Learning the Signature of Memorization in Autoregressive Language Models

David Ilić **Kostadin Cvejoski** **David Stanojević** **Evgeny Grigorenko**
JetBrains Research
{david.ilic, kostadin.cvejoski, david.stanojevic, evgeny.grigorenko}@jetbrains.com

Abstract

We introduce a learned membership inference attack that transfers zero-shot to architectural families absent from its training set, enabled by the observation that fine-tuning any model on any corpus yields abundant labeled data, since membership is known by construction. This decouples attack training from target identity, replacing target fidelity with source diversity: a single classifier, trained once on diverse public fine-tuning runs, generalizes to unseen architectures without retraining. We observe that fine-tuning language models produces a signature of memorization detectable across architectural families and data domains. We train a membership inference classifier exclusively on transformer-based models. It transfers zero-shot to Mamba (state-space), RWKV-4 (linear attention), and RecurrentGemma (gated recurrence), achieving 0.963, 0.972, and 0.936 AUC respectively, each evaluation combining an architecture and dataset never seen during training. These four families share no computational mechanisms, their only shared property is gradient descent on cross-entropy loss. Even simple likelihood-based methods exhibit strong transfer, confirming the signature exists independently of the detection method. Our method, Learned Transfer MIA (LT-MIA), captures this signal most effectively by reframing membership inference as sequence classification over per-token distributional statistics. On transformers, LT-MIA achieves 2.8$\times$ higher true positive rate at 0.1% false positive rate than the strongest baseline. The method also transfers to code (0.865 AUC) despite training only on natural language texts. Our results imply that leakage from memorization accompanies cross-entropy training across every architecture we tested. Code and trained classifier are available at this GitHub URL.

1 Introduction

Membership inference attacks (MIAs) determine whether a specific text appeared in a model's training data, with applications in privacy auditing, copyright detection, and data governance. Existing MIA methods for language models include designed statistics that threshold model outputs [Yeom et al., 2018, Shi et al., 2024, Carlini et al., 2021, 2022, Ilić et al., 2026] and learned classifiers trained on shadow models [Shokri et al., 2017, Song and Shmatikov, 2019]. Designed methods require no training but are bounded by the designer's intuition about what signals membership. Learned methods have been applied to text models [Song and Shmatikov, 2019, Hisamoto et al., 2020] but use target-only features and target-coupled supervision, and whether a learned classifier can transfer across architectural families by combining reference comparison with diverse, target-independent sources has not been tested.

We introduce a learned membership inference attack for fine-tuned language models that transfers zero-shot to architectural families absent from its training set. The key insight is that MIA has abundant labeled training data by construction: fine-tuning any model on any corpus yields perfect

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

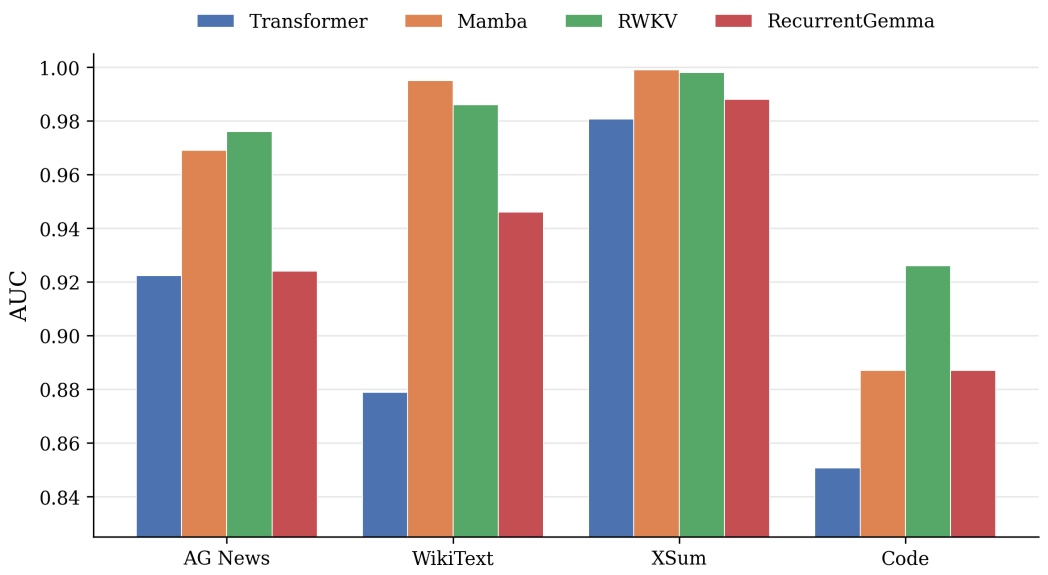

Figure 1: LT-MIA transfers from transformers to non-transformer architectures. Blue bars show held-out transformer results (mean over 7 models), while the remaining bars show zero-shot transfer to Mamba-2.8B (state-space), RWKV-4-3B (linear attention), and RecurrentGemma-2B (gated recurrence). The classifier was trained exclusively on transformers and textual data.

membership labels. This decouples attack training from target identity: rather than building shadow models that replicate a specific target, we train once on diverse public fine-tuning runs and transfer to unseen targets, including architectural families entirely absent from training.

Our membership inference attack, Learned Transfer MIA (LT-MIA), trained exclusively on transformer language models, transfers zero-shot to architectures it has never seen, including state-space models with no attention mechanism. LT-MIA achieves 0.963 area under the ROC curve (AUC) on Mamba-2.8B [Gu and Dao, 2023], 0.972 on RWKV-4-3B [Peng et al., 2023], and 0.936 on RecurrentGemma-2B [Botev et al., 2024] when detecting whether a text was used during fine-tuning. This transfer is not an artifact of our method's design: even simple loss thresholding (Loss) achieves 0.867 AUC on Mamba, and reference-based loss calibration (RefLoss) reaches 0.892.

The four architectural families share no computational mechanisms. Transformers compute attention through pairwise token interactions with quadratic complexity [Vaswani et al., 2017]. Mamba uses selective state spaces that evolve hidden states through input-dependent dynamics, with no attention mechanism whatsoever [Gu and Dao, 2023]. RWKV replaces softmax attention with linear recurrence, enabling constant-memory inference [Peng et al., 2023]. RecurrentGemma combines gated linear recurrence with local attention windows [Botev et al., 2024]. These models differ in how they process context, how they scale with sequence length, and how information flows through layers. Their only commonality is the training objective: gradient descent on cross-entropy loss.

The implication is that the memorization signature appears in the output distribution and does not depend on the computational mechanism. Cross-entropy training is accompanied by a distributional shift toward training data: every model we tested assigns higher likelihood to training samples relative to its pre-trained state, and this shift is detectable regardless of how the model computes that likelihood. Feature ablation confirms that comparison features (those relating target to reference model outputs) dominate across all four families, with the same feature importance hierarchy appearing in transformers, state-space models, linear attention, and gated recurrence alike.

To train our classifier, we reframe membership inference as sequence classification over per-token features. Given a text and logit-level black-box access to a fine-tuned target model and its pre-trained reference, we extract features at each token position capturing how the two models' predictions diverge. A lightweight transformer classifier processes this feature sequence, learning to weight positions adaptively rather than aggregating to a single statistic. Sequence modeling contributes

5.0 AUC percentage points over mean pooling (Section 5), with the transformer's advantage likely stemming from its adaptive weighting ability.

Training diversity proves essential. We train on 30 model-dataset combinations with 600 samples each, which substantially outperforms training on 1 combination with 18,000 samples (0.860 vs. 0.796 AUC on held-out combinations), despite identical data volume. Diversity filters model-specific artifacts (tokenizer idiosyncrasies, architectural biases in logit scales), retaining only the memorization signal shared across models. This filtering may explain why transfer succeeds: by learning to ignore transformer-specific patterns during training, the classifier relies on the likelihood shift that appears in every model we tested.

The signal itself exists across architectures, data domains, and attacks, but LT-MIA captures it most effectively. On held-out transformer combinations, LT-MIA achieves 0.908 mean AUC, outperforming the strongest baseline (EZ-MIA) by 2.1 points. The gap widens at stricter operating points: 1.6$\times$ higher TPR at 1% FPR and 2.8$\times$ at 0.1% FPR. The method also transfers to code (0.865 AUC) despite training only on natural language text.

We make four contributions. First, we train the attack on many fine-tuning runs that vary in both model and data, so that one classifier, fixed before any target is seen, transfers to new targets, including architectural families absent from its training set. Second, we demonstrate that memorization produces an architecture-invariant signature: a detector trained on transformers transfers zero-shot to state-space models, linear attention, and gated recurrence, with all tested likelihood-based methods exhibiting this transfer. Third, we introduce Learned Transfer MIA (LT-MIA), a learned classifier over per-token reference comparisons that outperforms every baseline we evaluate. Fourth, we demonstrate that training diversity, not pure scale, enables generalization: 30 combinations with 600 samples each outperforms 1 combination with 18,000 samples, and show that sequence classification over per-token features outperforms aggregation-based methods, contributing 5.0 AUC percentage points. Our results imply that privacy risk from memorization is present in every cross-entropy-trained model we tested, regardless of architecture.

2 Related work

Yeom et al. [2018] connected membership inference to overfitting and proposed thresholding the target model's per-sample loss. Subsequent work refined the target-only statistic by normalizing against compression baselines [Carlini et al., 2021] and by selecting low-probability token subsets [Shi et al., 2024, Zhang et al., 2024]. Each method designs a scalar from the target model's output alone, bounding performance by the designer's intuition about what signals membership. Duan et al. [2024] showed that these methods reach near-random performance on pretraining data at scale, confining practical membership inference to the fine-tuning setting.

Comparing the fine-tuned model's output to a pre-trained reference separates the fine-tuning signal from baseline likelihood. RefLoss [Carlini et al., 2022] thresholds the loss difference between the two models. Subsequent reference-based methods compare at finer granularity, including EZ-MIA [Ilić et al., 2026], which aggregates probability shifts at error positions, and neighborhood and window-based approaches [Mattern et al., 2023, Chen et al., 2026]. The reference signal is substantially more discriminative than target-only statistics (Table 1), but these remain designed decision rules. LiRA [Carlini et al., 2022] constructs a learned likelihood-ratio test from per-candidate shadow models, achieving strong results on classifiers but at a per-target cost that is prohibitive at LLM scale.

Shokri et al. [2017] introduced the shadow model paradigm, training models that replicate the target's setting and then training a classifier on their outputs. The shadows must resemble the target in architecture and data distribution, can only be built once the target is known, and are discarded after each attack. Song and Shmatikov [2019] applied this to text-generation models, training a classifier on token-rank features from the target model alone, while Hisamoto et al. [2020] tested shadow-model attacks on sequence-to-sequence models with limited success. In each case the classifier reads a target-only signal, and supervision requires replicating some aspect of the target's training setting.

Salem et al. [2019] relaxed the target-coupling requirement, showing that a single unrelated shadow suffices and that attacks transfer across datasets, including from MLP to CNN targets. However, architecture covaries with dataset in their evaluation, preventing isolation of cross-architecture transfer, and a threshold on a single posterior statistic (their Adversary 3) performs nearly as well as the

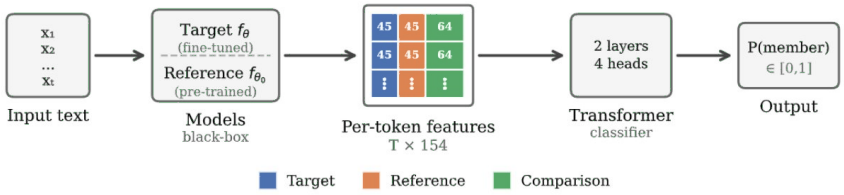

Figure 2: LT-MIA pipeline. Given a text sample and logit-level black-box access to a fine-tuned target model and its pre-trained reference, we extract 154-dimensional feature vectors at each token position capturing how the two models' predictions diverge. A lightweight transformer encoder processes this feature sequence to predict membership.

learned variant, indicating that the learned component contributes little in their setting. The signal remains target-only.

Our work combines learned classification with reference comparison, trained on diverse public fine-tuning runs chosen independently of any target. We claim priority for neither reference comparison nor cross-architecture transfer as a concept, both of which appear in the work above. What is new is a learned classifier over per-token reference comparisons for fine-tuned language models, evaluated on architectural families that vary independently of dataset and are entirely absent from training. Feldman [2020] showed that memorization is necessary for near-optimal generalization on long-tailed distributions, a property of the learning problem rather than of model capacity, motivating the expectation that the resulting distributional signature should transfer across architectures trained on the same objective.

3 Method

Figure 2 illustrates the LT-MIA pipeline. Given a text sample and logit-level black-box access to both a fine-tuned target model and its pre-trained reference, we extract a feature vector at each token position and classify the resulting sequence.

3.1 Threat model

We consider an adversary performing fine-tuning membership inference: given a candidate text x, determine whether x was included in the fine-tuning dataset $\mathcal{D}_{\text{ft}}$.

Adversary capabilities. The adversary has logit-level black-box query access to the fine-tuned target model f_θ and a pre-trained reference model f_{θ_0} sharing the same architecture. Both models return full vocabulary logits at each token position. The adversary can query them with arbitrary inputs.

Adversary knowledge. The adversary knows the model's architecture family (to identify the appropriate reference checkpoint) and possesses candidate texts to query, for instance, a copyright holder checking whether their content was used for fine-tuning.

What is not required. The adversary does not need white-box access, knowledge of the training data distribution, or the ability to train shadow models of the target.

Train-once-attack-anywhere. Our threat model decouples attack training from target identity. The adversary trains a classifier once on diverse public models, then deploys against any target, including architectures never seen during training, without retraining. This is the capability our experiments validate.

3.2 Per-token feature extraction

Let f_θ denote the fine-tuned target model and f_{θ_0} the pre-trained reference sharing the same architecture. For input sequence $x = (x_1, \ldots, x_T)$, we extract 154 features at each position t, capturing how the two models' predictions diverge.

The features, normalized for scale invariance, include:

- per-token losses from both models and their difference
- global statistics (mean, standard deviation of each loss term, total log-likelihood ratio) broadcast to every position
- the logit and rank of the ground-truth token x_t in both distributions
- top-20 and bottom-20 logits from each model
- cross-model ranks: for each model's top-20 and bottom-20 tokens, their rank in the other model's distribution

3.3 Architecture & training

The feature sequence is projected to dimension $d = 128$, combined with sinusoidal positional encodings [Vaswani et al., 2017], and processed by a 2-layer transformer encoder with 4 attention heads and feedforward dimension 256. A learned query computes attention over encoder outputs, that is then fed as a weighted sum into a two-layer MLP classifier. In total, the attack model has fewer than 500K parameters.

We construct training data by first fine-tuning 10 transformer-base models on 3 text corpora, and then extracting features for samples included and not included in fine-tuning. We sample uniformly across the 30 model-dataset combinations during training. The objective for training the attack model is binary cross-entropy, optimized with AdamW [Loshchilov and Hutter, 2019] (learning rate 3×10^{-4}, batch size 1024) for 30 epochs. We select the checkpoint with the highest AUC on the validation dataset.

4 Experiments

We evaluate LT-MIA on held-out model-dataset combinations, with particular focus on transfer to architectures not seen during training.

4.1 Experimental setup

Data. Training uses 10 transformer models (listed in Appendix B) fine-tuned on 3 text corpora, yielding 30 model-dataset combinations with 18,000 samples each (540,000 total). Evaluation uses 7 held-out transformers (Appendix B) plus three non-transformer architectures: Mamba-2.8B (state-space), RWKV-4-3B (linear attention), and RecurrentGemma-2B (gated recurrence). Each is evaluated on 4 datasets: AG News [Zhang et al., 2015], WikiText-103 [Merity et al., 2016], XSum [Narayan et al., 2018], and Swallow-Code [Fujii et al., 2025]. There is zero overlap between training and evaluation combinations (Appendix B). Each document is truncated to its first 128 tokens, and shorter documents are padded with masks that are propagated through both feature extraction and the classifier.

Baselines. We compare against training-free methods that use target-model statistics (Loss [Yeom et al., 2018], Min-K%++ [Zhang et al., 2024], Zlib [Carlini et al., 2021]) and reference-based methods that compare target to pre-trained outputs: RefLoss [Carlini et al., 2022] and EZ-MIA [Ilić et al., 2026], which aggregates probability shifts at positions where the target model errs. All reference-based methods use the pre-trained checkpoint as reference.

Cost. At query time, target-only methods (Loss, Min-K%++, Zlib) require one forward pass through the target model; reference-based methods (RefLoss, EZ-MIA, LT-MIA) require two, one through the target and one through the reference. LT-MIA adds a single pass through the 500K-parameter classifier, approximately 5,000$\times$ smaller than our smallest target model, making per-query cost effectively identical across all reference-based methods. LT-MIA's only additional expense is one-time classifier training (approximately 30 GPU-hours, Appendix B), amortized across all future targets. The compared baselines have no analogous training cost, but also no mechanism to improve with more computation.

Metrics. We report AUC and true positive rate at 1% and 0.1% false positive rate. Low-FPR metrics are critical because realistic attacks query many candidates, making tail calibration essential.

Table 1: AUC by method and architecture. All methods are evaluated on held-out combinations, only LT-MIA involves training. Means for transformers computed over 7 models $\times$ 4 datasets, non-transformer means computed over 4 datasets each.

Method	Transformer	Mamba	RWKV	RecurrentGemma	Mean
Loss	0.622	0.867	0.786	0.860	0.784
Min-K%++	0.585	0.814	0.706	0.813	0.730
Zlib	0.611	0.837	0.761	0.836	0.761
RefLoss	0.760	0.892	0.915	0.800	0.842
EZ-MIA	0.887	0.931	0.971	0.915	0.926
LT-MIA	**0.908**	**0.963**	**0.972**	**0.936**	**0.945**

Table 2: True positive rate at 1% and 0.1% false positive rate across 4 datasets.

TPR @ 1% FPR					
Method	Transformer	Mamba	RWKV	RecurrentGemma	Mean
Loss	0.036	0.356	0.108	0.185	0.171
RefLoss	0.056	0.346	0.308	0.094	0.201
EZ-MIA	0.230	0.526	0.439	0.273	0.367
LT-MIA	**0.375**	**0.569**	**0.593**	**0.399**	**0.484**
TPR @ 0.1% FPR					
Method	Transformer	Mamba	RWKV	RecurrentGemma	Mean
Loss	0.009	0.151	0.027	0.074	0.065
RefLoss	0.014	0.148	0.056	0.046	0.066
EZ-MIA	0.061	**0.240**	0.277	**0.226**	0.201
LT-MIA	**0.171**	**0.240**	**0.292**	0.166	**0.217**

4.2 Results

Tables 1 and 2 present results across four architectural families.

LT-MIA significantly outperforms all baselines on every family (vs. EZ-MIA: Wilcoxon signed-rank, $p < 0.001$). On held-out transformers, LT-MIA achieves 0.908 mean AUC, exceeding EZ-MIA (0.887) by 2.1 points and RefLoss (0.760) by 14.8 points. The same pattern holds for Mamba (0.963 vs. 0.931), RWKV (0.972 vs. 0.971), and RecurrentGemma (0.936 vs. 0.915).

All likelihood-based methods transfer. Transfer to non-transformer architectures is not an artifact of LT-MIA's design. Loss achieves 0.867 AUC on Mamba, RefLoss achieves 0.892, while EZ-MIA reaches 0.931 on Mamba and 0.971 on RWKV. The memorization signature exists independently of detection method: LT-MIA captures it most effectively, but the signal is architecture-invariant.

LT-MIA's advantage grows at strict thresholds. Table 2 shows that the gap between LT-MIA and baselines widens at low false positive rates. On transformers, LT-MIA achieves 1.6$\times$ higher TPR@1% than EZ-MIA (0.375 vs. 0.230) and 2.8$\times$ higher TPR@0.1% (0.171 vs. 0.061). EZ-MIA's TPR@1% collapses to near-zero on WikiText-103 for several models while LT-MIA remains robust (Appendix F).

Domain transfer. LT-MIA achieves 0.865 mean AUC on Swallow-Code despite training only on natural language text, demonstrating robustness beyond architectural transfer.

5 Analysis

The experiments demonstrate that LT-MIA transfers across architectures, but they do not explain why. We analyze three factors: feature importance, training diversity, and sequence modeling.

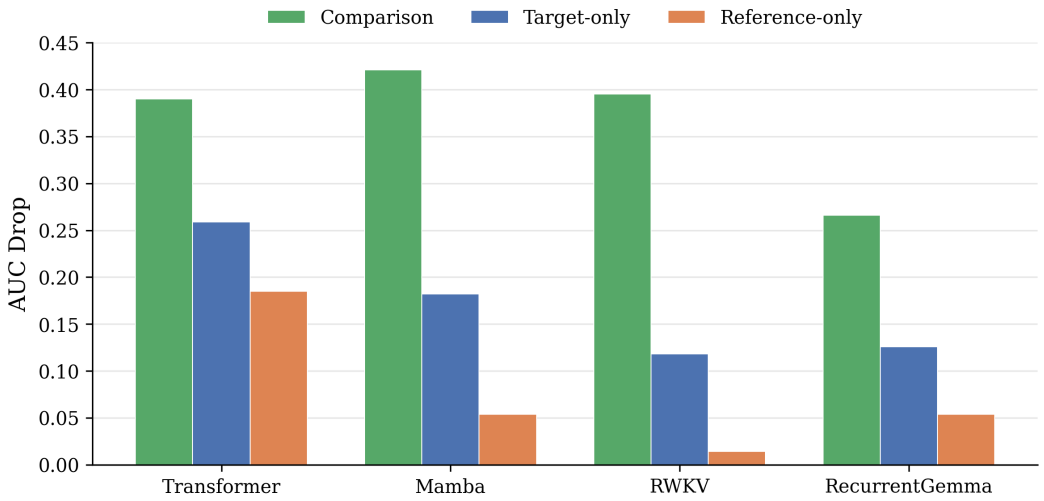

Figure 3: Feature importance measured as AUC drop when each feature group is ablated. Comparison features (relating target to reference model outputs) dominate across all four architectural families. The consistent hierarchy (Comparison $>$ Target-only $>$ Reference-only) confirms the membership signal is relational: what matters is how fine-tuning changed the model, not the model's behavior in isolation.

5.1 Feature importance is consistent across families

Figure 3 shows permutation importance, the AUC drop when each feature group's values are randomly shuffled, breaking its association with the label. Comparison features (those relating target to reference model outputs, Section 3.2) dominate for every architectural family, contributing substantially more than target-only or reference-only features. This hierarchy holds for transformers (0.390 vs. 0.259 vs. 0.185), Mamba (0.421 vs. 0.182 vs. 0.054), RWKV (0.395 vs. 0.118 vs. 0.014), and RecurrentGemma (0.266 vs. 0.126 vs. 0.054).

The consistency is striking: four families with no shared computational mechanisms exhibit the same feature importance pattern. The membership signal is fundamentally relational: what matters is not the target model's behavior in isolation, but how it has changed from its pre-trained state. This relational signal is present in every cross-entropy-trained model we tested, consistent with transfer succeeding for the reason that the signal is tied to the training objective rather than the architecture.

5.2 Diversity enables transfer

A classifier trained on one model-dataset combination might learn artifacts specific to that setting: tokenizer idiosyncrasies, architectural biases in logit scales, or domain-specific patterns. Training on diverse combinations should filter out such artifacts, leaving only signal that generalizes.

Figure 4 confirms this hypothesis. Training on a single combination overfits to model-specific artifacts, evident in the 20.2-point generalization gap. With 30 combinations, the classifier is forced to find signal that works across many transformers, filtering out transformer-specific patterns and retaining only the memorization signal shared across training sources. This filtering may explain why transfer succeeds despite training exclusively on transformers.

5.3 Sequence modeling captures positional structure

Prior methods aggregate per-token statistics into fixed-length representations before classification, discarding positional information. We test whether sequence modeling captures patterns that aggregation misses.

Table 3 compares four classifier architectures trained on identical features. The transformer achieves 5.0 points higher AUC than the mean-pooled MLP (0.925 vs. 0.875). Notably, the flattened MLP

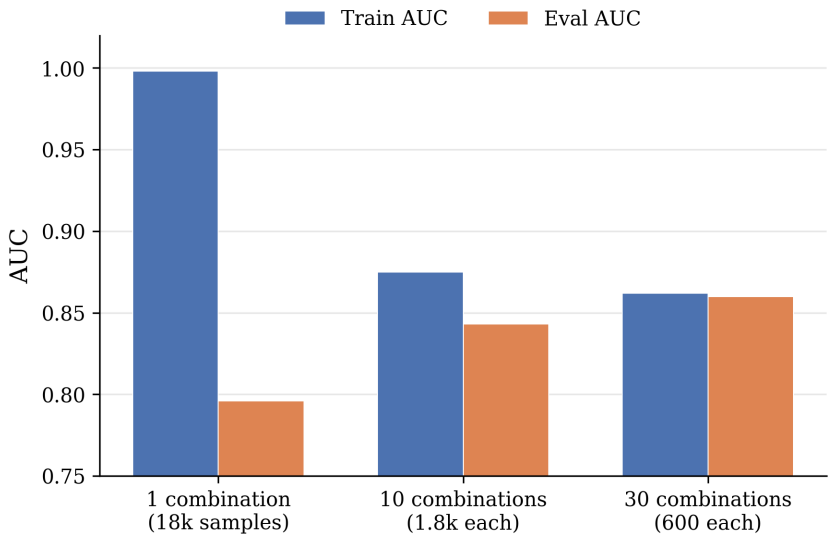

Figure 4: Effect of training diversity on generalization, with total samples fixed at 18,000. "Train AUC" is evaluated on held-out samples from training model-dataset combinations (in-distribution), while "Eval AUC" is evaluated on entirely different model-dataset combinations (out-of-distribution). All metrics use samples never seen during classifier training. Training on one combination yields near-perfect in-distribution AUC (0.998) but poor transfer (0.796). With 30 combinations, the gap shrinks from 20.2 points to 0.2 points.

Table 3: Classifier architecture comparison on held-out combinations. Sequence modeling contributes 5.0 AUC points over pooling.

Classifier	Train AUC	Eval AUC
Transformer (sequence)	0.914	**0.925**
MLP (flattened)	0.885	0.877
MLP (mean pooled)	0.885	0.875
Logistic Regression	0.838	0.839

(0.877) performs comparably to the mean-pooled MLP despite retaining positional information, suggesting the transformer's advantage stems from attention's ability to weight positions adaptively rather than position preservation alone.

6 Discussion

6.1 From coupled to target-independent learned attacks

Learned membership inference for text models is not new: Song and Shmatikov [2019] trained classifiers on token-rank features from shadow models, and Hisamoto et al. [2020] tested shadow-model attacks on sequence-to-sequence models. Both use target-only features and target-coupled supervision, and in both cases the learned component adds little over simple statistics of the target model's output.

LT-MIA differs in two respects. The supervision source is target-independent: rather than building shadows that replicate a specific target, we fine-tune diverse public checkpoints and extract features from all of them, collecting abundant labeled data before any target is known. The feature representation is relational: 109 of 154 features per position compare the target model's output to its pre-trained reference, capturing the distributional shift induced by fine-tuning rather than the target's behavior in isolation.

6.2 Scaling

Our current setup uses a 500K parameter classifier, 30 model-dataset combinations, and 154 features. All of these can be scaled. Standard supervised-learning scaling predicts performance gains from more training combinations, larger classifier capacity, and richer feature representations. Our diversity ablation (Section 5) already shows that 30 combinations substantially outperform 1 combination at fixed data volume, but the gains from 100 or 1000 combinations remain unexplored. This paper demonstrates proof-of-concept, and the performance ceiling is likely higher; finding the scaling laws that govern learned membership inference remains an open question.

6.3 Beyond cross-entropy fine-tuning

We validated the memorization signature on supervised fine-tuning with cross-entropy loss and tested whether it persists through a subsequent preference-optimization stage. Fine-tuning Pythia-2.8B on AG News with the standard SFT recipe, then running DPO on the SFT checkpoint ($\beta = 0.1$, lr $= 5 \times 10^{-7}$, 1 epoch), yields 0.985 AUC for detecting SFT membership from the DPO checkpoint (Appendix I). DPO's own preference data is not detectable: querying DPO members against held-out examples yields 0.549 AUC with the pre-trained reference and 0.496 with the SFT checkpoint. The SFT result shows that cross-entropy memorization is not erased by preference alignment, but the DPO null result does not isolate the training objective as the cause: DPO ran at a learning rate $20\times$ below the mildest recipe in Appendix H (which itself produced the lowest AUC of 0.819), so weak optimization pressure is a sufficient explanation. Whether continual pretraining, instruction tuning, or other training methods produce a detectable signature of their own data remains open.

6.4 Defense implications

If the memorization signature is a general consequence of gradient-based training, can it be defended? Differential privacy provides formal guarantees but severely degrades utility at the noise levels required for meaningful protection [Abadi et al., 2016, Carlini et al., 2022]. An open question is whether training modifications can hide the signature without destroying model performance, or whether this represents a fundamental tradeoff between learning from data and leaking information about that data. Our finding that the signature transfers across architectural families suggests that defenses targeting specific computational mechanisms (attention patterns, state dynamics) are unlikely to succeed, though further research is necessary.

7 Limitations

Our method struggles when the reference model already assigns high likelihood to the target text. On WikiText-103 with large models (Llama-2-7B: 0.781 AUC, Pythia-6.9B: 0.796 AUC), performance drops below our 0.908 average. WikiText-103 likely appears in these models' pretraining corpora: when the reference model already "knows" a text, fine-tuning produces a smaller likelihood shift, weakening the signal. This is fundamental to reference-based membership inference: these methods detect the change induced by training, and when training induces little change, detection is inherently harder.

Several practical constraints define the method's scope. Like all reference-based attacks, LT-MIA requires logit-level black-box access to a pre-trained checkpoint with the same architecture as the target, which is available for open-weight models but not closed APIs. Our experiments validate the method on fine-tuning. Whether it extends to pretraining membership or different training methods remains untested. Transfer to code achieves 0.865 AUC versus 0.927 on text, still above all baselines but with reduced performance.

8 Broader impact

Membership inference research is inherently dual-use. The same method enabling a copyright holder to verify unauthorized use of their content also enables an adversary to infer sensitive training data membership. We believe this work favors defenders, but acknowledge the tension.

Beneficial applications. LT-MIA enables copyright holders to check whether their content was used for fine-tuning, supports organizational compliance with data use agreements, allows model developers to audit memorization before release, and provides regulators a tool for enforcement (e.g., GDPR Article 15 right-of-access requests).

Why characterization helps defenders. The cross-architecture vulnerability exists independent of our method: Loss achieves 0.867 AUC on Mamba, EZ-MIA reaches 0.931, all without training. We did not introduce this transfer, just characterized it. Knowing the signature is architecture-invariant tells defenders not to invest in architectural migration (switching to state-space models will not help) and to focus instead on other types of defense. A single audit tool can now check any autoregressive model family. Additionally, our method's constraints (reference model access, full vocabulary logits, fine-tuning only) limit applicability to open-weight models, while closed APIs remain unaffected.

Acknowledgments and disclosure of funding

This work was carried out and funded by JetBrains, the employer of all authors. All experiments ran on compute provided by JetBrains. The authors received no third-party funding or support for this work. The authors declare no competing interests.

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*, pages 308–318, 2016.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR, 2023.

Aleksandar Botev, Soham De, Samuel L Smith, Anushan Fernando, George-Cristian Muraru, Ruba Haroun, Leonard Berrada, Razvan Pascanu, Pier Giuseppe Sessa, Robert Dadashi, Léonard Hussenot, Johan Ferret, Sertan Girgin, Olivier Bachem, Alek Andreev, Kathleen Kenealy, Thomas Mesnard, Cassidy Hardin, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Armand Joulin, Noah Fiedel, Evan Senter, Yutian Chen, Srivatsan Srinivasan, Guillaume Desjardins, David Budden, Arnaud Doucet, Sharad Vikram, Adam Paszke, Trevor Gale, Sebastian Borgeaud, Charlie Chen, Andy Brock, Antonia Paterson, Jenny Brennan, Meg Risdal, Raj Gundluru, Nesh Devanathan, Paul Mooney, Nilay Chauhan, Phil Culliton, Luiz Gustavo Martins, Elisa Bandy, David Huntsperger, Glenn Cameron, Arthur Zucker, Tris Warkentin, Ludovic Peran, Minh Giang, Zoubin Ghahramani, Clément Farabet, Koray Kavukcuoglu, Demis Hassabis, Raia Hadsell, Yee Whye Teh, and Nando de Freitas. Recurrentgemma: Moving past transformers for efficient open language models. *arXiv preprint arXiv:2404.07839*, 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 *USENIX Security Symposium*, pages 2633–2650, 2021.

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *IEEE Symposium on Security and Privacy*, pages 1897–1914, 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. *arXiv preprint arXiv:2601.02751*, 2026.

Nolan Dey, Gurpreet Gosal, Zhiming, Chen, Hemant Khachane, William Marshall, Ribhu Pathria, Marvin Tom, and Joel Hestness. Cerebras-GPT: Open compute-optimal language models trained on the cerebras wafer-scale cluster. *arXiv preprint arXiv:2304.03208*, 2023.

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

Vitaly Feldman. Does learning require memorization? a short tale about a long tail. In *Symposium on Theory of Computing*, pages 954–959, 2020.

Kazuki Fujii, Yukito Tajima, Sakae Mizuki, Hinari Shimada, Taihei Shiotani, Koshiro Saito, Masanari Ohi, Masaki Kawamura, Taishi Nakamura, Takumi Okamoto, Shigeki Ishida, Kakeru Hattori, Youmi Ma, Hiroya Takamura, Rio Yokota, and Naoaki Okazaki. Rewriting pre-training data boosts llm performance in math and code, 2025.

Gemma Team. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Sorami Hisamoto, Matt Post, and Kevin Duh. Membership inference attacks on sequence-to-sequence models: Is my data in your machine translation system? *Transactions of the Association for Computational Linguistics*, 8:49–63, 2020. doi: 10.1162/tacl_a_00299. URL `https://aclanthology.org/2020.tacl-1.4/`.

David Ilić, David Stanojević, and Kostadin Cvejoski. Powerful training-free membership inference against autoregressive language models. *arXiv preprint arXiv:2601.12104*, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

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

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Rishabh Misra. News category dataset. *arXiv preprint arXiv:2209.11429*, 2022.

MosaicML NLP Team. MPT-7B: A new standard for open-source, commercially usable llms. `https://www.mosaicml.com/blog/mpt-7b`, 2023.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, 2018.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The RefinedWeb dataset for Falcon LLM: Outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*, 2023.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Leon Derczynski, Xingjian Du, Matteo Grella, Kranthi Gv, Xuzheng He, Haowen Hou, Przemyslaw Kazienko, Jan Kocon, Jiaming Kong, Bartłomiej Koptyra, Hayden Lau, Jiaju Lin, Krishna Sri Ipsit Mantri, Ferdinand Mom, Atsushi Saito, Guangyu Song, Xiangru Tang, Johan Wind, Stanisław Woźniak, Zhenyuan Zhang, Qinghua Zhou, Jian Zhu, and Rui-Jie Zhu. RWKV: Reinventing RNNs for the transformer era. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14048–14077, 2023.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. ML-Leaks: Model and data independent membership inference attacks and defenses on machine learning models. In *Network and Distributed Systems Security Symposium*, 2019.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. Detecting pretraining data from large language models. *arXiv preprint arXiv:2310.16789*, 2024.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *IEEE Symposium on Security and Privacy*, pages 3–18, 2017.

Congzheng Song and Vitaly Shmatikov. Auditing data provenance in text-generation models. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 196–206. ACM, 2019. doi: 10.1145/3292500.3330885.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Ben Wang and Aran Komatsuzaki. GPT-J-6B: A 6 billion parameter autoregressive language model. `https://github.com/kingoflolz/mesh-transformer-jax`, 2021.

Wikimedia Foundation. Wikimedia downloads. `https://dumps.wikimedia.org`, 2024.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *IEEE Computer Security Foundations Symposium*, pages 268–282, 2018.

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

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

A Full results

Tables 4–9 present complete results for all model-dataset combinations.

Table 4: Full results on held-out transformers: AUC.

Model	Dataset	Loss	Min-K%++	Zlib	RefLoss	EZ-MIA	LT-MIA
GPT-2	AG News	0.745	0.704	0.717	0.790	**0.960**	0.945
GPT-2	WikiText	0.745	0.696	0.713	0.814	0.971	**0.980**
GPT-2	XSum	0.768	0.719	0.760	0.956	**0.994**	0.991
GPT-2	Code	0.618	0.608	0.618	0.608	**0.795**	0.761
Pythia-2.8B	AG News	0.603	0.565	0.584	0.740	0.872	**0.911**
Pythia-2.8B	WikiText	0.585	0.558	0.583	0.645	0.867	**0.944**
Pythia-2.8B	XSum	0.618	0.581	0.609	0.862	0.911	**0.975**
Pythia-2.8B	Code	0.596	0.552	0.586	0.717	0.857	**0.865**
Pythia-6.9B	AG News	0.638	0.597	0.615	0.796	0.917	**0.942**
Pythia-6.9B	WikiText	0.617	0.582	0.606	0.684	**0.911**	0.796
Pythia-6.9B	XSum	0.649	0.607	0.638	0.902	0.957	**0.990**
Pythia-6.9B	Code	0.626	0.573	0.612	0.761	0.909	**0.911**
OPT-2.7B	AG News	0.581	0.554	0.564	0.740	0.879	**0.908**
OPT-2.7B	WikiText	0.578	0.556	0.578	0.623	0.852	**0.862**
OPT-2.7B	XSum	0.628	0.592	0.621	0.899	0.940	**0.979**
OPT-2.7B	Code	0.561	0.540	0.562	0.605	0.692	**0.751**
Llama-2-7B	AG News	0.627	0.575	0.599	0.815	0.917	**0.948**
Llama-2-7B	WikiText	0.529	0.520	0.540	0.568	0.733	**0.781**
Llama-2-7B	XSum	0.671	0.620	0.671	0.930	0.972	**0.988**
Llama-2-7B	Code	0.624	0.571	0.609	0.784	0.902	**0.912**
Phi-2	AG News	0.585	0.556	0.568	0.698	0.845	**0.886**
Phi-2	WikiText	0.573	0.548	0.572	0.620	**0.859**	0.823
Phi-2	XSum	0.624	0.592	0.614	0.792	0.889	**0.952**
Phi-2	Code	0.573	0.534	0.562	0.732	0.824	**0.851**
Mistral-7B	AG News	0.595	0.551	0.573	0.787	0.883	**0.916**
Mistral-7B	WikiText	0.599	0.569	0.601	0.689	0.910	**0.966**
Mistral-7B	XSum	0.660	0.615	0.660	0.933	0.952	**0.989**
Mistral-7B	Code	0.594	0.547	0.578	0.777	0.856	**0.903**

B Experimental configuration

Fine-tuning Configuration. All target models were fine-tuned for 3 epochs with batch size 16, learning rate 5×10^{-5}, and sequence length 128 tokens. For most transformer models (Mistral-7B, Phi-2, OPT-2.7B, and all training-set transformers except DistilGPT-2), we used LoRA with rank $r = 16$, $\alpha = 32$, and dropout 0.05, applied to all attention projections and feed-forward layers. Other transformers are fully fine-tuned. Mamba, RWKV, and RecurrentGemma were fully fine-tuned without LoRA. Each model-dataset combination used 10,000 member and 10,000 non-member samples, with 5% held out for validation and 5% for testing. The checkpoint with lowest validation loss was selected for feature extraction.

Compute. All experiments were conducted on a single NVIDIA H200 GPU. We estimate total compute at approximately 9×10^{18} FLOPs (70 fine-tuning runs, feature extraction over 70 model-dataset combinations, and classifier training), corresponding to approximately 30 GPU-hours including overhead.

Table 10 documents the complete separation between training and evaluation. There is zero overlap in models or datasets.

C Feature definitions

Each input sample is a tensor of shape $(128, 154)$: 154 feature channels at each of 128 token positions. Features are extracted from both the fine-tuned target model and the pre-trained reference model.

Table 5: Full results on held-out transformers: TPR @ 1% FPR.

Model	Dataset	Loss	Min-K%++	Zlib	RefLoss	EZ-MIA	LT-MIA
GPT-2	AG News	0.018	0.026	0.014	0.020	**0.408**	0.274
GPT-2	WikiText	0.084	0.024	0.092	0.018	0.058	**0.632**
GPT-2	XSum	0.122	0.100	0.104	0.090	**0.868**	0.756
GPT-2	Code	0.024	0.010	0.010	0.010	0.030	**0.054**
Pythia-2.8B	AG News	0.014	0.020	0.012	0.042	0.132	**0.257**
Pythia-2.8B	WikiText	0.014	0.012	0.020	0.010	0.016	**0.492**
Pythia-2.8B	XSum	0.036	0.022	0.030	0.120	0.430	**0.653**
Pythia-2.8B	Code	0.034	0.040	0.036	0.028	**0.180**	0.129
Pythia-6.9B	AG News	0.012	0.024	0.012	0.052	0.208	**0.319**
Pythia-6.9B	WikiText	0.032	0.016	0.022	0.010	0.036	**0.137**
Pythia-6.9B	XSum	0.058	0.036	0.056	0.168	0.670	**0.826**
Pythia-6.9B	Code	0.038	0.042	0.040	0.078	**0.250**	0.217
OPT-2.7B	AG News	0.020	0.018	0.010	0.036	0.218	**0.264**
OPT-2.7B	WikiText	0.010	0.012	0.024	0.010	0.000	**0.262**
OPT-2.7B	XSum	0.040	0.040	0.022	0.050	0.350	**0.696**
OPT-2.7B	Code	0.018	0.028	0.010	0.014	0.024	**0.064**
Llama-2-7B	AG News	0.024	0.016	0.012	0.072	0.252	**0.317**
Llama-2-7B	WikiText	0.020	0.018	0.022	0.012	0.020	**0.130**
Llama-2-7B	XSum	0.044	0.040	0.058	0.136	0.628	**0.809**
Llama-2-7B	Code	0.060	0.026	0.048	0.074	0.144	**0.228**
Phi-2	AG News	0.012	0.014	0.010	0.050	0.166	**0.196**
Phi-2	WikiText	0.030	0.016	0.030	0.012	0.010	**0.187**
Phi-2	XSum	0.024	0.016	0.010	0.056	0.252	**0.465**
Phi-2	Code	0.038	0.006	0.026	0.034	**0.124**	0.098
Mistral-7B	AG News	0.012	0.018	0.010	0.052	0.140	**0.257**
Mistral-7B	WikiText	0.036	0.028	0.016	0.016	0.082	**0.666**
Mistral-7B	XSum	0.056	0.028	0.050	0.082	0.540	**0.848**
Mistral-7B	Code	0.052	0.028	0.020	0.122	0.192	**0.259**

Target model features (45 channels). Per-token cross-entropy loss (1); logits of target's top-20 tokens (20); logits of target's bottom-20 tokens (20); logit assigned to ground-truth next token (1); log-normalized rank of ground-truth token (1); sequence-level mean and standard deviation of loss, broadcast to all positions (2).

Reference model features (45 channels). Same structure as target features: per-token loss (1); reference's logits for the *target's* top-20 and bottom-20 tokens (40); ground-truth logit and rank (2); global statistics (2).

Comparison features (64 channels). Per-token loss difference $\ell_t^{\text{tgt}} - \ell_t^{\text{ref}}$ (1); sequence-level mean and standard deviation of loss difference (2); total log-likelihood ratio $\sum_t (\ell_t^{\text{tgt}} - \ell_t^{\text{ref}})$ broadcast to all positions (1); cross-model ranks for target's top-20 tokens in reference distribution (20); cross-model ranks for reference's top-20 tokens in target distribution (20); cross-model ranks for target's bottom-20 tokens (20).

Normalization. Logit features are locally normalized by subtracting the maximum within each comparison group. Rank features are log-transformed and normalized by $\log(V + 1)$ where V is vocabulary size.

D Classifier architecture ablation

Table 11 presents full results for the classifier architecture comparison. All variants are trained on identical features from 30 model-dataset combinations (540,000 samples total); only the classifier architecture differs.

Sequence modeling contributes 5.0 AUC points over pooling (0.925 vs. 0.875).

Table 6: Full results on held-out transformers: TPR @ 0.1% FPR.

Model	Dataset	Loss	Min-K%++	Zlib	RefLoss	EZ-MIA	LT-MIA
GPT-2	AG News	0.008	0.008	0.006	0.008	**0.124**	0.044
GPT-2	WikiText	0.008	0.002	0.006	0.000	0.002	**0.273**
GPT-2	XSum	0.032	0.048	0.016	0.016	**0.426**	0.340
GPT-2	Code	0.000	0.002	0.000	0.000	0.000	**0.006**
Pythia-2.8B	AG News	0.006	0.008	0.006	0.000	0.034	**0.064**
Pythia-2.8B	WikiText	0.002	0.000	0.000	0.004	0.000	**0.291**
Pythia-2.8B	XSum	0.008	0.004	0.006	0.022	0.092	**0.262**
Pythia-2.8B	Code	0.018	0.018	0.000	0.002	0.036	**0.045**
Pythia-6.9B	AG News	0.008	0.008	0.006	0.002	0.064	**0.073**
Pythia-6.9B	WikiText	0.004	0.002	0.002	0.004	0.000	**0.035**
Pythia-6.9B	XSum	0.010	0.004	0.006	0.028	0.012	**0.567**
Pythia-6.9B	Code	0.020	0.012	0.004	0.002	0.026	**0.051**
OPT-2.7B	AG News	0.008	0.008	0.006	0.020	0.012	**0.056**
OPT-2.7B	WikiText	0.000	0.000	0.000	0.000	0.000	**0.101**
OPT-2.7B	XSum	0.006	0.002	0.006	0.010	0.066	**0.298**
OPT-2.7B	Code	0.002	0.004	0.002	0.002	0.000	**0.015**
Llama-2-7B	AG News	0.010	0.012	0.006	0.030	**0.138**	0.061
Llama-2-7B	WikiText	0.004	0.006	0.006	0.002	0.000	**0.033**
Llama-2-7B	XSum	0.010	0.004	0.010	0.060	0.374	**0.580**
Llama-2-7B	Code	0.020	0.016	0.000	0.002	0.000	**0.051**
Phi-2	AG News	0.008	0.012	0.006	0.010	0.000	**0.033**
Phi-2	WikiText	0.004	0.004	0.004	0.008	0.000	**0.057**
Phi-2	XSum	0.002	0.002	0.000	0.016	0.046	**0.190**
Phi-2	Code	0.012	0.002	0.010	0.004	**0.018**	**0.019**
Mistral-7B	AG News	0.012	0.010	0.006	0.002	0.000	**0.036**
Mistral-7B	WikiText	0.012	0.008	0.008	0.000	0.000	**0.428**
Mistral-7B	XSum	0.006	0.002	0.010	0.024	0.214	**0.667**
Mistral-7B	Code	0.016	0.002	0.010	0.006	0.008	**0.092**

Table 7: Full results on non-transformer architectures: AUC.

Model	Dataset	Loss	Min-K%++	Zlib	RefLoss	EZ-MIA	LT-MIA
Mamba-2.8B	AG News	0.949	0.888	0.918	0.975	**0.983**	0.969
Mamba-2.8B	WikiText	0.966	0.886	0.897	0.947	0.979	**0.995**
Mamba-2.8B	XSum	0.988	0.949	0.977	0.999	0.999	**0.999**
Mamba-2.8B	Code	0.565	0.534	0.555	0.646	0.763	**0.887**
RWKV-3B	AG News	0.783	0.703	0.748	0.931	0.978	**0.976**
RWKV-3B	WikiText	0.792	0.704	0.753	0.865	0.965	**0.986**
RWKV-3B	XSum	0.860	0.766	0.853	0.993	0.998	**0.998**
RWKV-3B	Code	0.708	0.651	0.690	0.871	**0.943**	0.926
RecurrentGemma-2B	AG News	0.867	0.848	0.829	0.760	0.849	**0.924**
RecurrentGemma-2B	WikiText	0.860	0.835	0.822	0.759	0.914	**0.946**
RecurrentGemma-2B	XSum	0.938	0.881	0.939	0.898	0.985	**0.988**
RecurrentGemma-2B	Code	0.775	0.687	0.754	0.783	**0.911**	0.887

E Diversity ablation

Table 12 presents full results for the training diversity ablation. Total training samples are held constant at 18,000; only the number of model-dataset combinations varies.

Training on a single combination achieves near-perfect training metrics (0.998 AUC) but generalizes poorly (0.796 AUC). With 30 combinations, the train-eval gap shrinks from 20.2 points to 0.2 points, demonstrating that diversity filters model-specific artifacts.

Table 8: Full results on non-transformer architectures: TPR @ 1% FPR.

Model	Dataset	Loss	Min-K%++	Zlib	RefLoss	EZ-MIA	LT-MIA
Mamba-2.8B	AG News	0.150	0.150	0.114	0.316	**0.718**	0.206
Mamba-2.8B	WikiText	0.462	0.154	0.544	0.040	0.330	**0.900**
Mamba-2.8B	XSum	0.774	0.518	0.800	0.998	0.986	**0.996**
Mamba-2.8B	Code	0.036	0.028	0.022	0.028	0.070	**0.174**
RWKV-3B	AG News	0.092	0.060	0.026	0.084	0.402	**0.430**
RWKV-3B	WikiText	0.066	0.048	0.090	0.022	0.050	**0.624**
RWKV-3B	XSum	0.198	0.072	0.212	0.924	0.946	**0.981**
RWKV-3B	Code	0.076	0.046	0.058	0.202	**0.358**	0.338
RecurrentGemma-2B	AG News	0.110	0.082	0.014	0.104	0.038	**0.392**
RecurrentGemma-2B	WikiText	0.120	0.070	0.260	0.038	0.014	**0.316**
RecurrentGemma-2B	XSum	0.360	0.160	0.410	0.154	0.748	**0.750**
RecurrentGemma-2B	Code	0.150	0.054	0.144	0.080	**0.290**	0.138

Table 9: Full results on non-transformer architectures: TPR @ 0.1% FPR.

Model	Dataset	Loss	Min-K%++	Zlib	RefLoss	EZ-MIA	LT-MIA
Mamba-2.8B	AG News	0.044	**0.052**	0.012	0.000	0.018	0.018
Mamba-2.8B	WikiText	0.070	0.020	0.038	0.002	0.004	**0.300**
Mamba-2.8B	XSum	0.480	0.242	0.460	0.592	**0.888**	0.582
Mamba-2.8B	Code	0.010	0.018	0.002	0.002	0.048	**0.058**
RWKV-3B	AG News	0.010	0.010	0.008	0.002	**0.098**	0.060
RWKV-3B	WikiText	0.024	0.012	0.010	0.000	0.000	**0.231**
RWKV-3B	XSum	0.060	0.024	0.048	0.222	**0.820**	0.727
RWKV-3B	Code	0.014	0.020	0.014	0.002	**0.190**	0.148
RecurrentGemma-2B	AG News	0.025	0.025	0.005	0.001	0.022	**0.040**
RecurrentGemma-2B	WikiText	0.033	0.013	0.022	0.001	0.001	**0.110**
RecurrentGemma-2B	XSum	0.212	0.090	0.208	0.179	**0.726**	0.500
RecurrentGemma-2B	Code	0.026	0.025	0.020	0.002	**0.153**	0.012

F Calibration analysis

EZ-MIA's TPR@1% collapses to near-zero on WikiText for most models despite achieving moderate-to-high AUC. LT-MIA maintains substantially higher TPR across all models, explaining why its advantage grows at stricter operating thresholds.

G Pretraining/fine-tuning data overlap

Overlap between pretraining and fine-tuning corpora could in principle inflate measured memorization if the attack confuses pretraining familiarity with fine-tuning membership. The experimental design controls for this by construction. Members and non-members in each evaluation cell are drawn from the same underlying pool (10,000 of each, Appendix B), and pretraining exposure is determined before this split, so any overlap with the pretraining corpus is identically distributed across both classes. Under the null hypothesis of no fine-tuning effect the classes are exchangeable and every attack's expected AUC equals 0.5 regardless of contamination level, meaning that any observed lift above 0.5 must originate in fine-tuning. The per-token comparison features reinforce this control by measuring the distributional shift from the pre-trained reference, canceling likelihood that both models share from pretraining.

H Transfer across fine-tuning recipes

To test generalization across recipes, we fine-tuned Pythia-2.8B on WikiMIA under ten configurations spanning learning rates 1×10^{-5} to 2×10^{-4}, epoch counts 1 to 10, and both LoRA and full fine-tuning, then evaluated the released classifier zero-shot without retraining.

Table 10: Training and evaluation split. Training uses 10 transformer models $\times$ 3 datasets = 30 combinations. Evaluation uses 7 held-out transformers + 3 non-transformers $\times$ 4 datasets = 40 combinations.

Training Models (10)	DistilGPT-2 [Sanh et al., 2019], GPT-2-XL [Radford et al., 2019], Pythia-1.4B [Biderman et al., 2023], Cerebras-GPT-2.7B [Dey et al., 2023], GPT-J-6B [Wang and Komatsuzaki, 2021], Gemma-2B [Gemma Team, 2024], Qwen2-1.5B [Bai et al., 2023], MPT-7B [MosaicML NLP Team, 2023], Falcon-RW-1B, Falcon-7B [Penedo et al., 2023]
Training Datasets (3)	News Category Dataset [Misra, 2022], Wikipedia [Wikimedia Foundation, 2024], CNN/DailyMail [Hermann et al., 2015]
Eval Transformers (7)	GPT-2, Pythia-2.8B, Pythia-6.9B, OPT-2.7B, Llama-2-7B, Phi-2, Mistral-7B
Eval Non-Transformers (3)	Mamba-2.8B (state-space), RWKV-3B (linear attention), RecurrentGemma-2B (gated recurrence)
Eval Datasets (4)	AG News [Zhang et al., 2015], WikiText-103 [Merity et al., 2016], XSum [Narayan et al., 2018], Swallow-Code [Fujii et al., 2025]

Table 11: Classifier architecture ablation. The transformer processes features as a sequence; MLP and logistic regression flatten to a fixed-length vector; mean-pooled MLP averages features across positions before classification.

Classifier	Train AUC	Eval AUC	Train TPR@1%	Eval TPR@1%
Transformer (sequence)	0.914	**0.925**	0.601	**0.459**
MLP (flattened)	0.885	0.877	0.483	0.343
MLP (mean pooled)	0.885	0.875	0.506	0.319
Logistic Regression	0.838	0.839	0.351	0.237

AUC tracks optimization pressure monotonically, from 0.819 at the mildest recipe to 1.000 at the strongest, and the LoRA-versus-full-fine-tuning comparison at matched hyperparameters yields 0.996 versus 0.998.

I Persistence of SFT memorization through DPO

We fine-tuned Pythia-2.8B on AG News with the standard SFT recipe, then ran DPO on the SFT checkpoint (HuggingFaceH4/ultrafeedback_binarized, $\beta = 0.1$, lr $= 5 \times 10^{-7}$, 1 epoch) and applied the released classifier zero-shot to the DPO checkpoint.

SFT memorization survives DPO intact. DPO's own preference data is not detectable: querying DPO members against held-out examples yields 0.549 AUC with the pre-trained reference and 0.496 with the SFT checkpoint.

Table 12: Effect of training diversity on generalization with fixed total samples.

Combinations	Samples/Combo	Train AUC	Eval AUC	Train TPR@1%	Eval TPR@1%
1	18,000	0.998	0.796	0.992	0.185
10	1,800	0.875	0.843	0.371	0.244
30	600	0.862	**0.860**	0.409	**0.284**

Table 13: Calibration comparison: LT-MIA vs. EZ-MIA on WikiText (where calibration gap is largest).

Model	LT-MIA AUC	LT-MIA TPR@1%	EZ-MIA AUC	EZ-MIA TPR@1%
GPT-2	0.980	0.632	0.971	0.058
Pythia-2.8B	0.944	0.492	0.867	0.016
Pythia-6.9B	0.796	0.137	0.911	0.036
OPT-2.7B	0.862	0.262	0.852	0.000
Llama-2-7B	0.781	0.130	0.733	0.020
Phi-2	0.823	0.187	0.859	0.010
Mistral-7B	0.966	0.666	0.910	0.082
Mamba-2.8B	0.995	0.900	0.979	0.330
RWKV-3B	0.986	0.624	0.965	0.050
RecurrentGemma-2B	0.946	0.316	0.914	0.014

NeurIPS paper checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction (Section 1) state four contributions that are directly supported by experimental results in Section 4 and analysis in Section 5. Limitations on scope (e.g., fine-tuning only, reference model requirement) are discussed in Section 6.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 6 discusses limitations including degraded performance when the reference model already assigns high likelihood (WikiText-103 with large models), the requirement for a pre-trained reference checkpoint, restriction to fine-tuning scenarios, and reduced performance on code vs. natural language.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.

Table 14: LT-MIA AUC across fine-tuning recipes (Pythia-2.8B / WikiMIA, zero-shot).

Method	Epochs	lr = 1×10^{-5}	lr = 5×10^{-5}	lr = 2×10^{-4}
LoRA	1	0.819	0.952	0.997
	3	0.927	0.996	1.000
	10	0.989	1.000	1.000
Full FT	3	—	0.998	—

Table 15: SFT membership detection through a subsequent DPO stage (Pythia-2.8B, AG News).

Method	AUC	TPR @ 1% FPR	TPR @ 0.1% FPR
Loss	0.934	0.090	0.012
RefLoss	0.961	0.382	0.014
EZ-MIA	0.912	**0.528**	0.064
LT-MIA	**0.985**	0.523	**0.099**

- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not include formal theoretical results or proofs. The contributions are empirical.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.

- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 3 describes the full method (feature extraction, classifier architecture, training procedure). Appendix B specifies all fine-tuning hyperparameters, model lists, and data splits. Appendix C defines all 154 features. Code is included as an anonymized ZIP in the supplementary material.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code and the trained classifier are included as an anonymized ZIP file in the supplementary material. All datasets used are publicly available and cited with their original papers.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.

- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 3.3 specifies the optimizer (AdamW), learning rate, batch size, and training epochs for the attack classifier. Appendix B provides full fine-tuning configuration (epochs, batch size, learning rate, LoRA parameters, sequence length) and the complete train/eval split (Table 10). Checkpoint selection criterion (lowest validation loss) is stated.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report a Wilcoxon signed-rank test ($p < 0.001$) for the main comparison between LT-MIA and EZ-MIA (Section 4.2). Means for transformer results are computed over 7 models $\times$ 4 datasets (28 combinations), providing robustness through breadth of evaluation rather than repeated runs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix B reports that all experiments were conducted on a single NVIDIA H200 GPU, with total compute estimated at approximately 9×10^{18} FLOPs (approximately 30 GPU-hours including overhead). The breakdown covers fine-tuning, feature extraction, and classifier training.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conforms to the NeurIPS Code of Ethics. We discuss the dual-use nature of membership inference attacks and potential negative societal impacts in Section 8. All datasets and models used are publicly available and used in accordance with their licenses.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Section 8 discusses both beneficial applications (copyright verification, compliance auditing, memorization detection) and potential negative impacts (adversarial inference of training data membership). We argue that the cross-architecture vulnerability exists independently of our method and that characterizing it benefits defenders.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: The released artifact is a lightweight classifier (<500K parameters) for detecting memorization, not a generative model. Section 8 discusses the dual-use nature and argues the method primarily benefits defenders (copyright holders, auditors). The method's constraints (requiring reference model access and full vocabulary logits) inherently limit its applicability to open-weight models.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All datasets and models are cited with their original papers (Table 10). Known dataset licenses: WikiText-103 (CC-BY-SA 3.0), CNN/DailyMail (Apache 2.0), News Category Dataset (CC-BY 4.0), Wikipedia (CC-BY-SA 3.0), Swallow-Code (Llama 3.3 Community License, subject to The Stack v2 terms). AG News and XSum are listed as "unknown" license on distribution platforms; AG News is provided for non-commercial research use per its creators, and XSum is widely used in NLP research. All model checkpoints are publicly released under their respective open-weight licenses. All use is consistent with research purposes.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The released code and trained classifier are included as an anonymized ZIP in the supplementary material, with a README describing usage. The classifier architecture, training procedure, and feature definitions are fully specified in Section 3 and Appendices C–E.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: This work does not involve research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core method does not involve LLMs as an important, original, or non-standard component. The language models studied are the subject of membership inference, not a component of the attack method itself.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.