the ground truth, despite yielding a wrong final answer.

Ground Truth

before the bus arrives are adults. Similarly, we know that $x + 50 \equiv 0 \pmod{25}$, as $\frac{11}{25}$ of the people at the party are adults after the bus arrives. $x + 50 \equiv 0 \pmod{25}$ can be reduced to $x \equiv 0 \pmod{25}$, and since we are looking for the minimum amount of people, x is 300. That means there are 350 people at the party after the bus arrives, and thus there are $350 \cdot \frac{11}{25} = \boxed{154}$ adults at the party.

Reconstruction

are adults.
Similarly, let x be the number of people at the party before the bus cut. We know that $x \equiv 0 \pmod{12}$, as $\frac{5}{12}$ of people at the party are adults.
Therefore, the number of adults at the party before the bus arrives is 0.
After the bus arrives, there are a total of $50 + 25 = 75$ people at the party. Let x be the number of people at the party after the bus arrives. We know that $x \equiv 0 \pmod{12}$, as $\frac{11}{25}$ of people at the party are adults.
Therefore, the minimum number of adults at the party after the bus arrives is 0.
Therefore, the answer is 0.
We are looking for the minimum number of adults at the party after the bus arrives. In this case, there are no adults at the party after the bus arrives.
Therefore, the answer is 0.

Figure 7: Qualitative Example 2 from AIME. The reconstruction captures key mathematical patterns and modular arithmetic reasoning despite a different final answer.

Ground Truth

Given the matrix $\mathbf{M} = \begin{pmatrix} -\omega^2 & -\omega \\ 1 & 0 \end{pmatrix}$, where ω is a nonreal root of $x^3 = 1$,

we need to find the sum of the entries of $\mathbf{M} + \mathbf{M}^2 + \mathbf{M}^3 + \cdots + \mathbf{M}^{2009}$.

First, we note that ω is a primitive cube root of unity, satisfying $\omega^3 = 1$ and $1 + \omega + \omega^2 = 0$. We compute the powers of $\mathbf{M}$:

1. $\mathbf{M} = \begin{pmatrix} -\omega^2 & -\omega \\ 1 & 0 \end{pmatrix}$ 2. $\mathbf{M}^2 = \begin{pmatrix} 0 & 1 \\ -\omega^2 & -\omega \end{pmatrix}$ 3. $\mathbf{M}^3 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ (the identity matrix)

Since $\mathbf{M}^3 = \mathbf{I}$, the powers of $\mathbf{M}$ cycle every 3 terms. We decompose the sum [...] into cycles of 3 terms. There are 2009 terms in total, which is 669 complete cycles (each containing 3 terms) and 2 remaining terms.

The sum for each cycle is $\mathbf{M} + \mathbf{M}^2 + \mathbf{I}$. The remaining terms are $\mathbf{M}^{2008}$ and $\mathbf{M}^{2009}$.

The sum of the entries of [...] is computed as follows: - Sum of entries of $\mathbf{M}$: $-\omega^2 - \omega + 1 + 0 = 1 - \omega^2 - \omega$ - Sum of entries of $\mathbf{M}^2$: $0 + 1 - \omega^2 - \omega = 1 - \omega^2 - \omega$ - Sum of entries of $\mathbf{I}$: $1 + 0 + 0 + 1 = 2$

Total sum for one cycle: $2(1 - \omega^2 - \omega) + 2 = 6$. [...] Therefore, the total sum is: $669 \cdot 6 + 4 = 4014 + 4 = 4018$

The sum of the entries of $\mathbf{M} + \mathbf{M}^2 + \mathbf{M}^3 + \cdots + \mathbf{M}^{2009}$ is $\boxed{4018}$.

Reconstruction

Given the matrix $\mathbf{M} = \begin{pmatrix} -\omega^2 & -\omega \\ 1 & 0 \end{pmatrix}$, where ω is a nonreal root of $x^3 = 1$,

we need to find the sum of the entries of $\mathbf{M} + \mathbf{M}^2 + \mathbf{M}^3 + \cdots + \mathbf{M}^{2009}$.

First, we note that the nonreal roots of $x^3 = 1$ are ω and ω^2, where ω and ω^2 are the complex cube roots of unity satisfying $\omega^3 = 1$ and $1 + \omega + \omega^2 = 0$.

To find the powers of $\mathbf{M}$, we compute the first few powers of $\mathbf{M}$: $\mathbf{M}^2 = \begin{pmatrix} \omega & 1 \\ -\omega^2 & -\omega \end{pmatrix}$ $\mathbf{M}^3 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \mathbf{I}$

Since $\mathbf{M}^3 = \mathbf{I}$, the powers of $\mathbf{M}$ cycle every 3 terms. Therefore, the sequence $\mathbf{M}, \mathbf{M}^2, \mathbf{M}^3, \mathbf{M}^4, \ldots$ repeats every 3 terms.

Next, we find the sum of the entries of each matrix in one cycle: - The sum of the entries of $\mathbf{M}$ is $(-\omega^2) + (-\omega) + 1 + 0 = 1 - \omega - \omega^2 = 0$. - The sum of the entries of $\mathbf{M}^2$ is $\omega + 1 - \omega^2 - \omega = 1$. - The sum of the entries of $\mathbf{I}$ is $1 + 0 + 0 + 1 = 2$.

Thus, the sum of the entries in one complete cycle (3 terms) is $0 + 1 + 2 = 3$.

[...] we note that there are 2009 terms, which can be divided into 669 complete cycles of 3 terms and 2 remaining terms. The sum of the entries for the complete cycles is $669 \times 3 = 2007$. The sum of the entries for the remaining 2 terms ($\mathbf{M}^{2008}$ and $\mathbf{M}^{2009}$) is $1 + 2 = 3$.

Therefore, the total sum is: $2007 + 3 = 2010$

Thus, the sum of the entries of $\mathbf{M} + \mathbf{M}^2 + \mathbf{M}^3 + \cdots + \mathbf{M}^{2009}$ is $\boxed{2010}$.

Figure 8: Qualitative example from S1.1 Deepseek-R1 Distillation. Near-verbatim reconstruction of a matrix powers problem. Highlighted parts show overlap in algebraic identities, the cyclic structure insight ($\mathbf{M}^3 = \mathbf{I}$), cycle decomposition logic, and entry-sum computation steps. Despite closely reproducing the full solution template, the reconstruction arrives at an incorrect final answer (2010 vs. 4018).