LEAKY STUDENTS: MEMBERSHIP INFERENCE AGAINST ON-POLICY DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

On-policy distillation (OPD) trains a student to match a teacher's next-token distributions on student-generated trajectories. However, privileged information supplied to the teacher for OPD training may contain sensitive data. Whether the student leaks private information about the records supplied to the teacher during distillation remains poorly understood. To the best of our knowledge, we present the first systematic study of membership inference in this setting. We find that fresh student trajectories expose sparse membership signals that fixed reference-answer losses often miss. These signals are mixed with probability changes caused by training on other records. We introduce Leaky, which samples fresh trajectories from the target model and compares its token log-probabilities with the maximum across matched reference models trained without the candidate records. It applies Leaky ReLU to the resulting gaps, preserving positive gaps and down-weighting negative gaps as an approximate correction for incidental positive gaps in non-members. Across fifteen targets spanning mathematics, medical question answering, and code generation, Leaky outperforms all evaluated baselines and achieves mean AUROC 0.875, compared with 0.614 for the strongest baseline on each target in the main evaluation. On the same sampled trajectories, the strongest baseline achieves mean AUROC 0.826. These results show that students trained through OPD can expose the membership of records used for teacher supervision, even when fixed reference-answer losses provide little evidence of membership.

1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) has become a key paradigm for improving the reasoning capabilities of large language models (LLMs) (Lambert et al., 2024; DeepSeek-AI, 2025). However, RLVR typically relies on sparse, outcome-based rewards that provide limited token-level feedback. OPD addresses this limitation by providing dense, token-level teacher supervision on trajectories sampled from the student's own policy, mitigating the train–inference distribution mismatch of fixed-sequence distillation (Lu & Thinking Machines Lab, 2025; Agarwal et al., 2024). Recent methods additionally give the teacher *privileged information*, such as reference answers or expert demonstrations, that is absent from the student's input (Zhao et al., 2026; Shenfeld et al., 2026; Ye et al., 2026). The student learns from the teacher's distributions on its sampled trajectories without directly training to reproduce the privileged reference.

When these references contain sensitive information, excluding them from the student's input does not remove their influence on the released model. For example, a teacher might use confidential clinical annotations to supervise a student that sees only the case description. Those annotations shape the teacher's token-level feedback and, consequently, the student's parameter updates. If the resulting changes retain information specific to a training record, the student may reveal that the record was used, even without reproducing its annotation. An attacker who already knows a case and its annotation could thereby learn whether that case contributed to an institution's training collection. This is a membership privacy risk: the sensitive fact is the record's inclusion in training, rather than necessarily its contents (Shokri et al., 2017).

Prior work shows that distillation can leak membership information about both teacher and student training data (Jagielski et al., 2023; Zhang et al., 2025b). Membership inference attacks (MIAs) on language models use selected token scores (Zhang et al., 2025a), reference-model calibration (Tao

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

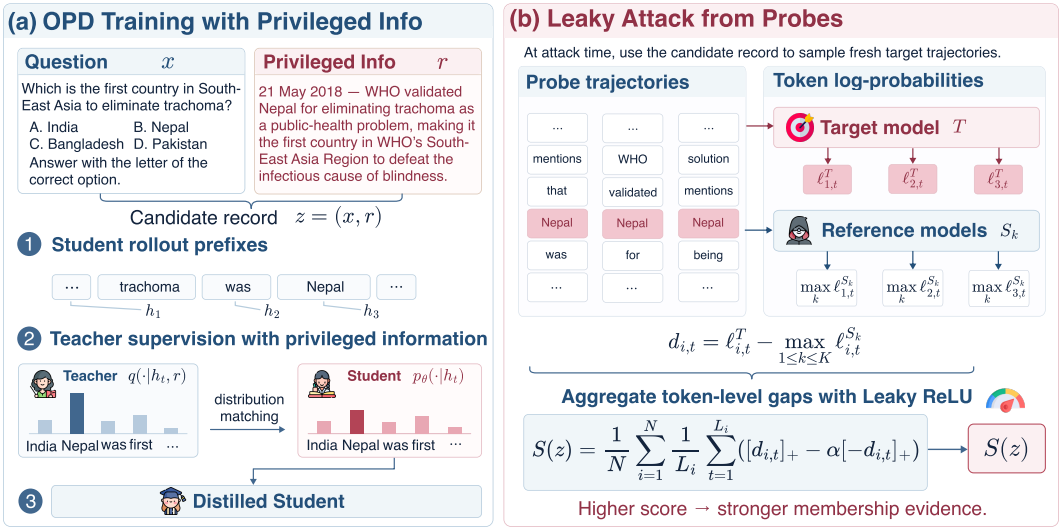


Figure 1: Overview. (a) In OPD with privileged information, the student samples rollouts from the prompt x alone, and the teacher, which also sees the reference answer r, supplies the next-token distributions the student matches. (b) Leaky samples trajectories from the target conditioned on the candidate records, scores each token under the target and K reference models trained without the candidate, and aggregates the resulting gaps with a Leaky ReLU into the membership score .

& Shokri, 2025), and aggregation over text windows (Chen et al., 2026). Related work also audits RLVR through changes in sampled generations (Liu et al., 2025). These findings motivate examining membership leakage in the reference-conditioned OPD setting, where a record's reference answer guides teacher feedback on the student's own trajectories.

OPD poses a particular challenge for this evaluation. Unlike supervised fine-tuning (SFT) and fixed-sequence knowledge distillation (KD), OPD matches the teacher's next-token distributions on student-generated trajectories rather than fixed target texts. Even for the same training record, trajectories can vary across iterations, so the record's influence may span different prefixes and continuations. Because OPD does not directly maximize the fixed reference answer's likelihood, that likelihood may miss membership signals in the student's generation behavior. Conventional loss-based MIAs may therefore provide limited separation between members and non-members, as illustrated in Figure 2(a). This motivates our central question: ***Can fresh student trajectories reveal which records were used for OPD training?***

We address this question with the following contributions:

- **The First study of membership inference in OPD.** To the best of our knowledge, we present the first systematic study of membership inference on OPD training records (Figure 1(a)). We show that fresh student trajectories reveal membership information through local token-probability differences between the target and matched reference models, which fixed reference-answer losses can miss.
- **Leaky, a membership inference attack tailored to OPD.** We introduce Leaky (Figure 1(b)), a reference-based attack that infers membership from fresh trajectories sampled from the target model for each candidate record. Instead of scoring the record's fixed reference answer, Leaky calibrates token-level log-probabilities on these trajectories against matched reference models and aggregates the resulting gaps asymmetrically to extract membership signals. The attack requires no access to the trajectories used during training.
- **Evaluation across three domains.** We evaluate Leaky on 15 target models spanning mathematics, medical question answering, and code generation. Leaky outperforms all evaluated baselines on every target, improving mean AUROC from 0.614 for the strongest baseline on each target to 0.875. We further demonstrate that membership leakage persists even under text-only prefix queries, show that synthetic auxiliary data can support effective attacks, and evaluate the privacy–utility trade-offs of LoRA and DP-LoRA defenses.

2 PRELIMINARIES

2.1 ON-POLICY DISTILLATION

Knowledge distillation trains a student to match a teacher's output distribution (Hinton et al., 2015). In on-policy distillation (OPD), the student samples a trajectory $\hat{y} \sim p_\theta(\cdot \mid x)$ for a prompt x and learns from the teacher's next-token distributions on the sampled trajectory (Agarwal et al., 2024; Lu & Thinking Machines Lab, 2025). As the student policy changes during training, subsequent trajectories are sampled from the updated policy. Recent work provides the teacher with privileged information that is absent from the student's input during training (Penaloza et al., 2026; Zhao et al., 2026; Shenfeld et al., 2026). We focus on records $z = (x, r)$, where x is a prompt and r is a reference answer used as privileged information. The student samples trajectories conditioned on x, while the teacher additionally receives r. At each prefix $\hat{y}_{<t}$, the teacher provides $q(\cdot \mid x, r, \hat{y}_{<t})$ as supervision for the student's distribution $p_\theta(\cdot \mid x, \hat{y}_{<t})$. For example, forward-KL distillation uses the token-level objective

$$\mathcal{L}_t = D_{\mathrm{KL}}(q(\cdot \mid x, r, \hat{y}_{<t}) \,\|\, p_\theta(\cdot \mid x, \hat{y}_{<t})) \,. \tag{1}$$

The reference answer influences training through the teacher's distribution; the student is not directly trained to maximize $\log p_\theta(r \mid x)$.

2.2 THREAT MODEL

Given a target model T and a candidate record $z = (x, r)$, the attacker aims to determine whether z belongs to the OPD training set $\mathcal{D}$ used to train T. Both the prompt x and the reference answer r are available to the attacker. In the main gray-box setting, the attacker can sample from the target and query the conditional log-probability of a specified token given a prompt and trajectory prefix. The target's parameters, gradients, and training trajectories are unavailable to the attacker. The attacker has access to the pretrained model used to initialize the target, the teacher model, and the OPD training configuration. We also assume access to auxiliary data drawn from the same task distribution, with the evaluated membership candidates excluded. We consider two additional settings: text-only black-box access without token probabilities, and limited auxiliary data. These settings are evaluated separately in Section 4.

3 LEAKY: MEMBERSHIP INFERENCE AGAINST OPD

Leaky is a reference-model-based membership inference attack against models trained with OPD. The attacker first trains K reference models on auxiliary data drawn from the same distribution as the training data. For each candidate record, the attacker then samples trajectories from the target with the prompt and privileged information, compares the target's token log-probabilities with the highest log-probability across all the reference models at each position, and uses the resulting gaps aggregated with Leaky ReLU as the membership score. We first motivate our design and then describe the attack procedure in this section.

3.1 UNDERSTANDING MEMBERSHIP SIGNALS IN OPD

Loss-based membership inference (Yeom et al., 2018) scores a candidate record (x, r) by the target model's loss on the reference answer r given the prompt x. SFT minimizes exactly this loss, so the loss can help distinguish members from non-members. OPD optimizes a different objective: the student matches the teacher's next-token distributions by minimizing the per-token distillation loss $\mathcal{L}_t$ (Eq. 1) on trajectories sampled from its own policy. Training therefore shapes how the student generates without directly raising the likelihood of the fixed reference answer. As a result, a record can leave a trace in the student's generation behavior while its reference-answer loss barely changes. On the Qwen3-1.7B mathematics target in Table 1, trained on OpenThoughts-Math, member and non-member losses largely overlap; the Loss attack achieves an AUROC of only 0.516 (Figure 2(a)).

We further examine whether membership signals exist on sampled trajectories. For each candidate record (x, r), we sample eight trajectories from the target model conditioned on (x, r) and average the token loss first within each trajectory and then across the eight trajectories. The resulting signals

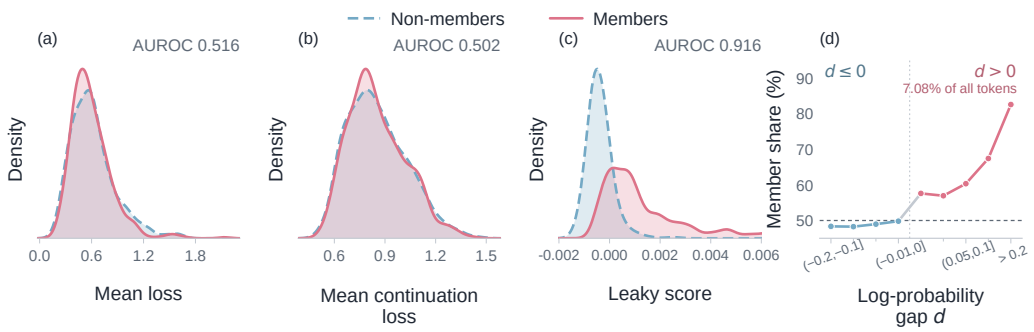


Figure 2: Membership signals in OPD. (a,b) Mean loss on fixed answers and sampled continuations provides limited separation between members and non-members. (c) Leaky separates the two groups more clearly. (d) Positive token-level gaps account for only 7.08% of all tokens, and larger positive gaps generally have a higher member share.

largely overlap between members and non-members, yielding an AUROC of 0.502 (Figure 2(b)). To understand what the average loss may miss, we examine both the source of token-probability increases and their distribution across token positions. First, the average loss reflects training on *all* records, not only the candidate: other records of the same task can produce comparable probability gains, so a low loss alone does not indicate that the candidate was used for training. Extracting a membership signal therefore requires *calibrating* the target's token probabilities against models trained without the candidate, so that shared changes are accounted for. Second, stochastic sampling can produce different trajectories for the same (x, r), and training on a member record does not necessarily increase the probability of every sampled token. Averaging over all positions may therefore hide probability changes concentrated at particular positions. Prior work also shows that aggregating selected token-level scores can be more effective than averaging the loss over all tokens (Shi et al., 2024; Zhang et al., 2025a). We examine (1) how to calibrate token probabilities using reference models and (2) which token positions carry membership signals and how to aggregate their scores.

(1) Shared changes in token probabilities. To separate the effect of the candidate from these shared changes, we need to know what training without the candidate already produces. The pretrained base model cannot provide this: its difference from the target reflects the effect of training on every record, not only the candidate. Following Zarifzadeh et al. (2024) and Tao & Shokri (2025), we train K reference models on auxiliary data that exclude the candidate records. At each position of a sampled trajectory, the target and the reference models score the same token under the same prompt and preceding tokens. A high target probability thus counts as membership signal only when it exceeds the probabilities of all reference models at that position. The challenge is that such positions are sparse and can also arise for non-members; we examine this empirically next.

(2) Membership signal in token-level gaps. We define the gap d as the target's token log-probability minus the highest reference log-probability. A positive gap means that the target exceeds every reference model at that position. In the same mathematics experiment, only 7.08% of scored tokens have positive gaps. Within the positive-gap region, the member share generally increases with the gap and reaches approximately 83% in the highest-gap interval. In contrast, the member share among negative-gap positions is 49.3% (Figure 2(d)). These observations motivate treating positive gaps as the primary membership signal. Non-members also produce positive gaps, however, so a positive gap cannot be attributed entirely to training on the candidate. Greater variation in token predictions across models can produce both larger positive and larger negative gaps, even for non-members, which suggests using the aggregated negative contribution as an approximate correction for such variation. Most scored tokens have negative gaps, and the magnitude of this contribution varies across records. Giving negative gaps equal weight may obscure membership differences carried by the positive gaps. We therefore retain positive gaps at full weight and assign negative gaps a small weight through Leaky ReLU, defined in Section 3.2. Further details are in Appendix A.2.

3.2 Attack Method

Leaky consists of three steps: (1) trajectory sampling from the target, (2) reference-model calibration of token log-probabilities to account for shared changes, and (3) Leaky ReLU aggregation of the resulting gaps.

Step 1: Trajectory sampling. Given a candidate record $z = (x, r)$, we sample N trajectories $y_1, \ldots, y_N$ from the target model T conditioned on (x, r). Sampling trajectories from the target allows us to examine its generation behavior after OPD. The attack samples these trajectories at attack time; it does not need the trajectories sampled during training.

Step 2: Reference-model calibration. The goal of this step is to identify token positions where the target's probability is higher than what models trained without the candidate can achieve. For each sampled token $y_{i,t}$, we compute the conditional log-probability under the target T and under each reference model S_k:

$$\ell_{i,t}^{M} = \log p_M(y_{i,t} \mid x,\, y_{i,<t}), \qquad M \in \{T, S_1, \ldots, S_K\}. \tag{2}$$

All models are conditioned on the prompt x and the trajectory prefix $y_{i,<t}$. The reference answer r is used during trajectory sampling (Step 1) but is not included in the scoring input. Because training on other records can raise the same token probabilities that training on the candidate raises, we compare the target against reference models trained without the candidate (Zarifzadeh et al., 2024; Tao & Shokri, 2025). At each position, we compute the gap between the target's log-probability and the *highest* reference log-probability:

$$d_{i,t} = \ell_{i,t}^{T} - \max_{1 \le k \le K} \ell_{i,t}^{S_k}. \tag{3}$$

We take the maximum rather than the mean because a target probability above the mean may still fall within the range of the reference models. A positive $d_{i,t}$ means that the target exceeds *every* reference model at that position, providing a stricter membership signal; Figure 2(d) shows a higher member share for positive gaps. The maximum is taken independently at each position, so the reference model that attains it can differ across positions.

Step 3: Leaky ReLU aggregation. Let L_i be the number of scored tokens in the i-th trajectory. The membership score is

$$S(z) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{L_i} \sum_{t=1}^{L_i} \left([d_{i,t}]_+ - \alpha \, [-d_{i,t}]_+ \right), \tag{4}$$

where $[a]_+ = \max\{a, 0\}$ denotes the positive part. The transformation $[a]_+ - \alpha \, [-a]_+$ is the leaky rectified linear unit (Leaky ReLU) (Maas et al., 2013), which reduces to ReLU (Nair & Hinton, 2010; Glorot et al., 2011) when $\alpha = 0$. We apply Leaky ReLU to each token-level gap before averaging. Positive gaps are retained unchanged, whereas negative gaps are scaled by the small coefficient α. We include the downweighted negative component as an approximate correction to the positive contribution. We first average over all L_i scored tokens within each trajectory and then average the N trajectory scores with equal weight. A higher $S(z)$ indicates stronger membership signal. Figure 2(c) shows the distributions of $S(z)$ for members and non-members. The full procedure is summarized in Algorithm 1.

4 Experiments

We evaluate Leaky against OPD-trained models across mathematics, medical question answering, and code generation, using five models from three model families. We compare it with existing MIAs and report downstream task utility alongside attack performance. We further examine black-box access and evaluate LoRA (Hu et al., 2022) and DP-LoRA (Yu et al., 2021) as potential defenses. Finally, we study the effects of reference-model count and trajectory generation, assess membership leakage under on-policy self-distillation (OPSD) (Zhao et al., 2026), and evaluate the use of synthetic auxiliary data for reference-model training.

4.1 Experimental Setup

Models and datasets. We evaluate five students (Qwen3-1.7/4B (Yang et al., 2025), Qwen3.5-2/4B (Qwen Team, 2026), and Gemma-4-E2B (Gemma Team et al., 2026)) across three model

Algorithm 1 Leaky: membership inference using sampled trajectories

Require: Candidate record $z = (x, r)$; target T; reference models $S_1, \ldots, S_K$ trained without z; number of trajectories N; negative slope α
Ensure: Membership score $S(z)$

```
Sample y_1, ..., y_N from T, conditioned on (x, r).
for i = 1, ..., N do
    Let L_i be the number of scored tokens in y_i.
    for t = 1, ..., L_i do
        Compute ℓ^M_{i,t} = log p_M(y_{i,t} | x, y_{i,<t}) for M ∈ {T, S_1, ..., S_K}.
        d_{i,t} ← ℓ^T_{i,t} − max_k ℓ^{S_k}_{i,t}
    end for
    s_i ← (1/L_i) Σ_{t=1}^{L_i} ( [d_{i,t}]_+ − α [−d_{i,t}]_+ )
end for
return (1/N) Σ_{i=1}^{N} s_i
```

families on OpenThoughts-Math (Guha et al., 2026), MedMCQA (Pal et al., 2022), and TACO (Li et al., 2023), covering mathematics, medical question answering, and code generation. Each candidate pool contains 1,024 records: 512 members and 512 non-members. All candidates are excluded from reference-model training. We describe the data splits in Appendix B.1, and the details of the training configs are listed in Appendix B.3

Metrics. We report AUROC (Fawcett, 2006) and TPR at 1%FPR (Carlini et al., 2022) in the main tables, with 5% and 10% FPR results in Appendix C. Downstream utility is Avg@8 on AIME 2024 (Mathematical Association of America, 2024), MedQA-USMLE (Jin et al., 2021), and LiveCodeBench v6 (Jain et al., 2025) at the attacked checkpoint, with identical evaluation settings for the base and trained models. Scoring and evaluation details are in Appendices B.4 and B.5.

Baselines. We compare Leaky with reference-free attacks, including LOSS (Yeom et al., 2018), ZLIB (Carlini et al., 2021), Min-K% (Shi et al., 2024), Min-K%++ (Zhang et al., 2025a), and ICP-MIA (Lu et al., 2026), and reference-based attacks, including Ratio (Watson et al., 2021), WBC (Chen et al., 2026), EZ-MIA (Ilic et al., 2026), LiRA (Carlini et al., 2022), RMIA (Zarifzadeh et al., 2024), InfoRMIA (Tao & Shokri, 2025), and DIBA (Liu et al., 2025). Ratio, WBC, EZ-MIA, and DIBA use the base model, while LiRA, RMIA, and InfoRMIA use the same eight trained reference models as Leaky. All methods are evaluated on the same candidate records. Implementation details and hyperparameters are provided in Appendix B.2.

4.2 Main Results

Attack performance. Table 1 compares Leaky with the evaluated baselines across mathematics, medical question answering, and code generation. Leaky achieves the highest AUROC and TPR at 1% FPR on all fifteen targets, demonstrating its effectiveness both overall and at a low false-positive rate. Averaged over these targets, Leaky improves AUROC from 0.614 for the strongest baseline on each target to 0.875. Several baselines improve substantially when applied to the same sampled trajectories used by Leaky (Table 10), further supporting our motivation to examine membership signals in the student's generation behavior rather than relying on fixed reference-answer losses. Under this comparison, Leaky still outperforms all evaluated baselines on every target, supporting the effectiveness of its token-level reference-model calibration and leaky ReLU aggregation. All attacked models also outperform their base models on the corresponding downstream benchmarks (Table 7). These results show that our OPD training can indeed improve downstream task performance while being vulnerable to membership inference attacks.

Black-box attack. Leaky requires token log-probabilities, which a text-only interface may not provide. We adapt it to this setting by querying the target with trajectory prefixes and recording whether its greedy prediction matches the sampled token. These binary indicators replace token log-probabilities in the gap computation and leaky ReLU aggregation; Appendix C.6 provides the scoring formula. As shown in Table 2, the text-only attack remains effective across all fifteen targets,

Table 1: Membership inference against OPD: AUROC and TPR @ 1% FPR, leading zeros omitted, best per column in bold. Leaky scores eight target-generated trajectories per candidate. Q3/Q3.5: Qwen3/Qwen3.5; G4: Gemma-4-E2B. Results at 5% and 10% FPR are in Appendix C.

Method	Mathematics					Medical QA					Code generation				
	Q3 1.7B	Q3 4B	Q3.5 2B	Q3.5 4B	G4 E2B	Q3 1.7B	Q3 4B	Q3.5 2B	Q3.5 4B	G4 E2B	Q3 1.7B	Q3 4B	Q3.5 2B	Q3.5 4B	G4 E2B
AUROC															
LOSS	.516	.513	.502	.502	.505	.553	.552	.542	.545	.530	.546	.532	.543	.550	.559
ZLIB	.528	.525	.497	.497	.503	.509	.511	.503	.505	.504	.616	.590	.612	.616	.622
Min-K%	.521	.518	.499	.500	.506	.540	.542	.532	.544	.518	.535	.512	.533	.539	.540
Min-K%++	.521	.522	.509	.509	.515	.492	.508	.511	.515	.522	.492	.454	.502	.526	.442
ICP-MIA	.522	.542	.530	.518	.511	.539	.543	.530	.534	.524	.552	.510	.529	.544	.516
Ratio	.502	.497	.473	.504	.497	.486	.469	.519	.513	.470	.470	.489	.516	.552	.529
WBC	.497	.504	.477	.501	.517	.499	.476	.500	.495	.479	.469	.459	.569	.618	.447
EZ-MIA	.506	.517	.469	.496	.493	.534	.492	.482	.487	.481	.466	.489	.522	.556	.543
DIBA	.513	.489	.495	.488	.472	.623	.599	.589	.518	.565	.544	.535	.484	.570	.556
LiRA	.661	.660	.534	.564	.497	.712	.622	.637	.573	.478	.496	.562	.577	.574	.506
RMIA	.662	.670	.518	.520	.488	.698	.589	.630	.570	.480	.512	.563	.526	.572	.509
InfoRMIA	.666	.663	.561	.573	.517	.740	.649	.613	.579	.496	.507	.558	.577	.583	.527
Leaky	**.916**	**.922**	**.913**	**.925**	**.832**	**.966**	**.848**	**.868**	**.837**	**.826**	**.847**	**.843**	**.901**	**.846**	**.839**
TPR at 1% FPR															
LOSS	.006	.004	.008	.014	.006	.018	.018	.016	.006	.012	.006	.002	.006	.008	.004
ZLIB	.004	.006	.021	.023	.018	.014	.012	.010	.010	.006	.004	.010	.002	.002	.008
Min-K%	.006	.004	.008	.014	.006	.002	.010	.004	.004	.004	.006	.002	.006	.010	.004
Min-K%++	.000	.002	.006	.010	.006	.012	.014	.010	.018	.012	.002	.004	.004	.004	.018
ICP-MIA	.006	.016	.010	.004	.006	.006	.014	.010	.016	.008	.016	.002	.002	.012	.006
Ratio	.010	.012	.010	.012	.004	.006	.010	.016	.010	.004	.012	.004	.004	.021	.021
WBC	.010	.012	.002	.006	.014	.016	.010	.023	.027	.004	.008	.008	.014	.061	.014
EZ-MIA	.004	.006	.014	.014	.010	.008	.010	.020	.010	.004	.010	.006	.037	.021	.000
DIBA	.014	.020	.006	.010	.012	.016	.039	.018	.020	.006	.018	.016	.012	.014	.012
LiRA	.066	.055	.035	.014	.008	.148	.045	.023	.021	.018	.012	.035	.021	.020	.018
RMIA	.084	.068	.041	.025	.010	.164	.029	.041	.076	.008	.016	.035	.016	.027	.008
InfoRMIA	.064	.070	.039	.016	.023	.150	.070	.053	.025	.000	.014	.035	.033	.027	.021
Leaky	**.418**	**.576**	**.477**	**.582**	**.207**	**.740**	**.449**	**.512**	**.461**	**.350**	**.434**	**.396**	**.578**	**.527**	**.270**

Table 2: Black-box membership inference on the fifteen targets in Table 1. We report AUROC and TPR at 1% FPR for the text-only prefix-query attack.

Target model	Mathematics		Medical QA		Code generation	
	AUROC	TPR@1% FPR	AUROC	TPR@1% FPR	AUROC	TPR@1% FPR
Qwen3-1.7B	.776	.273	.854	.389	.747	.213
Qwen3-4B	.796	.289	.691	.211	.687	.146
Qwen3.5-2B	.774	.191	.768	.342	.740	.238
Qwen3.5-4B	.812	.309	.737	.189	.788	.412
Gemma-4-E2B	.727	.152	.698	.189	.660	.084

including at low false-positive rates. Restricting access to token probabilities alone therefore does not prevent membership leakage in OPD.

4.3 Potential Defenses

We evaluate LoRA (Hu et al., 2022) and DP-LoRA (Yu et al., 2021) as potential defenses on Qwen3-1.7B with a Qwen3-8B teacher across the three domains. DP-LoRA applies record-level gradient clipping and Gaussian noise (Abadi et al., 2016) at a nominal privacy budget of $\varepsilon \approx 8$. Training data and evaluation candidates are held fixed within each domain, and each configuration uses eight reference models trained with the corresponding procedure. Training settings, privacy accounting, and additional results are provided in Appendix D.

Table 3: Defenses on Qwen3-1.7B (teacher Qwen3-8B): change in Avg@8 over the base (percentage points), AUROC, and TPR at 1% FPR of Leaky with eight matched reference models. Full fine-tuning is the Table 1 cell; DP-LoRA is at nominal $\varepsilon \approx 8$. Configurations are in Appendix D.

	Mathematics			Medical QA			Coding		
Training	ΔUtil.	AUROC	TPR	ΔUtil.	AUROC	TPR	ΔUtil.	AUROC	TPR
Full fine-tuning	+5.00	0.9158	0.418	+3.98	0.9665	0.740	+0.57	0.8467	0.434
LoRA	+2.50	0.6455	0.098	+2.46	0.7868	0.268	+0.14	0.6585	0.146
DP-LoRA ($\varepsilon \approx 8$)	+1.67	0.4903	0.014	+0.04	0.5044	0.023	+0.07	0.5097	0.027

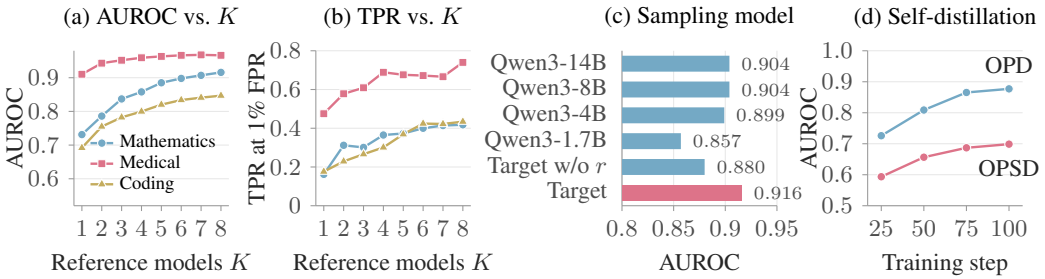


Figure 3: Ablations on Qwen3-1.7B. (a) AUROC and (b) TPR at 1% FPR against the number of reference models K in the three domains; $K = 8$ is Table 1, and the other students are in Figures C.1 and C.2. (c) Mathematics AUROC. (d) Mathematics AUROC across checkpoints for OPD with the Qwen3-8B teacher and for OPSD with the frozen initial student.

Defense performance. Table 3 reports attack performance alongside changes in downstream utility relative to the base model. LoRA reduces both AUROC and TPR at 1% FPR compared with full fine-tuning across all three domains, but Leaky still achieves AUROCs of 0.646–0.787, indicating that membership information remains detectable. LoRA also retains positive utility gains, although these are smaller than those of full fine-tuning. Adding clipping and noise brings AUROC close to chance and reduces TPR at 1% FPR to 0.014–0.027, but further limits the downstream gains. DP-LoRA retains a 1.67 percentage-point improvement in AIME Avg@8 over the base model, whereas its gains on medical QA and code generation are below 0.1 percentage points. These results illustrate a trade-off between resistance to Leaky and downstream utility in the evaluated configurations.

4.4 Ablation Study

We examine the effects of reference-model count, trajectory generation, and self-distillation. Additional ablations of reference-model calibration and token aggregation are provided in Appendix C.2.

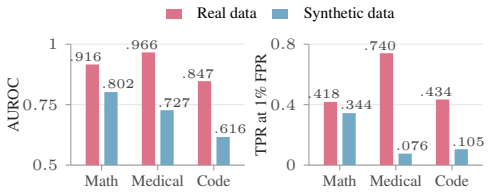
Number of reference models. Figures 3(a,b) show the effect of reference-model count on Qwen3-1.7B across the three domains, with results for the remaining students in Appendix C.4. Increasing K from one to eight improves both AUROC and TPR at 1% FPR on all fifteen targets, with a mean AUROC gain of 0.15. Four reference models already capture most of this gain on fourteen targets, although eight achieve higher AUROC on every target. These results indicate that a small reference ensemble provides useful calibration, while additional models further improve attack performance.

Trajectory generation. Figure 3(c) compares trajectory-generation choices on the mathematics Qwen3-1.7B target, with the target and reference models held fixed. Trajectories generated by the target, the training teacher, and two alternative models yield similar AUROCs of 0.899–0.916, whereas the frozen base model yields 0.857. Providing the reference answer during target sampling improves AUROC from 0.880 to 0.916. These results support target-generated trajectories as an effective choice and show that reference-answer conditioning strengthens, but is not necessary for, the observed membership signal. Further details are provided in Appendix C.3.

OPSD. We further evaluate Leaky under OPSD setup, where the teacher is a frozen copy of the initial student conditioned on the problem and reference answer (Zhao et al., 2026; Shenfeld et al., 2026).

Table 4: Auxiliary data for reference-model training under the Table 1 protocol; the real-data and OpenThoughts-Math columns are the Table 1 cells. (a) Qwen3-1.7B targets, reference models trained on real or Qwen3.5-27B-synthesized records. (b) Mathematics targets, reference models trained on OpenThoughts-Math or DeepMath-103K.

(a) Synthetic auxiliary data



(b) External mathematics data

	OpenThoughts-Math		DeepMath-103K	
Target	AUROC	TPR	AUROC	TPR
Qwen3-1.7B	.916	.418	.772	.164
Qwen3-4B	.922	.576	.767	.193
Qwen3.5-2B	.913	.477	.740	.135
Qwen3.5-4B	.925	.582	.884	.361
Gemma-4-E2B	.832	.207	.723	.123

On mathematics Qwen3-1.7B, the OPSD run has lower AUROC than the external-teacher OPD run at every evaluated checkpoint, while AUROC increases with training under both procedures (Figure 3(d)). These results show that membership leakage also persists in the evaluated self-distilled student. Full settings and results are provided in Appendix C.5.

4.5 Auxiliary Data for Reference Models

When auxiliary data from the target's training distribution are limited, we consider synthetic records and an external dataset for reference-model training. For synthesis, Qwen3.5-27B (Qwen Team, 2026) generates new problems and reference answers based on candidate records, without using membership labels. These records support effective attacks in mathematics and medical question answering, while the weaker performance in coding may reflect the difficulty of jointly matching problem requirements and program structure (Table 4(a)). Replacing OpenThoughts-Math auxiliary data with DeepMath-103K (He et al., 2025b) still yields effective attacks across all five mathematics targets (Table 4(b)). This shows that reference models can capture membership signals using auxiliary data from another dataset in the same domain.

5 Related Work

Privacy in distillation. Distillation can expose membership information about teacher and student training data (Jagielski et al., 2023; Zhang et al., 2025b). We study reference-conditioned OPD, where reference answers guide teacher feedback rather than serve as fixed student targets. Our empirical auditing complements private training methods such as DP-OPD (Khadem et al., 2026).

Membership inference in language models. Existing attacks exploit token selection (Shi et al., 2024; Zhang et al., 2025a) and reference-model calibration (Carlini et al., 2022; Zarifzadeh et al., 2024). Related methods examine token-level evidence through InfoRMIA (Tao & Shokri, 2025), local windows through WBC (Chen et al., 2026) and directional scoring in EZ-MIA (Ilic et al., 2026). Other work audits RLVR through sampled generations (Liu et al., 2025) and studies membership inference without target probabilities (Kaneko et al., 2025; He et al., 2025a). In contrast to these approaches, Leaky extracts membership signals from fresh student trajectories through token-wise maximum-reference calibration and asymmetric aggregation.

6 Conclusion

We study membership leakage in reference-conditioned on-policy distillation and show that fresh student trajectories reveal signals that fixed reference-answer losses can miss. We introduce Leaky, which combines token-level reference-model calibration with asymmetric aggregation and outperforms all evaluated baselines across fifteen targets in three domains. Leakage remains detectable through text-only trajectory-prefix queries, while DP-LoRA reduces Leaky's AUROC to near chance with smaller utility gains in the evaluated settings. These findings show that excluding reference answers from the student's training input does not by itself ensure membership privacy, motivating privacy audits that examine generation behavior rather than relying on fixed-answer likelihoods.

AI USE STATEMENT

In this work, we used generative AI tools to implement and execute parts of the experimental pipeline, namely to write and debug the job scripts for training, trajectory sampling, and scoring, and to launch and monitor these runs. We also used a generative model to produce synthetic data: the synthetic auxiliary data of Section 4.5 are generated by Qwen3.5-27B from the candidate records, as described there. We have not used generative AI tools to develop theoretical models or conceptual frameworks, formulate mathematical claims, provide ingredients for or write proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, clean or reformat datasets, or interpret results; translation and qualitative or thematic data analysis are not applicable to this work. Additionally, we used generative AI tools to edit the paper for readability, to create and debug software code, and to search for and identify relevant literature. We have reviewed all AI-assisted work: AI-edited text was checked by the authors for meaning and accuracy; AI-assisted code was reviewed and tested by the authors, and the experiment runs were verified against their logs and outputs; and all literature searched by AI tools was read by the authors before being cited. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies membership inference against language models trained with on-policy distillation. Its purpose is to characterize a privacy risk of this training paradigm so that practitioners can assess and mitigate it. All experiments use publicly available checkpoints and public datasets (OpenThoughts-Math, MedMCQA, and TACO); the medical data consist of examination questions rather than patient records, and no personal, proprietary, or human-subject data are involved. The attacked targets and reference models are trained by us, and we do not attack any deployed system or expose training data belonging to third parties. Membership inference is a dual-use technique: the same procedure could be applied to a deployed OPD-trained model. The attack, however, requires the base model and the training configuration as described in the threat model of Section 2.2, and it builds on components that are already standard in the membership inference literature. We also evaluate LoRA and DP-LoRA as defenses in Section 4.3 and Appendix D. We believe that documenting this leakage, together with the conditions under which it is reduced, benefits practitioners who train models on sensitive data more than it enables misuse.

REPRODUCIBILITY STATEMENT

The attack is fully specified in Section 3.2, with the token-level gap and the aggregation defined in Equations (3) and (4) and the complete procedure in Algorithm 1. Appendix A analyzes the roles of reference-model calibration and the negative slope and reports the sensitivity to the slope. Appendix B.1 describes the datasets, the filtering steps, and the construction of the member and non-member candidate sets; Appendix B.2 specifies every baseline; Appendix B.3 lists the student–teacher pairings and the training configurations (Tables 5 and 6); Appendix B.4 gives the sampling and scoring settings of Leaky and of its black-box variant; and Appendix B.5 gives the utility evaluation and decoding settings. Defense configurations, including the DP accounting, are in Appendix D; and Appendix B.6 lists the hardware and software versions. Code for training, trajectory sampling, scoring, and evaluation is available in an anonymous repository at `https://anonymous.4open.science/r/LEAKY_STUDENTS-8B11`.

REFERENCES

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 308–318, 2016.

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, volume 2024, pp. 21246–21263, 2024.

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.

Yuetian Chen, Yuntao Du, Kaiyuan Zhang, Ashish Kundu, Charles Fleming, Bruno Ribeiro, and Ninghui Li. Window-based membership inference attacks against fine-tuned large language models. *CoRR*, abs/2601.02751, 2026. doi: 10.48550/ARXIV.2601.02751. URL https://doi.org/10.48550/arXiv.2601.02751.

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *CoRR*, abs/2501.12948, 2025. doi: 10.48550/ARXIV.2501.12948. URL https://doi.org/10.48550/arXiv.2501.12948.

Tom Fawcett. An introduction to ROC analysis. *Pattern Recognit. Lett.*, 27(8):861–874, 2006. doi: 10.1016/J.PATREC.2005.10.010. URL https://doi.org/10.1016/j.patrec.2005.10.010.

Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, et al. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In Geoffrey J. Gordon, David B. Dunson, and Miroslav Dudík (eds.), *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, volume 15 of *JMLR Proceedings*, pp. 315–323. JMLR.org, 2011. URL http://proceedings.mlr.press/v15/glorot11a/glorot11a.pdf.

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, et al. OpenThoughts: Data recipes for reasoning models. In *International Conference on Learning Representations*, volume 2026, pp. 108059–108130, 2026.

Yu He, Boheng Li, Liu Liu, Zhongjie Ba, Wei Dong, Yiming Li, Zhan Qin, Kui Ren, and Chun Chen. Towards label-only membership inference attack against pre-trained large language models. In *34th USENIX Security Symposium (USENIX Security 25)*, pp. 1609–1628, 2025a.

Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. DeepMath-103K: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning, 2025b. URL https://arxiv.org/abs/2504.11456.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

David Ilic, David Stanojevic, and Kostadin Cvejoski. Powerful training-free membership inference against fine-tuned autoregressive language models. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 14077–14093. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.ACL-LONG.640. URL https://doi.org/10.18653/v1/2026.acl-long.640.

Matthew Jagielski, Milad Nasr, Katherine Lee, Christopher A Choquette-Choo, Nicholas Carlini, and Florian Tramer. Students parrot their teachers: Membership inference on model distillation. *Advances in Neural Information Processing Systems*, 36:44382–44397, 2023.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=chfJJYC3iL.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? A large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421, 2021.

Masahiro Kaneko, Youmi Ma, Yuki Wata, and Naoaki Okazaki. Sampling-based pseudo-likelihood for membership inference attacks. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, volume ACL 2025 of *Findings of ACL*, pp. 8894–8907. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.FINDINGS-ACL.465. URL https://doi.org/10.18653/v1/2025.findings-acl.465.

Fatemeh Khadem, Sajad Mousavi, Yi Fang, and Yuhong Liu. DP-OPD: differentially private on-policy distillation for language models. *CoRR*, abs/2604.04461, 2026. doi: 10.48550/ARXIV.2604.04461. URL https://doi.org/10.48550/arXiv.2604.04461.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In Jason Flinn, Margo I. Seltzer, Peter Druschel, Antoine Kaufmann, and Jonathan Mace (eds.), *Proceedings of the 29th Symposium on Operating Systems Principles, SOSP 2023, Koblenz, Germany, October 23-26, 2023*, pp. 611–626. ACM, 2023. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tülu 3: Pushing frontiers in open language model post-training. *CoRR*, abs/2411.15124, 2024. doi: 10.48550/ARXIV.2411.15124. URL https://doi.org/10.48550/arXiv.2411.15124.

Rongao Li, Jie Fu, Bo-Wen Zhang, Tao Huang, Zhihong Sun, Chen Lyu, Guang Liu, Zhi Jin, and Ge Li. TACO: Topics in algorithmic code generation dataset. *arXiv preprint arXiv:2312.14852*, 2023.

Chen Lin, Kedi Chen, and Wei Zhang. ReNIO: Reweighting negative trajectory importance for LLM on-policy distillation. *arXiv preprint arXiv:2606.23104*, 2026.

Yule Liu, Heyi Zhang, Jinyi Zheng, Zhen Sun, Zifan Peng, Jiaheng Wei, Tianshuo Cong, Yilong Yang, and Xinlei He. Auditing data membership in reinforcement learning with verifiable rewards. *arXiv preprint arXiv:2511.14045*, 2025.

Kevin Lu and Thinking Machines Lab. On-policy distillation. *Thinking Machines Lab: Connectionism*, 2025. doi: 10.64434/tml.20251026. https://thinkingmachines.ai/blog/on-policy-distillation.

Zhexi Lu, Hongliang Chi, Nathalie Baracaldo, Swanand Ravindra Kadhe, Yuseok Jeon, and Lei Yu. In-context probing for membership inference in fine-tuned language models. In *33rd Annual Network and Distributed System Security Symposium, NDSS 2026, San Diego, California, USA, February 23-27, 2026*. The Internet Society, 2026.

Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, 2013.

Mathematical Association of America. American invitational mathematics examination (AIME) 2024. https://artofproblemsolving.com/wiki/index.php/2024_AIME_I, 2024. Problems from AIME I and AIME II, 2024.

Ilya Mironov. Rényi differential privacy. In *30th IEEE Computer Security Foundations Symposium, CSF 2017, Santa Barbara, CA, USA, August 21-25, 2017*, pp. 263–275. IEEE Computer Society, 2017. doi: 10.1109/CSF.2017.11. URL https://doi.org/10.1109/CSF.2017.11.

Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted Boltzmann machines. In Johannes Fürnkranz and Thorsten Joachims (eds.), *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pp. 807–814. Omnipress, 2010. URL https://icml.cc/Conferences/2010/papers/432.pdf.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. MedMCQA: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Conference on health, inference, and learning*, pp. 248–260. PMLR, 2022.

Emiliano Penaloza, Dheeraj Vattikonda, Nicolas Gontier, Alexandre Lacoste, Laurent Charlin, and Massimo Caccia. Privileged information distillation for language models. *arXiv preprint arXiv:2602.04942*, 2026.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-distillation enables continual learning. *CoRR*, abs/2601.19897, 2026. doi: 10.48550/ARXIV.2601.19897. URL https://doi.org/10.48550/arXiv.2601.19897.

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

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

Jiashu Tao and Reza Shokri. (Token-Level) InfoRMIA: Stronger membership inference and memorization assessment for LLMs. *CoRR*, abs/2510.05582, 2025. doi: 10.48550/ARXIV.2510.05582. URL https://doi.org/10.48550/arXiv.2510.05582.

Lauren Watson, Chuan Guo, Graham Cormode, and Alex Sablayrolles. On the importance of difficulty calibration in membership inference attacks. *arXiv preprint arXiv:2111.08440*, 2021.

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

Tianzhu Ye, Li Dong, Xun Wu, Shaohan Huang, and Furu Wei. On-policy context distillation for language models. *CoRR*, abs/2602.12275, 2026. doi: 10.48550/ARXIV.2602.12275. URL https://doi.org/10.48550/arXiv.2602.12275.

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.

Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, Karthik Prasad, Mani Malek, John Nguyen, Sayan Ghosh, Akash Bharadwaj, Jessica Zhao, Graham Cormode, and Ilya Mironov. Opacus: User-friendly differential privacy library in PyTorch. *CoRR*, abs/2109.12298, 2021. URL https://arxiv.org/abs/2109.12298.

Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, et al. Differentially private fine-tuning of language models. *arXiv preprint arXiv:2110.06500*, 2021.

Sajjad Zarifzadeh, Philippe Liu, and Reza Shokri. Low-cost high-power membership inference attacks. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pp. 58244–58282. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/zarifzadeh24a.html.

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

Ziqi Zhang, Ali Shahin Shamsabadi, Hanxiao Lu, Yifeng Cai, and Hamed Haddadi. Membership and memorization in LLM knowledge distillation. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 20074–20084. Association for Computational Linguistics, 2025b. doi: 10.18653/V1/2025.EMNLP-MAIN.1015. URL https://doi.org/10.18653/v1/2025.emnlp-main.1015.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark W. Barrett, and Ying Sheng. SGLang: Efficient execution of structured language model programs. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/724be4472168f31ba1c9ac630f15dec8-Abstract-Conference.html.

A Understanding Leaky

A.1 Role of Reference-Model Calibration

OPD changes token predictions through both the candidate record and other training records. The target's probability increase relative to its initialization therefore includes changes shared across records. Reference models trained with the same procedure but without the candidate help account for these shared changes. On the mathematics Qwen3-1.7B target analyzed in the main text, replacing the base-model log-probability with the mean across trained reference models raises AUROC from 0.513 to 0.853, with the candidates, trajectories, and averaging procedure fixed. This comparison supports using trained models for calibration.

A.2 Negative Gaps as a Correction

A model trained without the candidate can still exceed all reference models at some positions. The positive contribution may therefore increase for two reasons: membership can give the target an additional lead, and larger disagreement among models can produce a larger chance lead. Keeping only positive gaps includes both contributions in the score. Negative gaps measure how far the target trails the same reference and can help account for the latter contribution.

To describe this effect, consider a fixed prefix and token. As an idealized model, suppose that the log-probabilities of models trained without the candidate satisfy

$$\ell_j = \mu + \sigma Z_j, \qquad \sigma > 0, \tag{5}$$

where the Z_j are independent and identically distributed. The location μ and scale σ may vary across records and positions, while the standardized distribution remains the same. Let $m_k = \mathbb{E}[\max_{1\le j\le k} Z_j]$, assuming finite first moments. Write M_K for the maximum log-probability among the K reference models and $d = \ell_T - M_K$. The positive gap is the increase in the maximum when the target is added, while the negative magnitude is the target's shortfall from this enlarged maximum:

$$[d]_+ = \max(\ell_T, M_K) - M_K, \tag{6}$$

$$[-d]_+ = \max(\ell_T, M_K) - \ell_T. \tag{7}$$

Their expectations are

$$\mathbb{E}[[d]_+] = \sigma(m_{K+1} - m_K), \qquad \mathbb{E}[[-d]_+] = \sigma(m_{K+1} - m_1). \tag{8}$$

Both share the scale factor σ. For a nonzero expected negative magnitude, setting

$$c_K = \frac{m_{K+1} - m_K}{m_{K+1} - m_1} \quad \text{gives} \quad \mathbb{E}[[d]_+ - c_K\,[-d]_+] = 0. \tag{9}$$

Under this model, subtracting a scaled negative contribution removes the mean chance lead associated with model disagreement. This gives the leaky ReLU form and explains a possible correction role for negative gaps. The coefficient depends on both the number of references and the standardized distribution.

The derivation concerns fixed scoring positions. We examine its relevance on the actual target-generated trajectories by letting each reference model in turn act as the target against the other seven. None of these reference models was trained on any candidate. In the medical Qwen3-1.7B experiment, we measure disagreement by averaging the tokenwise standard deviation of reference log-probabilities. The Spearman correlation between the positive score and this disagreement measure, averaged over the eight choices of acting target, decreases from 0.331 to 0.011 after subtracting the negative contribution with weight 0.04. Computing the score and disagreement on separate sets of trajectories gives a similar reduction. These observations support a correction that weakens the tendency of positive scores to increase with model disagreement.

A.3 Choice of the Negative Slope and Empirical Results

Approximating the standardized predictions Z_j by independent standard normal variables gives $c_8 \approx 0.041$ in Equation (9). This approximation provides a basis for a negative slope of about 0.04. We use $\alpha = 0.04$ for all targets.

We evaluate the effect of the slope while keeping the candidates, sampled trajectories, reference models, and token scores fixed. Figure A.1 shows the results for each model and domain. On target-generated trajectories, introducing the negative contribution raises mean AUROC over the fifteen targets from 0.843 at $\alpha = 0$ to 0.875 at $\alpha = 0.04$, close to the highest tested mean of 0.877 at $\alpha = 0.05$.

On target-generated trajectories, every tested slope in $\{0.04, 0.05, 0.06, 0.08\}$ outperforms $\alpha = 0$ on all fifteen targets. The best tested slope varies across targets, but the improvement extends over neighboring values. These results support using a common small negative slope and show that the gain does not depend on choosing exactly 0.04. The figure also reports teacher-generated trajectories as a supplementary comparison.

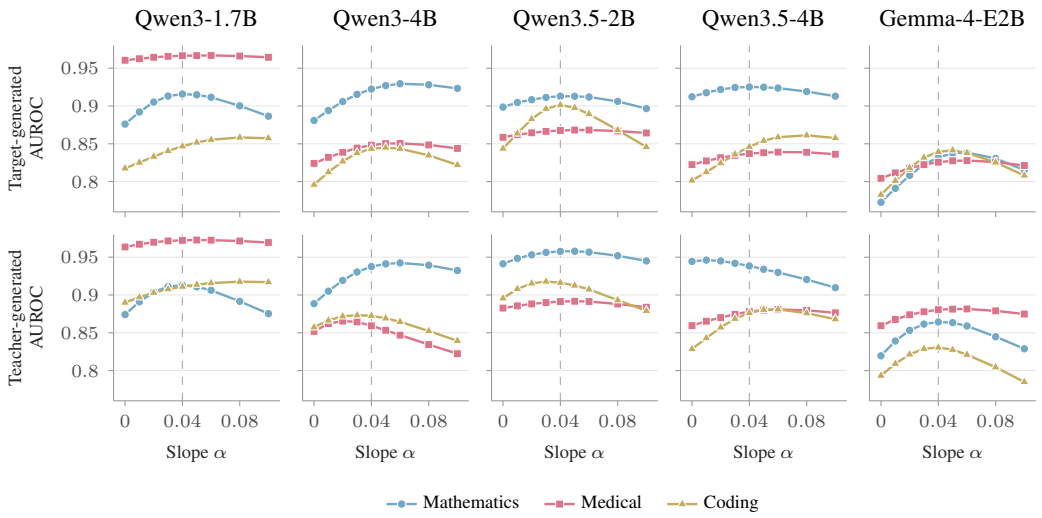


Figure A.1: Gray-box AUROC against the rectifier slope α for each target, on target-generated (top) and teacher-generated (bottom) trajectories; the dashed line marks $\alpha = 0.04$.

B Experimental Details

B.1 Datasets and Membership Evaluation

We evaluate membership inference on mathematics, medical question answering, and code generation. Each candidate record $z = (x, r)$ contains a task input x and a reference answer r. Membership is determined by whether the record was included in the target's OPD training set. For each evaluation, we initially construct a candidate pool of 1,024 records, containing 512 members and 512 non-members. All evaluated candidates are excluded from reference-model training. Comparisons between attack methods on the same target use the same candidate set.

Mathematics. We use OpenThoughts-Math, with 29,434 mathematical problems and their reference solutions. The problem statement forms x, and the corresponding solution forms r. Each target is trained on 512 problems, all of which are included as members in the attack evaluation. We sample 512 non-members from the remaining problems, excluding records used for target or reference-model training. This gives a balanced initial candidate pool of 1,024 problem–solution pairs.

Medical question answering. We use the training split of MedMCQA. We retain questions with a single correct answer among four options and an explanation of at least 20 words, and remove duplicate questions. The input x contains the question, the four options labeled A–D, and an instruction to return the correct option letter. The reference answer r contains the provided explanation followed by the correct option. Each target is trained on 16,384 records, including the 512 evaluated members. We select 512 non-members from the same filtered dataset, excluding records used for target or reference-model training. All medical targets use the same member and non-member candidates.

Code generation. We use TACO, which contains programming problems collected from multiple sources, including Codeforces and AtCoder. The input x is the problem statement, and the reference answer r is the associated Python solution formatted as a code block. Each target is trained on 8,339 records. We select 512 members from the training set and 512 non-members from outside it. To account for differences in problem composition, we divide records into strata according to their source, whether a function name is specified, and problem-statement length. Length is measured in characters and divided into three groups using the training-pool terciles. We select equal numbers of members and non-members within each stratum and match their available difficulty labels. This construction balances the two groups with respect to source, interface format, length group, and annotated difficulty.

B.2 Baselines

We evaluate all baselines on the same candidates as Leaky. **LOSS** (Yeom et al., 2018) uses the mean token-level loss on the reference answer, while **ZLIB** (Carlini et al., 2021) normalizes the total loss by the compressed byte length of the reference answer. **Min-K%** (Shi et al., 2024) averages the lowest 20% of token log-probabilities. **Min-K%++** (Zhang et al., 2025a) uses the same selection ratio after standardizing each token's log-probability using the corresponding vocabulary distribution.

Ratio (Watson et al., 2021), **WBC** (Chen et al., 2026), and **EZ-MIA** (Ilic et al., 2026) use the frozen checkpoint that initializes the target as the reference model. Ratio measures the reduction in reference-answer loss relative to this model. WBC compares target and reference losses over sliding windows, using ten window sizes spanning 2–40 tokens and averaging the fraction of windows in which the target has lower loss. EZ-MIA restricts the comparison to positions where the target's greedy prediction differs from the reference token and computes the ratio of positive to negative target-minus-reference log-probability differences.

LiRA (Carlini et al., 2022), **RMIA** (Zarifzadeh et al., 2024), and **InfoRMIA** (Tao & Shokri, 2025) use the same eight reference models as Leaky, trained without the evaluated candidates. We use offline LiRA with the loss statistic, standardized by its mean and standard deviation across reference models. We use offline RMIA with $\gamma = 1$. For InfoRMIA, the reference distribution is the arithmetic mean of the reference probability distributions; we use the token-level score with mean aggregation across tokens.

ICP-MIA (Lu et al., 2026) uses the reference-probe variant with ten probes per candidate. We retrieve problem–answer pairs from a held-out pool using cosine similarity between all-MiniLM-L6-v2 embeddings, excluding the evaluated candidates. Each probe is added separately as an in-context demonstration, and the attack measures the resulting change in the target's reference-answer log-probability.

DIBA (Liu et al., 2025) samples eight trajectories per candidate from each of the target and base models at temperature 0.5. It combines correctness features with features measuring differences in the models' generation behavior, using a random-forest and logistic-regression stacking classifier. We train the classifier with membership labels under five-fold stratified cross-validation and compute attack results from pooled out-of-fold predictions. For coding, the two correctness features are fixed to zero, leaving five varying features.

B.3 Models and Training

Student–teacher pairings. We evaluate five students from three model families: Qwen3-1.7B and Qwen3-4B (Yang et al., 2025), Qwen3.5-2B and Qwen3.5-4B (Qwen Team, 2026), and Gemma-4-E2B (Gemma Team et al., 2026). Each student is trained separately on mathematics, medical question answering, and code generation, giving fifteen target models. Each target uses a larger teacher from the same model family. Table 5 lists the student–teacher pairings for each domain.

OPD training. All targets in Table 1 use full-parameter fine-tuning with a frozen teacher. The student receives the problem x, while the teacher receives both x and the reference answer r. During training, the student samples trajectories from its current policy at temperature 1.1 and learns from the teacher's next-token distributions on these trajectories. The distillation loss uses a generalized Jensen–Shannon divergence with interpolation weight β and per-token clipping. Mathematics experiments use 126 training steps with a global batch size of 32. Medical and coding experiments use

Table 5: Student and teacher models used for OPD in each domain.

Student	Mathematics teacher	Medical teacher	Coding teacher
Qwen3-1.7B	Qwen3-8B	Qwen3-8B	Qwen3-8B
Qwen3-4B	Qwen3-14B	Qwen3-14B	Qwen3-14B
Qwen3.5-2B	Qwen3.5-27B	Qwen3.5-9B	Qwen3.5-9B
Qwen3.5-4B	Qwen3.5-27B	Qwen3.5-27B	Qwen3.5-27B
Gemma-4-E2B	Gemma-4-26B-A4B	Gemma-4-26B-A4B	Gemma-4-26B-A4B

512 training steps with a global batch size of 128. Table 6 summarizes the training configurations. All teacher models use thinking mode, while student models use non-thinking mode.

Table 6: OPD training configurations. Training steps indicate the total training budget.

Student	Interpolation weight (β)	Learning rate	Training steps	Global batch size
Mathematics				
Qwen3-1.7B	0	5e-6	126	32
Qwen3-4B	0	2e-6	126	32
Qwen3.5-2B	0.5	1e-6	126	32
Qwen3.5-4B	0.5	1e-6	126	32
Gemma-4-E2B	0	3e-6	126	32
Medical question answering				
Qwen3-1.7B	0	1e-5	512	128
Qwen3-4B	0	4e-6	512	128
Qwen3.5-2B	0.5	2e-6	512	128
Qwen3.5-4B	0.5	1e-6	512	128
Gemma-4-E2B	0.5	5e-6	512	128
Code generation				
Qwen3-1.7B	0	2e-6	512	128
Qwen3-4B	0	2e-6	512	128
Qwen3.5-2B	0.5	1e-6	512	128
Qwen3.5-4B	0.5	2e-6	512	128
Gemma-4-E2B	0.5	5e-6	512	128

B.4 Leaky Implementation

Trajectory sampling. For each candidate record $z = (x, r)$, we sample eight trajectories from the target model conditioned on the problem x and reference answer r. We use a sampling temperature of 1.1, top-p of 0.95, and top-k of 20. These trajectories are generated at attack time.

Token scoring and aggregation. We score each sampled token using the target and eight reference models, conditioned on the same problem x and trajectory prefix. The reference answer r is used for sampling and is not separately appended to the scoring input. Token log-probabilities are computed at temperature 1.1. We compute the membership score using the leaky ReLU aggregation defined in Section 3.2, first averaging over all scored tokens within each trajectory and then averaging the trajectory scores with equal weight. We use the same negative slope $\alpha = 0.04$ for all targets. Appendix A.3 explains this choice and reports the sensitivity to α.

Blackbox implementation. The black-box setting uses one greedy next-token query per scored prefix, giving $Q(z) = \sum_i L_i$ scoring queries per candidate. We score up to 1,024 positions per trajectory for Qwen3 and all coding targets, and up to 512 for Qwen3.5 and Gemma-4 mathematics and medical targets. With eight trajectories, the mean number of scoring queries per candidate ranges from 4,045 to 8,192 across targets; these counts exclude trajectory generation and local reference-model computation. Appendix C.6 defines the score and reports the results.

B.5 Downstream Utility

Table 7: Downstream utility (Avg@8, %) of the evaluated targets; Δ is the change from the base in percentage points. Training steps indicate the total training budget.

Student	Training steps	Base	Trained	Δ (pp)
Mathematics: AIME 2024				
Qwen3-1.7B	126	47.50	52.50	+5.00
Qwen3-4B	126	72.92	75.00	+2.08
Qwen3.5-2B	126	31.25	42.50	+11.25
Qwen3.5-4B	126	85.00	87.08	+2.08
Gemma-4-E2B	126	41.93	45.00	+3.07
Medical QA: MedQA-USMLE				
Qwen3-1.7B	512	49.00	52.98	+3.98
Qwen3-4B	512	65.63	70.38	+4.75
Qwen3.5-2B	512	60.17	66.01	+5.84
Qwen3.5-4B	512	83.77	86.69	+2.92
Gemma-4-E2B	512	63.58	65.00	+1.42
Code generation: LiveCodeBench v6				
Qwen3-1.7B	512	31.00	31.57	+0.57
Qwen3-4B	512	45.79	46.71	+0.92
Qwen3.5-2B	512	20.14	22.79	+2.65
Qwen3.5-4B	512	55.43	57.57	+2.14
Gemma-4-E2B	512	43.50	44.57	+1.07

Table 8: Utility decoding per base/target pair (not used for attack scoring). Coding uses SGLang (Zheng et al., 2024) for Qwen3-1.7B and Qwen3.5 and vLLM (Kwon et al., 2023) otherwise.

Model family	Decoding	Output tokens
Mathematics: reasoning enabled; 30 AIME questions		
Qwen3	$T = 1.0$, top-$p = 0.95$, no top-k	38,912
Qwen3.5	$T = 1.0$, top-$p = 0.95$, top-$k = 20$, presence penalty 1.5	81,920
Gemma-4	$T = 1.0$, top-$p = 0.95$, top-$k = 64$	32,768
Medical: reasoning disabled; 1,273 MedQA-USMLE questions		
All models	$T = 0.7$, top-$p = 0.8$, top-$k = 20$, presence penalty 1.5	32,768
Coding: reasoning enabled; 175 LiveCodeBench v6 questions		
Qwen3	$T = 0.6$, top-$p = 0.95$, top-$k = 20$	32,768
Qwen3.5	$T = 1.0$, top-$p = 0.95$, top-$k = 20$, presence penalty 1.5	81,920
Gemma-4	$T = 1.0$, top-$p = 0.95$, top-$k = 64$	32,768

B.6 Hardware and Software

Hardware. All experiments run on single servers of an NVIDIA GB200 NVL72 system. Each server has four NVIDIA B200 GPUs with 184 GiB of HBM3e memory each, connected by NVLink, two 72-core NVIDIA Grace CPUs, and about 1.5 TB of host memory. OPD training of targets and reference models, including the LoRA and DP-LoRA jobs, uses all four GPUs of a server, with the teacher served on the same GPUs. Trajectory sampling, scoring, and downstream utility evaluation use one to four GPUs per job.

Software. Jobs run in the NVIDIA NGC vLLM containers (releases 25.12 and 26.07; Python 3.12). OPD training builds on the ReNIO implementation (Lin et al., 2026) with Transformers 4.57.1, TRL 0.26.0, Accelerate 1.11.0 in bf16 mixed precision, and DeepSpeed 0.18.2; the teacher's next-token distributions are computed with vLLM (Kwon et al., 2023). LoRA uses PEFT 0.17.1, and DP-LoRA uses Opacus 1.6.0 (Yousefpour et al., 2021) for per-record clipping, noise addition,

and privacy accounting. Trajectory sampling and token log-probability scoring for Leaky and all baselines use vLLM. Downstream utility decoding uses vLLM or SGLang 0.5.18 (Zheng et al., 2024) as listed in Table 8. Our code is available at https://anonymous.4open.science/r/LEAKY_STUDENTS-8B11.

C Additional Results and Ablations

C.1 Additional Attack Results

We further evaluate attack performance at 5% and 10% FPR to complement the results at 1% FPR in the main table. As shown in Table 9, Leaky achieves the highest TPR among the evaluated methods on all fifteen targets at both FPR levels. Its advantage thus remains under more relaxed false-positive constraints.

Table 9: TPR at 5% and 10% FPR for the methods in Table 1. Leading zeros omitted; Q3/Q3.5: Qwen3/Qwen3.5; G4: Gemma-4-E2B.

Method	Mathematics					Medical QA					Code generation				
	Q3 1.7B	Q3 4B	Q3.5 2B	Q3.5 4B	G4 E2B	Q3 1.7B	Q3 4B	Q3.5 2B	Q3.5 4B	G4 E2B	Q3 1.7B	Q3 4B	Q3.5 2B	Q3.5 4B	G4 E2B
TPR at 5% FPR															
LOSS	.045	.035	.049	.070	.055	.039	.053	.047	.049	.041	.033	.043	.033	.035	.047
ZLIB	.043	.039	.059	.051	.045	.051	.049	.041	.051	.049	.049	.057	.055	.082	.082
Min-K%	.043	.039	.049	.070	.055	.043	.035	.043	.039	.029	.031	.037	.041	.045	.053
Min-K%++	.031	.037	.062	.047	.053	.047	.041	.059	.053	.061	.035	.037	.035	.061	.057
ICP-MIA	.047	.074	.029	.043	.035	.047	.045	.035	.064	.047	.072	.027	.037	.031	.029
Ratio	.039	.051	.043	.035	.039	.041	.033	.039	.047	.045	.039	.021	.045	.070	.057
WBC	.049	.053	.016	.031	.049	.045	.041	.064	.061	.033	.023	.023	.086	.139	.031
EZ-MIA	.029	.047	.037	.047	.047	.049	.064	.041	.053	.041	.039	.064	.102	.068	.000
DIBA	.039	.066	.035	.049	.035	.088	.104	.107	.055	.072	.064	.066	.057	.090	.064
LiRA	.180	.168	.084	.078	.053	.260	.141	.160	.125	.045	.072	.102	.117	.117	.078
RMIA	.189	.191	.111	.066	.047	.273	.119	.125	.156	.031	.064	.092	.070	.115	.062
InfoRMIA	.172	.170	.104	.104	.068	.311	.164	.123	.125	.057	.070	.086	.098	.117	.078
Leaky	.662	.693	.701	.762	.434	.865	.559	.646	.641	.508	.568	.572	.689	.643	.467
TPR at 10% FPR															
LOSS	.078	.070	.123	.107	.111	.107	.105	.107	.113	.076	.072	.090	.070	.088	.094
ZLIB	.105	.088	.104	.115	.104	.090	.100	.088	.082	.102	.119	.170	.129	.160	.178
Min-K%	.070	.074	.117	.111	.111	.094	.092	.078	.104	.086	.090	.096	.074	.092	.084
Min-K%++	.098	.096	.129	.102	.092	.102	.090	.119	.121	.094	.082	.062	.080	.094	.098
ICP-MIA	.107	.158	.084	.107	.094	.121	.123	.104	.113	.090	.109	.084	.082	.082	.070
Ratio	.102	.104	.086	.076	.066	.090	.072	.086	.102	.113	.062	.055	.086	.104	.119
WBC	.096	.078	.057	.072	.098	.096	.080	.104	.105	.090	.059	.045	.154	.195	.062
EZ-MIA	.068	.086	.082	.076	.092	.102	.113	.113	.117	.107	.061	.117	.158	.123	.000
DIBA	.102	.098	.078	.080	.076	.211	.205	.193	.119	.121	.107	.107	.100	.168	.117
LiRA	.287	.242	.154	.137	.094	.363	.223	.256	.229	.088	.133	.158	.166	.207	.131
RMIA	.295	.293	.154	.133	.072	.379	.213	.248	.229	.104	.129	.186	.168	.188	.109
InfoRMIA	.268	.271	.174	.160	.105	.391	.250	.201	.201	.125	.141	.184	.156	.229	.170
Leaky	.758	.799	.805	.811	.557	.928	.666	.703	.686	.600	.633	.643	.752	.697	.590

C.2 Baselines and Scoring Rules on the Sampled Trajectories

We apply the baselines and several scoring rules to the same candidates and eight target-generated trajectories used in Table 1. Each statistic is computed per trajectory, under the student prompt and scoring temperature of Table 1, and then averaged over the eight trajectory scores. This comparison examines whether scoring the sampled trajectories is sufficient, and how reference models and token aggregation affect membership inference.

Baselines on the sampled trajectories. Table 10 reports the baselines of Table 1 on the trajectories instead of the reference answer; DIBA, which generates its own rollouts, is not included. Scores without OPD-trained reference models remain close to chance on the trajectories (AUROC 0.47–0.61), whereas LiRA, RMIA, and InfoRMIA gain substantially from trajectory scoring: InfoRMIA rises from 0.740 to 0.949 on medical Qwen3-1.7B and becomes the strongest baseline, with a mean AUROC of 0.826 over the fifteen targets. Leaky still exceeds InfoRMIA on every target, by 0.004–0.100 AUROC (mean 0.049), and has the highest TPR at 1%, 5% and 10% FPR on all fifteen targets.

Table 10: Baselines applied to the eight target-generated trajectories that Leaky scores, on the candidates of Table 1: AUROC and TPR at 1%, 5% and 10% FPR, leading zeros omitted, best per column in bold. Q3/Q3.5: Qwen3/Qwen3.5; G4: Gemma-4-E2B.

Method	Mathematics					Medical QA					Code generation				
	Q3 1.7B	Q3 4B	Q3.5 2B	Q3.5 4B	G4 E2B	Q3 1.7B	Q3 4B	Q3.5 2B	Q3.5 4B	G4 E2B	Q3 1.7B	Q3 4B	Q3.5 2B	Q3.5 4B	G4 E2B
AUROC															
LOSS	.502	.477	.489	.506	.495	.536	.549	.568	.557	.550	.534	.536	.547	.551	.544
ZLIB	.504	.472	.490	.507	.487	.505	.530	.558	.546	.540	.531	.543	.552	.555	.553
Min-K%	.501	.473	.492	.510	.496	.531	.548	.572	.559	.544	.530	.536	.557	.556	.545
Min-K%++	.486	.472	.491	.492	.495	.491	.482	.564	.550	.499	.510	.503	.558	.566	.512
ICP-MIA	.531	.511	.521	.526	.502	.576	.525	.570	.540	.547	.531	.525	.591	.576	.563
Ratio	.513	.565	.583	.559	.533	.492	.491	.539	.511	.491	.513	.477	.506	.516	.519
WBC	.508	.545	.565	.606	.520	.504	.505	.597	.582	.479	.520	.488	.528	.581	.503
EZ-MIA	.503	.529	.522	.519	.503	.514	.545	.583	.570	.531	.503	.546	.542	.566	.524
LiRA	.851	.856	.814	.872	.717	.935	.841	.839	.790	.779	.714	.774	.731	.765	.752
RMIA	.849	.843	.729	.744	.711	.933	.842	.835	.790	.777	.711	.753	.705	.764	.749
InfoRMIA	.874	.877	.835	.905	.746	.949	.844	.841	.803	.784	.762	.801	.802	.792	.777
Leaky	**.916**	**.922**	**.913**	**.925**	**.832**	**.966**	**.848**	**.868**	**.837**	**.826**	**.847**	**.843**	**.901**	**.846**	**.839**
TPR at 1% FPR															
LOSS	.004	.008	.004	.008	.008	.023	.006	.016	.012	.004	.012	.014	.004	.004	.012
ZLIB	.012	.006	.010	.008	.006	.025	.008	.021	.012	.006	.020	.008	.016	.006	.025
Min-K%	.010	.010	.006	.012	.008	.008	.006	.020	.012	.002	.020	.004	.006	.004	.014
Min-K%++	.006	.002	.006	.002	.006	.020	.000	.025	.029	.012	.021	.004	.014	.012	.033
ICP-MIA	.016	.025	.018	.020	.006	.021	.008	.012	.012	.020	.006	.012	.016	.012	.018
Ratio	.000	.023	.045	.012	.018	.008	.014	.023	.008	.006	.004	.020	.012	.004	.008
WBC	.008	.027	.039	.031	.021	.018	.014	.037	.020	.014	.004	.014	.010	.025	.014
EZ-MIA	.008	.016	.018	.008	.014	.012	.018	.020	.014	.008	.004	.016	.006	.016	.023
LiRA	.352	.289	.207	.258	.113	.486	.352	.400	.283	.193	.174	.141	.172	.342	.180
RMIA	.373	.361	.162	.215	.135	.572	.373	.426	.299	.207	.184	.139	.162	.326	.172
InfoRMIA	.334	.418	.359	.516	.135	.613	.383	.451	.375	.270	.242	.246	.299	.355	.250
Leaky	**.418**	**.576**	**.477**	**.582**	**.207**	**.740**	**.449**	**.512**	**.461**	**.350**	**.434**	**.396**	**.578**	**.527**	**.270**
TPR at 5% FPR															
LOSS	.051	.037	.057	.035	.053	.082	.055	.084	.078	.068	.084	.088	.062	.064	.092
ZLIB	.049	.045	.049	.049	.043	.072	.045	.082	.076	.035	.098	.094	.059	.068	.072

Continued on next page

Table 10 (continued)

Method	Mathematics					Medical QA					Code generation				
	Q3 1.7B	Q3 4B	Q3.5 2B	Q3.5 4B	G4 E2B	Q3 1.7B	Q3 4B	Q3.5 2B	Q3.5 4B	G4 E2B	Q3 1.7B	Q3 4B	Q3.5 2B	Q3.5 4B	G4 E2B
Min-K%	.051	.027	.053	.041	.051	.080	.051	.092	.068	.055	.068	.064	.068	.061	.084
Min-K%++	.053	.049	.027	.053	.023	.053	.066	.080	.062	.043	.057	.043	.076	.068	.061
ICP-MIA	.059	.070	.061	.059	.055	.090	.023	.074	.051	.055	.053	.066	.062	.082	.078
Ratio	.029	.094	.094	.066	.086	.061	.055	.094	.047	.041	.041	.061	.080	.062	.053
WBC	.055	.080	.076	.076	.070	.053	.047	.117	.076	.053	.039	.061	.041	.064	.055
EZ-MIA	.072	.080	.047	.059	.072	.059	.047	.098	.074	.053	.047	.051	.047	.061	.072
LiRA	.490	.508	.432	.480	.225	.732	.510	.535	.473	.387	.297	.348	.340	.434	.295
RMIA	.518	.502	.242	.355	.230	.729	.553	.535	.465	.383	.307	.365	.275	.443	.297
InfoRMIA	.559	.586	.506	.668	.312	.752	.514	.549	.516	.400	.369	.436	.441	.498	.371
Leaky	**.662**	**.693**	**.701**	**.762**	**.434**	**.865**	**.559**	**.646**	**.641**	**.508**	**.568**	**.572**	**.689**	**.643**	**.467**
TPR at 10% FPR															
LOSS	.088	.076	.104	.100	.102	.133	.137	.166	.145	.127	.146	.154	.127	.123	.146
ZLIB	.105	.102	.098	.115	.088	.131	.121	.139	.125	.121	.154	.146	.125	.129	.174
Min-K%	.102	.070	.119	.107	.102	.148	.141	.156	.141	.113	.145	.152	.127	.139	.143
Min-K%++	.117	.092	.094	.094	.076	.094	.098	.135	.129	.074	.094	.094	.113	.129	.088
ICP-MIA	.117	.119	.104	.129	.096	.145	.107	.162	.109	.137	.105	.119	.143	.117	.152
Ratio	.092	.141	.164	.174	.154	.113	.098	.152	.092	.107	.098	.105	.107	.109	.102
WBC	.094	.143	.156	.152	.125	.123	.113	.207	.174	.105	.104	.092	.115	.143	.121
EZ-MIA	.115	.127	.084	.113	.125	.102	.135	.174	.150	.109	.102	.119	.146	.104	.123
LiRA	.576	.594	.557	.633	.385	.826	.605	.633	.545	.457	.418	.459	.418	.508	.432
RMIA	.594	.611	.363	.434	.391	.822	.646	.625	.547	.479	.422	.451	.389	.533	.441
InfoRMIA	.674	.658	.602	.758	.410	.857	.604	.643	.609	.525	.496	.529	.525	.559	.496
Leaky	**.758**	**.799**	**.805**	**.811**	**.557**	**.928**	**.666**	**.703**	**.686**	**.600**	**.633**	**.643**	**.752**	**.697**	**.590**

Scoring rules. Table 11 contains four groups. The first uses the target's log-probabilities: their mean, Min-K%, and Min-K%++, with the latter two selecting the lowest 20% of token scores. The second averages signed gaps using three calibration choices: the frozen base model and the mean or maximum log-probability across the trained reference models. The third and fourth keep the reference mean and the reference maximum, respectively, and change the aggregation, so that the two trained-reference calibrations are crossed with the same aggregations. Min-20% gap averages the lowest 20% of signed gaps within each trajectory; unlike Min-K%, it selects tokens by calibrated gaps rather than target log-probabilities. ReLU retains only positive gaps, while Leaky also includes negative gaps with the small weight used in the main experiments.

Table 11: Comparison of scoring rules on the same target-generated trajectories. AUROC and TPR at 1% FPR are averaged over fifteen targets. All scores using trained reference models use the same eight models.

Reference	Scoring rule	Mean AUROC	Mean TPR at 1% FPR
None	Target log-probability mean	0.529	0.009
None	Min-K% (20%)	0.530	0.009
None	Min-K%++ (20%)	0.511	0.013
Base model	Mean gap	0.520	0.014
Reference mean	Mean gap	0.801	0.282
Reference maximum	Mean gap	0.658	0.068
Reference mean	ReLU	0.774	0.293
Reference mean	Leaky	0.777	0.294
Reference maximum	Min-20% gap	0.481	0.008
Reference maximum	ReLU	0.843	0.431
Reference maximum	**Leaky**	**0.875**	**0.465**

Averaging the target's log-probabilities or selecting low-scoring tokens provides limited discrimination on the sampled trajectories. Reference-model calibration substantially improves performance, and the calibration and aggregation choices interact. Under the signed mean, the reference mean is the better reference (0.801 against 0.658 for the maximum, higher on all fifteen targets), and rectification does not help it: ReLU and Leaky lower its mean AUROC by 0.027 and 0.025 and improve only three of the fifteen targets. Under the reference maximum, ReLU and Leaky raise mean AUROC from 0.658 to 0.843 and 0.875 on all fifteen targets, and with either rectifier the maximum exceeds the mean on all fifteen targets, by 0.069 and 0.099 on average. Neither the maximum nor the rectifier contributes on its own; the gain of Leaky over the best signed-mean rule comes from their combination. This is consistent with the model of Appendix A.2: for a target trained without the candidate, the expected gap to the reference mean is zero for any disagreement scale σ, so both signs of the gap carry information and discarding the negative part loses signal, whereas the expected gap to the maximum is $-\sigma(m_K - m_1)$, so its signed average tracks how much the references disagree at each record and position rather than membership. Rectification removes this shortfall term and keeps the positions where the target exceeds every reference, and the small negative weight corrects the remaining chance lead.

C.3 Trajectory Generation

Leaky evaluates membership on sampled trajectories, so its effectiveness may depend on how these trajectories are generated. We examine two design choices: (1) whether informative trajectories need to be generated by the training teacher, or can also be obtained from the target and other models; and (2) whether providing the reference answer during generation improves membership inference over sampling from the problem alone. These comparisons evaluate our choice to sample from the target conditioned on both the problem and reference answer. We keep the target, eight reference models, candidate records, and scoring rule fixed, and sample eight trajectories per candidate under each setting.

Sampling model. For the mathematics Qwen3-1.7B target, Figure 3(c) compares trajectories generated by the training teacher (Qwen3-8B), two alternative models (Qwen3-14B and Qwen3-4B), the frozen base model, and the target. All five models receive the problem and reference answer. The teacher, the two alternative models, and the target yield similar AUROCs, ranging from 0.899 to 0.916, while the base model yields 0.857. On this target, effective attack trajectories can thus be obtained from several models, including the target itself. Sampling from the target directly examines the generation behavior shaped by OPD.

Reference-answer conditioning. We then keep the target as the sampling model and compare trajectories generated from the problem alone with those generated from both the problem and reference answer. In both cases, the target and reference models score the trajectories using only the

problem and the trajectory prefix as context. Providing the reference answer during generation increases AUROC from 0.880 to 0.916. Reference-answer conditioning therefore improves the attack on this target, while the problem-only result shows that membership information remains detectable without it.

C.4 Number of Reference Models

We vary the number of reference models to examine how much calibration can be obtained with a smaller ensemble. For each target in Table 1, we keep its candidates, eight target-generated trajectories, and scoring procedure fixed, including $\alpha = 0.04$. For each $K = 1, \ldots, 8$, we use the first K models from the same set of eight reference models. The curves therefore describe a fixed sequence of reference-model subsets. Figure 3(a,b) presents Qwen3-1.7B; Figures C.1 and C.2 below show the other four students across the three domains.

Even a single reference model provides membership information, with AUROC ranging from 0.612 to 0.910 across the fifteen targets. Increasing K from one to eight raises AUROC by 0.056–0.251. Four reference models recover more than half of this gain on fourteen targets, and eight reference models achieve higher AUROC than four on every target. These results show that a small ensemble already provides useful calibration, while additional reference models further improve overall discrimination. TPR at 1% FPR is also higher with eight reference models than with one on every target, but fluctuates more across intermediate values of K. Thus, an additional reference model does not necessarily improve detection at this particular FPR.

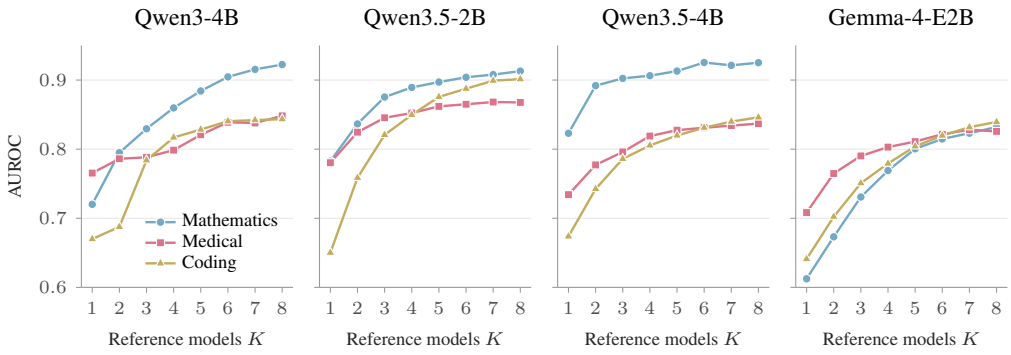


Figure C.1: AUROC against the number of reference models K for the four students not in Figure 3(a). Each K uses the first K of the same eight reference models, so $K = 8$ is Table 1.

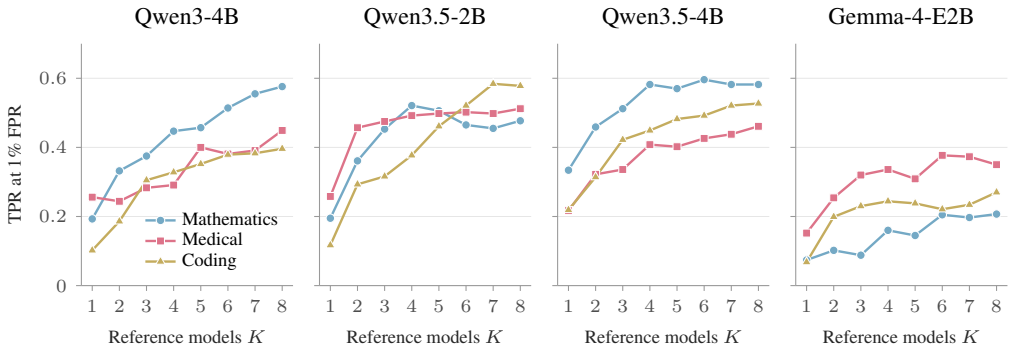


Figure C.2: TPR at 1% FPR along the sweeps of Figure C.1.

C.5 On-Policy Self-Distillation

We next examine whether membership leakage persists when the teacher is a frozen copy of the initial student. In on-policy self-distillation (OPSD) (Zhao et al., 2026; Shenfeld et al., 2026), this teacher receives the problem and reference answer, while the student samples training trajectories from the problem alone. We compare OPSD with external-teacher OPD using Qwen3-1.7B on OpenThoughts-Math. The OPD arm uses Qwen3-8B as teacher and is a separate run of the main mathematics configuration.

The two runs are trained on different samples of 512 OpenThoughts-Math records, so each run has its own candidate set of 512 members and 512 non-members and its own eight reference models trained with the corresponding procedure; the runs are compared at matched training steps on cohorts of equal size. We evaluate checkpoints at 25, 50, 75, and 100 training steps. At each checkpoint, the target generates eight trajectories per candidate, and Leaky uses the reference models from the same training step with $\alpha = 0.04$.

Table 12: Membership inference under on-policy self-distillation and external-teacher OPD on mathematics Qwen3-1.7B. Leakage increases across the evaluated checkpoints under both training procedures. TPR is measured at 1% FPR.

Training step	OPSD AUROC	OPSD TPR	OPD AUROC	OPD TPR
25	0.5935	0.090	0.7259	0.260
50	0.6565	0.090	0.8088	0.336
75	0.6869	0.146	0.8654	0.424
100	0.6987	0.150	0.8771	0.449

As shown in Table 12, AUROC increases with training under both procedures, and the OPSD run has lower AUROC than the OPD run at every evaluated checkpoint. Its AUROC nonetheless rises from 0.594 to 0.699 (OPD: from 0.726 to 0.877), showing that self-distillation still leaves detectable membership information. A control that replaces the target with a reference model yields AUROCs of 0.490–0.533 across the checkpoints of both runs. Together, these observations show that membership leakage is also present in the evaluated self-distilled student as it learns from reference-conditioned supervision.

C.6 Blackbox setting

We examine whether the target's predictions still reveal membership when its token probabilities are unavailable. The interface accepts a problem and trajectory prefix and returns one greedy next token. The attacker first samples trajectories from the target conditioned on the problem and reference answer, and then compares the target's returned tokens with the predictions of its own reference models.

Black-box score. Let $h_{i,t} = (x, y_{i,<t})$ be the prefix preceding a sampled token $y_{i,t}$. Define $E_{i,t} = 1$ when the target's greedy prediction at this prefix equals $y_{i,t}$, and zero otherwise. Similarly, let $G_{i,t} = 1$ when none of the reference models predicts $y_{i,t}$, and zero otherwise. We compute

$$S_{\mathrm{BB}}(z) = \frac{\sum_i \sum_{t=1}^{L_i} \left[G_{i,t}E_{i,t} - \alpha(1 - G_{i,t})(1 - E_{i,t})\right]}{\sum_i L_i}, \qquad \alpha = 0.04. \tag{10}$$

The positive term counts positions where only the target predicts the sampled token. The negative term counts positions where at least one reference model predicts it but the target does not. This preserves the asymmetric treatment of target advantages and shortfalls using discrete predictions. The score pools all scored positions across trajectories, with every position contributing to the denominator.

Evaluation and additional results. We use the same fifteen targets and candidate sets as the main experiments, with eight target-generated trajectories and eight reference models. For scoring, the target and reference models receive the problem and trajectory prefix. We compute the score from

greedy decisions at temperature zero, using only the identity of the predicted token. Implementation details are given in Appendix B.4. Table 13 supplements the main black-box results with TPR at 5% and 10% FPR. Replacing the target with one reference model and using the remaining seven as references gives AUROCs of 0.474–0.541.

Table 13: Additional black-box results at 5% and 10% FPR for the fifteen targets in Table 2.

Student	TPR at 5% FPR	TPR at 10% FPR
Mathematics		
Qwen3-1.7B	0.393	0.533
Qwen3-4B	0.453	0.543
Qwen3.5-2B	0.344	0.486
Qwen3.5-4B	0.469	0.568
Gemma-4-E2B	0.283	0.396
Medical question answering		
Qwen3-1.7B	0.590	0.648
Qwen3-4B	0.283	0.350
Qwen3.5-2B	0.453	0.523
Qwen3.5-4B	0.324	0.439
Gemma-4-E2B	0.293	0.365
Code generation		
Qwen3-1.7B	0.352	0.469
Qwen3-4B	0.250	0.322
Qwen3.5-2B	0.352	0.410
Qwen3.5-4B	0.488	0.557
Gemma-4-E2B	0.221	0.311

Number of sampled trajectories. To examine how the number of trajectories affects detection, we use the first one, two, four, or eight trajectories from each candidate's sampled set while keeping the eight reference models and scoring rule fixed. Figure C.3 shows that eight trajectories yield higher AUROC and TPR than one on all fifteen targets. Additional trajectories provide more positions at which to compare target and reference predictions and identify membership signals.

D Defense Evaluation

D.1 Defense Setup

We evaluate whether LoRA and DP-LoRA reduce membership leakage from OPD. The experiments use Qwen3-1.7B with a Qwen3-8B teacher across mathematics, medical question answering, and code generation. Within each domain, the training data and evaluation candidates remain fixed across full fine-tuning, LoRA, and DP-LoRA. Each configuration has its own target and eight reference models trained with the corresponding procedure. We evaluate Leaky using eight trajectories sampled from each target for every candidate.

LoRA. Both LoRA configurations apply rank-64 adapters ($\alpha = 128$) to all linear layers of the transformer blocks. LoRA without DP uses the training budget of full fine-tuning (126 steps with a global batch size of 32 for mathematics; 512 steps with a global batch size of 128 for medical QA and coding), no gradient clipping, and the interpolation weight of the corresponding full fine-tuning cell.

DP-LoRA. DP-LoRA trains the same adapters with per-record gradient clipping and Gaussian noise at a nominal privacy budget of $\varepsilon = 8$ with $\delta = 1/n$, where n is the number of training records; the noise multiplier is computed with the Rényi-DP accountant (Mironov, 2017) of Opacus (Yousefpour et al., 2021) for each training configuration. Mathematics and coding use the schedule of full fine-tuning with clipping norm $C = 1$ and a learning rate of 10^{-4} (noise multipliers 0.698 and 0.60). For

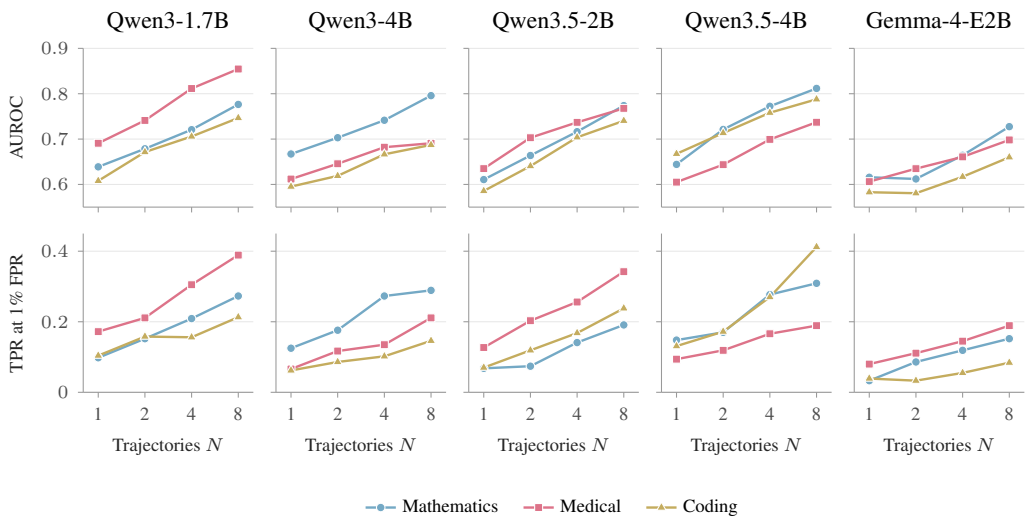


Figure C.3: Effect of the number of target-generated trajectories per candidate across all fifteen targets. Eight reference models and $\alpha = 0.04$ are fixed throughout. The eight-trajectory results match Table 2.

medical QA, we report a configuration at the same privacy budget and the same number of processed records that uses $C = 0.25$ and a learning rate of 3×10^{-4}.

D.2 Full Results

Table 14 supplements Table 3 with TPR at 5% and 10% FPR.

Table 14: Defenses on Qwen3-1.7B: change in Avg@8 over the base (percentage points), AUROC, and TPR at 1%, 5%, and 10% FPR of Leaky ($K = 8$, $\alpha = 0.04$).

Domain	Training	ΔUtil.	AUROC	TPR@1%	TPR@5%	TPR@10%
Mathematics	Full fine-tuning	+5.00	0.9158	0.418	0.662	0.758
	LoRA	+2.50	0.6455	0.098	0.180	0.277
	DP-LoRA	+1.67	0.4903	0.014	0.049	0.092
Medical QA	Full fine-tuning	+3.98	0.9665	0.740	0.865	0.928
	LoRA	+2.46	0.7868	0.268	0.396	0.502
	DP-LoRA	+0.04	0.5044	0.023	0.070	0.111
Code generation	Full fine-tuning	+0.57	0.8467	0.434	0.568	0.633
	LoRA	+0.14	0.6585	0.146	0.236	0.338
	DP-LoRA	+0.07	0.5097	0.027	0.059	0.115

D.3 Results across Attack Configurations

The main results show that LoRA reduces attack performance while retaining detectable membership information, and DP-LoRA brings AUROC close to chance. We examine whether this observation depends on the default Leaky configuration by changing the negative slope or the number of reference models on the three domains. Within each training configuration, the candidates and eight target-generated trajectories remain fixed. We compare the default score ($K = 8$, $\alpha = 0.04$) with positive-only aggregation ($\alpha = 0$), and with four or one reference model while retaining $\alpha = 0.04$.

Table 15: Attack performance under LoRA and DP-LoRA across four scoring configurations. Entries report AUROC / TPR at 1% FPR.

Domain	Attack configuration	LoRA	DP-LoRA
Mathematics	Leaky ($K = 8, \alpha = 0.04$)	0.6455 / 0.098	0.4903 / 0.014
	ReLU ($K = 8, \alpha = 0$)	0.5881 / 0.084	0.4874 / 0.016
	Leaky ($K = 4, \alpha = 0.04$)	0.5866 / 0.076	0.4856 / 0.008
	Leaky ($K = 1, \alpha = 0.04$)	0.5511 / 0.039	0.4903 / 0.008
Medical QA	Leaky ($K = 8, \alpha = 0.04$)	0.7868 / 0.268	0.5044 / 0.023
	ReLU ($K = 8, \alpha = 0$)	0.7689 / 0.240	0.5025 / 0.021
	Leaky ($K = 4, \alpha = 0.04$)	0.7577 / 0.229	0.5000 / 0.010
	Leaky ($K = 1, \alpha = 0.04$)	0.6966 / 0.141	0.5041 / 0.018
Code generation	Leaky ($K = 8, \alpha = 0.04$)	0.6585 / 0.146	0.5097 / 0.027
	ReLU ($K = 8, \alpha = 0$)	0.6309 / 0.107	0.5131 / 0.027
	Leaky ($K = 4, \alpha = 0.04$)	0.6292 / 0.074	0.5172 / 0.035
	Leaky ($K = 1, \alpha = 0.04$)	0.5653 / 0.041	0.5399 / 0.014

Table 15 shows that LoRA remains above chance across these configurations, with AUROC ranging from 0.551 to 0.787. For DP-LoRA, AUROC stays between 0.486 and 0.540. The low attack AUROC under DP-LoRA therefore persists when the negative contribution is removed or the reference ensemble is reduced.