A Statistical and Multi-Perspective Revisiting of the Membership Inference Attack in Large Language Models

Bowen Chen[1], **Namgi Han**[1], **Yusuke Miyao**[1,2]
[1]Department of Computer Science, The University of Tokyo
[2]Research and Development Center for Large Language Models, National Institute of Informatics
{bwchen, hng88, yusuke}@is.s.u-tokyo.ac.jp

Abstract

The lack of data transparency in Large Language Models (LLMs) has highlighted the importance of Membership Inference Attack (MIA), which differentiates trained (member) and untrained (non-member) data. Though it shows success in previous studies, recent research reported a near-random performance in different settings, highlighting a significant performance inconsistency. We assume that a single setting does not represent the distribution of the vast corpora, causing members and non-members with different distributions to be sampled and causing inconsistency. In this study, instead of a single setting, we statistically revisit MIA methods from various settings with thousands of experiments for each MIA method, along with a study in text features, embedding, threshold decision, and decoding dynamics of members and non-members. We found that (1) MIA performance improves with model size and varies with domains, while most methods do not statistically outperform baselines, (2) Though MIA performance is generally low, a notable amount of differentiable member and non-member outliers exists and vary across MIA methods, (3) Deciding a threshold to separate members and non-members is an overlooked challenge, (4) Text dissimilarity and long text benefit MIA performance, (5) Differentiable or not is reflected in the LLM embedding, (6) Members and non-members show different decoding dynamics. [1]

1 Introduction

Large Language Models (LLMs) (Minaee et al., 2024) are trained with terabyte-level corpora (Chowdhery et al., 2023) that are automatically collected, even the data creators themselves can hardly give instance-level analysis over the collected corpora (Biderman et al., 2022). Such a situation has led to several issues, such as the data leakage of evaluation benchmarks (Sainz et al., 2023) and personal information (Yao et al., 2024).

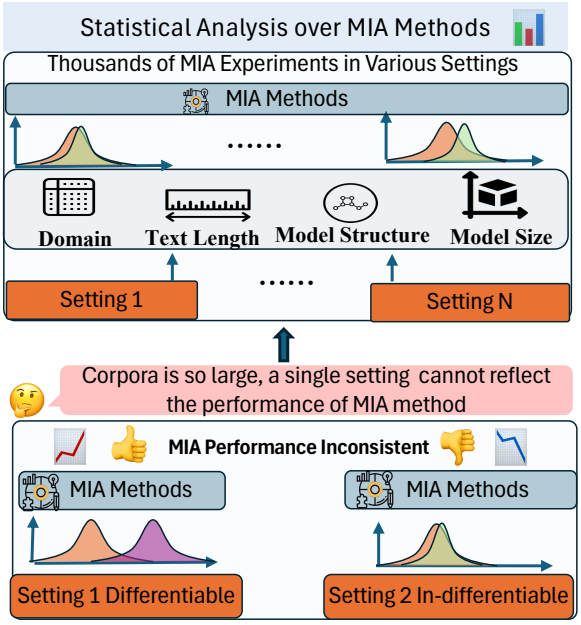

Figure 1: Sample with different settings may result in MIA performance inconsistency.

Those concerns inspired the Membership Inference Attacks (MIA) research in LLMs (Hu et al., 2022). Given a set of examples, MIA focuses on differentiating members (trained) and non-members (untrained) by calculating a feature value for each example and splitting them using a threshold. Those methods operate on the LLM outputs like generated tokens, probability distributions, losses, etc., and utilize them to distinguish between members and non-members. Despite the success reported in their studies, recent studies also suggested that these methods behave randomly in another MIA experiment setting, or that such benchmarks can be easily cheated (Duan et al., 2024; Das et al., 2024). Those negative results raised an inconsistency with respect to the performance of MIA methods, e.g., *do those MIA*

[1]Implementation of MIA methods used in this study is at https://github.com/llm-jp/llm-jp-membership-inference

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 22854–22874

methods really work or not ?

We see such an inconsistency comes from the distribution of the enormous pre-train corpora size, which is possible that members and non-members sampled from one setting could be totally different from another setting, leading to inconsistent results. In this study, instead of a single setting, we evaluate MIA methods statistically from multiple perspectives, e.g., the split methods, domains, text length, model sizes, and models. This led to thousands of MIA experiments for one MIA method and enabled a statistical analysis for MIA methods. Additionally, we conducted an in-depth analysis to study how the text feature, embedding, threshold decision, and decoding dynamics in members and non-members relate to MIA. Results show:

(I) MIA performance improves with model size and varies with domains, while most methods do not statistically outperform baselines.

(II) While MIA performance is generally low, a notable amount of differentiable member and non-member outliers exist and vary across MIA methods, connecting the inconsistency regarding the MIA performance.

(III) The threshold to separate members and non-members changes with model size and domains, raising it as an overlooked challenge when using MIA in the real world.

(IV) While the actual relation varies based on MIA methods, MIA performance generally positively relates to text length and text dissimilarity between members and non-members.

(V) Whether members and non-members are differentiable is reflected in LLM embedding with an emergent change in a larger model that makes them more separable. Specifically, the last-layer embedding used by the current MIA methods has a low embedding separability.

(VI) Domains with high MIA performance show a faster increase in accumulated entropy difference for members and non-members.

2 Related Works

2.1 Membership Inference Attack Methods

Gray-Box Method This method requires the intermediate outputs to be observable, like loss, token probability, etc. Carlini et al. (2021) calculated the loss difference with another reference model, with the assumption that if two models are trained under two samples of the same distribution, then the loss of non-members should be significantly different. Mink-$k\%$ (Shi et al., 2023) calculates the average log-likelihood of the tokens with the bottom-$k\%$ output probabilities, suggesting non-member text has more outliers and thus higher negative log-likelihood. Zhang et al. (2024b) improved Mink-$k\%$ by standardizing with variance and mean. Zhang et al. (2024c) compared predicted token probabilities against actual token probabilities from open corpora, in which the member data should have a closer distribution distance. Additionally, some methods alter the input text, like token swapping (Ye et al., 2024) or adding prefixes (Xie et al., 2024), with the hypothesis that the likelihood of member data should be influenced more by such text alternation.

Black-Box Method This method only observes the output tokens from the LLM. Dong et al. (2024) calculated a variant of edit distance with multiple generations from the LLM, with the hypothesis that those generations of a member text should have a smaller lexical distance compared to non-member text. Additionally, Kaneko et al. (2024) made a similar hypothesis while they evaluated the semantic similarity using the embedding model.

2.2 Membership Inference Attack Analysis

Regarding the MIA analysis, some research (Maini et al., 2021; Carlini et al., 2022) suggests the MIA difficulty increases with model size and corpora size. Zhang et al. (2024a) showed with toy data that it is hard to reliably operate an MIA method under a certain false positive rate. Meeus et al. (2024) found that some MIA benchmarks are flawed, which can be easily cheated by just checking the word differences (Das et al., 2024). Duan et al. (2024) evaluated Gray-Box MIA methods in the train and test sets of pre-train corpora of an LLM, where they behave almost randomly.

Those negative findings show an inconsistency with the performance reported by the previous MIA methods. We assume such inconsistency comes from the sampled member and non-member distribution under different settings, which could be totally different due to the enormous size of the corpora, leading to this inconsistency. In this study, instead of using a single setting, we create various settings, leading to thousands of experiments for one MIA method. This allows a statistical-level analysis of MIA methods from multiple perspectives, which shows new findings and connects the MIA performance inconsistency.

Baseline Methods	
Loss (Yeom et al., 2018)	Collects the loss value $L(M_t; x)$ for each input instance.
Refer (Carlini et al., 2021)	Calculates the loss gap $L(M_t; x) - L(M_r; x)$ between the attacked model M_t and a reference model M_r.
Gradient	Collects the gradient value $G(M_t; x)$ for each input instance.
Zlib (Carlini et al., 2021)	Calibrates the loss by the Zlib compression entropy $Z(x)$ of the input text, computed as $\frac{L(M_t;x)}{Z(x)}$.
Token Distribution Based Methods	
Hypothesis: non-member text contains more rare tokens or has a distribution whose average log-likelihood differs from that of member text.	
Min-k% Prob (Shi et al., 2023)	Calculates the average log-likelihood of $\mathrm{Bot}(x)$ tokens in the bottom-k% decoding probabilities in the whole input as $\frac{1}{E}\sum_{t_i \in \mathrm{Bot}(x)} \log p(t_i \mid t_1, \ldots, t_{i-1})$, where E is the number of bottom-k% tokens.
Min-k% Prob++ (Zhang et al., 2024b)	Standardizes Min-k% Prob with its mean and standard deviation to stabilize the value range.
DC-PDD (Zhang et al., 2024c)	Computes the divergence between decoded token probabilities and a pre-computed large corpus frequency.
Text Alternation Based Methods	
Hypothesis: the log-likelihood of altered member text is more sensitive than that of non-member text.	
EDA-PAC (Ye et al., 2024)	For an original text x and its perturbed text $\hat{x}$ created by token swapping, it calculates their differences in the average log-likelihood for top-k% and bottom-k% tokens. The hypothesis is that token swapping affects member texts more.
ReCall (Xie et al., 2024)	Creates a non-member prefix p and concatenates it with x to compute the score $\frac{LL(x\|p)}{LL(x)}$, where LL is the average log-likelihood. Member texts are hypothesized to be more sensitive to this prefix perturbation.
Black-Box Methods	
Hypothesis: the generated continuations of member-text are more similar to the actual continuations than those of non-member texts.	
SaMIA (Kaneko et al., 2024)	Inputs a partial prefix of the text and generates multiple continuations, then computes the average semantic similarity between generated and actual continuations.
CDD (Dong et al., 2024)	Generates multiple continuations for a given prefix, then calculates a variant of the lexical edit distance between the generated continuations and the actual continuation to derive a "peakiness" score.

Table 1: Summary of MIA Methods used in this study.

3 Experiments Setting

Given a model M and text set $X = \{x_0 \ldots x_n\}$, where each text x contains $\{t_0 \ldots t_m\}$ tokens. A MIA method calculates feature scores $S = f(M; X) = \{s_0 \ldots s_n\}$ for every text instance. A threshold t will be selected to classify whether x_i belongs to the training data D of the model M. Data that are in the D ($x_i \in D$) are called member data, otherwise called non-member data. MIA methods used in this study are in Table 1.[2]

3.1 Datasets

We use one existing benchmark and sample data from pre-train corpora with various settings.

WikiMIA (Shi et al., 2023) contains Wikipedia text sampled at the timestamp of 2023/10. Text samples before the time stamp are member text, and those after it are non-member text. This benchmark has been used by several MIA methods (Kaneko et al., 2024; Zhang et al., 2024c), which shows that its member and non-member splits are separable. Thus, we use it as a separable MIA benchmark in some experiments for reference.

Pile (Gao et al., 2020) corpora contain texts collected from domains like arXiv, GitHub, Freelaw, PubMed, DM Math, etc, with the train, test, and validation sets. The train set text and test set text can be treated as member text and non-member text, which will be the main focus of this study.

Dolma (Soldaini et al., 2024) corpora contain 3 trillion tokens from various domains. Unlike Pile, most of the sub-domains in Dolma only provide train splits. Therefore, we can only select domains where valid splits are clearly illustrated according to the introduction and training configuration files, which are arXiv, Open Web Math, Wikipedia, pes2o, Algebraic Stack, and Code Search Net. [3]

3.2 MIA Experiments Construction

We provide three split methods to construct the member and non-member sets.

Truncate Split (Duan et al., 2024) creates the member set and non-member set by truncating texts into a fixed range. We extend it by setting a length range of 100 from 0 to 1000.

Complete Split samples member and non-member texts whose whole length is in a text range that follows the Truncate Split.

Relative Split calculates the ten-percentile text length range based on the test set of each domain. The member and non-member texts are sampled from those ten-percentile length ranges. Each split method is applied to all domains in the Pile, with a minimum of 100 examples for both members and non-members. As text distribution varies by

[2] We did not apply the Refer method in OLMo as no previous research on the suitable reference model for OLMo.

[3] https://github.com/allenai/OLMo

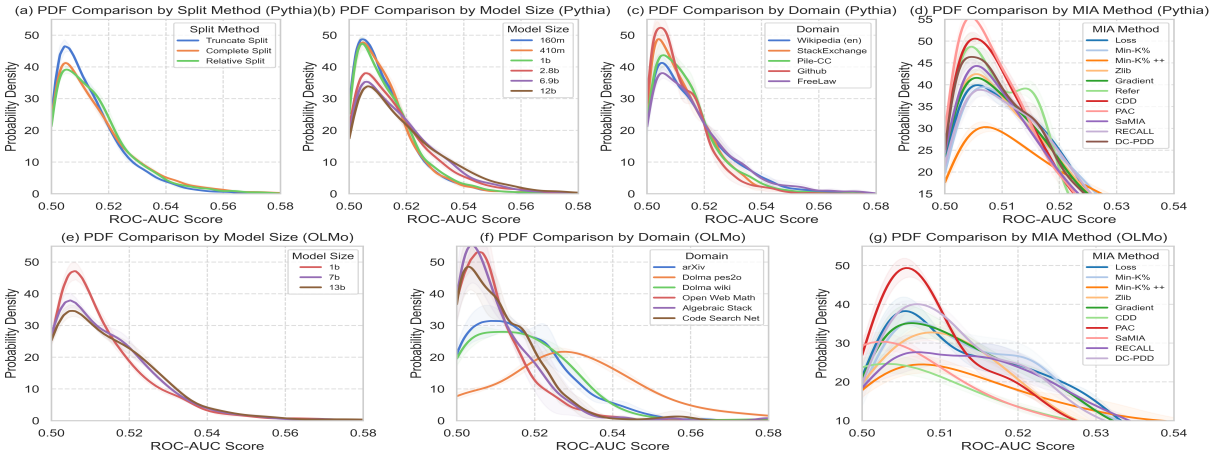

Figure 2: ROC-AUC probability density in different dimensions while fixing other dimensions. Less area on the left side means statistically better MIA performance. Shade area means variance from random seeds. We enlarge Figures (d) and (f) to increase readability due to the number of MIA methods.

domain, not every domain meets this criterion. [4] Moreover, most Dolma validation data are released as a one-row array which we cannot recover their original untruncated texts. Therefore, the Complete and Relative Splits are not available for the OLMo model. In total, we collected nearly 100 GBs of member and non-member texts sampled from different settings for MIA experiments. [5]

Our statistical evaluation contains (1) more domains (compared to WikiMIA(Shi et al., 2023), ArxivMIA (Duan et al., 2024), BookMIA (Shi et al., 2023)), (2) wider text length range (compared to MIMIR (Duan et al., 2024)), (3) considered the truncation method and domain-specific sampling, in which the previous MIA settings only considered absolute truncated situations, which ignored the domain-dependent length and how semantic completeness affects MIA methods. We run on every length split on domains in that split for every model size in all random seeds, resulting in 5,760 experiments (4,860 in Pythia and 900 in OLMo) for a single MIA method.

3.3 Models

Pythia (Biderman et al., 2023) (160m, 410m, 1b, 2.8b, 6.9b, 12b) and OLMo (OLMo et al., 2024) (1b, 7b, 13b) trained on the Pile and Dolma are used where valid and test sets are non-members.

3.4 Evaluation Metric

ROC-AUC (Fawcett, 2006) iterates every threshold for binary classification to calculate the True Positive Rate (TPR) and False Positive Rate (FPR) to form a ROC curve. AUC is the area under this curve and is used to analyze MIA performance.

Davies-Bouldin Score (Shi et al., 2023) (DB Score) evaluates the separability of two clusters of embeddings. A lower value indicates a better separability. This is used to evaluate the separability of embeddings for members and non-members.

4 Results

We first provide a statistical analysis between ROC-AUC scores of MIA methods with multiple factors, which generally aligns with previous negative studies, while also revealing new findings. Following this, we also conduct an analysis of ROC-AUC outliers not captured by the statistical approach, offering both statistical and individual views of MIA. Next, we discuss the existence of a unified good threshold to analyze the practical effectiveness of MIA methods. We then investigate how MIA relates to input texts by examining its correlations with text length and similarity. Finally, we study MIA performance from the LLM structural level by analyzing embedding separability and decoding dynamics for members and non-members. [6]

4.1 Effect of Different Factors

We aggregate ROC-AUC scores between 0.50 and 0.58, containing most experiments, and calculate their probability density over the split method, model size, domain, or MIA methods while fixing

[4] Domains in each split are in Appendix Table 5b.
[5] OLMo Validation Data
[6] Appendix A contains all experiments in this study for unlisted domains or MIA methods in the main contents.

Method	Pythia					OLMo				
	Model			ROC-AUC		Model			ROC-AUC	
	410m	2.8b	12b	Max	Mean	1b	7b	13b	Max	Mean
Loss	12	20	30	.585	.561	2	5	9	.580	.559
Gradient	14	54	31	.631	.563	3	5	15	.557	.552
Refer	12	12	11	.572	.559	-	-	-	–	–
Zlib	12	22	47	.590	.562	17	16	21	.596	.567
Min-k%	12	25	48	.600	.562	4	8	15	.568	.555
Min-k% ++	11	94	173	.631	.564	13	9	19	.582	.559
DC-PDD	0	12	15	.575	.558	3	9	12	.563	.556
EDA-PAC	4	1	5	.573	.557	5	1	1	.567	.558
RECALL	15	24	33	.806	572	4	6	9	.923	.651
SaMIA	37	37	34	.647	.569	11	10	19	.658	.612
CDD	17	12	6	.604	.561	23	15	21	.742	.595

Table 2: The number of outliers across model sizes and MIA methods. Max and Mean are the maximum and average ROC-AUC scores of those outliers. The highest value in each column is underscored.

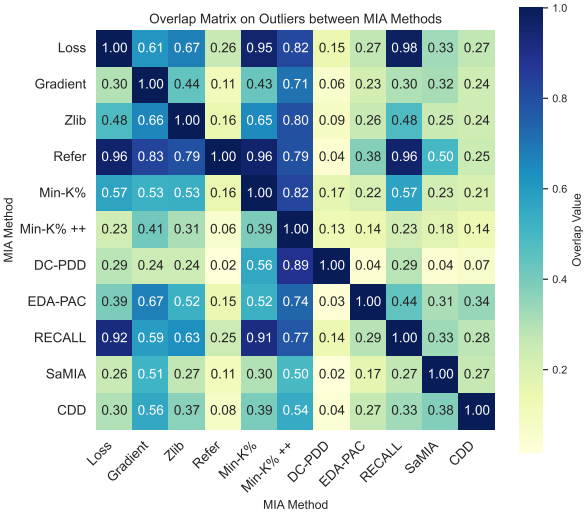

Figure 3: MIA outliers overlap matrix across methods.

the others in Figure 2. [7] While 0.50-0.52 occupies most probability densities, we still observe:[8]

(I) In Figure 2 (a), the commonly used Truncate Split shows the worst performance, while the Relative Split gives the best performance. Truncating a text may cause outlier words to be lost, and such contextual information loss affects MIA methods that rely on alternating the original members. It also affects the Black-Box method as the quality of generated tokens deteriorates.

(II) In Figure 2 (b) and (e), MIA performance improves with model size, particularly from 1b to 2.8b (Pythia) and 1b to 7b (OLMo), which contradicts previous findings that suggest it should decrease with model size. One possible explanation is that small models struggle with learning large corpora due to limited capacity, making most members behave like non-members, thus reducing MIA performance. As model capacity scales, more member texts are well learned, which starts to differ from non-members and enhances performance. However, this does not falsify previous research. If a very strong LLM that even fits well with non-member texts exists, it may again show a low MIA performance. In this case, the model size and MIA performance relation is an inverse U-curve.

(III) In Figure 2 (c) and (f), among shared domains across split methods, Wikipedia (en) and FreeLaw show statistically better performance compared to other domains in Pythia. In OLMo, Dolma Wiki, Dolma pes2o, and arXiv show better performance. We suggest this is related to token diversity. Domains related to codes (Github, StackExchange, etc.) or math (Open Web Math, Algebraic Stack) have less token diversity compared to those free text domains. This can be further validated within those free-text domains. Domains like Pile-CC or arXiv, which contain code/text mixes or mostly science papers, have lower performance compared to Wikipedia (en) and Dolma pes2o, which contain various text or paper topics.

(IV) In Figure 2 (d) and (g), the MIA method's performance is not consistent across models. Only Min-k% ++ and RECALL are still relatively better in both Pythia and OLMo. Other methods lie between these baselines, and their performances are within the variance from random seeds. Nevertheless, this does not reflect their peak performance in particular setups since the probability density assesses the overall effectiveness.

4.2 Analysis of Outliers in MIA

Though MIA performance is low, we observed notable outliers with higher differentiability (ROC-AUC > 0.55) not captured by the probability density. In Table 2, we list the number of these outliers across different MIA methods and model sizes, as well as their maximum and mean ROC-AUC values. Figure 3 then illustrates whether different MIA methods share the same outliers.

Outliers and Model Size Scaling (I) ROC-AUC outliers occupy only a small fraction overall (less than 10% even for Min-k% ++), aligning with reported low MIA performance in previous studies. However, the existence of outliers also helps ex-

[7] The analysis over the numerical values of probability mass is in Appendix Sec A.10

[8] We also formalized MIA as a hypothesis test for whether members and non-members are differentiable. The results are in Appendix Section A.6 and A.9.2.

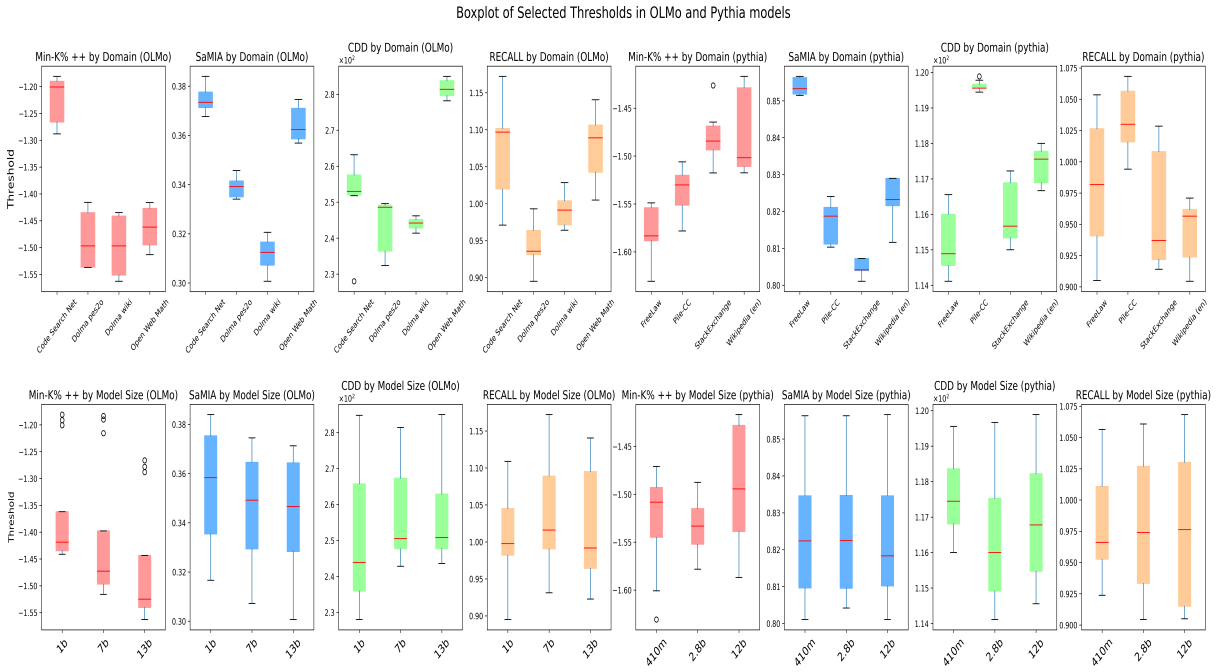

Figure 4: Threshold boxplot for different MIA methods across domains and model sizes in OLMo and Pythia.

plain cases where previous studies have observed positive results in which the RECALL method has the highest ROC-AUC performance of 0.81 and 0.92. Furthermore, this indicates that generally successful MIA methods (e.g., Min-$k\%$ ++) are not the best in every scenario, underscoring the importance of statistical MIA evaluations. (II) In most MIA methods, the number of outliers increases with model size, consistent with Section 4.1. As this trend is observed in both probability density (statistically) and outliers (individually), LLM internal structure may change in a way that favors MIA as size scales. Additionally, methods that do not rely only on internal states (e.g., SaMIA, CDD, Refer) appear less sensitive to size scaling.

Overlap between MIA Methods (I) Even the best-performing method (Min-$k\%$ ++) does not generally have high overlap outliers with others, which is only a 6% overlap with the Refer baseline. SaMIA and CDD, which rely solely on output tokens, exhibit a low overlap with the other methods as their features differ significantly from MIA methods relying on internal LLM states.[9] (II) While most methods do not outperform the baselines statistically, their low overlap implies that each method works in different situations, suggesting no "winner-takes-all" situation in MIA.

[9]As the size of outliers is small in OLMo due to its smaller experiment size, so we only analyze the Pythia models while we put the OLMo overlap matrix in the Appendix A.9.1.

4.3 Does a universal good threshold exist?

The ROC-AUC metric iterates feature values to differentiate members from non-members, but does not show how to decide a threshold and whether the threshold is generally effective. We split each set of member and non-member texts into train/validation sets with a 4:1 ratio and use the Geometric Mean (Youden, 1950) to find a threshold that balances the true positive rate and false positive rate. The distribution of this threshold across model sizes and domains is shown in Figure 4. We can obtain the following from those two figures:

(I) In the top figure, the threshold varies not just between domains but also within the same domain with the existence of outliers. In the bottom figure, the threshold changes with model sizes, as most MIA methods rely on the output likelihood, which is related to the model size. This trend can be observed in both OLMo and Pythia. The SaMIA, which relies on an external model to compare sentence similarity, is less affected by the model size, further confirming this point.

(II) These results show the generalizability of the MIA threshold as an overlooked challenge. A threshold may not work even within the same domain, may not transfer to another domain, and may not work in another model size, leading to a high possibility of performance deterioration when using the MIA method in the real world.

Method	Pythia: Text Length					Pythia: Text Similarity					OLMo: Text Length		OLMo: Text Similarity	
	Trunc	Comp	Rel	Avg	S-T	Trunc	Comp	Rel	Avg	S-T	Trunc	S-T	Trunc	S-T
Loss	.12	.23	.21	.17	✓	-.07	-.12	-.20	-.13	✓	.15	✓	–.28	✓
Refer	.01	-.11	.29	.11	✓	-.20	.05	-.33	-.16	✓	–	–	–	-
Zlib	.13	.35	.22	.23	✓	-.16	-.29	-.16	-.20	✓	.13	✓	<u>–.39</u>	✓
Min-k%	.30	.24	.34	.29	✓	-.21	-.17	-.22	-.20	✓	.20	✓	–.31	✓
Min-k% ++	<u>.57</u>	<u>.64</u>	<u>.48</u>	<u>.56</u>	✓	<u>-.50</u>	<u>-.53</u>	<u>-.43</u>	<u>-.48</u>	✓	.22	✓	<u>–.39</u>	✓
DC-PDD	.28	.14	.22	.21	✓	-.27	-.14	-.23	-.22	✓	.17	✓	–.25	✓
EDA-PAC	-.07	.11	.01	.06	✗	-.09	-.21	-.09	-.14	✓	.12	✓	–.16	✓
RECALL	.17	.23	.16	.21	✓	-.06	-.05	-.12	-.08	✗	<u>.23</u>	✓	–.05	✗
SaMIA	-.15	-.28	-.13	-.18	✗	-.17	-.14	-.13	-.14	✓	–.24	✗	–.10	✓
CDD	.15	-.16	.09	.03	✗	-.05	.02	-.41	-.15	✓	–.03	✗	–.35	✓
Avg	.16	.14	.17	**.16**	N/A	-.18	-.16	-.23	**-.19**	N/A	**.10**	N/A	**–.23**	N/A

Table 3: Average Spearman Correlation between ROC-AUC and Text Length/Text Similarity in OLMo and Pythia models across MIA methods in all domains. Trunc, Comp and Rel means Truncate, Complete and Relative split. S-T means whether the corresponding MIA method pass the significance test for the positive/negative correlation with Text Length/Text Similarity. We underscore the highest/lowest value in columns in Text Length/Text Similarity.

4.4 Text Similarity and Text Length

Previous studies showed text length (Zhang et al., 2024c) or token differences (Duan et al., 2024) contribute to the MIA, but with results induced from a single split, lacking general evidence. In this section, we calculate the Spearman correlation (Schober et al., 2018) between the ROC-AUC score with text length and the 7-gram overlap occurrence for every MIA split in Table 3.[10]

(I) For most of the MIA methods, its average correlation with length is positive, indicating longer text benefits MIA in general. However, it also varies based on split methods and MIA methods. We see that SaMIA and CDD showed a negative and near-zero correlation. For such Black-Box methods, the generated tokens will largely deviate from the actual continuation for both members and non-members in long text. The SaMIA used semantic comparison, which is affected more by such a deviation than the lexical distance of CDD.

(II) In the text similarity, we see a negative relation, indicating that token differences between members and non-members benefit the MIA performance. This partly explains the general low MIA performance, e.g., they detect word differences rather than member and non-member differences. Additionally, the general trend in correlation for text length and similarity is consistent in the Pythia and OLMo models.

(III) Additionally, some methods cannot pass the significance test ($p < 0.05$). SaMIA and CDD failed in the Text Length as generated tokens largely deviate when inputting long texts. RECALL inserts a prefix for both members and non-members that increases the text similarity at a running time, so it is naturally less sensitive to text similarity.

[10]We also calculated the correlation coefficient using other metrics in Appendix A.4 .

4.5 MIA and LLM Internal Strucutre

In this section, we study how MIA relates to the LLM structure to answer the question of *whether members and non-members are initially indistinguishable in the internal states*, and whether they exhibit different decoding dynamics, and whether such observations hold true in different LLMs.

4.5.1 Embedding Probing and Separability

We collect the average pooled hidden states for both members and non-members for each layer. Then, we use the DB Score and directly train a Transformer classifier to evaluate embedding separability as shown in Figure 5.

(I) The DB Score is around 10-20 for the differentiable splits, where the Transformer classifier obtains 70%–100% accuracy. By contrast, it reaches around 40 for the in-differentiable splits, where accuracy is near random. This indicates that differentiable/in-differentiable splits can be readily decided at the embedding level, and the degree of embedding separability is related to the MIA performance as it affects the behavior of LLM outputs.

(II) As model size increases, the DB Score curve exhibits emergent behavior on in-differentiable domains. PubMed and Pile-CC do not show a DB Score decrease at 410m, but their DB Scores suddenly drop in deeper layers at 2.8b size, indicating higher embedding separability between member and non-member embeddings, which is even more significant at 12b. A similar pattern is also observed in OLMo models for arXiv and Code Search Net when scaling from 1b to 7b and 7b to 13b. This jump in separability explains

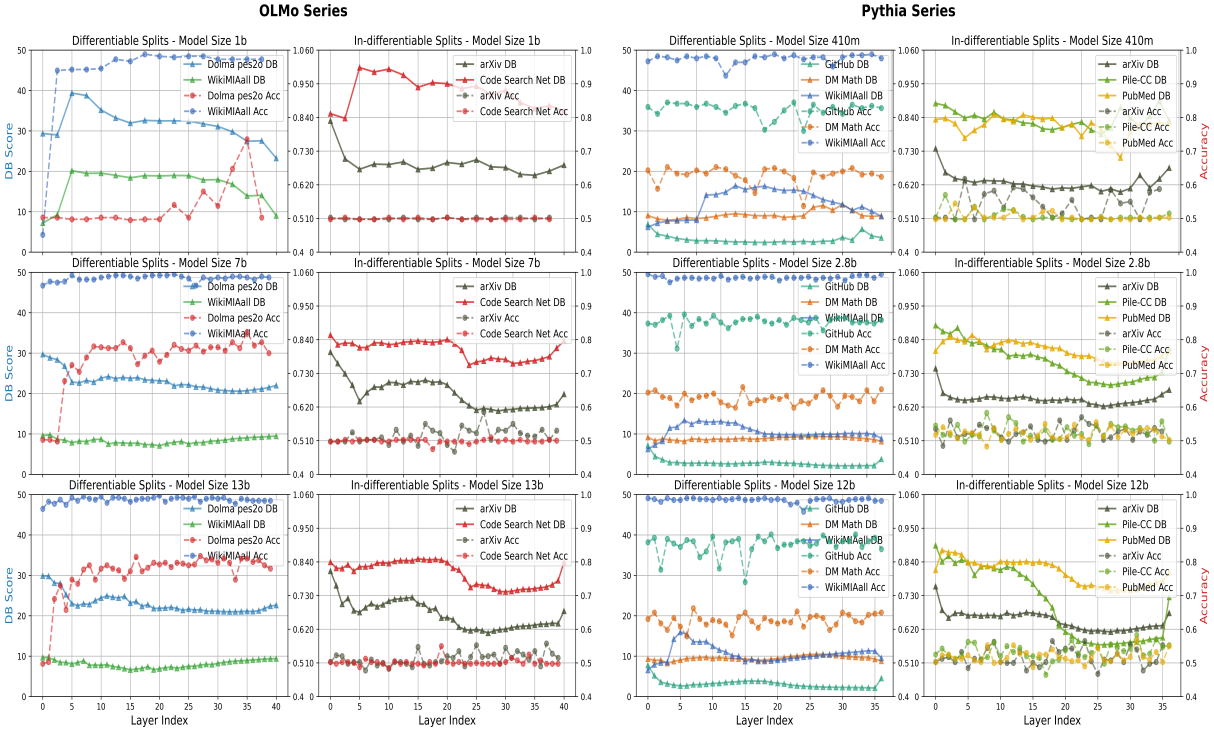

Figure 5: The DB Score (solid triangle line) and Transformer Classifier Accuracy (dotted circle line) on the member and non-member embeddings. Differentiable/In-differentiable outliers are selected from [DM Math, GitHub, WikiMIA]/[arXiv, Pile-CC, PubMed] for Pythia. Differentiable/In-differentiable outliers are selected from [WikiMIA, Dolma pes2o]/[arXiv, Code Search Net] for Pythia.

the ROC-AUC performance boost in Pythia (from 1b to 2.8b) and OLMo (from 1b to 7b), as larger models have more separable embeddings, increasing overall ROC-AUC performance.

(III) The DB Score rebounds to a high DB score in the final layer, indicating a decreased embedding separability. Given that many MIA methods rely on the computed outputs of the last layer (e.g., likelihood, tokens), this rebound helps explain why MIA performance is generally low: the final layer's embeddings are not well suited for the MIA task.

4.5.2 Generation Entropy Dynamics

We calculate the token entropy for members and non-members and their accumulated entropy difference $\Delta_t = \sum_{i=1}^{t} |H_i^{\text{mem}} - H_i^{\text{non}}|$ across decoding steps in Figure 6. The H_i^{mem} and H_i^{non} means the decoded entropy at step i for member texts and non-member texts, respectively.

(I) Firstly, we see that a low or high domain entropy (GitHub, Code Search Net) does not relate to its MIA performance in Figure 2. However, the domain-dependent entropy (decoding probability) means a domain-dependent log-likelihood, which explains the low threshold generalizability of Gray-Box methods. This also helps to explain the good performance of Min-$k\%$ ++ as it standardizes the log-likelihood of input tokens, erasing such domain or input text dependency.

(II) Though decoding entropy at each step does not show a clear relation with the MIA performance, the accumulated entropy difference increases during decoding, suggesting non-members have a statistically higher entropy compared to the member texts. Moreover, the domains with higher MIA performance (FreeLaw, Dolma pes2o in Figure 2) have a higher increasing speed in the accumulated entropy difference than other low MIA performance domains (GitHub, arXiv).

5 Discussion

Fairer MIA Evaluation While the statistical evaluation cannot be treated as a common evaluation benchmark, it does not mean that we are running out of options to evaluate MIA transparently and fairly. Future benchmarks in this direction should cover more domains and also report more statistical details like the token frequency of benchmarks by checking them against packages

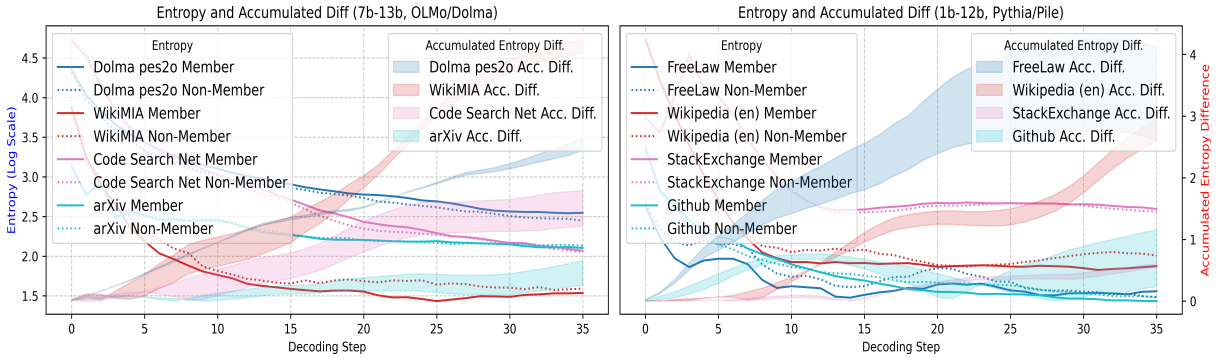

Figure 6: Entropy and accumulated entropy difference over decoding steps in different domains. We only draw average values for entropy to improve readability. The shaded area means the standard deviation across model sizes.

like infini-gram (Liu et al., 2024). Instead of claiming a benchmark can disable previous MIA methods, making the benchmark itself transparent is more important, since we could give an estimation based on such information.

Direction for Future MIA Methods While this study did not propose a new method for the MIA, we think the following directions are worthy of discussion. (I) Optimally choose the MIA method for different situations. Based on the results from the Figure 3, we can tell that there is no "winner-takes-all" situation in MIA, as we cannot expect one MIA hypothesis to work across the diverse situations of the pre-train corpora. Instead of trying to propose an MIA hypothesis that works for every situation, finding how to select a combination of methods based on situations is more promising. (II) While we are not able to assess the internal states for close source models, based on the research results presented in the Figure 5, the final embedding is not an optimal layer for the MIA task, it is may be worthy of developing methods that could integrate information from different layers as different layers have different functionality in LLM (Jin et al., 2025; Zhang et al., 2024d).

Stabilizing the Threshold Our results also suggested the instability of the threshold when using MIA methods. This poses a challenge for the MIA method to be used in the real world. We cannot decide on a threshold beforehand since we cannot predict what kind of texts the user will input. If a good uniform threshold cannot be decided, the performance of those MIA methods will largely deteriorate. To address this challenge, one solution is to standardize or normalize the output of the LLMs and operate on such normalized outputs instead of using the raw outputs of LLMs. This not only helps to address the threshold variance across domains but also may help address the threshold variance across different model sizes.

6 Conclusion

In this study, we statistically revisited MIA with an in-depth analysis from multiple perspectives to address the performance inconsistency of MIA methods reported in recent literature. We constructed a statistical evaluation of MIA methods without thousands of experiment for one method to profile their performances. Our results show that MIA performance improves with model size and varies across domains, with most MIA methods showing no advantage compared to baselines. Our results generally support previous negative results, but notable amounts of performance outliers make space for positive results, connecting the MIA performance inconsistency. We also found that deciding on a unified good threshold is an overlooked challenge. Additionally, long text and text dissimilarity benefit the MIA performance. The separability of members and non-members is also reflected in the LLM embedding with emergent change that benefits MIA in large models. The final layer used by current MIA methods may be suboptimal due to low embedding separability. Finally, differentiable members and non-members show faster accumulated entropy differences.

Limitations

The analysis of the results is based on the statistical level. This means we do not either support or negate the experiment results that were conducted in a single setting or benchmark in previous studies, e.g statistical analysis should be a standalone analysis. We cannot scale the model size further be-

cause Pythia and OLMo only provide model sizes up to 12b and 13b. Additionally, only very few LLMs have released their pre-train data, and their pre-train data is different, so it is hard to conduct such experiments on more models. [11]

Though we tried to extend the experiment scale further, the test size and valid data limited it. Their texts will be exhausted with further samplings and no longer satisfy the experiment requirements, where we want sampled members and non-members to be different each time.

Ethical Considerations

The original Pile data was reported to contain domains related to copyright issues, which are Books3, BookCorpus2, OpenSubtitles, YTSubtitles, and OWT2. We have made sure we did not conduct any MIA experiment on any of those domains, and we used processed Pile corpora that removed those domains, which are accessible online. [12] The Dolma data was collected properly according to their statements, and we also followed the corresponding instructions.

For other data we have used, we have made sure the usage aligns with the data license and its intended usage. Though we conducted experiments over the Pile and Dolma corpora, we did not observe any personal information or offensive content during the experiments.

Acknowledgements

This work was partially supported by the Institute of AI and Beyond of the University of Tokyo and by the "R&D Hub Aimed at Ensuring Transparency and Reliability of Generative AI Models" project of the Ministry of Education, Culture, Sports, Science and Technology.

References

Stella Biderman, Kieran Bicheno, and Leo Gao. 2022. Datasheet for the pile. *arXiv preprint arXiv:2201.07311*.

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

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

Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650. USENIX Association.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Debeshee Das, Jie Zhang, and Florian Tramèr. 2024. Blind baselines beat membership inference attacks for foundation models. *arXiv preprint arXiv:2406.16201*.

Yihong Dong, Xue Jiang, Huanyu Liu, Zhi Jin, Bin Gu, Mengfei Yang, and Ge Li. 2024. Generalization or memorization: Data contamination and trustworthy evaluation for large language models. *arXiv preprint arXiv:2402.15938*.

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

Tom Fawcett. 2006. An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861–874. ROC Analysis in Pattern Recognition.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Hongsheng Hu, Zoran Salcic, Lichao Sun, Gillian Dobbie, Philip S Yu, and Xuyun Zhang. 2022. Membership inference attacks on machine learning: A survey. *ACM Computing Surveys (CSUR)*, 54(11s):1–37.

Mingyu Jin, Qinkai Yu, Jingyuan Huang, Qingcheng Zeng, Zhenting Wang, Wenyue Hua, Haiyan Zhao, Kai Mei, Yanda Meng, Kaize Ding, Fan Yang, Mengnan Du, and Yongfeng Zhang. 2025. Exploring concept depth: How large language models acquire knowledge and concept at different layers?

[11] Most famous open-source LLMs, like the LLaMA series or Qwen series, did not release their pre-train data.

[12] https://huggingface.co/datasets/monology/pile-uncopyrighted

Masahiro Kaneko, Youmi Ma, Yuki Wata, and Naoaki Okazaki. 2024. Sampling-based Pseudo-Likelihood for Membership Inference Attacks. *arXiv preprint arXiv:2404.11262*.

Jiacheng Liu, Sewon Min, Luke Zettlemoyer, Yejin Choi, and Hannaneh Hajishirzi. 2024. Infini-gram: Scaling unbounded n-gram language models to a trillion tokens. *arXiv preprint arXiv:2401.17377*.

Pratyush Maini, Mohammad Yaghini, and Nicolas Papernot. 2021. Dataset inference: Ownership resolution in machine learning. *arXiv preprint arXiv:2104.10706*.

Matthieu Meeus, Igor Shilov, Shubham Jain, Manuel Faysse, Marek Rei, and Yves-Alexandre de Montjoye. 2024. SoK: Membership Inference Attacks on LLMs are Rushing Nowhere (and How to Fix It). *arXiv preprint arXiv:2406.17975*.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint arXiv:2402.06196*.

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

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

Patrick Schober, Christa Boer, and Lothar Schwarte. 2018. Correlation coefficients: Appropriate use and interpretation. *Anesthesia & Analgesia*, 126:1.

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

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Harsh Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Nathan Lambert, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Abhilasha Ravichander, Kyle Richardson, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Pete Walsh, Luke Zettlemoyer, Noah A. Smith, Hannaneh Hajishirzi, Iz Beltagy, Dirk Groeneveld, Jesse Dodge, and Kyle Lo. 2024. Dolma: an Open Corpus of Three Trillion Tokens for Language Model Pretraining Research. *arXiv preprint*.

Roy Xie, Junlin Wang, Ruomin Huang, Minxing Zhang, Rong Ge, Jian Pei, Neil Zhenqiang Gong, and Bhuwan Dhingra. 2024. Recall: Membership inference via relative conditional log-likelihoods. *arXiv preprint arXiv:2406.15968*.

Yifan Yao, Jinhao Duan, Kaidi Xu, Yuanfang Cai, Zhibo Sun, and Yue Zhang. 2024. A survey on large language model (llm) security and privacy: The good, the bad, and the ugly. *High-Confidence Computing*, 4(2):100211.

Wentao Ye, Jiaqi Hu, Liyao Li, Haobo Wang, Gang Chen, and Junbo Zhao. 2024. Data contamination calibration for black-box LLMs. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 10845–10861, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. 2018. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pages 268–282.

William J. Youden. 1950. Index for rating diagnostic tests. *Cancer*, 3(1):32–35.

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

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

Weichao Zhang, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Yixing Fan, and Xueqi Cheng. 2024c. Pretraining data detection for large language models: A divergence-based calibration method. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5263–5274, Miami, Florida, USA. Association for Computational Linguistics.

Yang Zhang, Yanfei Dong, and Kenji Kawaguchi. 2024d. Investigating layer importance in large language models. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 469–479, Miami, Florida, US. Association for Computational Linguistics.

A Appendix

A.1 Experiment Setting

The experiment was conducted on over 8 H100 CUDA devices. Experiments to run one gray-box method overall model sizes take roughly 2 days. Experiments to run one black-box method over one model size take roughly 20 days, which means

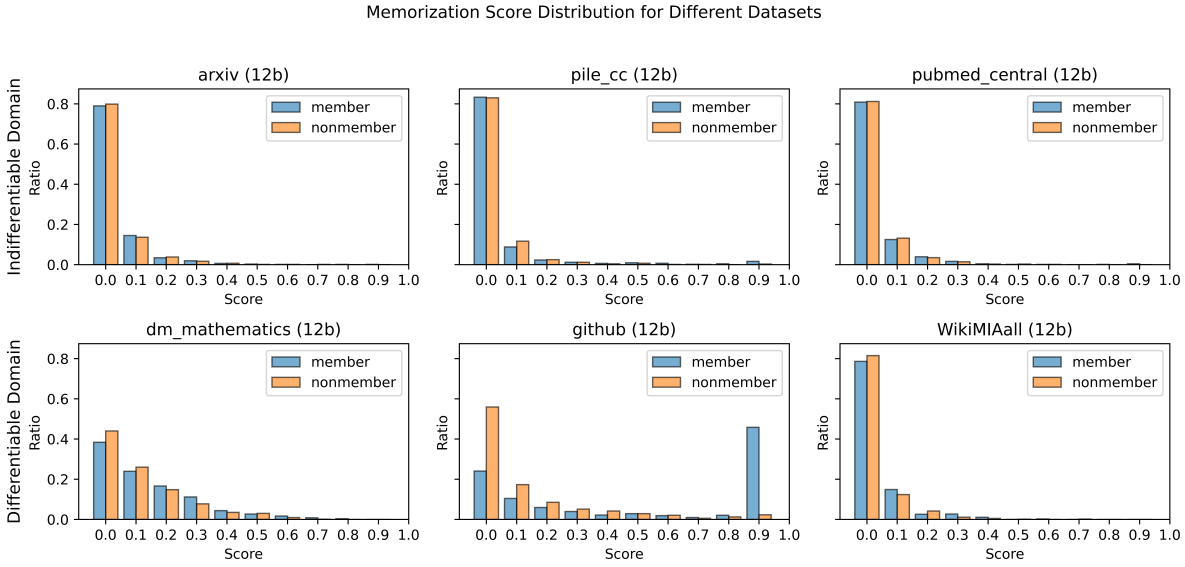

Figure 7: Sampled memorization score distribution in 12b model across domains.

the Black-Box methods require more heavy computation as they require the generation of tokens rather than directly taking the intermediate outputs like log-likelihood. Additionally, unlike Gray-Box methods that can input the entire sentence, the Black-Box requires the input of a partial sentence and then requires the LLM to generate the following continuations, in which the generation time will be largely delayed when the input sentence is long. The random seeds used to sample different member and non-member sets are [47103, 28103, 58320]. For Dolma experiments, we used five random seeds [47103, 28103, 58320, 423, 320] For calculating 7-gram overlap, we used Llama 2 and OLMo tokenizers to tokenize member and non-member texts in Pile and Dolma, respectively, following Liu et al. (2024). With tokenized texts, we counted the frequencies of 7-gram overlap between member and non-member texts in each split. We do not use the unique number of 7-gram overlap since we assumed that the frequencies of 7-gram overlap could explain more about text similarity. However, the frequencies of the 7-gram overlap appear to be dependent on the length of the target texts. For example, a member and non-member text length of 1,000 has possibly more members than a text length of 100. Therefore, we normalized it using the sum of the lengths of member and non-member texts to remove the effect brought by the text length.

The Transformer model that is used to predict member and non-member embedding is configured with 256 for its hidden dimension, 2 for the output prediction (member and non-member), 4 for the number of layers, and 8 for the number of attention heads. It is trained for four epochs with a binary classification objective to classify whether the given input is member text or non-member text.

The entropy is collected by inputting the previous N tokens and asking the LLM to generate the $N+1$ tokens, and then we input the previous $N+1$ tokens. This process is repeated before reaching the specified text length, which is set as 36 in this experiment.

A.2 Experiment Setting for MIA Method

For the reference model, we use the best reference model based on previous research (Duan et al., 2024). For the Min-k%and Min-k% ++, we choose K as 20, which means 20% of $\mathrm{Box}(x)$ are selected from the whole input tokens. This metric is used in their research paper and repositories.[13][14]

For the DC-PDD, there is no hyperparameter, and it relies on a pre-computed token frequency from corpora, which is not released at the time of writing. To reproduce this study, we used the infini-gram package [15] as the pre-computed frequency. However, their frequency is computed over the LLaMa tokenizer, which is different from that of

[13] https://github.com/zjysteven/mink-plus-plus

[14] https://github.com/swj0419/detect-pretrain-code

[15] https://infini-gram.io/pkg_doc.html

the Pythia tokenizer. We have to align their results, but this causes inevitable errors, which we cannot manage since the frequency is computed on a different tokenizer, and a sentence may be tokenized into different tokens based on the tokenizer.

For the EDA-PAC, the percentage of words that are swapped is set as 30%, and collect five perturbed sentences are collected.

For the RECALL, the number of shots (the number of prefixes) inserted into the input text is set as 12 while the maximum length is 1,000. If the prefix combines the input text above the max length of the model, we decrease the number of prompts gradually until the length is acceptable.

For the SaMIA, for a given input prefix, we generate the ten possible continuations with 0.8 temperature, which follows the setting in their original repository. [16] The model that is used to calculate the semantic similarity is BLUERT-20.[17] As the BLUERT-20 only accepts token lengths up to 512, we are only able to run it up to a length of 500 for this method.

For the CDD, for a given input prefix, we also generate ten possible continuations with a 0.8 temperature.

For both SaMIA and CDD methods, the maximum input tokens are 512 based on the input text length, and we generate the rest of the tokens based on the difference with the maximum text length setting in our experiments (1,000 tokens).

A.3 Available Domains in Each Split Method

We have the Truncate, Complete, and Relative split methods over the input text of all domains in the Pile corpora. We only keep those splits that have at least 100 examples for both member and non-member text at all text lengths. If a domain does not meet this requirement, it will be discarded. The available domains for all those split methods are presented in the following Table 5a and 5b.

For each MIA method, the results are run on all of its split methods, length range, model size, and random seeds. We are also able to see that the Complete splitting methods have the lease domains as the whole length of a text is a strict standard. Additionally, we also see that the Relative split has the most domains, as this split method suits the distribution of the target domains. Thus, most data are kept using this split method while following the text distribution.

A.4 Detailed Correlation Analysis

A.4.1 Kendall-Tau and Pearson Correlation Analysis in Pythia

For the correlation coefficient, we also use two common metrics, Kendall-Tau and Pearson, to evaluate it. The results are in Table 6 and Table 7. For the Pearson correlation, though it is originally applied to continuous values, as the text length is an increasing ordinal value, the Pearson correlation can also be used. From the results, we can see that:

There is a slight change due to a different computation method, which leads to a small fluctuation in results. For example, there is a decrease in the positive coefficient when compared to the Spearman correlation coefficient in the Kendall-Tau computation method.

However, the general trend in results remains unchanged, which generally aligns with the results in Table 3. The symbol of average value remains the same as in the original Spearman correlation analyses, which is still maintained in both Kendall-Tau and Pearson correlation analyses. The SaMIA retains its negative relation among other positive relations in Text Length, and Min-$k\%$ ++ still has both the highest and lowest values in the correlation coefficient for Text Length and Text Similarity.

A.4.2 Hypothesis Test over Correlation Analysis

To add further discussion on the analysis of the correlation coefficient, we conduct a significance test on the calculated correlation coefficients. For the text length, the null hypothesis is that there is no correlation between the ROC-AUC value and the targeted factor (text length or text similarity). For the alternative hypothesis, we had assumptions about an expected relation between ROC-AUC performance with text length (positive) and text similarity (negative). Therefore, we choose the one-sided alternative hypothesis rather than the two-sided with a hypothesis for a positive correlation coefficient in text length and a negative correlation in text similarity. We chose the commonly used P-value of 0.05 to conduct the significance test. Since per-domain has limited samples, which may not provide valid significance test results, we performed the MIA method significance test.

[16] https://github.com/nlp-titech/samia

[17] https://huggingface.co/lucadiliello/BLEURT-20

Name	Category	Calculated Feature
Loss	Gray-Box	Example Loss
Perplexity	Gray-Box	Example Perplexity
Gradient	Gray-Box	Example Gradient
Reference Model	Gray-Box	$\text{Loss}_{\text{Target}} - \text{Loss}_{\text{Reference}}$
Zlib Entropy	Gray-Box	$\frac{\text{Loss}}{\text{Entropy(Text)}}$
Min-k% Prob	Gray-Box	$\frac{1}{N}\sum_{i=1}^{N} \log P(w_i \mid w_{<i})$ for $w_i \in$ Bottom-K%
Min-k% Prob ++	Gray-Box	$\frac{\text{Min-}k\%\text{ Prob} - \mu}{\sigma}$
DC-PDD	Gray-Box	Compare the decoded log-likelihood with statistics from large corpora.
ReCaLL	Gray-Box	$\frac{LL(x\|p)}{LL(x)}$ when insert p text prefix
SaMIA	Black-Box	$\frac{1}{N}\sum_{i=1}^{N}$ SemanticDistance(g_i, a) .
CDD	Black-Box	$\frac{1}{N}\sum_{i=1}^{N}$ EditDistance(g_i, a).

Table 4: Collection of MIA methods evaluated in this study.

(a) Domains in shared split methods.

Split Method	**Shared Domains**
Truncated	Wikipedia (en), StackExchange, Pile-CC, GitHub, FreeLaw
Complete	Wikipedia (en), StackExchange, Pile-CC, GitHub, FreeLaw
Relative	Wikipedia (en), StackExchange, Pile-CC, GitHub, FreeLaw

(b) Domains in specific split methods.

Split Method	**Specific Domains**
Truncated	PubMed Central, HackNews, EuroParl, DM Mathematics, arXiv
Complete	USPTO Backgrounds
Relative	PubMed Central, NIH ExPorter, HackNews, Enron Mails, DM Mathematics, arXiv

Table 5: Domains included in different split methods across shared and specific datasets.

A.5 Memorization and MIA

A.5.1 Memorization Score Sample Distribution

In this figure, we saw that LLM does not show a very obvious distribution gap for most of the domains. However, we notice that in the GitHub domain, many texts show high memorization scores, meaning that most of the texts are well-memorized by the LLM. A similar trend is also observed in DM Mathematics, where most of the texts are not distributed in the low memorization score area (memorization score 0 - 0.2). A similarity between those two domains is that both of those two domains have low vocabulary diversity, considering that both DM Mathematics and GitHub are more oriented toward symbols (math) or fixed expressions (coding). Thus, it is easier for the LLM to memorize those texts.

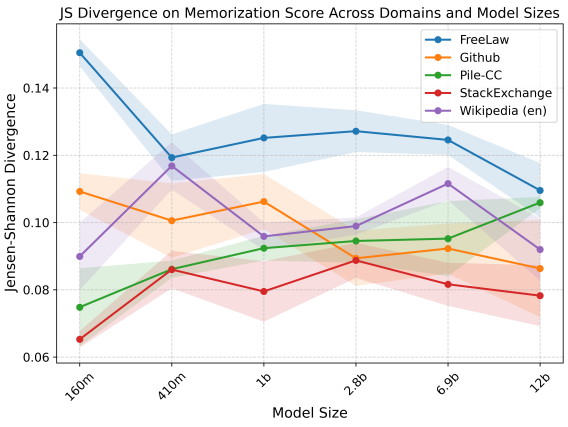

Figure 8: Memorization Score Distribution Divergence for Member and Non-Member Text

A.5.2 Memorization Score Distribution Distance

This section examines whether MIA performance is related to memorization by comparing the generated tokens with actual continuations when pro-

Method	Pythia: Text Length					Pythia: Text Similarity					OLMo: Text Length		OLMo: Text Similarity	
	Trunc	Comp	Rel	Avg	S-T	Trunc	Comp	Rel	Avg	S-T	Trunc	S-T	Trunc	S-T
Loss	.13	.14	.26	.18	✓	-.10	-.17	-.23	-.17	✓	.10	✓	-.20	✓
Refer	.14	.13	.22	.07	✓	-.13	-.12	-.12	.07	✓	–	-	–	-
Zlib	.18	.32	.19	.23	✓	-.19	-.13	-.15	-.16	✓	.15	✓	-.25	✓
Min-$k\%$	.30	.23	.36	.30	✓	-.09	-.11	-.18	-.13	✓	.22	✓	-.21	✓
Min-$k\%$ ++	.35	.35	.38	.36	✓	-.26	-.21	-.34	-.27	✓	.32	✓	-.28	✓
DC-PDD	.18	.08	.16	.14	✓	-.19	-.10	-.10	-.13	✓	.11	✓	-.17	✓
PAC	.05	.12	-.06	.04	✗	-.08	-.15	-.21	-.12	✓	.05	✗	-.10	✓
RECALL	.14	.14	.26	.18	✓	-.11	.07	.08	.02	✗	.19	✓	-.04	✗
SaMIA	-.12	-.04	-.06	-.08	✗	-.07	-.10	-.13	-.10	✓	-.18	✗	-.06	✗
CDD	-.02	.03	.03	.01	✗	-.12	-.13	-.11	-.12	✓	-.01	✗	-.24	✓
Avg	.13	.15	.17	.15	N/A	-.12	-.12	-.15	-.10	N/A	.10	N/A	-.19	N/A

Table 6: Kendall-Tau correlation coefficient between AUC and Text Length/Text Similarity across models.

Method	Pythia: Text Length					Pythia: Text Similarity					OLMo: Text Length		OLMo: Text Similarity	
	Trunc	Comp	Rel	Avg	S-T	Trunc	Comp	Rel	Avg	S-T	Trunc	S-T	Trunc	S-T
Loss	.10	11	.21	.14	✓	-.17	-.23	-.15	-.18	✓	.15	✓	-.16	✓
Refer	.12	.10	.28	.17	✓	-.05	-.20	-.32	-.19	✓	–	✗	–	✗
Zlib	.10	.23	.11	.11	✓	-.17	-.17	-.03	-.12	✓	.18	✓	-.31	✓
Min-$k\%$	.12	.20	.25	.19	✓	-.18	-.17	-.05	-.13	✓	.20	✓	-.20	✓
Min-$k\%$ ++	.26	.23	.22	.24	✓	-.17	-.26	-.32	-.25	✓	.22	✓	-.28	✓
DC-PDD	.13	.13	.16	.14	✓	-.17	-.18	-.13	-.16	✓	.17	✓	-.18	✓
EDA-PAC	.12	.20	-.10	.07	✗	-.11	-.31	-.04	-.15	✓	.12	✓	-.17	✓
RECALL	.08	.11	.17	.12	✓	-.15	.23	.11	.06	✗	.11	✓	-.01	✗
SaMIA	-.23	-.08	-.08	-.13	✗	-.18	-.04	.09	-.02	✗	-.09	✗	.03	✓
CDD	-.01	.02	.04	.02	✗	-.04	-.12	-.14	-.10	✗	.15	✓	-.25	✓
Avg	.08	.13	.13	.11	N/A	-.14	-.09	-.10	-.12	N/A	.12	N/A	-.19	N/A

Table 7: Pearson correlation coefficient between AUC and Text Length/Text Similarity across models.

moting 32 tokens, known as the K-extractable score, for both non-member and member text. We compute the distribution distance between the K-extractable score over the different domains on member and non-member text using JS divergence, as shown in Figure 8. We can see a correlation between MIA performance and the JS distribution difference between member and non-member texts. The FreeLaw has the highest JS Divergence score among those domains, suggesting that the memorization score distribution between member text and non-member text is large. This aligns with the ROC-AUC score density distribution in Figure 2. Additionally, we also see that GitHub and StackExchange have a low divergence, meaning their memorization score distribution between member text and non-member text is small, which makes it hard to differentiate.

A.6 Membership Inference Attack as Hypothesis Test

Besides directly analyzing the probability density function of the ROC-AUC scores, we also try to look at the MIA from a hypothesis test perspective. We treat the feature scores of member and non-member texts as two distributions and use a hypothesis test to verify whether those two distributions are the same or not. If a distribution passes such verification, it at least means the feature score distribution of the member is different from the feature distribution of the non-member text. Even though it does not guarantee any MIA performance, it does not directly evaluate MIA performance; it just shows whether those two distributions are the same or not. Such analysis at least provides a perspective to look at MIA differently. We divide the number of splits whose feature scores of member and non-member passed the verification into two distributions with the total number of splits. The results are presented from Table 8 to 10.

1. Similar to MIA performance, we observed that the number of splits that pass the hypothesis test increases with the model size. This confirms the analysis of the results of the ROC-AUC score using the probability density functions.

2. Further, we also see that in this evaluation metrics, the best-performed method Min-$k\%$ ++ does not also show the best performances in passing the hypothesis test. On the contrary, the best-performed MIA method is the Refer, which actually has the lowest performance in the ROC-AUC analysis. The reason is that the hypothesis test method does not evaluate whether the two examples are separate or not; it evaluates how those two distributions, consisting of the members and non-members, are the same distribution or not. This

Method	160m	410m	1b	2.8b	6.9b	12b
Loss	0.08	0.067	0.087	0.107	0.120	0.167
Min-$k\%$	0.08	0.087	0.073	0.153	0.207	0.220
Zlib	0.013	0.020	0.040	0.080	0.113	0.173
SaMIA	0.093	0.073	0.053	0.080	0.060	0.080
Min-$k\%$ ++	0.033	0.073	0.133	0.273	0.453	0.567
Refer	0.040	0.040	0.113	0.533	0.740	0.787
Grad	0.033	0.033	0.053	0.133	0.160	0.047
DC-PDD	0.073	0.080	0.087	0.160	0.140	0.240
CDD	0.033	0.060	0.093	0.027	0.087	0.060
RECALL	0.093	0.067	0.107	0.140	0.140	0.167

Table 8: Hypothesis test results across MIA methods and model size in the Relative Split method.

Method	160m	410m	1b	2.8b	6.9b	12b
Loss	0.033	0.053	0.080	0.120	0.140	0.193
Min-K	0.033	0.073	0.113	0.187	0.227	0.313
Zlib	0.040	0.027	0.040	0.107	0.167	0.260
SaMIA	0.180	0.153	0.160	0.153	0.140	0.140
Min-$k\%$ ++	0.053	0.027	0.093	0.293	0.460	0.553
Refer	0.027	0.020	0.147	0.527	0.733	0.760
Grad	0.020	0.060	0.047	0.073	0.067	0.060
DC-PDD	0.060	0.073	0.100	0.187	0.240	0.360
CDD	0.033	0.040	0.053	0.047	0.080	0.040
EDA-PAC	0.033	0.080	0.060	0.047	0.060	0.047
RECALL	0.053	0.060	0.087	0.127	0.133	0.207

Table 9: Hypothesis test results across MIA methods and model size in Truncate Split method.

means that they do not consider separating a specific example, but focus on identifying those two distributions.

3. Even though the hypothesis test does not provide a method to differentiate members and non-members specifically. It tells the performance of MIA from another perspective, whereas the previous worst-performing method could actually have the best performance. This shows the importance of evaluating the MIA method from multiple perspectives rather than only focusing on certain metrics, which could be misleading.

4. In this metric, we are also able to observe the same performance boost when transferring from the 1b to the 2.8b model. This aligns with the observation in the probability density analysis of ROC-AUC scores across dimensions, which confirms the emergent embedding change that we have discovered.

A.7 Detailed Results in Each Split Method

In this section, we present the detailed results for Truncate Split, Complete Split, and Relative Split. Each split contains all available domains. We shot the probability density in the Domain, Model Size, and MIA Method dimension in Figure 9.

1. In the first row, which shows the probability density over domains, we saw some more high-performance domains. For example, in the Truncate split, the EuroParl performs very well compared to other domains. One of the reasons may be that the EuroParl contains some non-English texts, which serve as an important feature for the member and non-member classification. Still, in the relative split, we are able to see more domains with relatively high MIA performance compared to other domains, which helps to explain why the Relative Split can give better performance.

Method	160m	410m	1b	2.8b	6.9b	12b
Loss	0.053	0.033	0.073	0.127	0.100	0.113
Min-K	0.087	0.080	0.080	0.153	0.147	0.247
Zlib	0.100	0.060	0.060	0.147	0.147	0.200
SaMIA	0.120	0.153	0.153	0.113	0.120	0.153
Min-$k\%$ ++	0.053	0.087	0.133	0.387	0.413	0.547
Refer	0.040	0.040	0.107	0.500	0.607	0.633
Grad	0.027	0.020	0.027	0.193	0.147	0.053
DC-PDD	0.067	0.073	0.073	0.200	0.213	0.300
CDD	0.040	0.093	0.060	0.060	0.013	0.060
EDA-PAC	0.067	0.060	0.047	0.067	0.033	0.027
RECALL	0.053	0.033	0.073	0.127	0.100	0.113

Table 10: Hypothesis test results across MIA methods and model size in Complete Split method.

2. In the second row, which shows the probability density over model sizes, we saw a uniform performance across different splits where the MIA performance positively scales with the model size.

3. In the third row, which shows the probability density over the different MIA methods. We are also able to observe some split-based differences. In the Relative and Complete split, we can see that the Min-$k\%$ ++ performs better than other methods. However, in the Truncate split, we see mixed results where most methods do not show obvious performance differences, where the Min-$k\%$ ++ is no longer significantly better than other methods.

A.8 Boxplot of the threshold for other MIA methods

In this section, we show the boxplot of the threshold for other MIA methods across model sizes and domains in Figure 10. From this figure, we can obtain the following:

1. The threshold still changes across domains, with the existence of outliers for all those methods. The Refer method shows an extreme trend where the threshold in each domain is totally different, indicating that a threshold decided from another domain totally fails to generalize to other domains.

2. Regarding the model size, we still observe that their thresholds change across model sizes. However, the Refer model has a stable threshold

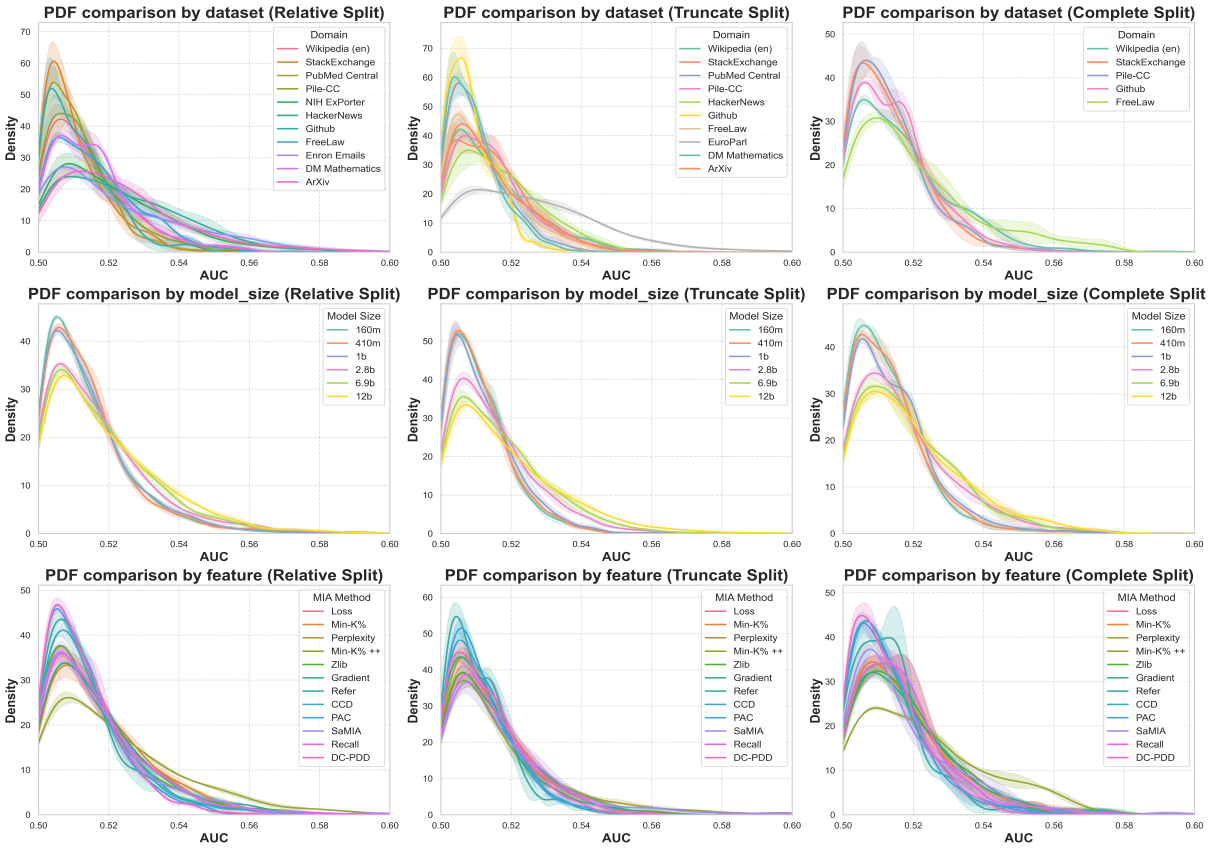

Figure 9: Detailed Results in Each Split Method

that is generalized well in other model sizes. This is probably because the referee relies on a reference model, which makes it less dependent on the target model. However, it still contains outliers, indicating that the threshold in one model size may not work in another model size, depending on the input members and non-members.

3. Additionally, while the trend is not general, we are able to see that the change of threshold is not random in some methods. We saw that the perplexity, zlib, Min-$k\%$, and DC-PDD all showed either a gradual increase or decreasing threshold values. This can increase the predictability of the threshold, making the decision of the threshold less random. However, even if it indicates some trend, it is still hard to make a correct prediction regarding how the threshold would change across model sizes.

A.9 Experiments on OLMo Series

A.9.1 Outliers Overlap Matrix

In this Figure 11, we are still able to conclude that there is no method that could have a high overlap with all other MIA methods. This confirms our conclusion that there is no "winner-takes-all" situation in the MIA study, as every hypothesis used by each MIA method has its pros and cons. Additionally, we are also able to see that a clear boundary exists between Gray-Box methods and Black-Box methods, where Gray-Box/Black-Box methods have a higher mutual overlap inside them compared to overlap with Black-Box/Gray-Box methods. This is due to the fundamental difference between those two types of methods, where Gray Box operates on the internal states, and Black-Box Box operates on the output tokens.

A.9.2 MIA as Hypothesis Test in OLMo

Method	1B	7B	13B
Loss	0.19	0.20	0.21
Gradient	0.14	0.16	0.18
Zlib	0.19	0.23	0.24
SaMIA	0.32	0.33	0.32
Min-$k\%$	0.23	0.24	0.24
Min-$k\%$ ++	0.20	0.24	0.29
DC-PDD	0.23	0.20	0.18
CDD	0.38	0.43	0.45
EDA-PAC	0.13	0.13	0.14
RECALL	0.19	0.20	0.22

Table 11: Ratio of experiments that passed hypothesis tests across model sizes in different MIA methods in the Truncate split in OLMo models.

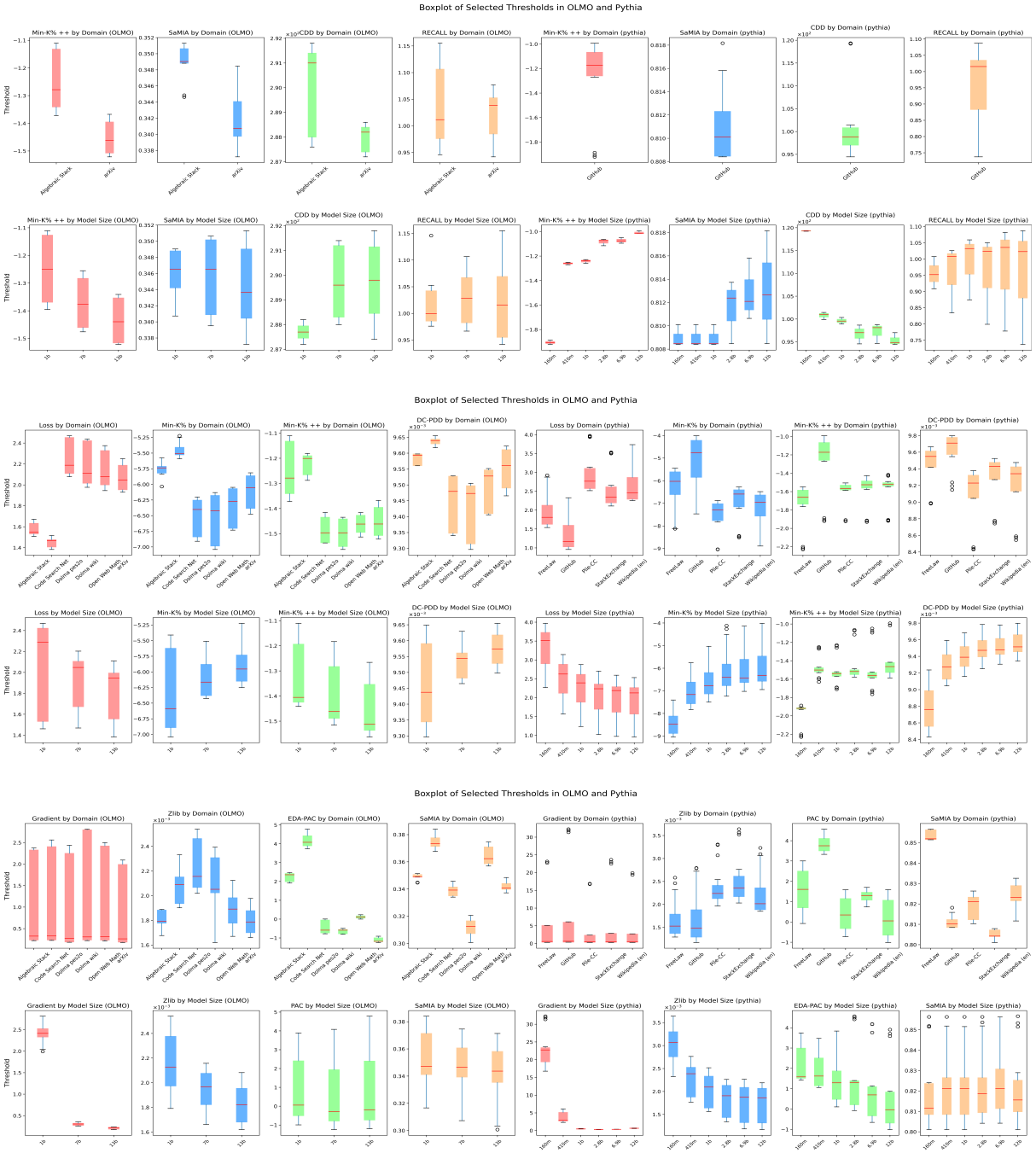

Figure 10: Threshold boxplot of other MIA methods across model sizes and domains in OLMo and Pythia

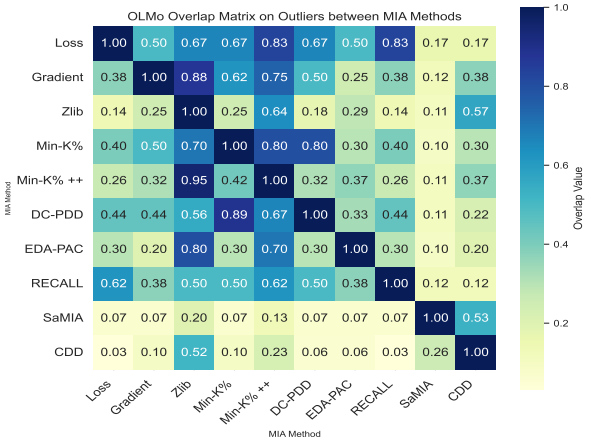

Figure 11: MIA outliers overlap matrix across methods in OLMo models.

In this experiment, we also conducted experiments using hypothesis tests in OLMo models. We can see that the results are a little bit different from the probability density function results. The reason may be that the experiment that passed the hypothesis test may already have a certain separability between members and non-members. Therefore, the results are based on those experiments that already have separability. In this hypothesis test, the best performance method is CDD, SaMIA, and Min-$k\%$ ++. We observed that Black-Box methods performed better than Gray-Box methods in this evaluation. One explanation may be that the OLMo method is a much stronger generation method compared to Pythia, as it is trained on much larger corpora. With a higher generation ability, the Black-Box method performs better than the Gray-Box method as LLM starts to behave differently in generations.

A.10 Numerical Results for Statistical Analysis

In this section, we provide the calculated numerical probability mass for the probability density results presented in the main content page. We have the following metrics:

MEAN: Average AUC value between 0.5–0.58 (outliers analysed in Section 4.2).

PROB MASS: Probability mass (area under the density curve) between 0.5–0.54. Lower values indicate fewer small AUC results.

Median: Median AUC value between 0.5–0.58.

Q1 (25%): 25th percentile of AUC results.

Q3 (75%): 75th percentile of AUC results.

STD: Standard deviation (provided only as a reference due to uneven distribution).

Bold values indicate the highest *Mean* and lowest *PROB MASS* within each table.

Table 12 Pythia Truncation The Relative split achieves the highest mean value and lowest probability mass, aligning with the experiment's conclusion. From the results, we can see there is a significant improvement in the probability mass when comparing the Relative split method and the Truncate split method, showing that using the Relative split could increase performance. However, the difference between Complete and Relative is very close; it shows whether semantically complete is a more important factor than comparing the sample relatively based on domain statistics. The reason may be that in the pre-training data of LLM, the domain is a very wide concept, usually with tens of gigabytes and much text noise. Thus, doing this relative sampling based on domain length does not perfectly make sure such sampling fits with the actual text length distribution; thus, the benefit in MIA performance is not obvious. Still, the completeness of semantics affects the MIA performance significantly.

Table 13 Pythia Model Size Mean performance increases consistently with model size. Notably, a significant drop in probability mass and a substantial increase in mean occur when scaling from 1b to 2.8b, confirming enhanced MIA performance with larger models. This significant boost also aligns with a similar change in the probability density figure in Figure 2 (b).

Table 14 Pythia Domain FreeLaw demonstrates the best performance (highest mean, lowest probability mass). Wikipedia (en) ranks second, while GitHub exhibits the lowest performance metrics. This analysis does not change when using the numerical values.

Table 15 Pythia Feature Min-K% ++ notably outperforms all other MIA methods, having the highest mean and significantly lowest probability mass. Min-K% and RECALL follow closely behind. The DC-PDD and EDA-PAC show no difference in the mean value and probability mass. Other methods show close or nearly the same performance as baseline methods (Loss, Zlib, Refer, Gradient), indicating that the reported good performance of some MIA methods in previous research may be an illusion.

TRUNCATION	MEAN	PROB MASS	MEDIAN	Q1 (25%)	Q3 (75%)	STD
Truncate	0.5130	0.9401	0.5107	0.5050	0.5182	0.0106
Complete	0.5156	0.9195	0.5127	0.5060	0.5184	0.0129
Relative	**0.5164**	**0.9030**	0.5106	0.5058	0.5185	0.0109

Table 12: Pythia model truncation analysis. Bold values denote the highest Mean and the lowest PROB MASS.

MODEL SIZE	MEAN	PROB MASS	MEDIAN	Q1 (25%)	Q3 (75%)	STD
160m	0.5115	0.9443	0.5093	0.5044	0.5164	0.0094
410m	0.5120	0.9485	0.5101	0.5047	0.5167	0.0096
1b	0.5126	0.9443	0.5105	0.5048	0.5180	0.0101
2.8b	0.5152	0.9089	0.5123	0.5058	0.5215	0.0125
6.9b	0.5164	0.9057	0.5136	0.5062	0.5236	0.0130
12b	**0.5175**	**0.8788**	0.5139	0.5065	0.5250	0.0141

Table 13: Pythia model size analysis.

Table 16 OLMo Size For the model size scaling in the OLMo model, we also saw the performance increase with the model size. The mean value increases, and the probability mass decreases. The probability mass and mean value change between 1b $\rightarrow$ 7b is also more obvious than that of 7b $\rightarrow$ 13b, showing a similar trend with the Pythia model series. Interestingly, in the probability mass metric, we are able to see that its performance is better than the comparable model size in the Pythia series. The 1b, 7b, and 13b model sizes in the OLMo series all have a lower probability mass, showing better statistical performance. As OLMo is an instruction-tuned LLM and Pythia is a pre-trained LLM, OLMo has a stronger ability in natural language; this suggests not only model scaling but also the post-instruction tuning may positively benefit the MIA performance. One possible reason for this may be the instruction tuning or additional training after the pre-train stage, which may help the LLM to recall its knowledge during the pre-train stage. Some research also discusses similar findings on how instruction/post-fine-tuning could help LLM reorganize its learned knowledge during pre-training and increase the chance of eliciting its knowledge learned in pre-training.[1, 2, 3, 4]

Table 17 OLMo Domain This table basically aligns with its corresponding Figure 2 (f), where the Dolma pes2o shows the best performance with the highest mean value and lowest probability mass. The reason is still about the text diversity. The pes2o contains various research papers comparing the arXiv, which only contains scientific research papers. The Code Search Net, Open Web Math, and Algebraic Stack have low textual diversity.

Table 18 OLMo Feature In the OLMo models, the absolute best method among evaluation methods does not exist. The Min-K% ++ has the best average, while the CDD has the best probability mass. We saw that SaMIA and CDD have much better results in the OLMo models; this is probably because OLMo is much better at generating tokens due to the large pre-train data size and instruction tuning, so it is more sensitive to black-box MIA methods like SaMIA and CCD, while methods like Min-K% ++ and RECALL still show relatively decent performance among those two model series compared to baselines. This shows that their hypothesis about how to split members and non-members may be more general.

DATASET	MEAN	PROB MASS	MEDIAN	Q1 (25%)	Q3 (75%)	STD
FreeLaw	**0.5162**	**0.8966**	0.5129	0.5059	0.5228	0.0137
Github	0.5117	0.9452	0.5094	0.5045	0.5168	0.0095
Pile-CC	0.5137	0.9352	0.5114	0.5056	0.5187	0.0108
StackExchange	0.5129	0.9418	0.5107	0.5048	0.5185	0.0103
Wikipedia (en)	0.5153	0.9087	0.5123	0.5056	0.5217	0.0126

Table 14: Pythia domain analysis.

FEATURE	MEAN	PROB MASS	MEDIAN	Q1 (25%)	Q3 (75%)	STD
Loss	0.5146	0.9220	0.5124	0.5057	0.5204	0.0116
Zlib	0.5141	0.9189	0.5114	0.5053	0.5200	0.0116
Gradient	0.5142	0.9148	0.5115	0.5053	0.5197	0.0118
Refer	0.5121	0.9409	0.5103	0.5046	0.5167	0.0099
Min-K%	0.5156	0.9083	0.5125	0.5061	0.5219	0.0124
Min-K% ++	**0.5204**	**0.8059**	0.5159	0.5069	0.5299	0.0166
DC-PDD	0.5130	0.9467	0.5109	0.5050	0.5184	0.0102
EDA-PAC	0.5110	0.9466	0.5090	0.5043	0.5155	0.0088
RECALL	0.5147	0.9103	0.5123	0.5058	0.5205	0.0117
SaMIA	0.5140	0.9021	0.5108	0.5049	0.5195	0.0121
CDD	0.5118	0.9421	0.5095	0.5047	0.5167	0.0096

Table 15: Pythia feature analysis.

MODEL SIZE	MEAN	PROB MASS	MEDIAN	Q1 (25%)	Q3 (75%)	STD
1b	0.5143	0.8938	0.5111	0.5047	0.5228	0.0148
7b	0.5153	0.8531	0.5123	0.5040	0.5225	0.0140
13b	**0.5159**	**0.8466**	0.5117	0.5044	0.5219	0.0132

Table 16: OLMo model size analysis.

DATASET	MEAN	PROB MASS	MEDIAN	Q1 (25%)	Q3 (75%)	STD
Algebraic Stack	0.5103	0.8860	0.5075	0.5032	0.5150	0.0102
arXiv	0.5155	0.8739	0.5133	0.5060	0.5230	0.0120
Code Search Net	0.5112	0.9038	0.5095	0.5036	0.5165	0.0100
Dolma pes2o	**0.5294**	**0.6440**	0.5286	0.5179	0.5407	0.0181
Dolma wiki	0.5164	0.8547	0.5153	0.5060	0.5244	0.0124
Open Web Math	0.5088	0.9074	0.5065	0.5028	0.5122	0.0084

Table 17: OLMo domain analysis.

FEATURE	MEAN	PROB MASS	MEDIAN	Q1 (25%)	Q3 (75%)	STD
Loss	0.5157	0.9075	0.5136	0.5057	0.5235	0.0120
Zlib	0.5158	0.7721	0.5129	0.5059	0.5229	0.0184
Gradient	0.5162	0.8907	0.5138	0.5062	0.5239	0.0123
Min-K%	0.5166	0.9053	0.5142	0.5063	0.5232	0.0123
Min-K% ++	**0.5168**	0.7108	0.5128	0.5061	0.5247	0.0137
DC-PDD	0.5153	0.9131	0.5124	0.5062	0.5218	0.0116
EDA-PAC	0.5132	0.9015	0.5091	0.5045	0.5189	0.0118
RECALL	0.5156	0.7228	0.5136	0.5056	0.5237	0.0121
SaMIA	0.5102	0.6511	0.5035	0.5000	0.5171	0.0135
CDD	0.5139	**0.6054**	0.5051	0.5000	0.5226	0.0187

Table 18: OLMo feature analysis.